VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

SOFTWARE ENGINEERING (CO3001)

Assignment

Restaurant POS 2.0

Advisor: Quản Thành Thơ
Students: Lê Quốc Anh - 1852006
Trần Việt Hoàng - 1852145
Nguyễn Văn Phúc Nghiệp - 1952354
Nguyễn Trần Anh Quân - 1952418
Lê Mai Quang Duy - 2052913

HO CHI MINH CITY, February 2022

Contents

1 Member list & Workload

No.	Fullname	Student ID	Problems	Percentage of work
1	Lê Quốc Anh	1852006	- The context of project - Expected to be done - Class diagram	20%
2	Trần Việt Hoàng	1852145	- Use case diagram - Table 1 - Sequence diagram	20%
3	Nguyễn Văn Phúc Nghiệp	1952354	- Functional requirements - Non-functional requirements - Sequence diagram	20%
4	Nguyễn Trần Anh Quân	1952418	- Stakeholders - Scope of project - Activity diagram	20%
5	Lê Mai Quang Duy	2052913	- Table 2 - Table 3 - Activity diagram	20%

2 Require Elicitation

2.1 Introduction

2.1.1 The context of project

Our client owns many restaurants and wants us to design a web-based POS system that can be applied to the existing restaurant's working system. The system is expected to serve business activities at the point of sale, especially during the covid pandemic.

- **For further details:**
  - Services: dine-in or take-away.
  - Payment Option: Credit card, cash.
  - Reservation: Customers can reserve table on the restaurant website, they can also pre-order food.
  - Menu, meal and price: can only be defined by the manager or the restaurant owner.
- **For business processes:**
  - Customers are prompted to declare their health status due to the pandemic beforehand using QR code. When finished they are allowed to get into the restaurant and choose the table (if the option is take-away, this step can be ignored).
  - Customers can either scan QR code or use a URL to view the menu on the website and complete their order.

- After that, the order will be automatically sent to the clerk for confirmation. The clerks then ask the kitchen staff to process the order.
- Order is carried out and customers will receive their meal.

2.1.2 Stakeholders

Some of the relevant stakeholders of the system:

- System managers: IT staffs who operate and maintain the system (website or server engineer).
- Restaurant supervisor: oversee all restaurant operations to ensure that restaurants run smoothly.
- Related baking services: keep track of money flow from customer's account to restaurant's account and calculate the extra fee and the tax if necessary.
- Customers: Order food, reserve table, give feedback.
- Restaurant staffs(manager, clerk): Manage receipts, list of food, payment, booking and customers service.

2.1.3 Expected to be done

The system is considered successfully implemented if it meets the conditions listed below:

- Design front-end for Menu, Shopping Cart, Login Cart.
- Design the list of dishes which are described clearly about name and kinds of food. Moreover, there are brief description about food, Star points, and previous customer's commands.
- Design Back-end for POS 2.0 systems by using noSQL.
- The Clerk or Manager can add/remove food from menu as well as all food's characteristics.

2.1.4 Scope of project

- **Justification:**
  - Managing restaurant can be easier by using POS system, which is developed based on web-platform.
  - POS system makes the reservation, payment more convenient and tracking profit more clearly.
  - The system, moreover, allows the restaurant owner to manage their staffs, food and revenue.
  - Being built in web-platform, the system guarantee the stability on several devices, security and convenience.
- **Description:** POS system provides basic functions such as food ordering, table reservation, online payment. It is almost the same as traditional restaurant but more flexible as everything in online.

- **Acceptance criteria:** In term of technology, the system is experienced and assess by the restaurant staffs. The project is considered to be complete if the rating point is more than 4 over fifth. In term of user experience, the project is considered to be complete if the user rating point is more than 4 over fifth.
- **Deliverable:**
  - Project manager has to provide the customer with documents related to implementation and maintaining the system such as SRS (system requirement), SDD (system design), STD (system testing document), some analysis about the system and the user manual.
  - Project has to be handed over to the customer after testing and acceptance.
  - Customer will be supported for implementation and maintaining for free in the first year.

2.2 Functional and non-functional requirements

2.2.1 Functional Requirements

- The system should allow the customers to scan the QR code or enter the URL in order to go to the restaurant's website.
- The system allows the customers to select a meal from many categories (size, type of food, special requirements,...) on the menu list and then order.
- The system should record feedback from the customer after collecting their orders.
- The system should allow the clerk to check and confirm the orders from customers.
- When the order is unavailable, the system should notify the customer.
- When there are errors, the system should notify the system manager.
- When the order is valid, the system will allow customer payment by credit card or cash, and record successful payments.
- Chiefs can see the valid orders.
- The clerks can delete finished orders.
- The customer can choose to receive a digital receipt or a physical receipt after the successful purchase. if they don't choose any options, they will receive a physical receipt automatically.
- The system should be usable from mobile devices(phones, tablets) including both Android and iOS or a normal computer/laptop which runs on either Windows, MacOS, or Linux.

2.2.2 Non-Functional Requirements

- The system should be able to provide both direct or non-direct contact between Clerks and Customers. Three main actions which are selecting food, ordering, and paying for the meal must be carried out on the user's device. The clerks only confirm the order and make recommendations towards the customer in direct contact if the customer wants or in case the food is out of order.

- The system should be extendable to different restaurants and adaptable to similar kinds of establishments.
- The system should be able to handle 300 orders/day.
- The system should notify the clerk to confirm the order of the customer within 3 seconds after the customer clicks the 'order' button.
- The order information should be sent to kitchen staff within 30 seconds after the order has been confirmed by the clerk.
- The system should always run for 18 hours continuously from 5 o'clock in the morning to 11 o'clock in the evening.
- The system should allow accidental downtime for at most 10 minutes.
- Only the clerks are allowed to confirm the customer's order.
- Only authenticated users such as administrators to modify, add and delete food items.
- The food cart of the customer hold at most 50 meals.

2.2.3 Use-case diagram

For the purpose of having a more vivid visualization of the system, we need to put it in a use case diagram in order to show us an illustration of the interaction between the system, to be more specific the fast-food ordering system, and its actors or related parties.

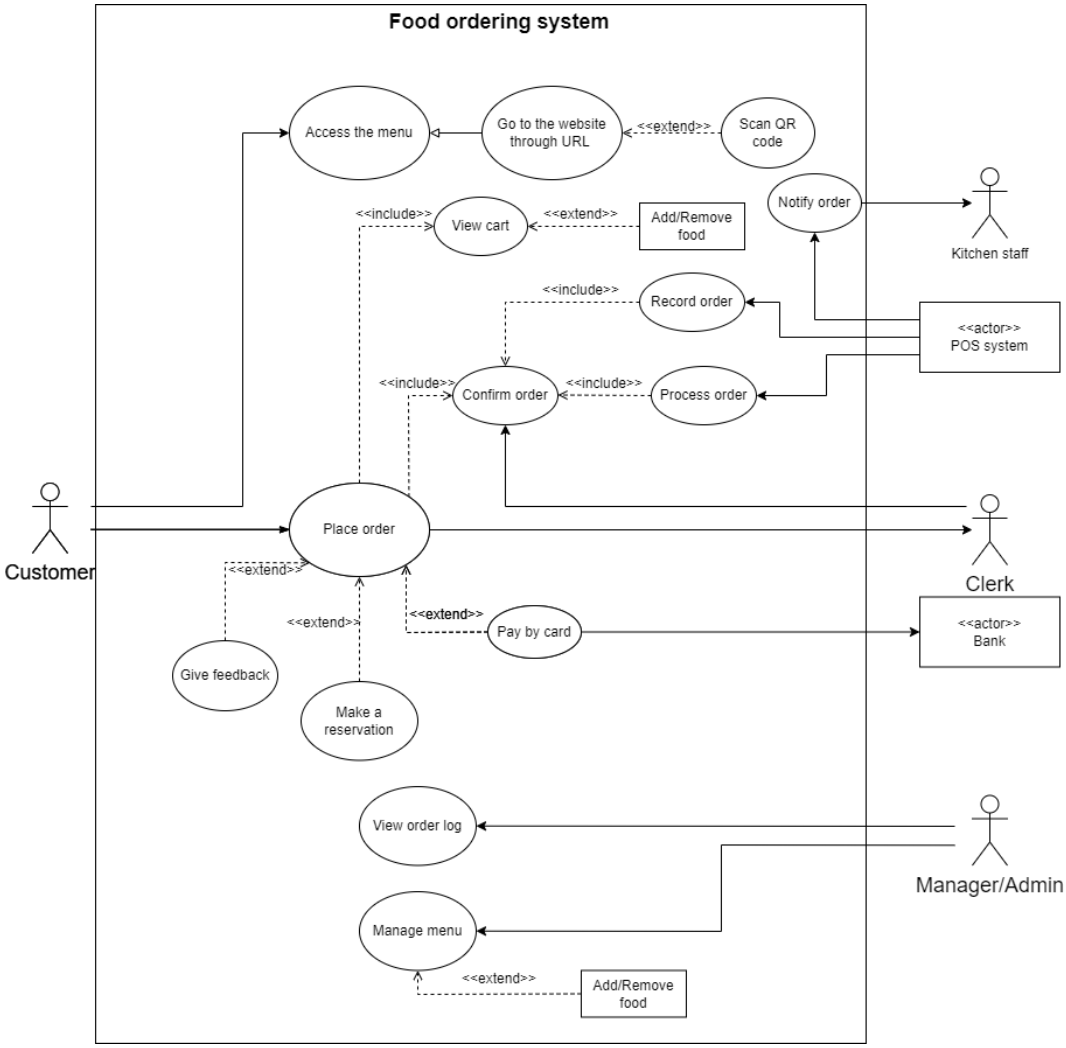


Figure 1: Use case diagram

2.3 Detailed use case description

2.3.1 Food Ordering Use-case Diagram

To be more specific, we choose the Place order use case of the fast-food ordering system to analyze.

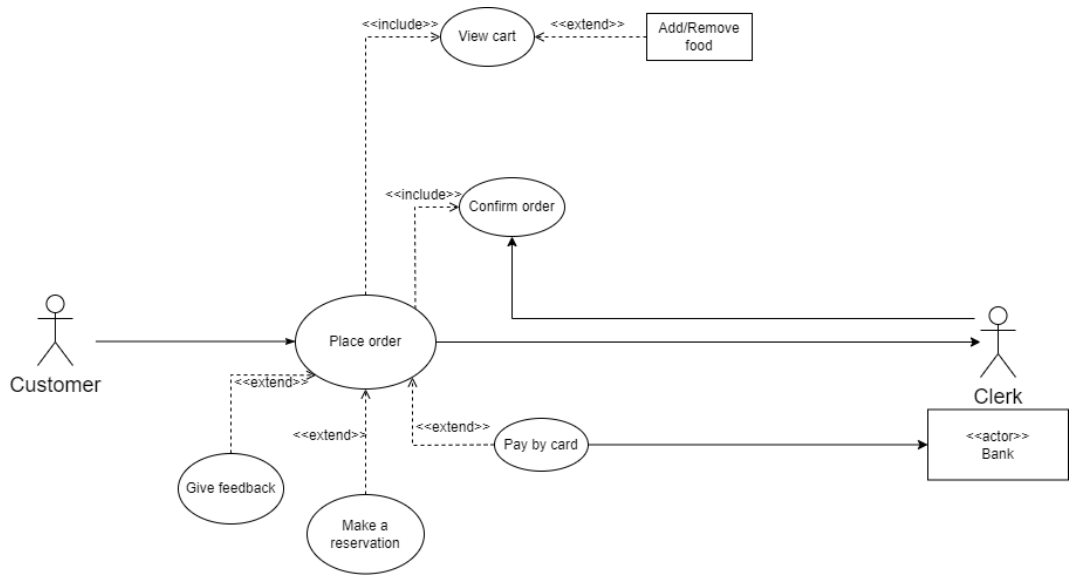


Figure 2: Use case diagram of Place order use case

Table 1: Table format for place order use case

Use case name	**Place order**
Related requirements	The customer can order the food or make a reservation on the website of the restaurant.
Goal of the context	A request of the customer to place order.
Preconditions	The customer must access the website of the restaurant.
Successful end condition	The order of the customer is processed by the system.
Failed end condition	A notification sent to the customer that their order cannot be placed at the moment.
Primary actors	The customer.
Secondary actors	Food ordering system.
Trigger	The customer asks the Food ordering system to place the order.
Main flow	1. include::ViewCart 2. include::ConfirmOrder 3. The customer pays for the order
Extension	3.1 The customer pays the order by credit card 4. Customer can make a reservation 5. Customer can send feedback to the system

As mentioned in above table, the Place order use case includes View cart use case and Confirm order use case together. Therefore, we will provide two additional tables for View cart and Confirm order use case in table 2 and table 3 respectively.

Table 2: Table format for View cart use case

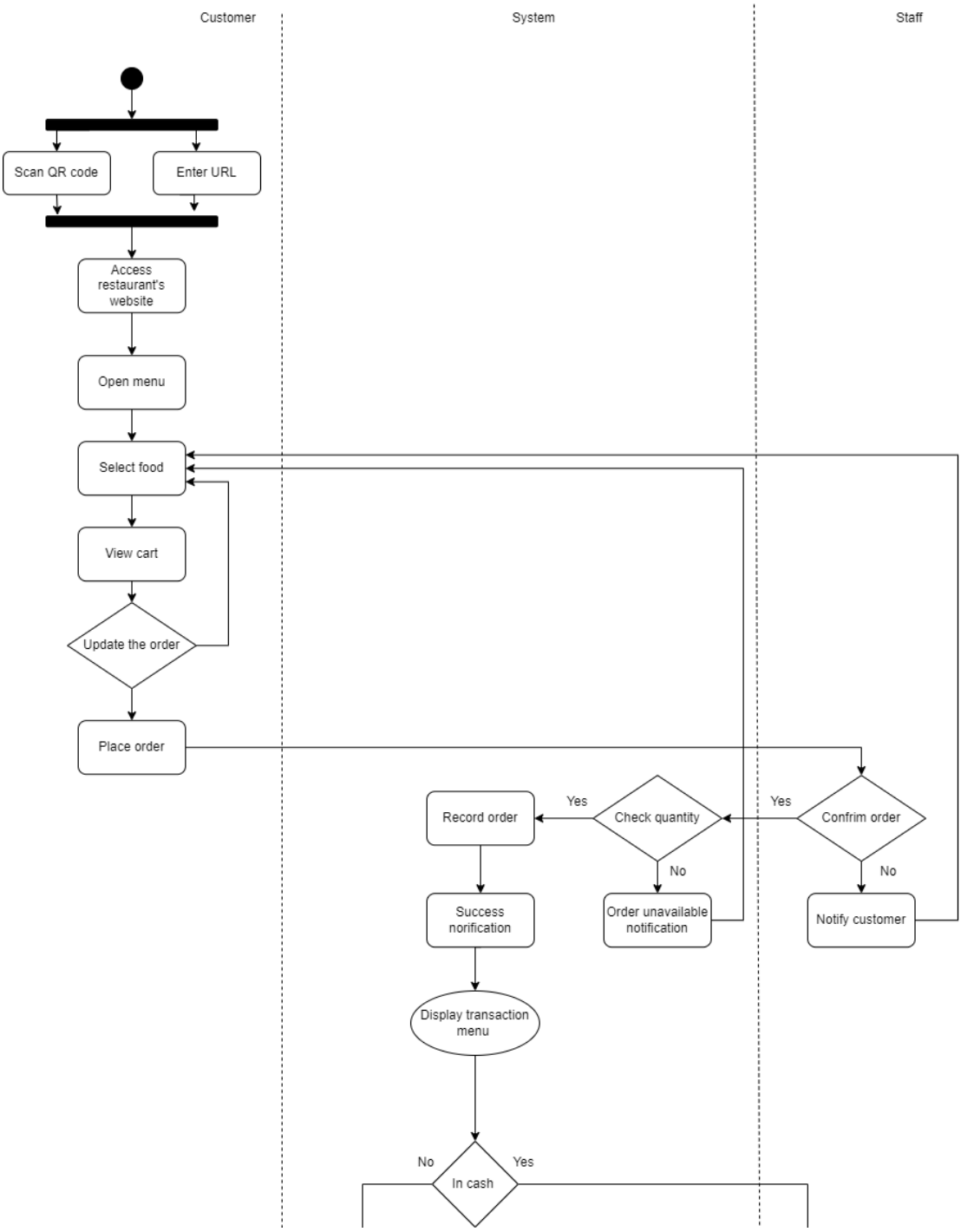
Use case name	**View cart**
Related requirements	the customer can view all the food that he/she has already chosen in their cart.
Goal of the context	A request of the customer to view the cart from system.
Preconditions	The customer must access the website of the restaurant.
Successful end condition	A cart displayed in the customers' devices.
Failed end condition	A notification sent to the customer that the cart cannot be loaded.
Primary actors	The customer.
Secondary actors	Food ordering system.
Trigger	The customer asks the Food ordering system to view the cart.
Main flow	1. The customer clicks on the "View cart" button on their device. 2. Food ordering system sends the cart information to the customer. 3. The customer go directly to the checkout.
Extension	2.1. The system cannot access the customers' cart to modify anything. 2.2. The request of the customer is rejected (in case of system is not working or some errors are happening). 3.1. Customer can add or remove the food in their cart.

Table 3: Table format for Confirm order use case

Use case name	**Confirm order**
Related requirements	The clerk can confirm the order placed by the customer.
Goal of the context	The order needs to be validated by the clerk before going to the food process.
Preconditions	The order has already placed by the customer.
Successful end condition	The system saves the order's information and sends a notification to the customer to notice that their order was successfully placed..
Failed end condition	A notification sent to the customer in order to notice that their order cannot be confirmed at the moment.
Primary actors	The clerk.
Secondary actors	Food ordering system.
Trigger	The clerk asks the Food ordering system to confirm the order.
Main flow	1. The clerk confirms the order on the system. 2. If successful, the system confirms the order and sends a notification to the customer.
Extension	2.1. If the order cannot be confirmed, the system rejects confirming the order. 2.2. The request of the customer is rejected (in case of system is not working or some errors are happening).

3 System Modeling

3.1 Activity diagram



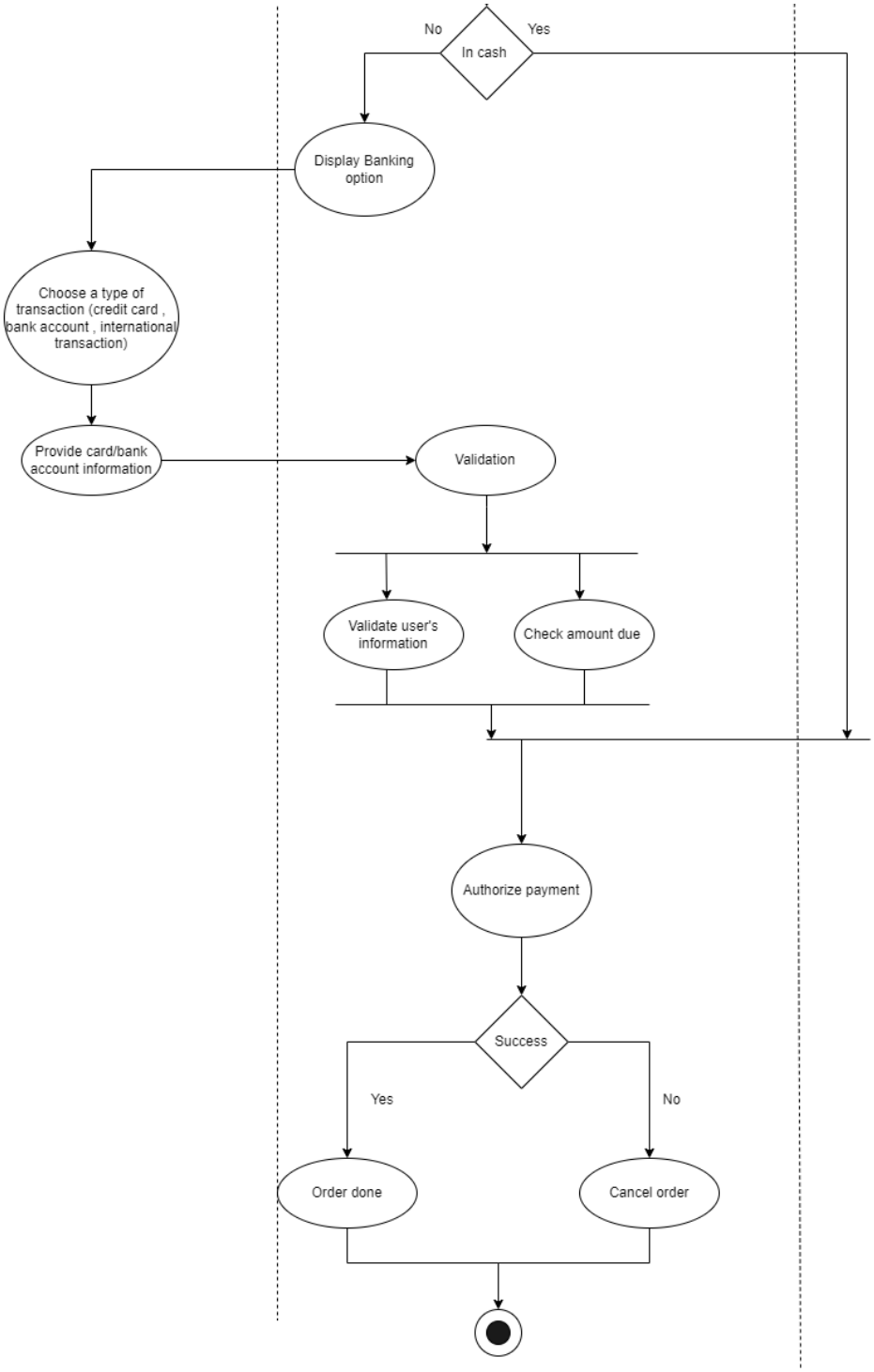


Figure 4: Activity diagram for the food ordering system

3.2 Sequence diagram

We decided to design the sequence diagram following these step:

1. The customer goes to the order page through the URL or the QR code and then the menu list will be displayed on this page in order for the customer to select the food. The customer can adjust their order in the cart and finally click the "Order" button to send their order to the clerk.

2. The clerk responsibility is to check and confirm the order, then sent it to the system to checks for its availability based on the system database.

3. In case of the order is available, the customer will be notified and redirected to the checkout page. On this page, the customer can choose a suitable payment method which is cash or credit card. If the customer chooses to pay by credit card, they will be asked to fill in their credit card's information, the information will be sent to the banking system for verification and check for the balance of the card.

    - If the balance is enough to pay for the order, the banking system will update the order payment status, make the transaction, as well as sent a records to the system database. Finally, a notification will be sent to the payment page to notify the customer about payment success.
    - Otherwise, the POS system will send a notification to the customer to notice that some failure happened in the payment process.

4. If the order is not available due to unconfirmed by the clerk or not enough stock in the system database, the POS system will send a message to the order page to notify the customers that their order is unavailable at this moment and redirect them to the order page to make new order.

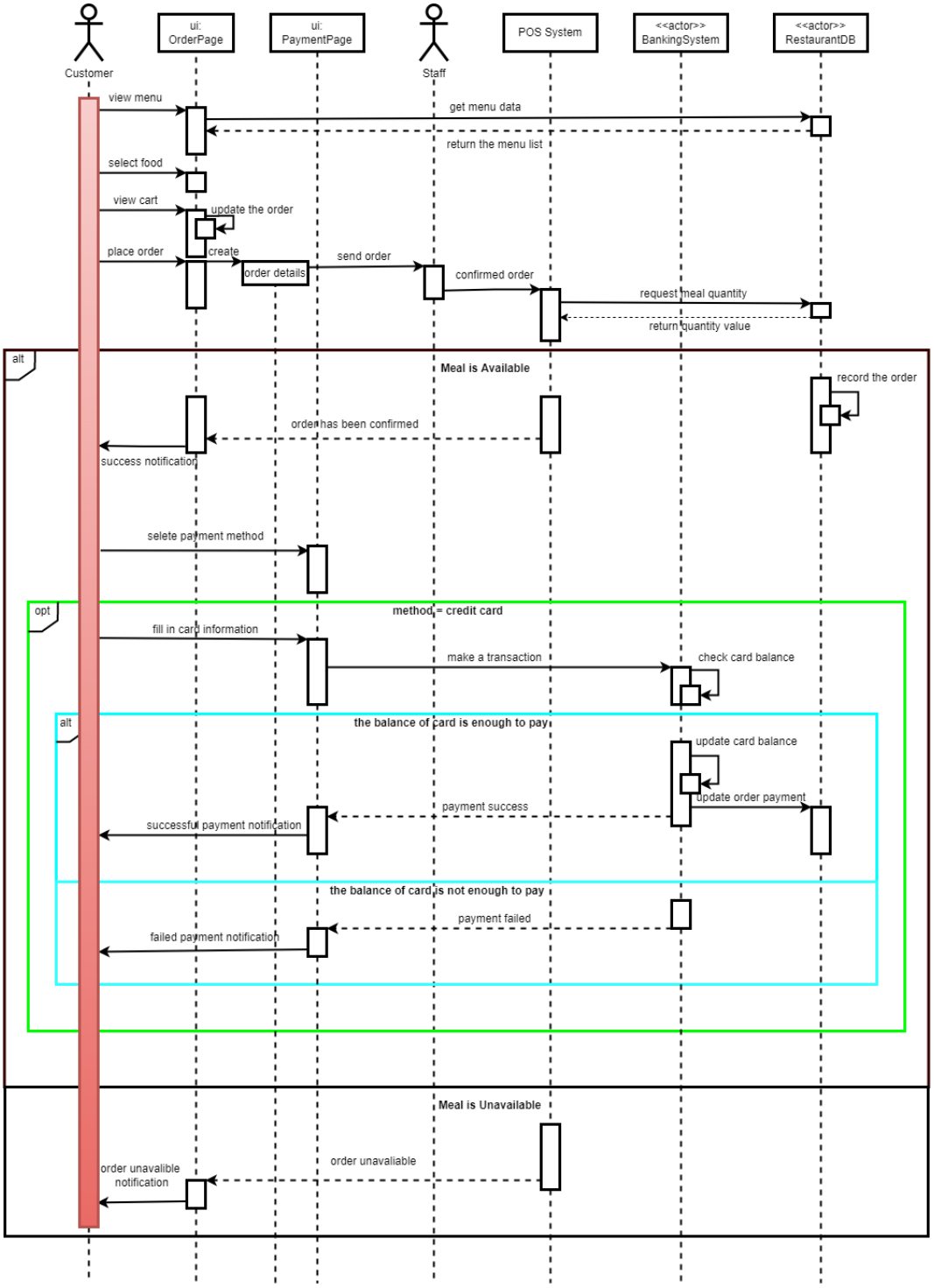


Figure 4: Sequence diagram for the food ordering system

3.3 Class diagram

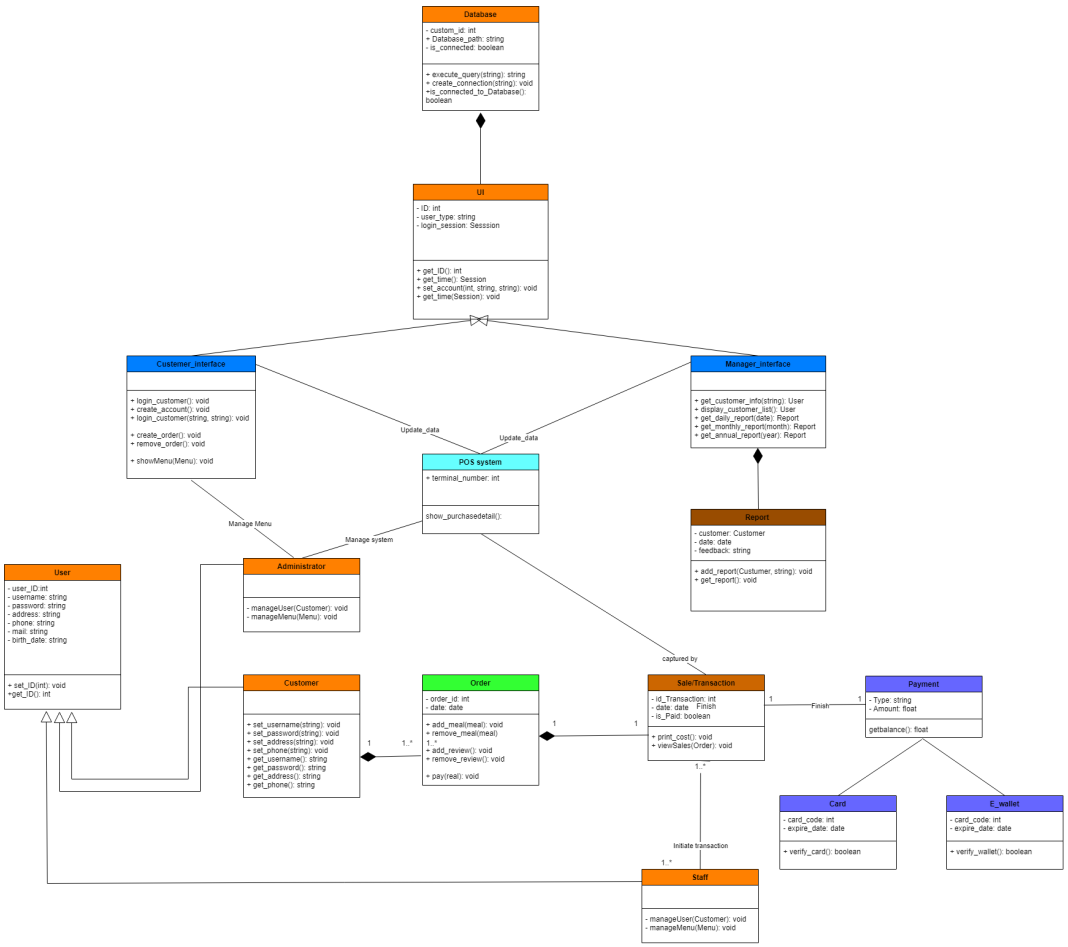


Figure 5: Class diagram for the whole fast food ordering system

4 Architecture design

4.1 Architectural approach

4.1.1 Introduction to Redux

Definition : The (MVC) is an architectural pattern that separates an application into three main logical components: the model, the view, and the controller. Each of these components are built to handle specific development aspects of an application. MVC is one of the most frequently used industry-standard web development framework to create scalable and extensible projects.

There are 3 components in MVC : **Model , View and Controller**.

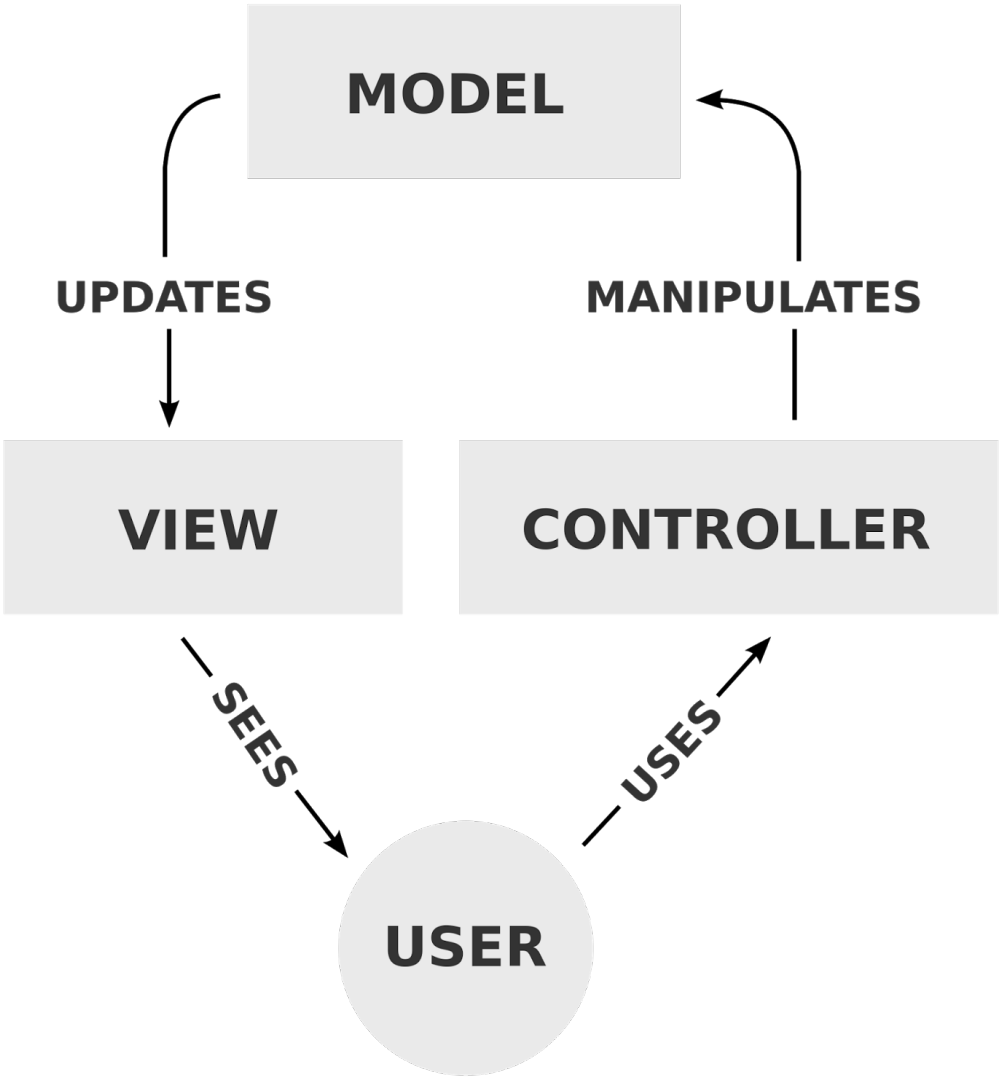


Figure 6: MVC Components

Model: The central component of the pattern. It is the application's dynamic data structure, independent of the user interface . It directly manages the data, logic and rules of the application.
View: Any representation of information such as a chart, diagram or table. Multiple views of the same information are possible, such as a bar chart for management and a tabular view for accountants.
Controller: Controller accepts input and converts it to commands for the model or view.

There are few advantages and disadvantages of MVC architecture as every architecture has. First, Let's see it's advantages.
Advantages of MVC architecture:

- Development of the application becomes faster.

- Easy for multiple developers to collaborate and work together.
- Easier to Update the application.
- Development of the application becomes faster

Disadvantages of MVC architecture:

- It is hard to understand the MVC architecture.
- Must have strict rules on methods.
- There is not much in the disadvantages part of the architecture. And the disadvantages are not so huge and are very easy to ignore in comparison with all the benefits we get.

4.1.2 Applying into our model

As we mentioned in the introduction of this section, the Model-View Controller is suitable for applying to our system. to be more specific, we would like to explain some features of MVC architecture that we applied to our system:

1. **Model:** takes the data of following classes from the database
    - *Food Item:*Each food item on the menu will have a unique ID, together with a description, information about the food, the number of available dishes the restaurant has at the moment, and the cost of that food item.
    - *Order:* Contain the unique Order ID that each order will have along with the date the order was made, each food item that the customer pick will be added to the order, and the customer can also remove an item or reduce the quantity of it, feedback can be added and the order ended when the customer selects the pay button.
    - *Transaction:* Transaction will take the Order ID as its ID since each Order ID is unique, the date and ID will be recorded and the customer will select the payment method that suits them. The amount they have to pay will also be shown.
    - *Account:* Each customer will have their own account to log in to, the account will make it easier for them to make payments, check bills, and along with other bonuses. They can still use a guess account if they choose not to sign up.
2. **View:** the main graphical user interface to interact with the system. Here we break the view into many different view components, each for a different user
    - *Customer view:* There will be a QR code and a website interface for the user to skim the menu and select food to add to the order. Customers can provide feedback and sign in or sign up on the website.
    - *Staff view:* This contains the POS system interface, which allows the staff to confirm the new incoming order, determine whether the order's dish can be served by the restaurant, record any errors that occur, and allow the staff to input order if the user pays in cash.
    - *Manager view:* Allow the manager to easily modify the menu and enter the number of dishes that the restaurant can serve so that the system can provide a better user experience for the customer. Furthermore, the system allows the manager to view the restaurant's report as well as the customer list.

3. **Controller:** the main role of the controller is to receive user action from the view and render the corresponding view, receive notification from the change of model and update the model according to the new view. The controller will contain all the logic flow of the restaurant.

    - *Customer controller:* The customer controller will have an order ID generator, it will handle the add or remove food item action in the order, there will be action to call the bank API for payment and a login and logout control.
    - *Staff controller:* The staff controller will mainly handle the order confirmation, record error if any happen, and notify the kitchen manager of incoming orders, and if the customer chooses to pay with cash, the staff will receive it and make changes to the order accordingly.
    - *Manager controller:* The manager controller will mainly handle administrative duty, modify menu and update dishes, generate reports and check customer information.

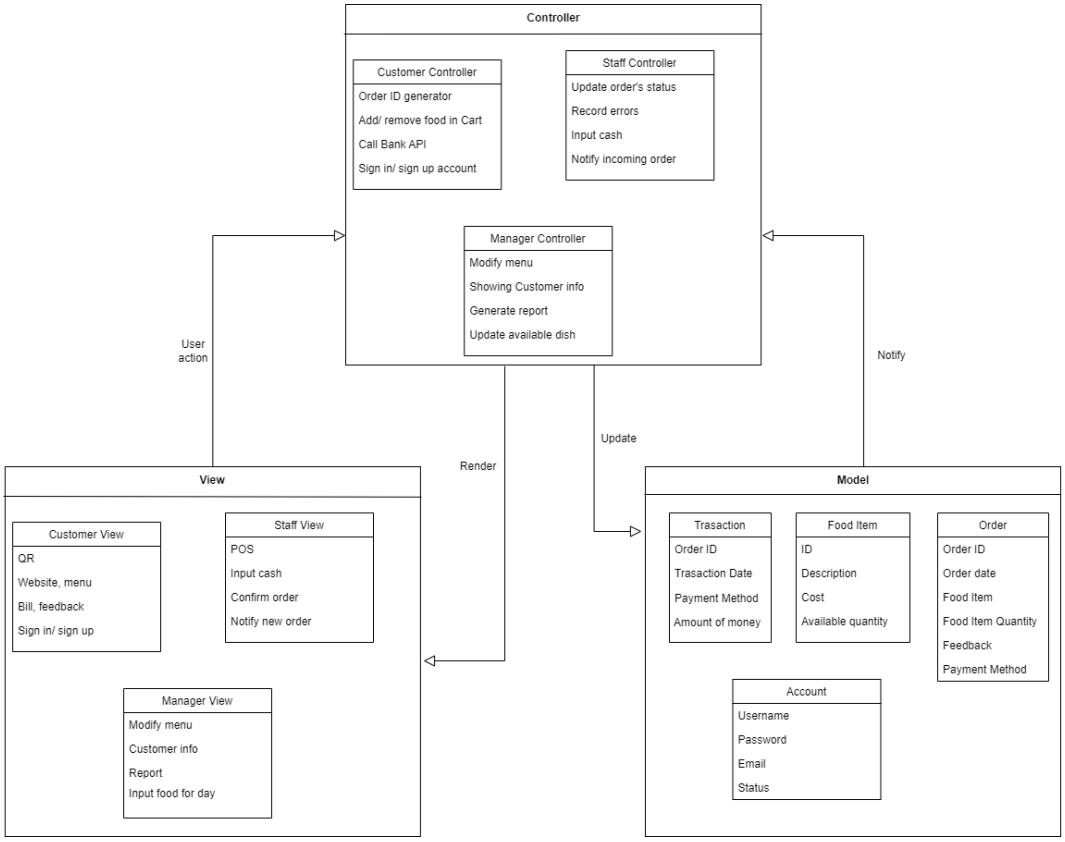


Figure 7: MVC diagram design

4.2 Implementation diagram

food_ordering_system.zip contain the two folders, pages, and component.

- The pages folder contains important pages of the food ordering system such as the home page, ordering page, and payment. This is what the Customers mostly see and interact with.

- The component folder consists of 5 files implementing 5 main components (foodcart.js, payment.js, fooditem.js, orders.js, possystem.js).

The system comprise of three main sub-systems: **WebRestaurant**, **Kitchen**, and**Accounting** and 2 separated components: **BankDatabase** and **POSSystem**.

Main subsystem:

1. The WebRestaurant subsystem contains the Food Cart and Payment:
    - The FoodCart component provides the interface for the customers to select and order food, access the cart, place order, and view order. Afterward, the staff will receive the order for confirmation.
    - The Payment component provides the interface for the customers to pay for the order after it had been placed and approved by the staff, it will interact with the Bank component if the customers choose to pay with a card, if not, the staff can take cash and give the bill in either case.
2. The Accounting subsystem will mainly deal with the orders from customers, taking into account, handling them, in combination with the POS system to notify the customers if food items is available or not.
3. The Kitchen subsystem is mainly used by the restaurant side, the restaurant can check for available food, quantity, ingredient, shipment, and other databases related.

Separated components to provide additional service:

- POS-System component provides OrderConfirmation and OrderDisplay interfaces for the customers to confirm their orders and display the order. It is the main control unit of the system.
- BankDatabase component provides the PaymentMethod interface for making transactions valid on the Payment component and recording the transaction. This will connect with the bank and is mainly used if the customers use a card for payment.

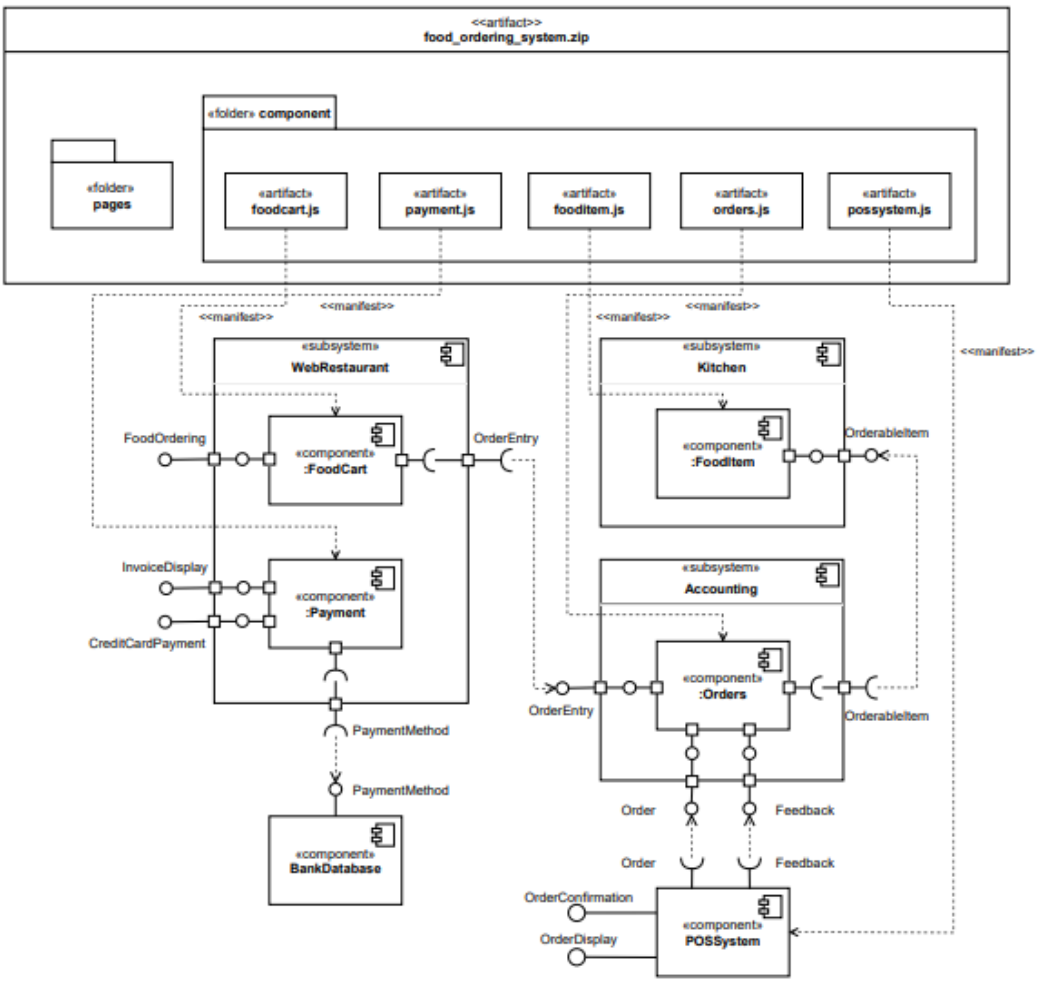


Figure 8: Implementation diagram

5 Implementation

5.1 Main page

After the customers access our website, the interface as shown will appear. In the top left corner of the screen is the name of the restaurant. Right below that is a row of fillers to help customers filter foods according to their needs, by price, rating, and the size of the dish, respectively.

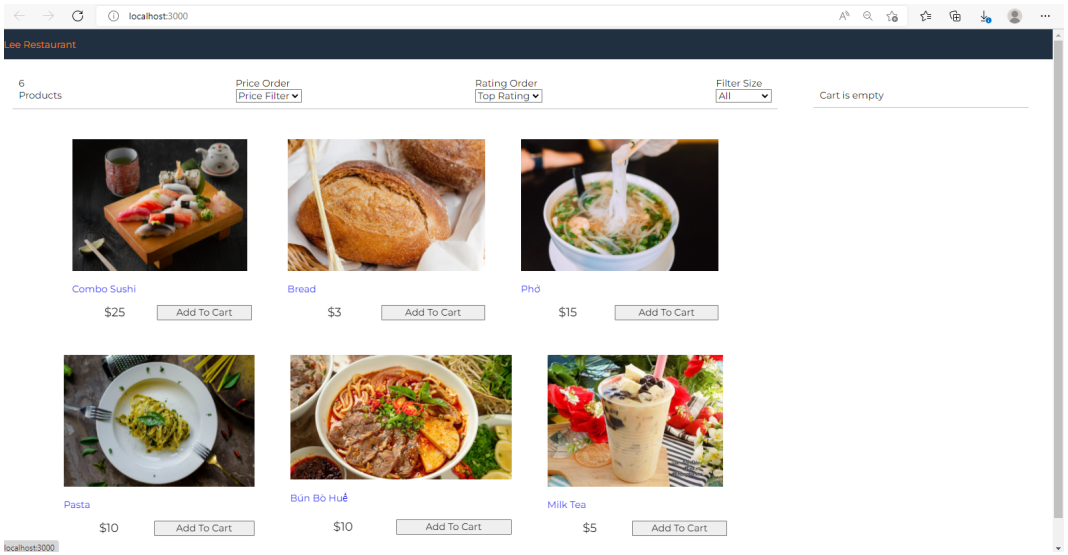


Figure 9: Home screen

On this interface, customers can click on each dish to see more detailed information about that dish.

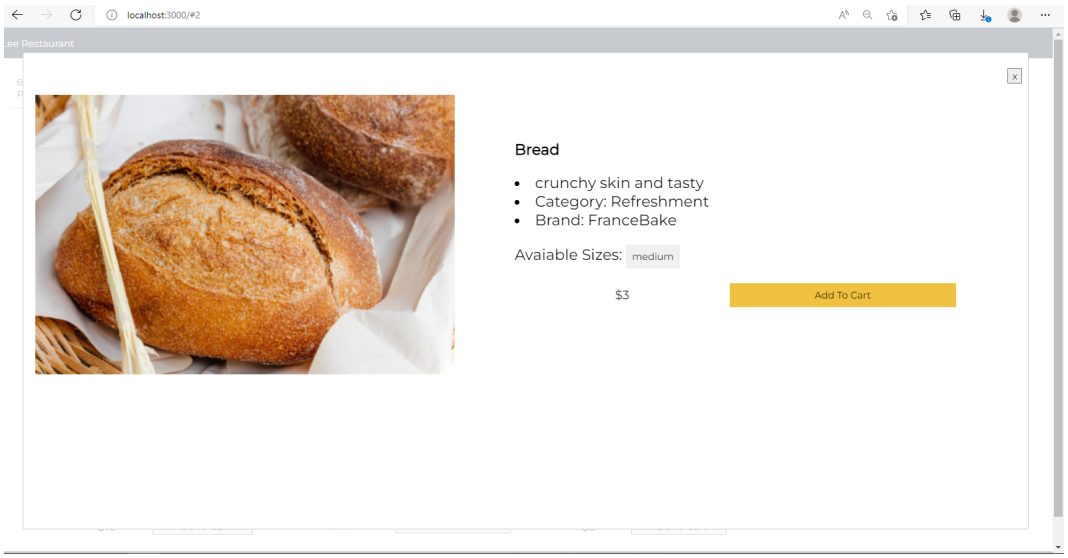


Figure 10: Food item information screen

Customers can click "Add To Cart" to add the item to the cart, customers can also click "Add To Cart" at the same item to increase the quantity of food in the cart. The total amount of the order will appear at the bottom of the cart, next to the "Proceed" button.

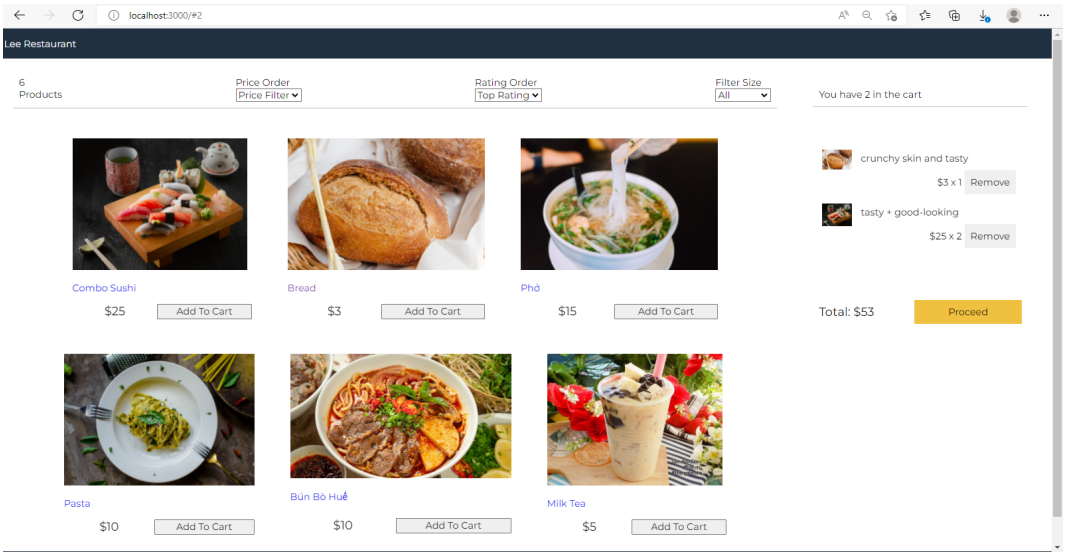


Figure 11: Cart screen

After completing the selection process, the customer can click "Proceed" to start the ordering process. In this step, we will ask the customer to log in to their account to proceed with the order payment.

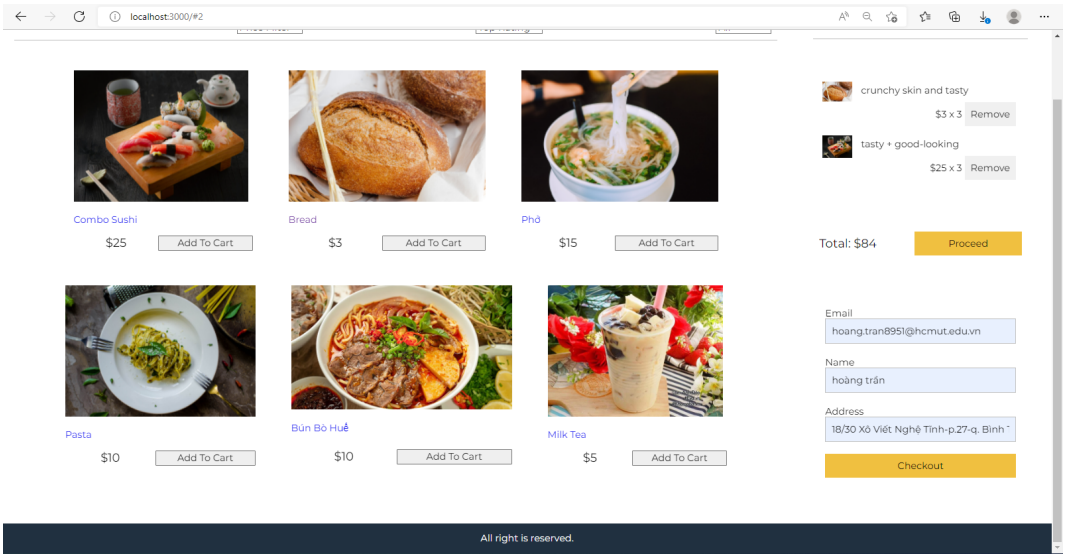


Figure 12: login screen