

Project Report

Grab Delivery App

Submited By

Woraviboon Mungmee 6504062610234

Pongporn Yampradit 6504062620078

Thanop Sangjan 6504062620051

Mok Maard 6504062620159

Computer Science

This report is part of SYSTEM ANALYSIS & DESIGN 040613302  
Semester 1 Year of Education 2023

Table of Contents

[Grabfood 7](#_Toc148907798)

[System Request 7](#_Toc148907799)

[Project Sponsor :**Asst.prof.Khantharat Anekboon Ph.D.** 7](#_Toc148907800)

[Business Need : 7](#_Toc148907801)

[Business Requirement **:** 7](#_Toc148907802)

[Business Value **:** 8](#_Toc148907803)

[Special Issues or Constraints : 8](#_Toc148907804)

[Feasibility : 1](#_Toc148907805)

[Technical feasibility (can we build it? ) 1](#_Toc148907806)

[Economic Feasibility (should we build it ?) 2](#_Toc148907807)

[Crash flow 3](#_Toc148907808)

[Legal Feasibility 5](#_Toc148907809)

[Scheduling Feasibility 5](#_Toc148907810)

[Functional Requirement : 6](#_Toc148907811)

[Login and Registration System 6](#_Toc148907812)

[Restaurant Page 6](#_Toc148907813)

[Order Management 6](#_Toc148907814)

[Menu Management 6](#_Toc148907815)

[Order Calculation 6](#_Toc148907816)

[Order Display 6](#_Toc148907817)

[Order History: 6](#_Toc148907818)

[Driver Status Updates 6](#_Toc148907819)

[Admin Panel 6](#_Toc148907820)

[Alert Notifications: 7](#_Toc148907821)

[Non-Functional Requirement : 8](#_Toc148907822)

[● Performance 8](#_Toc148907823)

[○ Response 8](#_Toc148907824)

[● Sercurity 8](#_Toc148907825)

[○ Privacy 8](#_Toc148907826)

[○ Action 8](#_Toc148907827)

[● Usability 8](#_Toc148907828)

[○ Customization 8](#_Toc148907829)

[○ Adaptability 9](#_Toc148907830)

[Requirement Gathering : 9](#_Toc148907831)

[● Rider 9](#_Toc148907832)

[○ Process Overview 9](#_Toc148907833)

[Desired Feature : 10](#_Toc148907834)

[Advantage: 10](#_Toc148907835)

[Disadvantage: 10](#_Toc148907836)

[● Restaurants 11](#_Toc148907837)

[○ Process Overview 11](#_Toc148907838)

[Desired Feature : 12](#_Toc148907839)

[Advantages: 12](#_Toc148907840)

[Disadvantage : 12](#_Toc148907841)

[Customer 12](#_Toc148907842)

[○ Process Overview 12](#_Toc148907843)

[Desired Feature : 12](#_Toc148907844)

[Advantages : 13](#_Toc148907845)

[Disadvantage : 13](#_Toc148907846)

[Project Schedule 14](#_Toc148907847)

[Usecase Diagram 16](#_Toc148907848)

[Use Case Description 18](#_Toc148907849)

[Use Case Description : Create Account 18](#_Toc148907850)

[Use Case Description : Login 19](#_Toc148907851)

[Use Case Description : Update Account 20](#_Toc148907852)

[Use Case Description : Delete Account 21](#_Toc148907853)

[Use Case Description : Create Cart 22](#_Toc148907854)

[Use Case Description : Submit Order in cart 23](#_Toc148907855)

[Use Case Description : Update Cart 24](#_Toc148907856)

[Use Case Description : Delete Order 25](#_Toc148907857)

[Use Case Description : Add Food 25](#_Toc148907858)

[Use Case Description : Update Food 27](#_Toc148907859)

[Use Case Description : Delete Food 28](#_Toc148907860)

[Use Case Description : Order History 29](#_Toc148907861)

[Use Case Description : Finish Order 30](#_Toc148907862)

[Use Case Description : Accept Customer Order 31](#_Toc148907863)

[Use Case Description : Revenue 32](#_Toc148907864)

[Use Case Description : Pay 33](#_Toc148907865)

[Use Case Description : Add food to Cart 34](#_Toc148907866)

[Use Case Description : Finish Order 35](#_Toc148907867)

[Use Case Description : Look up order history 36](#_Toc148907868)

[Use Case Description : Look up order 37](#_Toc148907869)

[Use Case Description : Chat 38](#_Toc148907870)

[Use Case Description : Receive Money 39](#_Toc148907871)

[Activity Diagram 42](#_Toc148907872)

[1. Create User 42](#_Toc148907873)

[2. Login 43](#_Toc148907874)

[43](#_Toc148907875)

[3. Look Up Order 44](#_Toc148907876)

[4. Chat 45](#_Toc148907877)

[**5. Update Account** 46](#_Toc148907878)

[6. Delete Account 47](#_Toc148907879)

[7. Receive Money 48](#_Toc148907880)

[8. Update Cart 48](#_Toc148907881)

[9. Delete Cart 49](#_Toc148907882)

[10. Add Food to Cart 50](#_Toc148907883)

[11. Submit Order in Cart 51](#_Toc148907884)

[12. Pay 52](#_Toc148907885)

[13. Accept Customer Order 54](#_Toc148907886)

[14. Finish Order 54](#_Toc148907887)

[15. Add Food 55](#_Toc148907888)

[16. Update Food 57](#_Toc148907889)

[17. Delete Food 58](#_Toc148907890)

[Sequence Diagram 59](#_Toc148907891)

[1. Submit Cart 59](#_Toc148907892)

[2. Update Food 59](#_Toc148907893)

[3. Create User 60](#_Toc148907894)

[4 . Chat 60](#_Toc148907895)

[Context Diagram 61](#_Toc148907896)

[Error! Bookmark not defined.](#_Toc148907897)

[Data Flow Diagram 62](#_Toc148907898)

[1. DFD level0 62](#_Toc148907899)

[2. DFD level 1 for process 2 63](#_Toc148907900)

[3. DFD level 1 for process 10 63](#_Toc148907901)

[ER Diagram 64](#_Toc148907902)

[Class Diagram 65](#_Toc148907903)

[Microsoft Access Table 66](#_Toc148907904)

[User Interface 70](#_Toc148907905)

[1. Sign Up interface 70](#_Toc148907906)

[2. Login Interface 72](#_Toc148907907)

[3. Restaurant Interface 74](#_Toc148907908)

[4. choose food 76](#_Toc148907909)

|  |
| --- |
| GrabfoodSystem Request |
| Project Sponsor :**Asst.prof.Khantharat Anekboon Ph.D.** |
| Business Need : This project has been developed to provide convenience to customers who wish to order food from restaurants without having to physically visit the restaurant. It also creates a new job opportunity, the role of a "Rider" who will handle the delivery of food from the restaurant to the customers. |
| Business Requirement **:**       This project will be divided into three components: customers, restaurants, and food delivery drivers. Customers**:**  * Customers can place food orders based on the participating restaurants listed on GrabFood.  Restaurants**:**  * Participating restaurants have the capability to prepare   food according to the orders placed by customers. For customers interested in becoming partners, they can apply for partnership using one email per one ID.   * Restaurants can also add food items to their menu.  Food Delivery Drivers (Grab Bike)**:**  * Accepting food order requests. * Utilizing GPS to navigate to the restaurant's location and the customer's destination. |
| Business Value **:**  * I can provide a maximum compensation of 47,187,342 within 5 years. * The highest number of participating businesses is 142,840 stores within 5 years. * Obtained customer usage behavior data can be utilized to create marketing strategies resulting in a 20% increase in sales. |
| Special Issues or Constraints :  * One email per account, once an email has been used for registration, it can’t be used for another registration |

# **Feasibility :**

detailed business case for the project

## Technical feasibility (can we build it? )

* User and analysis familiarity with the business application area
* Familiarity with technology
* Computer science students who study both fundamental and advanced programming may not encounter significant issues due to their strong familiarity with technology.
* Our team possesses working computers, internet access for research, and a cloud server dedicated to deploying projects.
* Project size
* Number of People: 4 individuals
* Duration: 3 months
* Customers can place food orders.
* Merchants can receive orders and add food items to their menu.
* Delivery drivers can accept orders and deliver them to their destinations.
* Compatibility with existing systems
* In the system for creating web applications, programming is developed primarily using the React language. It incorporates a database system and various other components.

In conclusion, creating a GrabFood-like website is achievable within a certain timeframe, given the available number of individuals and the current usable technologies.

## Economic Feasibility (should we build it ?)

* identify costs and benefits
* Investment costs : These include expenses related to computer hardware and equipment, network infrastructure, system design, and compensation.
* Expected Benefits : Potential Profit: There is a possibility of generating profits through the percentage allocation from participating merchants and delivery drivers. As the user base grows over time, potential profits could increase.
* Upon successful completion of this project, the potential for profitability is evident, and the business is likely to remain stable over time. This is due to its association with the four factors and human preference for convenience.

# **Crash flow**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Y0 | Y1 | Y2 | Y3 | Y4 | Y5 | Total |
| Benefit | | | | | | |  |
| Increase Used by Customer ( Processing Fee ) |  | 744,819 | 3,724,095 | 7,448,081 | 14,896,161 | 15,640,980 | 42,454,136 |
| Reduce Call Center Employee |  | 360,000.00 | 360,000.00 | 360,000.00 | 360,000.00 | 360,000.00 | 1,800,000.00 |
| Reduce Equipment Such As headset and telephone |  | 10,000.00 | 10,000.00 | 10,000.00 | 10,000.00 | 10,000.00 | 50,000.00 |
| Data from User to outline the Marketing plan |  | 10,000 | 20,000 | 40,500 | 100,000 | 590,000 | 760,500 |
| Advertisement Fee from Restaurant |  | 37,241 | 186,205 | 372,404 | 744,808 | 782,049 | 2,122,707 |
| Total Benefits |  | 1,162,060 | 4,300,300 | 8,230,985 | 16,110,969 | 17,383,029 | 47,187,342 |
| Development Costs | | | | | | | |
| 1 server | 1,250,000 | 0 | 0 | 0 | 0 | 0 | 1,250,000 |
| Development Labor | 4,800,000 | 0 | 0 | 0 | 0 | 0 | 4,800,000 |
| Software License | 20,000,000.00 | 0 | 0 | 0 | 0 | 0 | 20,000,000.00 |
| Total Development Costs | 26,050,000 | 0 | 0 | 0 | 0 | 0 | 26,050,000 |
| Operational Costs | | | | | | | |
| Maintenance service agreement | 0 | 1,000,000 | 1,000,000 | 1,000,000 | 1,000,000 | 1,000,000 | 5,000,000 |
| Operational Labor | 0 | 300,000 | 300,000 | 300,000 | 300,000 | 300,000 | 1,500,000 |
| Maintenance server | 0 | 100,000 | 100,000 | 100,000 | 100,000 | 100,000 | 500,000 |
| Total Operational Costs | 0 | 1,400,000 | 1,400,000 | 1,400,000 | 1,400,000 | 1,400,000 | 7,000,000 |
| Total Costs | 26,050,000 | 1,400,000 | 1,400,000 | 1,400,000 | 1,400,000 | 1,400,000 | 33,050,000 |
| Total Benefits - Total Costs | (26,050,000.00) | -237,940 | 2,900,300 | 6,830,985 | 14,710,969 | 15,983,029 | 14,137,342 |
| Cumulative Net Cash Flow | (26,050,000.00) | (26,287,940.05) | (23,387,640.30) | (16,556,655.78) | (1,845,686.73) | 14,137,342.28 | 28,274,684.55 |
| Return of Investment (ROI) | 42.78% |  |  |  |  |  |  |
| Break-Even Point | 4.13 Years |  |  |  |  |  |  |

## Legal Feasibility

* In reality, this project cannot be executed due to copyright constraints. However, as a case study, this project does not violate any laws whatsoever.

## Scheduling Feasibility

* The project submission is on October 16th, and the assessment date is September 13th, providing a total of 67 days for

completion. Considering that you are a student, you plan to work on it every Saturday and Sunday. With 67 days, which amounts to around 10 weeks, subtracting approximately 2 weeks for mid-term exams, you are left with 8 weeks. During these 8 weeks, you intend to work for 2 days per week, totaling 16 days. This approach seems feasible, especially since you already have a prototype in place from previous work. This will allow you to focus on refining and completing the additional requirements within the allotted time frame.

# **Functional Requirement :**

## Login and Registration System

For customers, restaurants, and drivers.

## Restaurant Page

Ability to categorize food types, specify quantities, and receive notifications for incoming orders.

## Order Management

Customers can cancel orders, and restaurants can modify and update their information.

## Menu Management

Restaurants can add, delete, and modify food items.

## Order Calculation

Calculate the individual and total prices of ordered food items.

## Order Display

Display customer orders or generate receipts (bills).

## Order History:

Store order history for reference.

## Driver Status Updates

Provide 3 driver statuses: searching for a driver, awaiting food, and delivering.

## Admin Panel

An admin page to manage all related information.

## Alert Notifications:

Alert boxes for actions like ordering and paymen

# **Non-Functional Requirement :**

## Performance

### Response

* Our system is designed to efficiently handle a high volume of incoming orders within a short timeframe, ensuring smooth and fast processing without any bottlenecks or slowdowns.
* We aim to provide seamless and swift responses to users, ensuring convenience and efficiency in meeting their needs.

## Sercurity

### Privacy

* We are committed to safeguarding all user personal data, preventing unauthorized access and ensuring data privacy.

### Action

* We have enhanced the payment security system to provide increased protection. Additionally, customers can input a PIN code to ensure a secure payment process.

## Usability

### Customization

* We offer the flexibility for users to customize the website theme according to their preferences and needs.

### Adaptability

* We provide the option to switch between a variety of languages to accommodate orders from different locations, ensuring a seamless ordering experience.

# **Requirement Gathering :**

## Rider

### Process Overview

1. Drivers await orders from customers. There are two methods for drivers to receive orders:

* If a driver is in automatic order-taking mode, the system will automatically assign orders to them.
* If a driver is not in automatic mode, the system will send a notification to prompt the driver to decide whether to accept or decline the order.

1. If a customer chooses to pay within the system, the system will automatically deduct the amount. Drivers do not need to handle payment to the restaurant. However, if the customer opts for cash payment or a deferred payment with the driver, the driver must have sufficient funds in their GrabPay account. This is necessary for the driver to pay the restaurant first before receiving the ordered items. Subsequently, the driver delivers the items to the customer and receives payment from them.

### Desired Feature :

Drivers want the money transfer system to be more reliable than before.

### Advantage:

The Grab app for drivers is compatible with both Android and iOS platforms, enabling drivers to use it comprehensively and access it easily.

### Disadvantage:

Transferring money from GrabPay to a bank account is relatively slow and takes at least 30 minutes or more.

**picture :**





## Restaurants

### Process Overview

1. Customers place their orders through the app. Once an order is placed, it is sent to the restaurant, and then the staff follows the items listed in the order as instructed.
2. If customers make a prepayment within the app, the system will automatically deduct the payment for the order. However, if payment hasn't been made, the customer must settle the payment via GrabPay before the order can proceed.

### Desired Feature :

When products are out of stock, a notification should be sent promptly to potential customers who are in the process of placing an order, ensuring real-time awareness.

### Advantages:

Facilitates sales without the need for a physical store, making it convenient to conduct business. Disadvantages: GrabFood charges a commission fee for food delivery, which reduces overall profits.

### Disadvantage :

The translation provided takes into consideration English grammar rules and structure. If you have any further questions or need additional assistance, feel free to ask.

## Customer

### Process Overview

1. Customers select the desired food items.
2. Customers choose their preferred payment method, with two options: cash payment and bank transfer.
3. Track the delivery status of the order from the courier.
4. Await the arrival of the ordered items.

### Desired Feature :

I would like the notification system to be more reliable. For instance, having the ability to notify users about the open or closed status of a restaurant to prevent orders from being placed when the restaurant is closed.

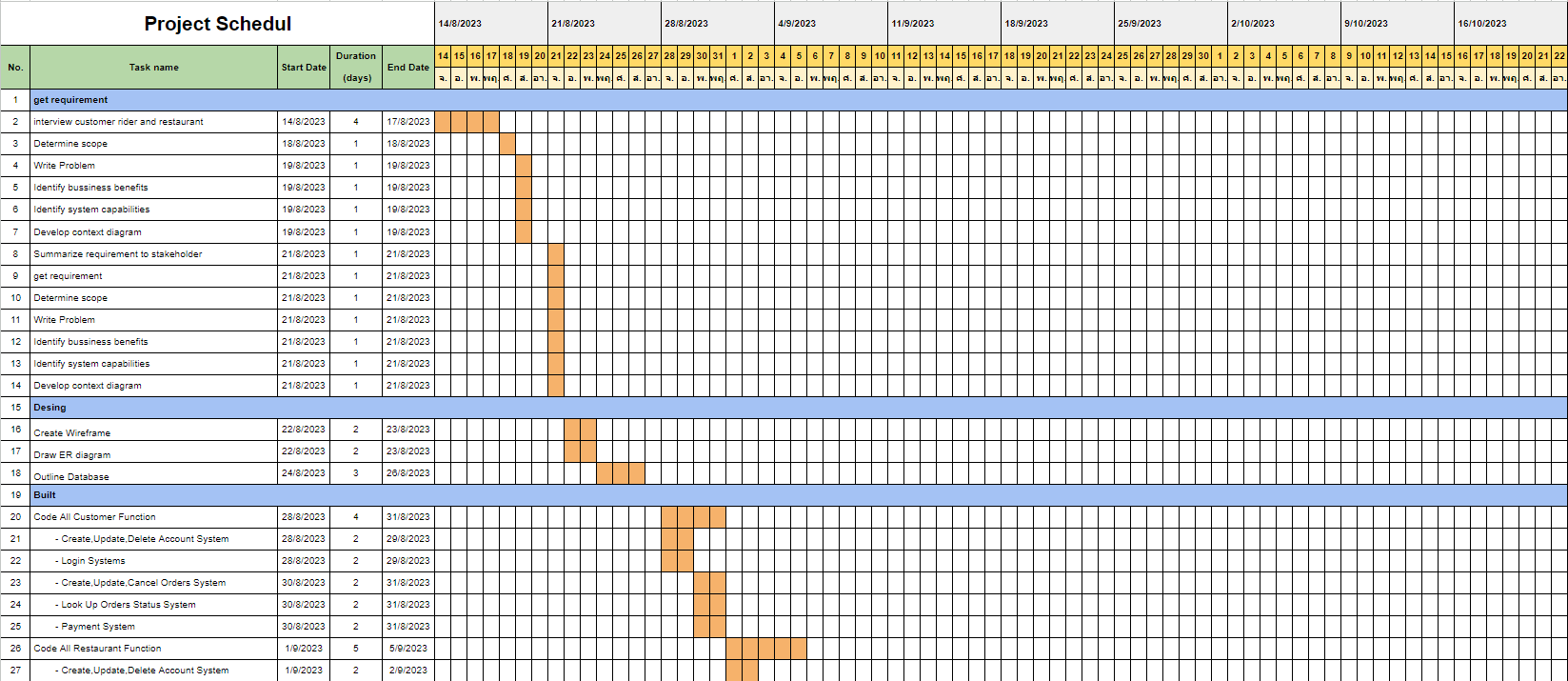
### Advantages :

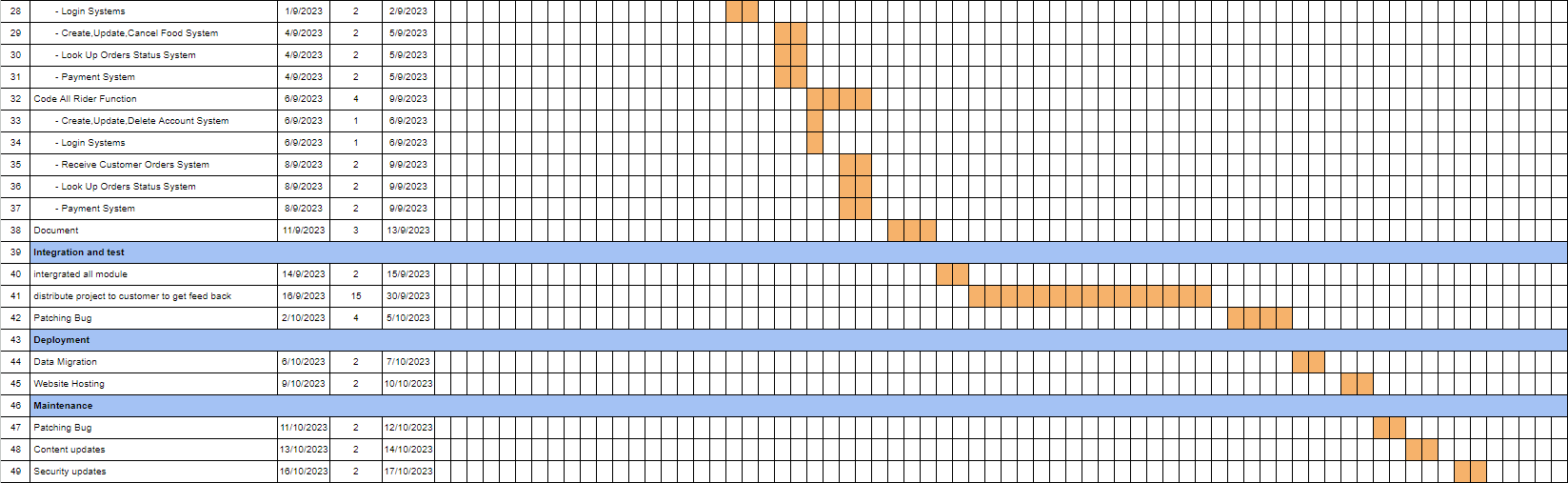
Easy food ordering, eliminating the need to physically wait at the restaurant. Disadvantages: Inability to verify food quality and the possibility of receiving incorrect orders.

### Disadvantage :

The translation provided takes into consideration English grammar rules and structure. If you have any further questions or need additional assistance, feel free to ask.

# **Project Schedule**





# **Usecase Diagram**

A diagram of a customer service

Description automatically generated

**A diagram of a delivery system

Description automatically generated**

# **Use Case Description**

## Use Case Description : Create Account

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Create Account | |
| **Scenario :** | Register to Grab | |
| **Triggering Event :** | User want to Register to Grab | |
| **Brief Description:** | It is a use case that allows a User to apply for an ID to log in. | |
| **Actors :** | Customer,Rider,Restaurant | |
| **Related Use Cases :** | - | |
| **Stakeholders :** | Customer,Rider,Restaurant | |
| **Preconditions :** | - | |
| **Post conditions :** | User has ID and Password to Login | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1. User press “Register” | 1.1 The system displays a screen for the user to apply for information. |
| 1. User fills in the form of the system. | 2.1 The system checks important information that has not been filled in. If not filled in, the system will notify the User. |
| 1. User press “Register” | 3.1 The system saved information |
| **Alternative Flow** | step 1,2 if User press “cancel” The system will bring user to homepage | |
| **Exception**  **Conditions :** | If the information is incomplete Must fill out the new system application information | |

## Use Case Description : Login

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Login | |
| **Scenario :** | User login | |
| **Triggering Event :** | User want to Login | |
| **Brief Description:** | It is a use case that allows a User to log in. | |
| **Actors :** | Customer ,Rider ,Restaurant | |
| **Related Use Cases :** | - | |
| **Stakeholders :** | NONE | |
| **Preconditions :** | Use case : Register | |
| **Post conditions :** | User has logged in | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1.Use case start when User press “Login” | 1.1 The system displays a screen for the user to fill in.  login information |
| 2.User fill out the from  (username ,password ) | 2.2 Check User login information |
| 3.User press “login” | 3.1 The system bring user to homepage |
| **Alternative Flow** | Step 1,2 if user press “ register ” System will dispaly use case : register | |
| **Exception**  **Conditions :** | If the information is incomplete Must fill out the new system application information | |

## Use Case Description : Update Account

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Update Account | |
| **Scenario :** | Edit Account | |
| **Triggering Event :** | User want to editing Account | |
| **Brief Description:** | It is the use case that allows the User to edit data in the system. | |
| **Actors :** | Customer ,Rider ,Restaurant | |
| **Related Use Cases :** | - | |
| **Stakeholders :** | NONE | |
| **Preconditions :** | Use case : Login , Register | |
| **Post conditions :** | User was edited User information | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1.Use case start when User  press “Edit” in profile page | 1.1 The system will display user information |
| 2.User Edit the information that needs to be corrected. |  |
| 3.User press “submit” | 3.1 the system check information from user |
| 3.2 the system saved information |
| **Alternative Flow** | step 1,2 if User press “cancel” The system will bring user to homepage | |
| **Exception**  **Conditions :** | Step : 3.1 If User enters information (phone number and Email) repeatedly, the system will display a message. “This username already exists” to the user | |

## Use Case Description : Delete Account

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Delete Account | |
| **Scenario :** | delet Account | |
| **Triggering Event :** | User want to delete Account | |
| **Brief Description:** | It is use case that allow User for deleting account | |
| **Actors :** | Customer ,Rider ,Restaurant | |
| **Related Use Cases :** | - | |
| **Stakeholders :** | NONE | |
| **Preconditions :** | Use case : Register , Login | |
| **Post conditions :** | User was deleted account | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1.Use case start when User  press “Delete Account” in profile page | 1.1The system display Message “Are you sure you want to delete your account?” |
| 2. user press “submit” | 2.1 The system will delete User account |
|  | 2.2 The system will bring user to homepage |
| **Alternative Flow** | step : 2 If the user does not press submit but then presses "Back", the system will take the user back to their profile page. | |
| **Exception**  **Conditions :** | NONE | |

## Use Case Description : Create Cart

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Create Cart | |
| **Scenario :** | Create Order | |
| **Triggering Event :** | User want to Order Food | |
| **Brief Description:** | It is use case that allow Customer to Order food | |
| **Actors :** | Customer | |
| **Related Use Cases :** | NONE | |
| **Stakeholders :** | NONE | |
| **Preconditions :** | use case : Login | |
| **Post conditions :** | User has Cart | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1.Use case start when User press “Add cart” | 1.1 The system will add food to Cart |
| **Alternative Flow** | NONE | |
| **Exception**  **Conditions :** | NONE | |

## Use Case Description : Submit Order in cart

| **Use Case Name :** | Submit Order in cart | |
| --- | --- | --- |
| **Scenario :** | Submit Order in cart | |
| **Triggering Event :** | User want to Submit order | |
| **Brief Description:** | It it use case that allow user to submit order | |
| **Actors :** | Customer | |
| **Related Use Cases :** | NONE | |
| **Stakeholders :** | NONE | |
| **Preconditions :** | use case : Login , Create Order | |
| **Post conditions :** | The customer has confirmed the food items selected in the Cart. | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1.Use case start when User  press “Submit Order” | 1.1 The system will display the product details page and order summary of the Order. |
| **Alternative Flow** | When the customer presses the “Back” button, the system will take them back to the Cart page again. | |
| **Exception**  **Conditions :** | NONE | |

## Use Case Description : Update Cart

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Update Cart | |
| **Scenario :** | Increase-decrease the number of products in the cart. | |
| **Triggering Event :** | Users want to increase-decrease the number of products in the cart. | |
| **Brief Description:** | It is a use case that allows the User to increase-decrease the number of products in the cart. | |
| **Actors :** | Customer | |
| **Related Use Cases :** | Use case : Create Order | |
| **Stakeholders :** | NONE | |
| **Preconditions :** | use case : Login | |
| **Post conditions :** | The numbers of product was increased or decreased | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1.Use case start when User  press “+ , - ” in Cart | 1.1 The system will reduce or increase  Food menu according to customer press |
| **Alternative Flow** | NONE | |
| **Exception**  **Conditions :** | NONE | |

## Use Case Description : Delete Order

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Delete Order | |
| **Scenario :** | Delete product in cart | |
| **Triggering Event :** | User want to Delete product in cart | |
| **Brief Description:** | it is use case that allow user to delete product in cart | |
| **Actors :** | Customer | |
| **Related Use Cases :** | Use case : 1. Create Order | |
| **Stakeholders :** | NONE | |
| **Preconditions :** | Use case : Login , Create Order | |
| **Post conditions :** | Order is deleted | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1.Use case start when User  press “Cancel” ใน Cart | 1.1 The system will cancel the products ordered in the Cart. |
| **Alternative Flow** | NONE | |
| **Exception**  **Conditions :** | NONE | |

## Use Case Description : Add Food

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Add Food | |
| **Scenario :** | Added food items to the shop menu. | |
| **Triggering Event :** | Stores like to add more as new food items come in. | |
| **Brief Description:** | It is a use case where the store adds food items to the storefront. | |
| **Actors :** | Restaurant | |
| **Related Use Cases :** | NONE | |
| **Stakeholders :** | NONE | |
| **Preconditions :** | Use case : Login , Add Food | |
| **Post conditions :** | New food items have been added to the menu. | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1.The use case will start when the merchant has clicked “Add Food”. | 1.1 The system will add new food items to the shop's menu page. |
| **Alternative Flow** | NONE | |
| **Exception**  **Conditions :** | NONE | |

## Use Case Description : Update Food

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Update Food | |
| **Scenario :** | Edit food items into the shop menu. | |
| **Triggering Event :** | Restaurant wants to edit the list of products in the store. | |
| **Brief Description:** | It is a use case that stores use to edit food items into the storefront. | |
| **Actors :** | Restaurant | |
| **Related Use Cases :** | NONE | |
| **Stakeholders :** | NONE | |
| **Preconditions :** | Use case : Login , Add Food | |
| **Post conditions :** | Edit menu items. | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1. The shop has pressed “Edit Food” | 1.1 The system will show a window to edit food item information. |
|  | 2. The restaurant has pressed “Save”. | 2.1The system will save the food items that the shop edits. |
| **Alternative Flow** | step : 2 If the user does not press "Save" but presses "Back", the system will take the user back to the restaurant page. | |
| **Exception**  **Conditions :** | None | |

## Use Case Description : Delete Food

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Delete Food | |
| **Scenario :** | Removed food items from the shop menu. | |
| **Triggering Event :** | Restaurants want to delete items in the store. | |
| **Brief Description:** | It is a use case where the store removes food items from the store. | |
| **Actors :** | Restaurant | |
| **Related Use Cases :** | NONE | |
| **Stakeholders :** | NONE | |
| **Preconditions :** | Use case : Login , Add Food | |
| **Post conditions :** | Order is deleted | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1. The store has pressed “Delete Food” | 1.1 Message display system “Are you sure you want to delete your food item?” |
|  | 2. Restaurant has pressed “Submit” | 2.1 The system will save the information. The store has removed food items from the front of the store. |
| **Alternative Flow** | step : 2 If the user does not press "Submit" but presses "Back", the system will take the user back to the restaurant page. | |
| **Exception**  **Conditions :** | None | |

## Use Case Description : Order History

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Order History | |
| **Scenario :** | Customers view order history | |
| **Triggering Event :** | Customers want to view order history and order status. | |
| **Brief Description :** | Customers click on "Order History" and view order history and order status. | |
| **Actors :** | Customer , Rider , Restaurant | |
| **Related Use Cases :** | NONE | |
| **Stakeholders :** | NONE | |
| **Preconditions :** | Use case : Submit order in cart | |
| **Post conditions :** | Users can view order history and status. | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1. The use case will start when the user presses "History” | 1.1 The system will display order information that the customer has ordered and its status. |
| **Alternative Flow** | 1.If the customer presses the back button, the system will take them back to the main page. | |
| **Exception**  **Conditions :** | NONE | |

## Use Case Description : Finish Order

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Finish Order | |
| **Scenario :** | Rider delivers food to the customer and wants to finish the order. | |
| **Triggering Event :** | Rider finishes delivering and wants to accept a new order. | |
| **Brief Description :** | When the rider delivers food successfully, the rider will finish the order. | |
| **Actors :** | Rider | |
| **Related Use Cases :** | None | |
| **Stakeholders :** | Rider | |
| **Preconditions :** | Riders have to login first, There must be order in the list. Riders have to accept orders first. | |
| **Post conditions :** | Riders can accept new orders.  System move order from rider list to rider history.  System saves history. | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1.The use case will start when the driver presses “Finish Order”. | 1.1 The order from the list will be moved to history.  2. The system will accept new purchases into the system. |
| **Alternative Flow** | NONE | |
| **Exception**  **Conditions :** | NONE | |

## Use Case Description : Accept Customer Order

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Accept Customer Order | |
| **Scenario :** | Rider wants to accept the order. | |
| **Triggering Event :** | Riders get notified about the order that is free to accept and want to get orders. | |
| **Brief Description :** | Rider clicks accept the order and prepare for delivery. | |
| **Actors :** | Rider | |
| **Related Use Cases :** | None | |
| **Stakeholders :** | Rider and Customer | |
| **Preconditions :** | Riders have to login first, There must be order in the list. | |
| **Post conditions :** | Riders can accept new orders.  System saves history. | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1. Rider click order when food was delivered to customer | 1.1 System moves order from list to history.  1.2 System allows riders to accept new orders. |
| **Alternative Flow :** | NONE | |
| **Exception**  **Conditions :** | NONE | |

## Use Case Description : Revenue

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Distribute Revenue | |
| **Scenario :** | Bank distribute money to an application account that | |
| **Triggering Event :** | Restaurant or Rider want to withdraw a revenue from their account. | |
| **Brief Description :** | System requests the bank to withdraw money according to the Restaurant or Rider request. | |
| **Actor(s) :** | Bank, Restaurant and Rider | |
| **Related Use Cases :** | Receive Money | |
| **Stakeholders :** | Restaurant, Bank and Rider | |
| **Preconditions :** | Rider or Restaurant must send a request to receive money. | |
| **Post conditions :** | Rider and Restaurant receive money | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1. Bank check a payment transaction. | 1.1 System send request to bank. |
| 1. Bank withdraw money from distribution account to Rider or Restaurant account whose send a request. | 2.1 System send status to Rider and Restaurant. |
| **Alternative Flow** | NONE | |
| **Exception**  **Conditions :** | NONE | |

## Use Case Description : Pay

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Pay | |
| **Scenario :** | A customer pays for food after clicking ‘Submit order’. | |
| **Triggering Event :** | A customer clicks ‘Submit order’. | |
| **Brief Description:** | When a customer clicks ‘Submit order’, the bank will send a transaction payment to the customer for them to pay. | |
| **Actors :** | Customer, Bank | |
| **Related Use Cases :** | Submit Order | |
| **Stakeholders :** | Customer, Rider, Restaurant | |
| **Preconditions :** | Customers have to submit the order before paying for food. | |
| **Post conditions :** | System notify the payment status. | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1.User clicks ‘Submit order’. | 1.1 System display payment method. |
| 2. Users choose a payment method. | 2.1 System request payment method that was chosen by customer to a bank. |
| 3. User pay for food. | 3.1 System sends payment transactions to the bank. |
|  | 3.2 System shows payment status. |
| **Alternative Flow** | NONE | |
| **Exception**  **Conditions :** | NONE | |

## Use Case Description : Add food to Cart

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Add food to Cart | |
| **Scenario :** | Customer want to add foods to a cart. | |
| **Triggering Event :** | Customer want to add foods to a cart. | |
| **Brief Description:** | Customer want to add food to a cart by clicking the button ‘add to cart’. | |
| **Actors :** | Customer | |
| **Related Use Cases :** | Use case : Create cart | |
| **Stakeholders :** | Restaurant | |
| **Preconditions :** | ~~Use case : Create cart~~ | |
| **Post conditions :** | Food order is added to a cart. | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1. User add food to cart. | 1.1 System will add that food to a cart. |
| **Alternative Flow** | NONE | |
| **Exception**  **Conditions :** | NONE | |

## Use Case Description : Finish Order

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Finish Order | |
| **Scenario :** | Customers get food and want to tell the system that they have received food already. | |
| **Triggering Event :** | Customers receive food. | |
| **Brief Description :** | The customer informs the system that they have received the food. | |
| **Actor(s) :** | Customer | |
| **Related Use Cases :** | NONE | |
| **Stakeholders :** | NONE | |
| **Preconditions :** | Customer must submit the order. | |
| **Post conditions :** | Rider and Restaurant receive money | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1. Bank receive money from Customer | 1.1 System distribute revenue to Rider and Customer |
| **Alternative Flow** | NONE | |
| **Exception**  **Conditions :** | NONE | |

## Use Case Description : Look up order history

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Look Up Order History | |
| **Scenario :** | User want to see order history. | |
| **Triggering Event :** | User want to see order history. | |
| **Brief Description:** | The user clicks 'History' to view their food order history. | |
| **Actors :** | Customer, Restaurant and Rider | |
| **Related Use Cases :** | NONE | |
| **Stakeholders :** | Customer, Restaurant and Rider | |
| **Preconditions :** | None | |
| **Post conditions :** | The user see their food order history. | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1. User click food order. | 1.1 System display food order history . |
| **Alternative Flow** | NONE | |
| **Exception**  **Conditions :** | NONE | |

## Use Case Description : Look up order

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Look Up Order | |
| **Scenario :** | Restaurant want to look a food order. | |
| **Triggering Event :** | Order was sent to restaurant. | |
| **Brief Description:** | The restaurant needs to review the order to determine what they need to cook. | |
| **Actors :** | Restaurant | |
| **Related Use Cases :** | NONE | |
| **Stakeholders :** | Customer, Restaurant and Rider | |
| **Preconditions :** | User must submit the food order first. | |
| **Post conditions :** | Restaurant get a list of what they need to cook. | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1. Restaurant is notified about order that is submitted by customer. | 1.1 System sends a food order list to restaurant. |
| 2. Restaurant open device to review order. | 2.1 System display food order. |
| **Alternative Flow** | NONE | |
| **Exception**  **Conditions :** | NONE | |

## Use Case Description : Chat

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Chat | |
| **Scenario :** | User want to chat with other parties ( Rider , Customer, Restaurant ) | |
| **Triggering Event :** | The user want to communicate with other parties to inquire about food status or another purpose. | |
| **Brief Description :** | The user want to communicate with other parties to inquire about food status or another purpose. | |
| **Actors :** | Customer, Rider and Restaurant | |
| **Related Use Cases :** | None | |
| **Stakeholders :** | Customer, Rider and Restaurant | |
| **Preconditions :** | None | |
| **Post conditions :** | Message from one user was sent to another user. | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1. User click ‘Chat’ button to communicate with that person. | 1.1 System navigate user to a chat page. |
| 1. The user initiates a chat with some party they wish to communicate with. | 2.1 System deliver message to receiver.  2.2. System store message in database. |
| **Alternative Flow** | NONE | |
| **Exception**  **Conditions :** | NONE | |

## Use Case Description : Receive Money

|  |  |  |
| --- | --- | --- |
| **Use Case Name :** | Receive Money | |
| **Scenario :** | Rider and Restaurant Receive money from Grab | |
| **Triggering Event :** | Rider and Restaurant finish the food order | |
| **Brief Description :** | Rider and Restaurant click to withdraw the money that is stored in the application account.  They can choose either withdraw all the money or select an amount. | |
| **Actors :** | Rider , Restaurant | |
| **Related Use Cases :** | None | |
| **Stakeholders :** | Rider, Restaurant | |
| **Preconditions :** | There must be money in the application account. | |
| **Post conditions :** | Rider and Restaurant receive payment share.  A balance in application is reduce due to withdraw amount that the choose. | |
| **Flow of Activities :** | **Actor** | **System** |
|  | 1. Actor go to a profile page | 1.1 System navigate actor to profile page |
| 2. Actor clicks withdraw the money. | 2.1 System display withdrawal modal.  2.2 System asks for an amount of money. |
| 3. Actor input a withdraw amount | 3.1 System asks to confirm a withdrawal. |
| 4. Actor confirm. | 4.1 System deducts money from an application account.  4.2 System request bank to send  money to the actor's account according to the amount of money that actor needs. |
| **Alternative Flow** | in step 4 : if the actor doesn’t confirm a withdrawal system will close the withdrawal modal. | |
| **Exception**  **Conditions :** | NONE | |

# **Activity Diagram**

## 1. Create User

A diagram of a software system

Description automatically generated with medium confidence

## 2. Login

## A diagram of a user flow Description automatically generated

## 3. Look Up Order

A screenshot of a diagram

Description automatically generated

## 4. Chat

A diagram of a customer service

Description automatically generated

## 5. Update Account

A diagram of a system

Description automatically generated

## 6. Delete Account

A diagram of a user flow

Description automatically generated

## 7. Receive Money

A screenshot of a computer screen

Description automatically generated

## 8. Update Cart

A screenshot of a diagram

Description automatically generated

## 9. Delete Cart

**A diagram of a customer order

Description automatically generated**

## 10. Add Food to Cart

A diagram of a customer

Description automatically generated

## 11. Submit Order in Cart

A screenshot of a diagram

Description automatically generated

## 12. Pay

A diagram of a payment method

Description automatically generated

## 13. Accept Customer Order

A screenshot of a diagram

Description automatically generated

## 14. Finish Order

A screenshot of a diagram

Description automatically generated

## 15. Add Food

A screenshot of a diagram

Description automatically generated

## 16. Update Food

A screenshot of a computer screen

Description automatically generated

## 17. Delete Food

A diagram of a restaurant

Description automatically generated

# **Sequence Diagram**

## 1. Submit Cart

## A diagram of a system Description automatically generated

## 2. Update Food

A diagram of a restaurant

Description automatically generated

## A diagram of a user flow Description automatically generated 3. Create User

## 4 . Chat

A diagram of a chat and a chat box

Description automatically generated

# **Context Diagram**

A diagram of a food order

Description automatically generated

# Data Flow Diagram

## 1. DFD level0

A diagram of a restaurant

Description automatically generated

## A diagram of a chat Description automatically generated 2. DFD level 1 for process 2

## 3. DFD level 1 for process 10

A diagram of a chat

Description automatically generated

# **ER Diagram**

A diagram of a customer service

Description automatically generated

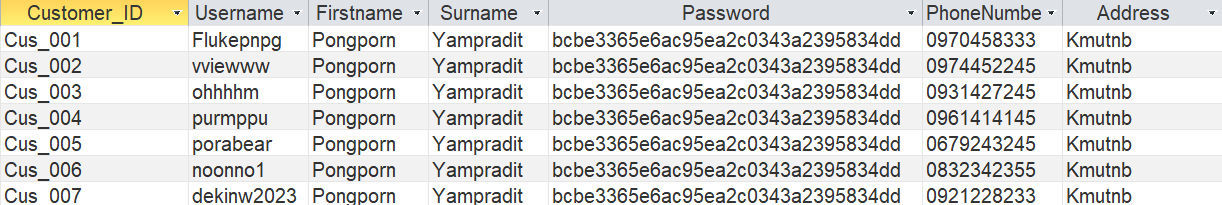
# **Class Diagram**

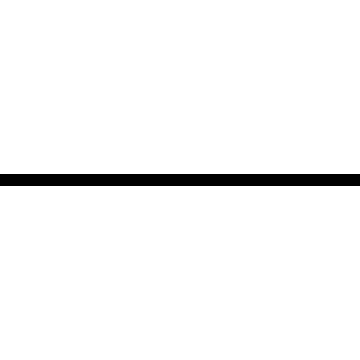
A diagram of a restaurant

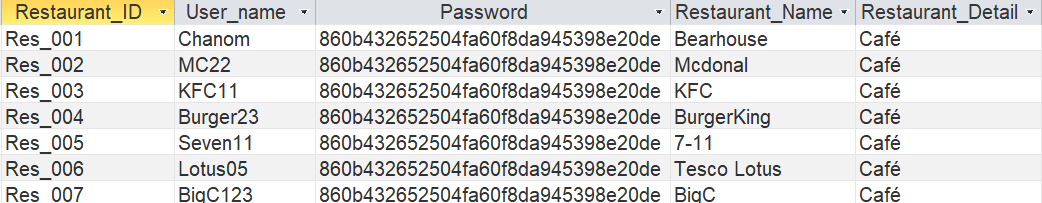
Description automatically generated

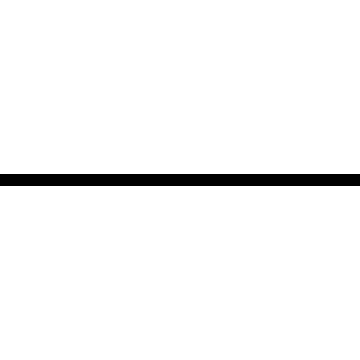
# **Microsoft Access Table**

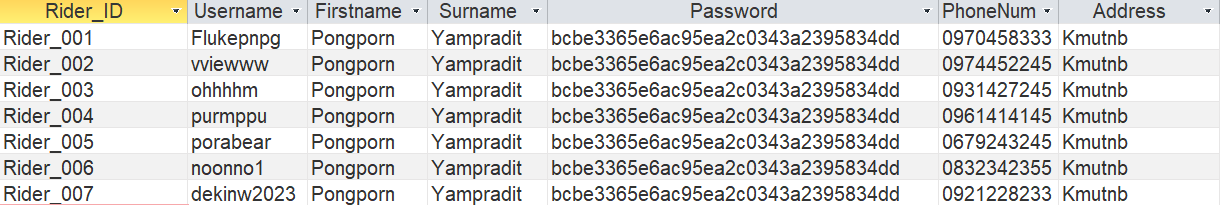
**Customer :**

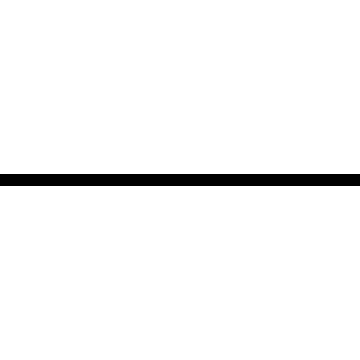


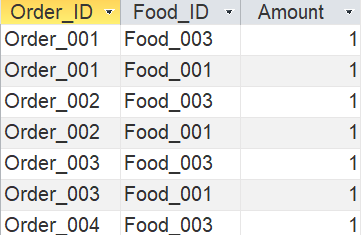
**Restaurant :**

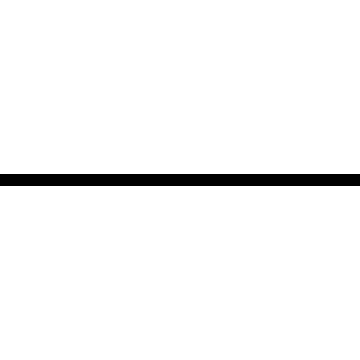


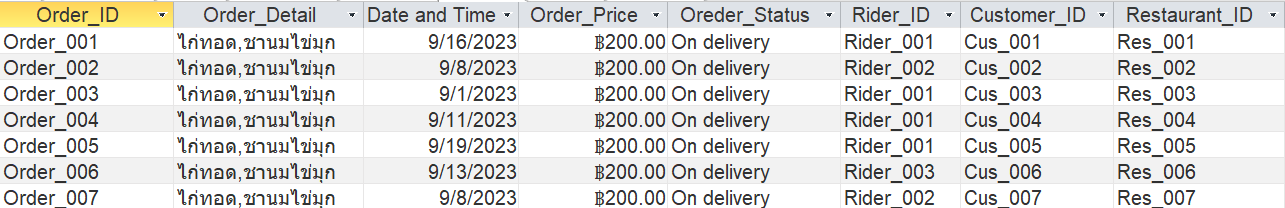
**Rider :**

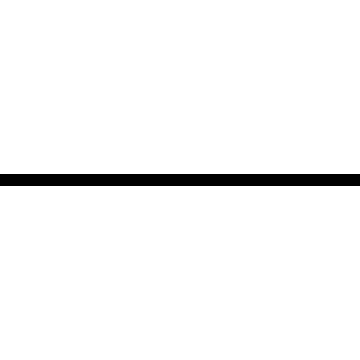


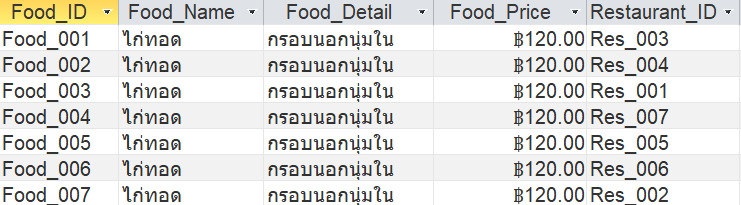
**Cart :**

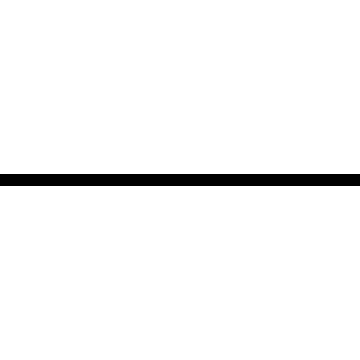
****

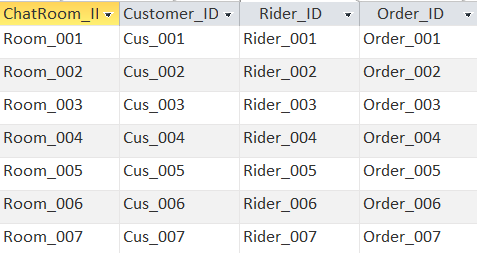
**Order :**

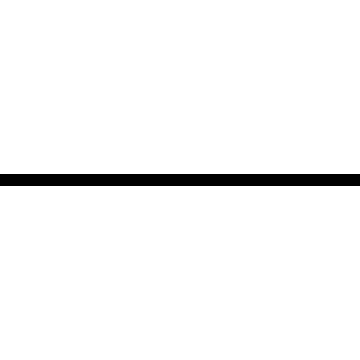


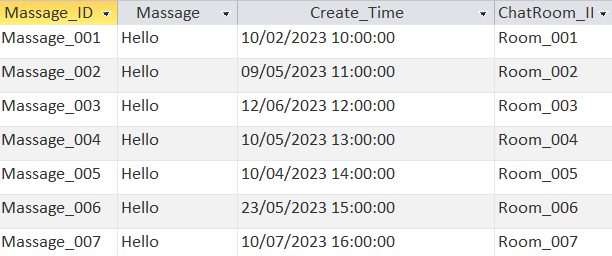
**Food :**

****

**Chat Room:**

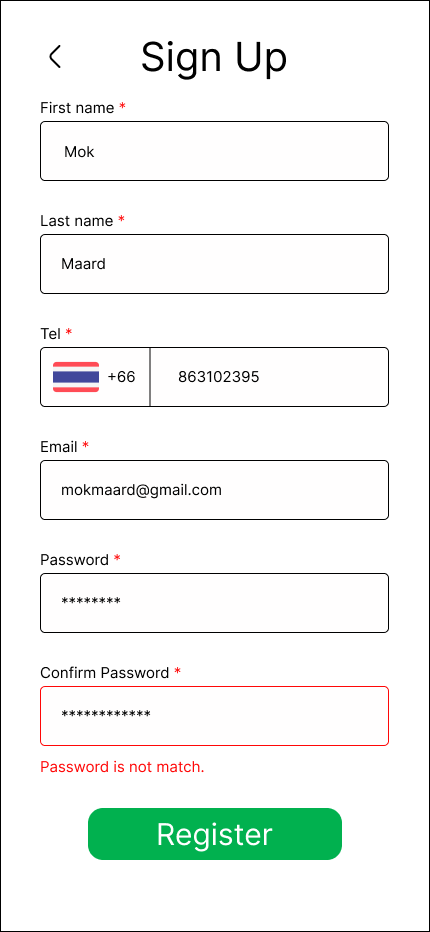
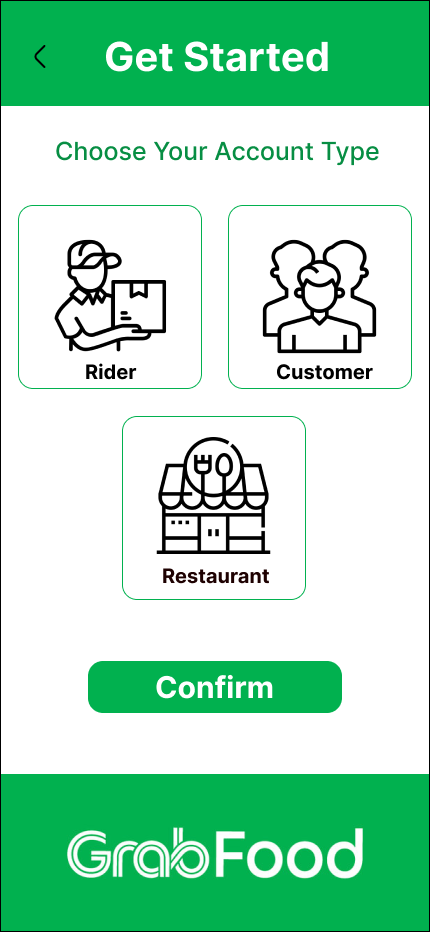
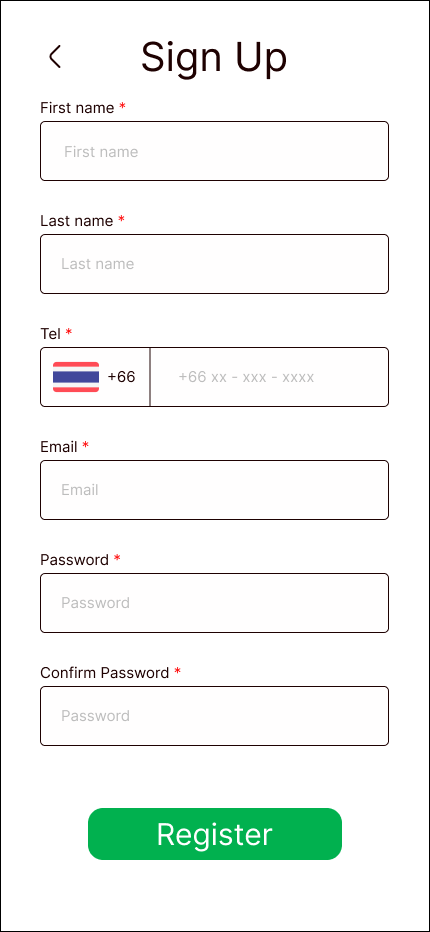
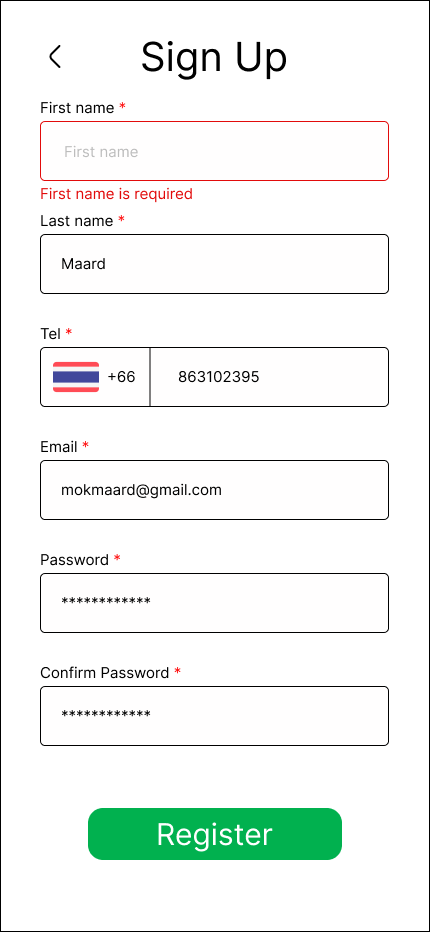
****

**Massage:**

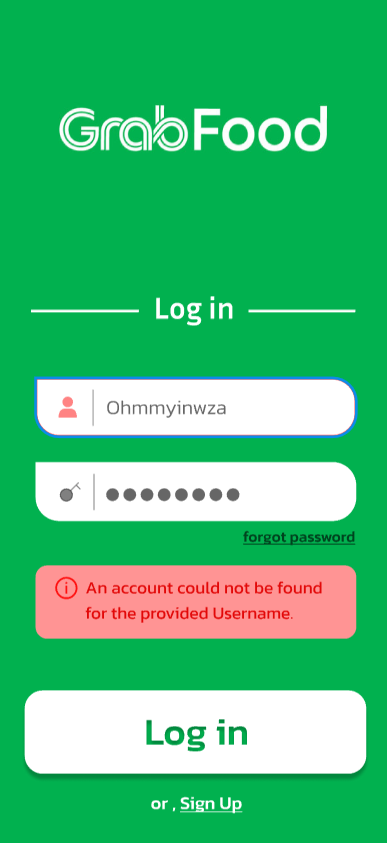
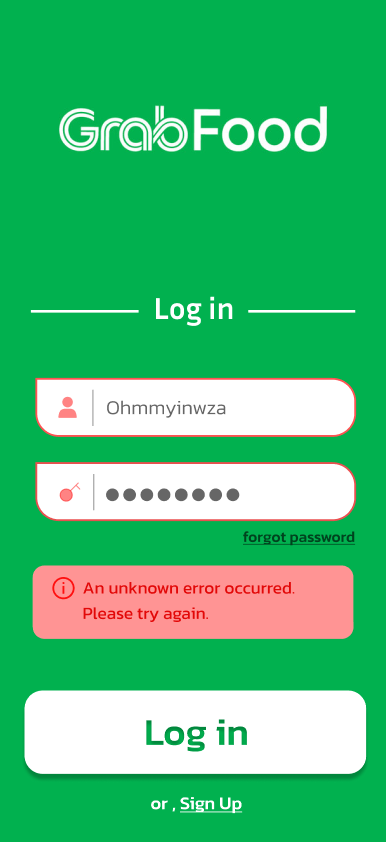
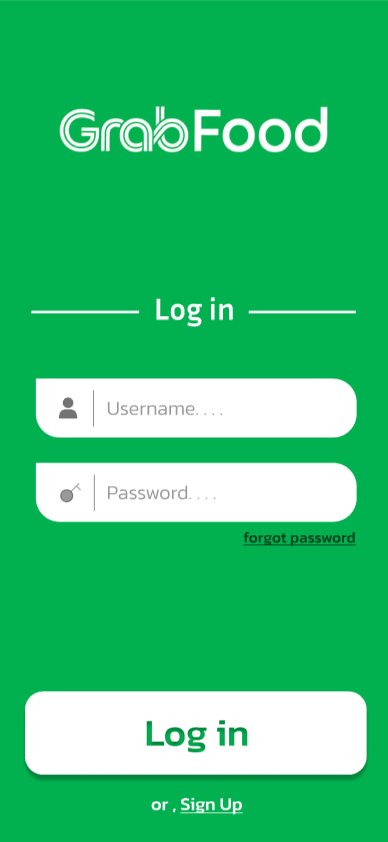
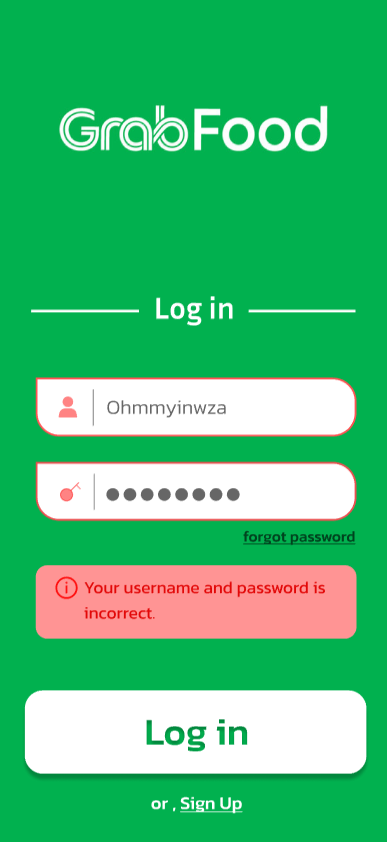
****

# **User Interface**

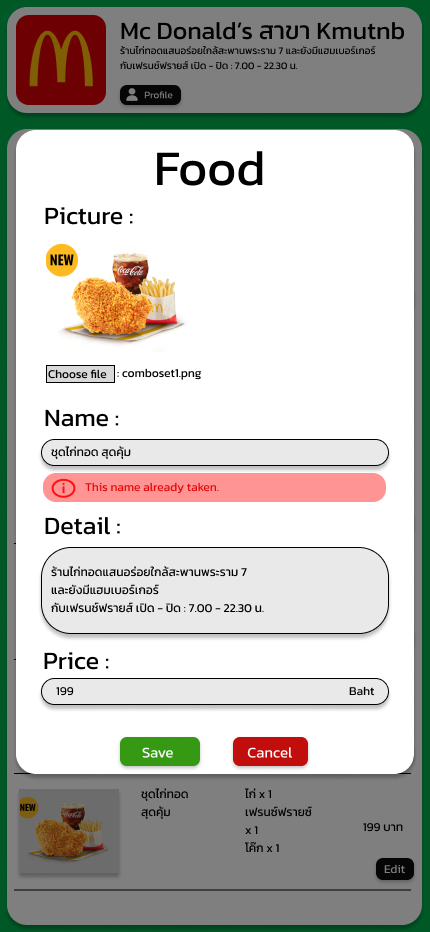
## Sign Up interface

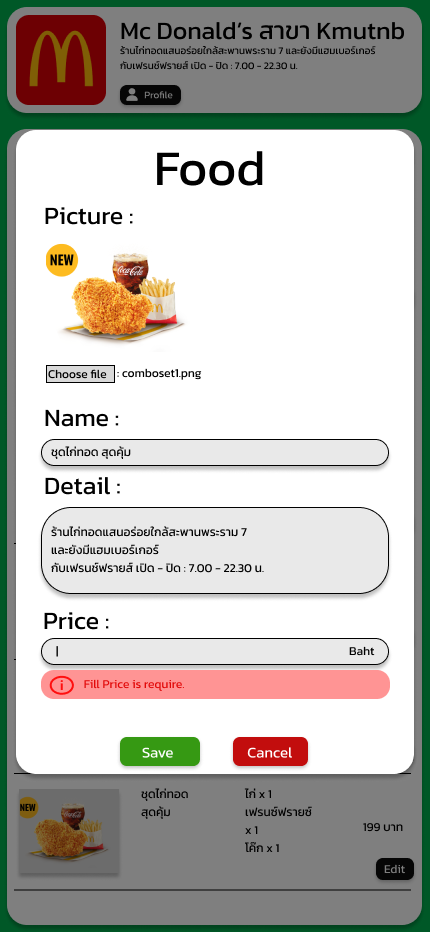


## Login Interface



## Restaurant Interface





A screenshot of a menu

Description automatically generated

## 4. choose food

