서울과학기술대학교
SEOULTECH

Software Engineering

Final Report

20102104 Kim Sua, 0213tndk@seoultech.ac.kr

20102122 Jeong Hyoan, goodhyoan@seoultech.ac.kr

20102125 Pi Yujin, 20102125@seoultech.ac.kr

IT Management, SeoulTech

Table of Contents

Abstract

"Re-dish" is a delivery service that aims to reduce the use of disposable items when using delivery services through using reusable containers. "Re-dish" lends, returns, washes containers. This service can contribute to reducing the use of disposable containers used when ordering delivery food. In order to implement this application , we organized work breakdown structures and defined functional and non-functional requirements. In functional requirements, restaurant owners can order containers to 'Redish' and deliver foods using those containers. After eating, customers can request a pick-up service to 'Re-dish' and 'Re-dish' pick-up, wash, and store in stocks. Non-functional requirements include background requirements such as database management and security. The scenario indicates how customers, restaurant owners, and 'Re-dish' employees use the app. Finally, we define domain specification and design UML diagram, user interface, database design including use case diagram, class diagram, and sequence diagram.

1. Necessity of the term project

Over the past two years, due to Covid19, the usage of delivery apps has increased exponentially. And also the usage of disposable products has also increased significantly. In this situation, many people are worried about the amount of disposable products and interested in ways to reduce the usage of disposable products. And our application 'reDish' can meet the needs for convenience of delivering service and the needs for preventing environmental pollution.

In terms of restaurants, owners suffer from rising prices and rising minimum wages. 'reDish' can reduce the necessity of hiring a person who cleans their containers. From the perspective of consumers, they do not have to handle leftover food and plastics from delivery. Users no longer need to feel guilty about disposable products using delivery services. These are the reasons why this app is essential in that both customers and restaurants are satisfied and can realize social value.

2. Goal of the term project

Our final goal is that many people can reduce the use of disposable products in their daily lives and participate in environmental protection by using our service (Re-dish app). The main service of our application is to lend Re-dish's own reusable container to the store continuously. It reduces the cost of purchasing disposable products every time,

reducing the financial burden. Furthermore, our goal is to stabilize small business owners by stabilizing brokerage fees.

Function of Re-dish (our service)	**Expected Result**
Only show stores which use reusable containers	Consumers can conveniently check stores which use reusable containers and use reusable containers without carrying them around themselves.
Return reusable containers through photo upload and deliver containers to the store	***consumer*'s point of view** : It is convenient because there is no need to clean or separate collection of waste. ***store*'s point of view :** Stores do not need to collect containers by themselves. Stores can cut down the cost of hiring delivery men and time can be shortened.
Wash reusable containers	Stores do not need to clean themselves. They can reduce the cost of hiring or buying separate dishwashers.
Show reusable container order history	***consumer*'s point of view** : Consumers can see how much they have participated in environmental protection. A circumstance is created in which people can participate continuously. ***store*'s point of view :** Store makes a good impression on consumers with the image of participating in environmental protection. It can create a virtuous cycle structure for both consumers and stores.

3. Problem Description

There are two people here who have come to use the 'Re-dish' application due to personal circumstances.

Stella: A 21-year-old living alone in Gongneung and usually orders a lot of delivery food. And she regrets the huge amount of disposable waste that comes after eating all the delivery food, but she is lazy to cook every day, and she is troubled these days because of the convenience of delivery food.

Emma: She is 40 years old and the owner of a salad shop in Gongneung. Her store is a delivery-only store, and whenever she buys disposable containers every month due to the large amount of delivery, she hates the adverse environmental impact of them, so she is thinking of using reusable containers instead of disposable ones. However, using reusable containers is not practicable due to the hassle of collecting and cleaning the containers again every order.

In this situation, they became aware of our service. So Emma downloaded the application, ran it, clicked the sign up button, entered her salad shop name by nickname, business registration number by ID, and authenticated by phone number. And after entering the location and phone number of the restaurant, she agreed to use the store information and became a member of reDish. Then she logged in, and entered greetings, store information, menu information, delivery tips (2500won) and review events. And she registered the proper container for each menu. And she ordered ten containers of one of them.

When Emma ordered the container, the administrator of reDish received an alarm to receive the container order and confirmed the order. And the quantity of the corresponding container in stock was reduced by 10. Then on October 31, 2022, reDish began delivery to Emma's store, and on November 1, the containers arrived at the restaurant. Then Emma received the container delivery alarm and clicked the confirm button.

Stella downloaded and ran the app on November 2. When Stella signed up and logged in, it was redirected to the store list. She wanted to eat salad, so she entered the salad in the search function, and then used the filter function to see shops with delivery tips of less than 3,000 won. Then Emma's salad shop appeared on the store list, and Stella, who clicked on the store, saw the menu information and put the chicken salad in the shopping cart. Then she entered the food delivery address and phone number, and filled out the store requests. Next, Stella selected the payment method by card, checked the total payment amount, and clicked the confirm button to complete the food order. A point equivalent to 0.5% of the amount she paid was set aside, and she paid 1,000 won in advance for the container deposit. Then Emma's restaurant received a notification of food order acceptance and confirmed it. Stella was notified of her acceptance of the food order, and about 30 minutes later, the salad she ordered arrived in a reusable container. Having finished eating the salad, Stella entered the pickup page to request a container pickup, selected Stella's store from the food order history and took a picture of the container in front of the door and uploaded it. And when she clicked the pick-up request button, she received a scheduled pick-up message.

When Stella's order came into Emma's store and delivery began, the reddish administrator checked that the location information of the container corresponding to the chicken salad moved from the restaurant to Stella's house. And he received a

notification of receiving a stella's pickup request. The reDish delivery man went to her house during the scheduled pick-up time to retrieve the container. And when he retrieved the container, he checked the type and quantity of the container to be picked up. When the information matched, the delivery man clicked the confirmation button, and the deposit of the container paid in advance by Stella was returned. Then, the responsibility for the loss of the container went to the staff who retrieved the container from Stella. The containers that were picked up were cleaned and the reDish updated the quantity of the washed containers in stock.

Emma, who used the reDish service, no longer had to purchase disposable containers for delivery, so she was able to reduce the cost of purchasing them, and it was very satisfactory to reduce environmental pollution caused by disposable containers without additional inconvenience. She therefore decided to continue using reDish's services. Stella didn't feel guilty every time she ate food because she didn't make disposable container garbage when she ordered food with reDish, and naturally her worries were solved.

4. Team Roles

- Sua : Necessity, functional requirements, Class diagram, Interface Design, Detailed Design

- Hyoan: Problem Description, Scenarios, Sequence diagram, Interface Design, Detailed Design

- Yujin : Goal, Nonfunctional requirements, User Interface Design, DB Design, Interface Design, Detailed Design

5. Functional Requirement Specification

Category	Requirement Identifier	Requirement Name
Membership (Mem)	FR – Mem - 1	Customer Sign up
	FR – Mem – 2	Restaurant Sign up
	FR – Mem – 3	Login
	FR – Mem – 4	Logout
	FR – Mem - 5	Withdrawal
Store List (SL) - Customer's Main Page	FR – SL - 1	Search Restaurant
	FR – SL – 2	Show Restaurant List
	FR – SL – 3	Select Food Category
	FR – SL - 4	Filtering Restaurant
Food Order (FO)	FR – FO - 1	Show Menu Information
	FR – FO – 2	Show Restaurant Information
	FR – FO – 3	Customer's Shopping Basket

	FR – FO – 4	Specify Food Order
	FR – FO - 5	Payment for Food Order
	FR – FO - 6	Food Order Reception
	FR – FO - 7	Food Order Cancellation
Pick – up (PU)	FR – PU - 1	Food Order History
	FR – PU - 2	Pick-up Request
Map (M)	FR – M - 1	Show Food Category on Map
	FR – M - 2	Show Restaurant Information on Map
Post (P) -Restaurant's Main Page	FR – P - 1	Upload Menu Information
	FR – P – 2	Upload Restaurant Information
	FR – P – 3	Revise Menu Information
	FR – P – 4	Revise Restaurant Information
	FR – P - 5	Review Management
Container Order (CO)	FR – CO - 1	Container Information

	FR – CO – 2	Container Order History
	FR – CO – 3	Restaurant's Shopping Basket
	FR – CO – 4	Specify Container Order
	FR – CO - 5	Payment for Container Order
	FR – CO - 6	Container Order Cancellation
My Page – customer (MPC)	FR – MPC - 1	Show Customer Basic Information
	FR – MPC – 2	Revise Customer Basic Information
	FR – MPC – 3	Permission Settings
	FR – MPC – 4	My Review Management
	FR – MPC - 5	Visualize Usage
	FR – MPC - 5	Customer point
My page – restaurant (MPR)	FR – MPR - 1	Show Restaurant Basic Information
	FR – MPR – 2	Revise Restaurant Basic Information
	FR – MPR - 3	Sales History

	FR – MPR - 3	Restaurant point
Notification (N)	FR – N - 1	Pick – up Schedule Alarm
	FR – N – 2	Return Request Alarm
	FR – N – 3	Order Receipt Alarm
	FR – N - 4	Container Delivery Completion Alarm
Administration (A)	FR – A - 1	Container Order Management
	FR - A - 2	Container Pick- up Request
	FR - A - 3	Container Washing and Inventory
	FR - A - 4	Inventory management
	FR - A - 5	Container Location Tracking and Inventory Control

Req. Identifier	**FR – Mem - 1**
Category	Membership
Req. Name	Customer Sign up
Actor	Customer
Details of Functional Requirement	1. User executes the application. 2. After clicking the 'Sign-up' button, users can fill up the application form.

	3. User can set nickname. 4. Users have to enter their phone number. It will be used as an ID. 5. If the phone number already exists, error occurs with message '이미 존재하는 전화번호입니다.' . 6. User can get certification using their phone number. 7. User can set passwords. (Condition : At least one English, number, and special character) 8. If the conditions are not satisfied, error occurs with message '비밀번호는 영어, 숫자, 특수문자를 각각 1 개 이상씩 포함하여야 합니다.' . 9. User have to enter a password two times. 10. Check the duplication of password pressing the 'Password Confirm' button. - If the password values are not same, error occurs with message '비밀번호가 일치하지 않습니다.' . 11. For finishing the sign-up process, user has to check agreement with personal information utilization and GPS information utilization.

Req. Identifier	**FR – Mem - 2**
Category	Membership
Req. Name	Restaurant Sign up
Actor	Restaurant
Details of Functional Requirement	1. User executes the application. 2. After clicking the 'Sign-up' button, user can fill up the application form. 3. User can put the name of the restaurant which will be set to the user's nickname. 4. User have to put their business number to be used as ID. - If the business ID already exists, error occurs with message '이미 등록된 사업자번호입니다.' . 5. User have to enter their phone number.

	- If the phone number already exists, error occurs with message '이미 존재하는 전화번호입니다.' . 6. User can get certification using their phone number. 7. User can put the information of the restaurant's telephone number and location information. 8. User can set passwords. (Condition : At least one English, number, and special character) - If the conditions are not satisfied, error occurs with message '비밀번호는 영어, 숫자, 특수문자를 각각 1 개 이상씩 포함하여야 합니다.' . 9. User have to enter a password two times. 10. Check the duplication of password pressing the 'Password Confirm' button. - If the password values are not same, error occurs with message '비밀번호가 일치하지 않습니다.' For finishing the sign-up process, user have to check agreement with restaurant information utilization.

Req. Identifier	**FR- Mem - 3**
Category	Membership
Req. Name	Login
Actor	Customer and Restaurant
Details of Functional Requirement	1. When the user is already logged in the application(using automation login function), skip login. 2. When the user is logged out or just joined the member, the application goes to the 'Log in' page. 3. User enters an ID and password and clicks the 'Log-in' button. - User can check the 'automation login' button. 4. If ID and password are well matched, log in completed, customer moves to 'store list' page and restaurant owner moves to 'post' page.

	- If ID or password is not correct, error occurs with message '아이디 또는 비밀번호가 일치하지 않습니다.' And shows the login page again. 5. If a user cannot remember their password, user can click 'password finding' button. - After the user input the registered telephone number and get certification, user can get a temporary password through a message. User can change a temporary password in 'My Page' after login to reDish. Customer can log in with the same ID at most 2 devices, but restaurant can log in with same ID at most 5 devices.

Req. Identifier	**FR- Mem - 4**
Category	Membership
Req. Name	Logout
Actor	Customer and Restaurant
Details of Functional Requirement	1. User can click the 'Logout' button in 'My Page'. 2. When user click 'Log out' button, pop up the message '로그아웃 하시겠습니까?' 3. When user click '네', continue log out procedure and show 'Login' page. When user click '아니오', user can see the previous page 'My Page'.

Req. Identifier	**FR- Mem - 5**
Category	Membership
Req. Name	Withdrawal
Actor	Customer and Restaurant
Details of Functional Requirement	1. Users can click the 'Delete Account' button in 'My Page'.

	2. When user click ‘ Delete Account’ button, pop up the message ‘정말로 탈퇴하시겠습니까?’ 3. User have to check the message ‘탈퇴하시면 reDish 와 관련된 모든 정보, 기록이 사라집니다.’ - If user do not check the button for this message, user cannot press the withdrawal ‘네’ button. 4. When user clicks ‘네’ button, user’s account will be logged out and removed. And user can see the ‘Log in’ page. When user clicks ‘아니오’, user can see the previous page ‘My Page’. 5. Remove all information of the user. ‘My Page’.

Req. Identifier	**FR – SL -1**
Category	Store List
Req. Name	Search Restaurant
Actor	Customer
Details of Functional Requirement	1. User can search specific restaurant’s name 2. Users can search for specific food. And user can see the restaurants which sell that food. - If there are no corresponding results, print the message ‘00 에 대한 검색 결과는 존재하지 않습니다.”

Req. Identifier	**FR – SL -2**
Category	Store List
Req. Name	Show Restaurant List
Actor	Customer
Details of Functional Requirement	1. This is the initial page when clicking ‘Store List’. 2. User can see the list of restaurants randomly. - With brief information (restaurant name, score, minimum order amount, delivery tip, avg of delivery time)

	3. When a user clicks on a specific restaurant, the user moves to the 'Order' page' and can see the specific information or menus.

Req. Identifier	**FR – SL -3**
Category	Store List
Req. Name	Select Food Category
Actor	Customer
Details of Functional Requirement	1. User can select food types. (Korean food, Western food, Japanese food, Chinese food, Snack food, Fast food, Café) 2. When a user chooses a specific food type, the user can only see the list of restaurants which offer that food.

Req. Identifier	**FR – SL -4**
Category	Store List
Req. Name	Filtering Restaurant
Actor	Customer
Details of Functional Requirement	1. User can filter the restaurant according to the conditions they want. (Minimum order amount, Delivery tip, The number of orders, Scores) 2. User can filter more than two conditions at the same time. 3. When the user sets the conditions (filtering), the user can only see the list of corresponding restaurants.

Req. Identifier	**FR – FO -1**
Category	Food Order
Req. Name	Show Menu Information

Actor	Customer
Details of Functional Requirement	1. When the user chooses the specific restaurant, the user can see all menus with restaurant information. 2. User can see the menu which is available and sold out. 3. User can also see the type of containers corresponding to each food.

Req. Identifier	**FR – FO - 2**
Category	Order
Req. Name	Show Restaurant Information
Actor	Customer
Details of Functional Requirement	1. When the user clicks the 'Restaurant Information 'button, the user can see the basic information of that restaurant. 2. Restaurant's basic information contains location, telephone number, business hours, brief introduction, delivery area, and customer's review. 3. Customer can click 'Menu' button to see menus.

Req. Identifier	**FR – FO - 3**
Category	Food Order
Req. Name	Customer's Shopping Basket
Actor	Customer
Details of Functional Requirement	1. User can put any kind of food in their shopping basket. 2. Shopping baskets can show information about the kinds of foods, quantity, and the total price of order. 3. When the user clicks the 'Order now' button, user can move to 'Specify Order' section. When the user clicks the 'Go back' button, the user can see the list of menus again.

Req. Identifier	FR – FO - 4
Category	Food Order
Req. Name	Specify Food Order
Actor	Customer
Details of Functional Requirement	1. When the user finishes selecting the menu, the user can click 'Order now' button and move to the order specification section. 2. User choose between delivery or take-out(packaging). 3. User have to set delivery destinations and phone numbers. - Registered basic delivery destinations enter automatically in the delivery destination section. In the case of a take-out user, he does not need to enter his delivery destination, only phone number. 4. User can check the order list with menus, quantity. 5. User can make requests to restaurants. 6. When the user clicks the 'Payment' button , the user can move to 'Payment for Food Order'. - If the delivery user's order amount is less than the restaurant's minimum order amount, 'Payment' button will not be activated.

Req. Identifier	FR – FO - 5
Category	Food Order
Req. Name	Payment for Food Order
Actor	Customer
Details of Functional Requirement	1. When user clicks the 'Payment' button, user can choose payment method. 2. Payment method consists of card or face-to-face payment. 3. User can register credit cards or accounts for convenient payment. 4. After choosing a payment method, the user can see the total price (food price + delivery tip).

	5. When the user clicks the 'Confirm' button, the payment will proceed. 6. If payment is completed successfully, the order will send to restaurant. 7. Customer can earn points as the 0.5% of payments. Points will be used like cash when ordering food through Re-dish. - The user has to pay the deposit and the price of the food together

Req. Identifier	**FR – FO - 6**
Category	Food Order
Req. Name	Food Order Reception
Actor	Restaurant
Details of Functional Requirement	1.When customers order food through Re-dish, restaurants can receive food orders through Re-dish. 2. Restaurants can accept or reject the order according to their situation. 3. Through this order information, Re-dish can track containers(type, quantity, etc.).

Req. Identifier	**FR – FO - 7**
Category	Food Order
Req. Name	Food Order Cancellation
Actor	Restaurant
Details of Functional Requirement	1.If a customer requests the cancellation of a food order or a restaurant wants to cancel the order for some reasons, restaurant can cancel the food order and retrieve the containers from customer. - Re-dish can track the container's location.

Req. Identifier	FR – PU - 1
Category	Pick - up
Req. Name	Food Order History
Actor	Customer
Details of Functional Requirement	1. User can see their own history of food order. 2. This history list shows the restaurant name, date and time of order, and order specification. 3. When the user clicks a specific order history, the user will move to the 'Pick-up Request' section of that order.

Req. Identifier	FR – PU - 2
Category	Pick - up
Req. Name	Pick-up Request
Actor	Customer
Details of Functional Requirement	1. When a user clicks a specific order in the 'Order History' section, the user can request 'Pick-up' service. 2. When user request 'Pick-up' service, pick- up location is set to the delivery destination automatically. User can change this location also. 3. User have to upload the pictures of containers being put out in front of their doors. - User can permit camera and album utilization. - User can upload the photos using a camera or album. 4. When the pick-up employees collect the containers and check the number of plates, user gets a deposit back.

Req. Identifier	FR – M - 1
Category	Map
Req. Name	Show Food Category on Map

Actor	Customer
Details of Functional Requirement	1. User can select food types. (Korean food, Western food, Japanese food, Chinese food, Snack food, Fast food, Café) 2. When a user choose specific food type, user can only see the location of restaurant which offers that food.

Req. Identifier	**FR – M - 2**
Category	Map
Req. Name	Show Restaurant Information on Map
Actor	Customer
Details of Functional Requirement	1. User can show the restaurant on the map. 2. When a user clicks a specific restaurant, user can see the brief information of that restaurant. 3. Brief information consists of restaurant name, food category, location, telephone number. 4. When the user clicks the '+' button in the brief information section, user will move to the 'Order – Menu Information' page.

Req. Identifier	**FR – P - 1**
Category	Post
Req. Name	Upload Menu Information
Actor	Restaurant
Details of Functional Requirement	1. Restaurant can upload menu information consists of food name, price, brief introduction, materials, the types of container and additional options.

Req. Identifier	**FR – P - 2**

Category	Post
Req. Name	Upload Restaurant Information
Actor	Restaurant
Details of Functional Requirement	1. Information of restaurant name, telephone number, and location will be set automatically using 'Sign-up' information. 2. Restaurant can set delivery tips by order prices. 3. User can enter brief sentences like review events. 4. User have to put in business times and holidays. 5. User can set available delivery areas.

Req. Identifier	**FR – P - 3**
Category	Post
Req. Name	Revise Menu Information
Actor	Restaurant
Details of Functional Requirement	1. User can revise food name, price, brief introduction, materials, the type of containers and additional options. 2. When a specific menu is out of stock, user can set 'sold out' mark. 3. User can delete specific menus which is not available anymore.

Req. Identifier	**FR – P - 4**
Category	Post
Req. Name	Revise Restaurant Information
Actor	Restaurant
Details of Functional Requirement	1. User can revise delivery tips, brief sentences, business times and holidays, and available delivery areas whenever they want.

Req. Identifier	FR – P - 4
Category	Post
Req. Name	Review Management
Actor	Restaurant
Details of Functional Requirement	1. User can reply to the customer's review. 2. User can change the scope of publication of the review can be set to 'private' so that only authors and users can see it. But the user cannot delete the review.

Req. Identifier	FR – CO - 1
Category	Container Order
Req. Name	Container Information
Actor	Restaurant
Details of Functional Requirement	1. User can see many kinds of container information. 2. When user clicks specific container, user can see specific information of container.(size, shape, price, pictures, available amount)

Req. Identifier	FR – CO - 2
Category	Container Order
Req. Name	Container Order History
Actor	Restaurant
Details of Functional Requirement	1. When the user clicks the 'Container Order History' button, the user can see lists of orders with container types, quantity, total price, and date. 2. When a user chooses a specific history order and clicks the ' To my basket' button, the restaurant's shopping basket will have the same order.

Req. Identifier	FR – CO - 3
Category	Container Order
Req. Name	Restaurant's Shopping Basket
Actor	Restaurant
Details of Functional Requirement	1. User can put any kind of container in their shopping basket. 2. Shopping baskets can show information about the kinds of containers, quantity, and the total price of order. 3. When the user clicks the 'Order now' button, user can move to the 'Specify Order' section. When the user clicks the 'Go back' button, the user can see the list of containers again. - User can order each container in units of 10. (10, 20,…100)

Req. Identifier	FR – CO - 4
Category	Container Order
Req. Name	Specify Container Order
Actor	Restaurant
Details of Functional Requirement	1. When the user finishes selecting containers, user can click the 'Order now' button and move to the order specification section. 2. User can choose the delivery date and time. - Reservations can be made up to 3 days in advance. And 'reDish' provides the day delivery also. 3. User have to set delivery destinations and phone numbers. - Registered restaurant's location enter automatically in the delivery destination section. 4. User can check the order list with containers, quantity. 5. User can put requests to 'reDish'. 6. When the user clicks the 'Payment' button , the user can move to 'Payment for Container Order'.

Req. Identifier	FR – CO - 5
Category	Container Order
Req. Name	Payment for Container Order
Actor	Restaurant
Details of Functional Requirement	1. When user's clicks the 'Payment' button, user can choose payment method. 2. User have to prepaid the order and choose the payment method consisting of cards or accounts. 3. User can register credit cards or accounts for convenient payment. 4. After choosing payment method, user can see the total price. 5. When user clicks the 'Confirm' button, the payment will proceed. 6. If payment is completed successfully, order will be sent to 'reDish'. - This cost is for rent. User pays for the rental container. 7.Restaurant can earn points as 0.5% of rental fee. Points will be used like cash when ordering containers through Re-dish.

Req. Identifier	FR – CO - 6
Category	Container Order
Req. Name	Container Order Cancellation
Actor	Restaurant
Details of Functional Requirement	1.(Preference Change) If a restaurant does not like the containers and wants to change the containers after receiving the order, the restaurant can request container order cancellation and pay delivery fee to Re-dish. 2. (Faulty) If the container is damaged during delivery, the restaurant can request partial replacement within one day for free.

Req. Identifier	FR – MPC - 1
Category	My Page - Customer
Req. Name	Show Customer Basic Information
Actor	Customer
Details of Functional Requirement	1. In 'My Page', users can see the registered information. (nickname, delivery destination, and phone number) 2. When the user clicks the 'Order History' button, the user can see the order history with restaurant name, date, and menu types. And when the user clicks specific order, the user can see more specific order information.

Req. Identifier	FR – MPC - 2
Category	My Page - Customer
Req. Name	Revise Customer Basic Information
Actor	Customer
Details of Functional Requirement	1. User can revise basic information. (nickname, password, and phone number) 2. User can see the list of delivery destinations and user can change default delivery destinations. User can add or revise delivery destinations.

Req. Identifier	FR – MPC - 3
Category	My Page - Customer
Req. Name	Permission Settings
Actor	Customer
Details of Functional Requirement	1. User can change location permission.

	- If the user does not agree with the utilization of location information, the user cannot use 'map' section. 2. User can change camera and album permission. - If the user does not agree with utilization of camera/ album permission, the user cannot request 'pick-up' service.

Req. Identifier	**FR – MPC - 4**
Category	My Page - Customer
Req. Name	My Review Management
Actor	Customer
Details of Functional Requirement	1. User can see history orders. 2. If a user wrote reviews of a specific order, the user can see the written review under each specific order. 3. If a user wants to write a review for a specific order, the user can write a review by clicking the specific order. 4. User can revise or delete written reviews.

Req. Identifier	**FR – MPC - 5**
Category	My Page - Customer
Req. Name	Visualize Usage
Actor	Customer
Details of Functional Requirement	1. When user clicks the 'Visualize Usage' button, user can see their contributions. - For example, approximate reusable container usage is calculated based on the user's order history. If user used a reusable container 10 times, display the tree picture with the message "You plant a tree !". 2. And user can also see the next contribution criteria for saving more disposable things and planting more trees.

Req. Identifier	FR – MPC - 6
Category	My Page - Customer
Req. Name	Customer point
Actor	Customer
Details of Functional Requirement	1. Customers can receive 0.5% of the payment as a point. 2. The point can be used like cash when ordering food through Re-dish.

Req. Identifier	FR – MPR - 1
Category	My Page - Restaurant
Req. Name	Show Restaurant Basic Information
Actor	Restaurant
Details of Functional Requirement	1. In 'My Page', user can see the registered information. (business number, restaurant name, location, restaurant's telephone number, and manager phone number)

Req. Identifier	FR – MPR - 2
Category	My Page - Restaurant
Req. Name	Revise Restaurant Basic Information
Actor	Restaurant
Details of Functional Requirement	1. User can revise basic information. (restaurant name, location, restaurant's telephone number, manager phone number, password) 2. User can see the list of delivery destinations and user can change the default delivery destination for container delivery. User can add or revise delivery destinations.

Req. Identifier	FR – MPR - 3
Category	My Page - Restaurant
Req. Name	Sales History
Actor	Restaurant
Details of Functional Requirement	1. User can see the list of sales history with order specification, date, and delivery destination. 2. User can see total sales benefits filtering by weekly or monthly.

Req. Identifier	FR – MPR - 4
Category	My Page - Restaurant
Req. Name	Restaurant point
Actor	Customer
Details of Functional Requirement	1. Restaurant can receive 0.5% of the payment as a point. 2. The point can be used like cash when ordering containers through Re-dish.

Req. Identifier	FR – N - 1
Category	Notification
Req. Name	Pick-up Schedule Alarm
Actor	Customer and Administrator
Details of Functional Requirement	**Customer :** 1. When customers upload the pictures and request 'pick-up', customers will receive the message when the next 'pick-up' time. Ex) Your container will pick up at 17:00 – 18:00 **Administrator :**

	1. When customers request pick up service, administrator will receive the message "그릇 회수 요청이 업로드 되었습니다." .

Req. Identifier	**FR – N - 2**
Category	Notification
Req. Name	Return Request Alarm
Actor	Customer
Details of Functional Requirement	1. If customer does not request pick-up service within 3 hours after ordering, customer will get message "그릇 수거 요청을 해주세요." 3 times for each order. (After 3 hours, 1 day, 3days)

Req. Identifier	**FR – N - 3**
Category	Notification
Req. Name	Order Receipt Alarm
Actor	Customer, Restaurant and Administrator
Details of Functional Requirement	**Customer :** - When a restaurant checks the food order and confirms it, customers can get an order acceptance or order reject message. **Administrator :** - When a restaurant owner orders a container, the administrator will receive the order and make sure whether to take an order or not. **Restaurant :** - When a customer orders food, the restaurant will receive the order and make sure whether to take an order or not - When an administrator checks the container order and confirms it, the restaurant can get order acceptance or order reject messages.

Req. Identifier	FR – N - 4
Category	Notification
Req. Name	Container Delivery Completion Alarm
Actor	Restaurant
Details of Functional Requirement	1. When 'reDish' finishes delivering the container to the restaurant, restaurant can get the messages "그릇 배달이 완료되었습니다." 2. User(restaurant) can click the 'confirm' button for confirming the container delivery.

Req. Identifier	FR – A - 1
Category	Administration
Req. Name	Container Order Management
Actor	Administrator
Details of Functional Requirement	1. Re-dish receive the container order and check the stocks of each container. 2. If there is enough stock, accept the order, and if there is not enough stock, reject the order 3. When the order is accepted, user receive the rental fee from restaurants. 4. User can see the basic restaurant information and delivery location. 5. After deliver containers to the restaurant, user click the complete button and send the message "배달이 완료되었습니다.". 6. After delivering containers, if restaurants request container cancellation, Re-dish employee confirms that and retrieve or re-deliver the containers according to restaurant's request.

Req. Identifier	FR – A - 2
Category	Administration

Req. Name	Container Pick- up Request
Actor	Administrator
Details of Functional Requirement	1. User receive the pick-up request alarm. 2. By the automatic path setting recommendation system, re-dish employees pick up the container. 3. Employee check and register the number of containers and the type of containers. 4. After pick-up the containers, employee press the pick-up complete button. 5. After collecting more than a certain amount of containers, employees deliver all containers to 'Re-dish washing station' and change the state to 'Washing now'.

Req. Identifier	**FR – A - 3**
Category	Administration
Req. Name	Container Washing and Inventory
Actor	Administrator
Details of Functional Requirement	1. Containers that have been cleaned are transferred to the stockroom. 2. The user updates the type of containers and the number of containers to the 'Re-dish' app. 3. This stock information will be used for receiving container orders.

Req. Identifier	**FR – A - 4**
Category	Administration
Req. Name	Inventory management
Actor	Administrator
Details of Functional Requirement	1. Re- dish can add or delete the amount of existing container (change the state)

	2. Re-dish can introduce and add the new container. - Re-dish has the ownership of the container.

Req. Identifier	**FR – A - 5**
Category	Administration
Req. Name	Container Location Tracking and Inventory Control
Actor	Administrator
Details of Functional Requirement	'reDish' can track the approximate location of containers. 1. 'reDish' can get container orders through the app and deduct the inventory according to each order. Re-dish can know the type and number of containers that have been moved to the specific restaurant. 2. When a customer's food order occurs, 'Re-dish' can track the container location using the delivery destination. 'Re-dish' has information about the types of containers depending on the type of food. So, Re-dish can track the container's type, amount, location through delivery order information. 3. 'Re-dish' can check the current status through uploaded pick up requests in real time. 4. 'Re-dish' picks up uploaded containers by themselves. - There is a fixed pick-up schedule for each area. 5. After pick-up containers, Re-dish employees update the status. And Re-dish can know the current status of the number, type, and approximate location of containers. 6. When pick-up employees arrive at 'Re-dish washing center', 'Re-dish' can check uncollected types and amounts of containers. - Re-dish has the menu and container type information, delivery order

	information and pick-up information. So Re-dish can specify the order and the location where the containers are lost. 7. After cleaning containers, the container will be in stock and Re-dish can update the inventory. The amount of containers are uploaded to 'Re-dish' inventory state. - Re-dish can check the state.

6. Nonfunctional Requirement Specification

Number	Category	Non - Functional Requirements	
NFR-1	Product requirements	Usability requirements	
NFR-2		Efficiency requirements	Performance requirements
NFR-3			Space requirements
NFR-4		Reliability requirements	
NFR-5		Portability requirements	
NFR-6	Organizational requirements	Delivery requirements	
NFR-7		Implementation requirements	
NFR-8		Standards requirements	
NFR-9	External requirements	Interoperability requirements	
NFR-10		Ethical requirements	

NFR-11		Legislative requirements	Privacy requirements
NFR-12			Safety requirements

Number	**NFR-1**
Category	Product requirements
Requirement Name	Usability requirements
Specification	- Wi-fi or mobile data connection are needed. - Korean and English will be provided as default language.- - When error occurs, error message will be shown and the application will be restarted. - All personal data is shown only to oneself - User interface will be easy to use even for those who are new user of application. - The application provides help or a manual for each function for helping user's use

Number	**NFR-2**
Category	Product requirements
Requirement Name	Efficiency requirements - Performance requirements
Specification	- Quick responsiveness and quick processing time should be guaranteed in less than a second. - When customer post or revise information, it is reflected directly in the customer's view. - After logging in, the time to retrieve information for that user from database must be within 10 seconds.

Number	**NFR-3**
Category	Product requirements
Requirement Name	Efficiency requirements - Space requirements

Specification	- Unused and unnecessary memory will be deleted automatically - User can upload up to 1MB (photo) file when they request pick-up.

Number	**NFR-4**
Category	Product requirements
Requirement Name	Reliability requirements
Specification	- The error should be less than 1%. - When logging in, the loading page is displayed without changing the screen until all information is retrieved from database.

Number	**NFR-5**
Category	Product requirements
Requirement Name	Portability requirements
Specification	- Since application was developed by ReactNative, there is no need to consider the device version.

Number	**NFR-6**
Category	Organizational requirements
Requirement Name	Delivery requirements
Specification	Download application with smartphone and tablets: - Android user can download application in Google Play Store. - IOS user can download application in App Store.

Number	**NFR-7**
Category	Organizational requirements
Requirement Name	Implementation requirements
Specification	Develop application with these tools:

	- ReactNative - React - MariaDB - Kotlin for android application - Swift for IOS application

Number	**NFR-8**
Category	Organizational requirements
Requirement Name	Standards requirements
Specification	- The user(customer/restaurant) shall authenticate themselves using their phone number or business number. - System identifies the user with phone number or business number and password.

Number	**NFR-9**
Category	External requirements
Requirement Name	Interoperability requirements
Specification	- Both Customer and store owner(restaurant) can use application with their smartphones or tablets. - In the case of store account, user can login with multiple smartphones or tablets.

Number	**NFR-10**
Category	External requirements
Requirement Name	Ethical requirements
Specification	*Customer version* - The application is for everyone that uses smartphone. - Register with user's phone number and certification with smartphone. *Restaurant version* - This application is only available to restaurant owners who have certified store information.

Number	NFR-11
Category	External requirements
Requirement Name	Legislative requirements - Privacy requirements
Specification	- Request for permission of personal and store information collection is needed when registration. - In my-page, Camera(album) and location permission settings can be changed. - Only customer's nickname information will be shared to others. - User can choose to use the safe number when ordering. - Personal information can't be used for other purposes.

Number	NFR-12
Category	External requirements
Requirement Name	Legislative requirements - Safety requirements
Specification	- With unique ID (phone number, store number) and password, user data will be managed to protect it. - When user inputs the data, application stores it immediately to make preparation for disappearance of data. - Data storage will be done in distributed manner.

6. Scenarios

Sign up	
Actors	User(restaurant and customer)
Initial condition	- Applications should be downloaded on a user's mobile device such as smartphones or tablets. - Mobile devices should be accessible to the Internet network and GPS
Final condition	Users get their account

Exceptional case	User is rejected by the administrator
Process	1. Sign up page is opened 2. User answers form 3. User checks the agreement box 4 User submits form 5. Account is verified in database 6. The user is notified about the successful registration 7. User is redirected to the log-in page

Login	
Actors	User(restaurant and customer)
Initial condition	- The application should be executed. - User should have an app account. - Mobile devices should be accessible to the network.
Final condition	The user logs in.
Exceptional case	- User tries to log in without registration. - ID or password doesn't match.
Process	1. The user runs the app 2. The user enters the ID and password. 3. The user clicks the login button. 4. User is redirected to the main page

Logout	
Actors	User(restaurant and customer)
Initial condition	- The user must be logged in to the application. - Mobile devices should be accessible to the network.
Final condition	The user is logged out of the application
Process	1. The user enters My Page. 2. The user clicks the Logout button.

	3. Click 'Yes' to the message 'Do you want to log out?' 4. User is redirected to the log in page

Withdrawal	
Actors	User(restaurant and customer)
Initial condition	The user must have an application account.
Final condition	The user's account is deleted from the database.
Process	1. User clicks 'Delete Account' button in 'My Page'. 2. A withdrawal confirmation message pops up on the page 3. The user checks the personal information deletion and presses the 'Yes' button. 4. User's account is removed from the database. 5. User is redirected to the sign up page

Store List	
Actors	Customer user
Initial condition	- The user must be logged in to the account. - The user's GPS function should be turned on
Final condition	- If filtered by the user, only the filtered restaurant list should be visible - If a category is selected, only the list of restaurants that correspond to the category must be visible
Exceptional case	If there are no restaurants in this category, no list appears
Process	1. The main page opens. 2. The user inputs the desired keyword using the search function. 3. Users see a random list of stores associated with search keywords.

	4. The user selects a food category or filters it with the desired conditions and then looks the sorted store list. 5. If the user clicks the map button, he or she checks nearby stores around his or her location.

Food order	
Actors	Customer user
Initial condition	The user's mobile device must be connected to the Internet.
Final condition	Food order is received at the store
Exceptional case	- Depending on the situation of the store (too busy or sold out of food), the user's order will be canceled. - If the user does not enter a detailed address - If the user cancel his order for food that arrived - If the container stock in the store is zero
Process	1. The user selects a specific store from the store list. 2. The user sees the information of the selected store and menu information. 3. The user selects the menu to order. 4. The selected menu is placed in the shopping basket. 5. The user clicks the 'Order Now' button. 6. The user chooses between take-out and delivery options. 7. The user inputs detailed information (delivery destination, phone number, and request). 8. The user presses the payment button and selects the payment method. 9.After looking at the total price (Food price + delivery fee + deposit for container), the payment is completed by pressing the confirm button. 10. Points equivalent to 0.5% of the cost of ordering food will be set aside.

	11.The order is sent to the restaurant and owner confirms it (order receipt alarm), the food order receipt alarm goes off to the customer. 12. The location information of the container corresponding to the menu ordered by the user is changed to the user's home. (Exception) If the user cancels the food ordered, restaurant canceled the order and the order details(history) is deleted. The location of the container is changed back to the store. The user's accumulated points are canceled. The container is then returned to the store and the deposit is returned. (Exception) If certain (partial) containers in the store are zero in stock, the menu cannot be ordered. (Exception) If the entire container of the store is zero in stock, the store is left out of the orderable store list.

Pick-up	
Actors	Customer user
Initial condition	- The customer user must have a container that hasn't been returned yet among the foods he ordered - The user's camera function and internet connection for photo upload should be possible
Final condition	The container pick up request is sent to the administrator.
Exceptional case	- When the user posts a picture other than a picture of a bowl - If the user selects a place that does not correspond to the pick-up area as the pick-up location

Process	1. User clicks specific order in 'Order History' 2. User confirms the restaurant information and pick-up location(take-out users enters a location) based on the specific order 3. User Permits to use camera and albums utilization. 4. The user takes a picture of the container and uploads it. 5. The user clicks the Pickup Request button 6. Administrator receives the message "그릇 회수 요청이 업로드 되었습니다." (pick up schedule alarm) 7. Customers receives the message when the next 'pick-up' time (pick up schedule alarm) 8. If customer does not request pick-up service within 3 hours after ordering, customer will get message "그릇 수거 요청을 해주세요." (return request alarm) 3 times for each order. (After 3 hours, 1 day, 3days) 9. The delivery man goes to retrieve the container, checks the type and quantity of the container, and then clicks the confirmation button. 10. The position of the container (state) is changed during recovery and is responsible for the container to him

Map	
Actors	Customer user
Initial condition	User's GPS permission must be allowed
Final condition	If the user selects a category, only nearby restaurants corresponding to the category are displayed on the map
Exceptional case	- If the user does not allow gps - If the user zooms in or out too much, the restaurant list is not displayed.
Process	1. The Map page opens.

	2. The user selects the food type. 3. Only restaurants that sell selected food type are displayed on the map. 4. After the user selects a restaurant, he or she sees the brief information about the restaurant. 5. The user clicks the + button and is redirected to the 'order -menu information' page.

Upload post	
Actors	Restaurant user
Initial condition	Restaurants must have accounts and mobile devices must be connected to the Internet.
Final condition	The post is uploaded.
Exceptional case	If any of the required information is not entered
Process	1. User inputs menu information consists of food name, price, brief introduction, materials, and additional options. 2. The user registers a container for each menu 3. User sets delivery tip by order prices, available delivery area. 4. User enters brief sentences like review events. 5. User puts business times and holiday. 6. User clicks the upload button.

Revise post	
Actors	Restaurant user
Initial condition	There should be previously uploaded posts and Internet connections should be possible.
Final condition	The revised information is reflected in the food order page
Exceptional case	Same as existing information
Process	1. User revise their information or delete specific menu which is not available anymore.

	2. User set 'sold out' mark on specific menu which is out stock. 3. User clicks the revise post button.

Review management	
Actors	Restaurant user
Initial condition	There should be a restaurant review
Final condition	The review reply is uploaded
Exceptional case	If there is a slang in the reply review
Process	1. Users see reviews on their stores. 2. User enters a reply to the review. 3. User clicks the Review Reply button

Container order	
Actors	Restaurant user
Initial condition	There must be a stock of containers to order, and the Internet must be connected.
Final condition	The container order is sent to the reDish.
Exceptional case	- If user enters a quantity larger than the inventory - The user presses the Order button without registering a payment method - If the store requests replacement of the container, or if the container is damaged
Process	1. The user can sees list of container. 2. The user clicks specific container. 3. The user sees the information of the selected container. 4. The user puts the containers he wants to buy in his shopping basket. 5. The user clicks the 'Order Now' button. 6. The user is redirected to order specification section

	7. The user inputs detailed information (delivery date, time, destination, phone number and requests) 8. The user presses the payment button and selects the payment method. 9. After looking at the total price, the payment is completed by pressing the confirm button. 10. Points equivalent to 0.5% of the payment amount will be accumulated. 11. The order is sent to 'reDish' and the container order receipt alarm goes off to the 'reDish'. 12. Reddish confirms it and restaurant will get container order receipt alarm 13. Inventory will be deducted from reDish's inventory by the quantity of containers ordered. 14. When 'reDish' finishes delivering the container to the restaurant, restaurant can get the messages "그릇 배달이 완료되었습니다." (container delivery completion alarm) 15. Restaurant user clicks the 'confirm' buttons for confirming the dish delivery. 16. The location information of the container is changed to the store. 17. Responsibility for the container goes to the store. 18.Upon completion of container delivery, the store is added to the list of available stores (Exception) The store cancels the whole order .Or if the restaurant request an exchange within a day, the container is exchanged for free (Exception) The container can only be exchanged in its entirety, and upon full exchange, the container is recovered as a reDish and delivered back to the store

My page-customer	
Actors	Customer user
Initial condition	Must be logged in

Final condition	Changes should be reflected in the database.
Exceptional case	- If the password does not match the existing password when attempting to change it - Unable to verify and modify information in case of network instability
Process	1. The user views his registered basic information. 2. The user clicks the Order History button. 3. The user clicks a specific order in the order history list 4. The user views the clicked order with the order details. 5. The user writes, revises, and deletes his review of the order. 6. The user changes his information or permission settings. 7. The user clicks 'Visualize Usage' button. 8. The user see his contributions with tree pic and message.

My page-restaurant	
Actors	Restaurant user
Initial condition	Must be logged in
Final condition	Changes should be reflected in the database.
Exceptional case	- When changing passwords, existing passwords do not match - In case of network instability, information cannot be checked and changed
Process	1. The user views his registered basic information. 2. The user changes his information and delivery location. 3. The user clicks the Sales History button. 4. User sees the list of sales history with order specification, date, and delivery destination.

	5. User sees total sales benefits filtering by weekly or monthly.

Administration	
Actors	Administrator
Initial condition	- Must be logged in - Location must be allowed
Final condition	Inventory update, pick-up, container delivery checked
Process	1. 'reDish' receives orders for containers from restaurants through the app. 2. 'reDish' subtracts the stock of the container according to the container order. 2. 'reDish' tracks the location of the container using the delivery address registered by the restaurant and the address of the user who ordered the food 3. 'reDish' checks the current status of the container in real time with the uploaded container pickup request. 4. 'reDish' picks up containers according to the schedule set for each region. 5. The delivery man checks the type and quantity of the container, and then clicks the confirmation button. 6. When the container is received after washing the container, 'reDish' updates the inventory.

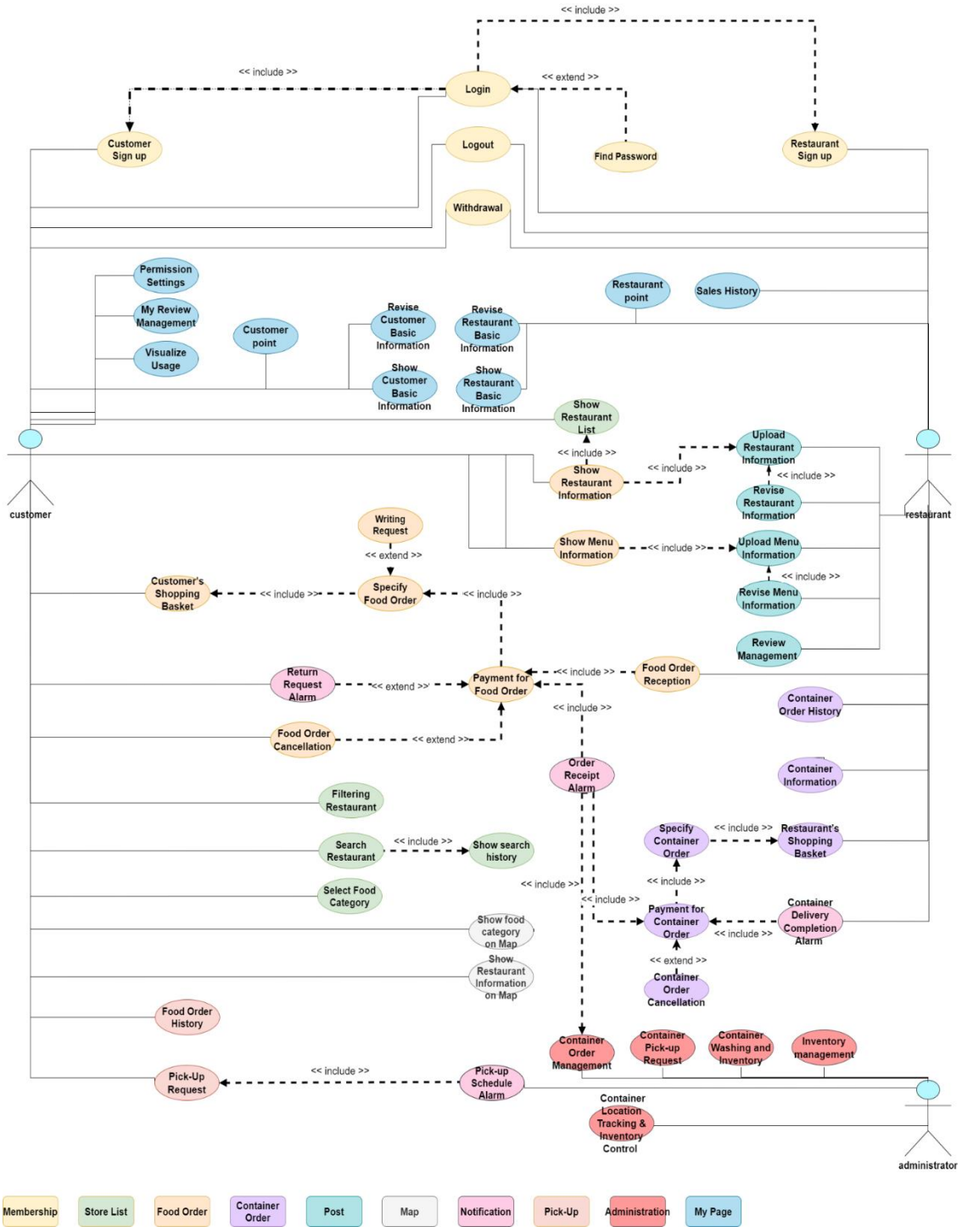
8. Domain Specification

The delivery industry has entered a boom period as social distancing has been strengthened due to COVID-19. According to Statistics Korea, the food delivery market grew about 2.6 times last year. Currently, delivery food orders amount to 2.7 million per day, and according to the Green Alliance, plastic delivery container waste generates at least 8.3 million per day.

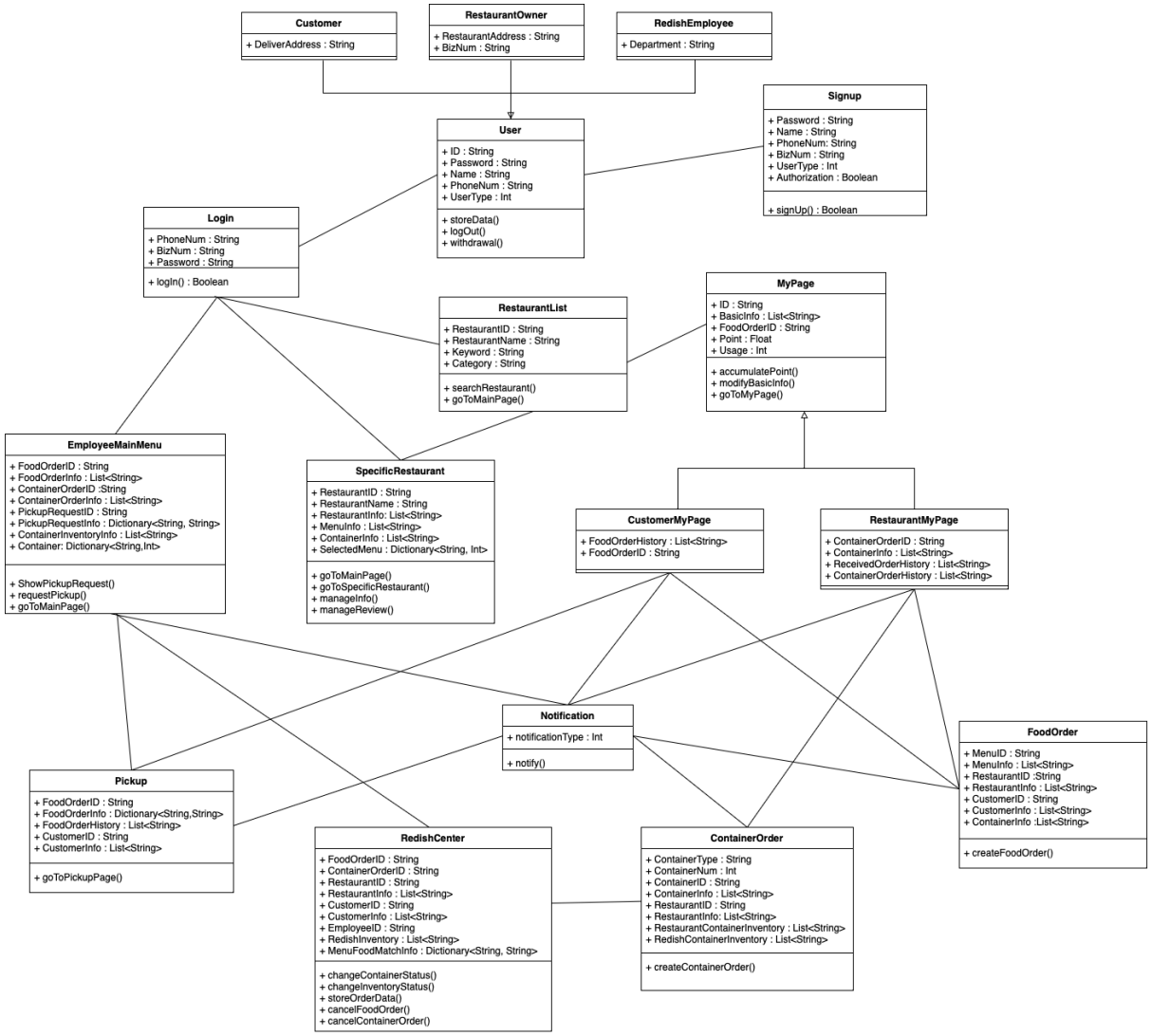
In this situation, the Ministry of Environment said it wants to limit the provision of disposable products when delivering food through the "Act on the Promotion of Resource Saving and Recycling" and sub-laws, which include regulations on disposable products, and the scope will gradually expand. For example, the use of plastic straws in cafes is already regulated.

9. UML Design

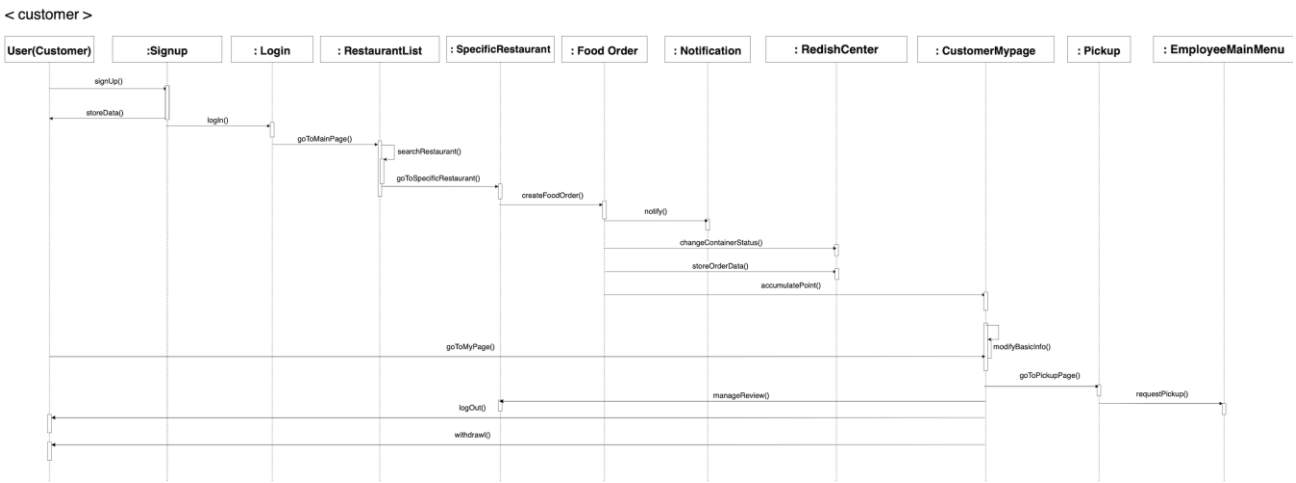
9.1. Use case diagram



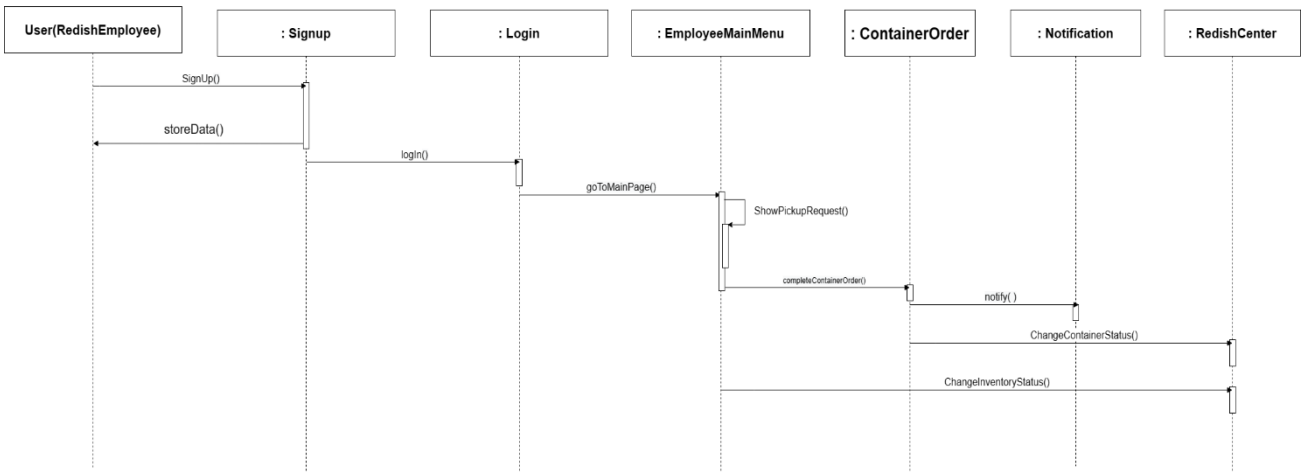
9.2. Class diagram



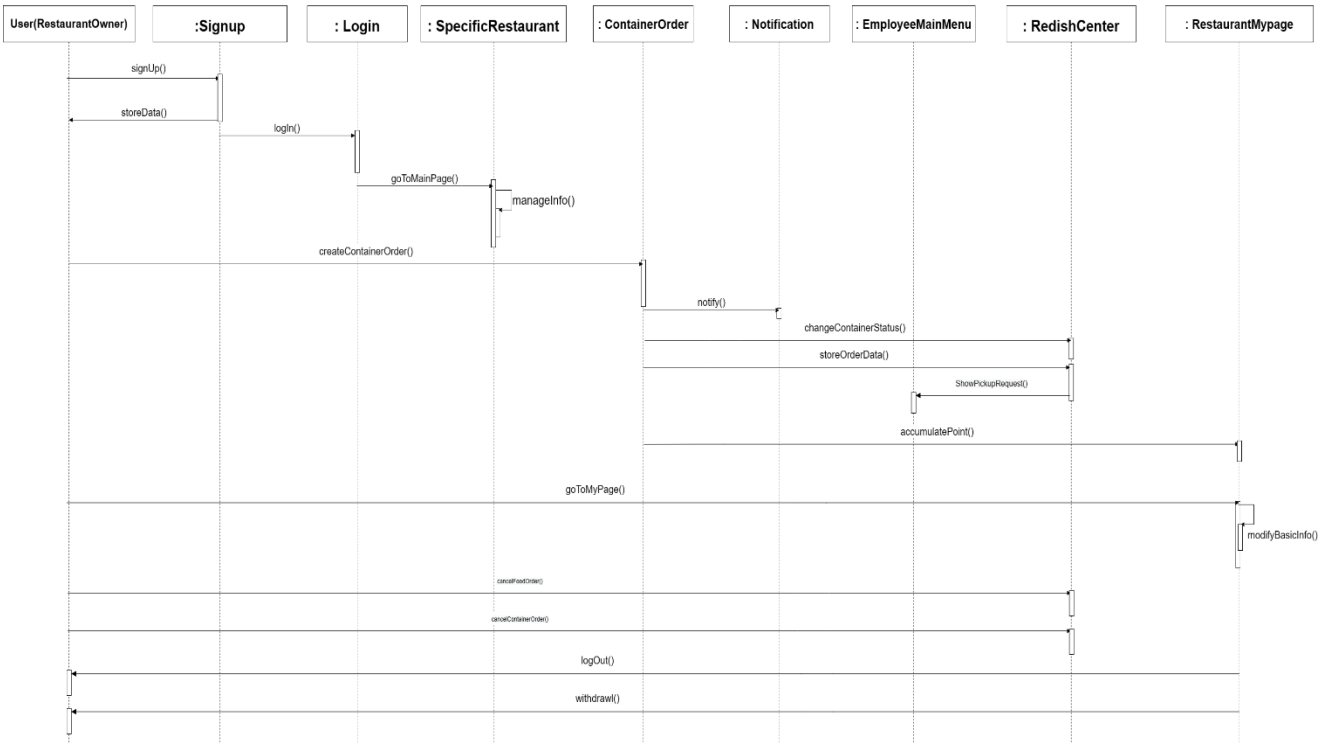
9.3. Sequence diagram



< RedishEmployee>

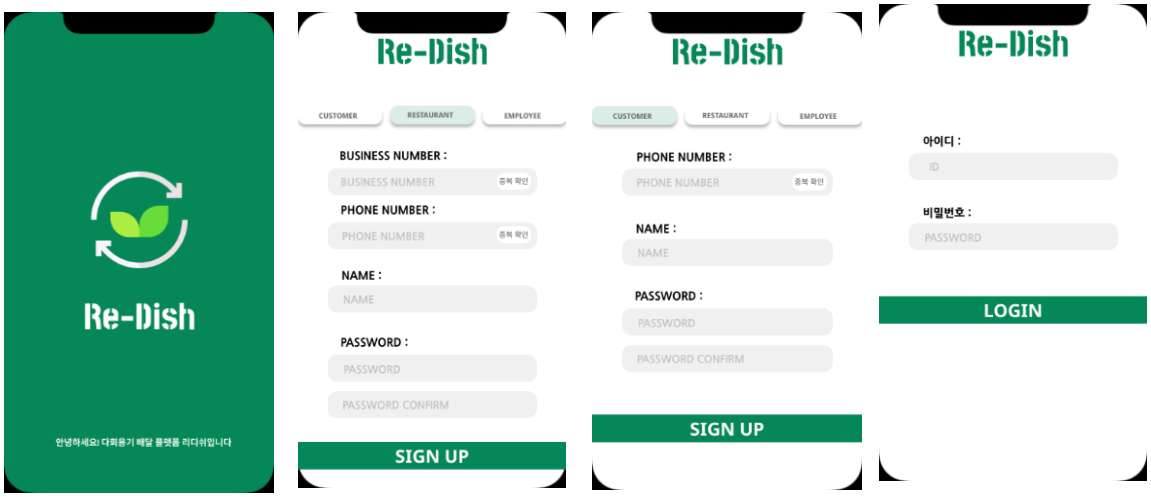


<Restaurant>

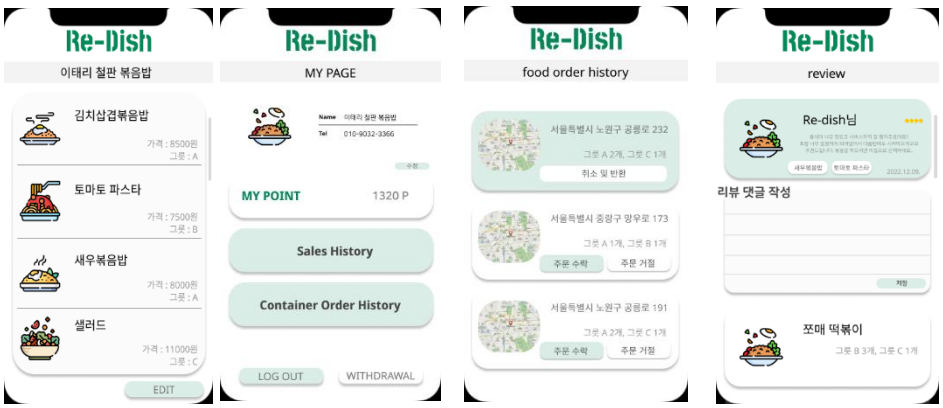


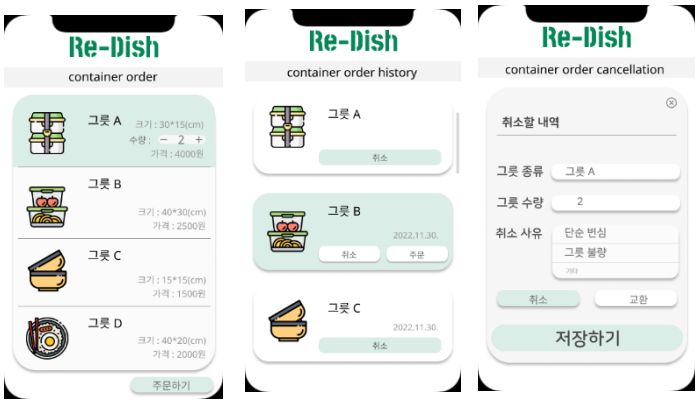
10. User Interface Design

Intro

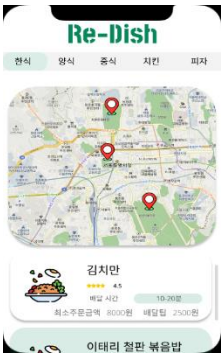


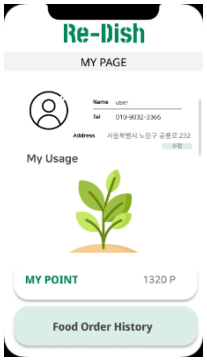
Page for restaurant

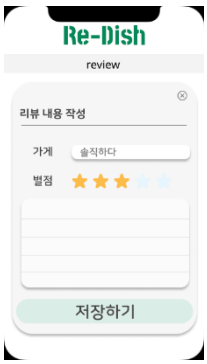




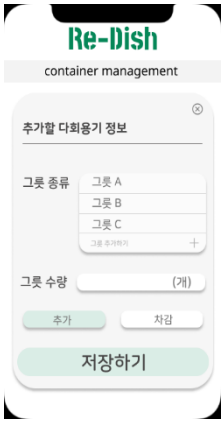
Page for customer







Page for administrator



11. Interface Design

NO.	Function (method)	parameter	return	process
1	signUp	String,Int	Boolean	Get the basic information by String type. If user sign-up succeeds, return True. Or if user sign-up fails, return false.
		PhoneNum,BizNum,Password, Name, UserType	True or False	

NO.	Function (method)	parameter	return	process
2	logIn	String	Boolean	Get the phone number and password by String type and return boolean type.
		PhoneNum, Password	True or False	

NO.	Function (method)	parameter	return	process
3	goToMainPage	Int	List	According to the user type, the kind of list is shown with needed information to the user's screen.
		UserType	RestaurantList or SpecificRestaurant or EmployeeMainMenu	

NO.	Function (method)	parameter	return	process
4	goToSpecificRestaurant()	String	List<String>	Return the information of the restaurant to be shown to the customer in
		RestaurantID	RestaurantList	

				the form of a list according to the restaurant ID

NO.	Function (method)	parameter	return	process
5	searchRestaurant	string	string	Returns a list of restaurants that are related to the keyword searched by the customer
		keyword	List<String>	

NO.	Function (method)	parameter	return	process
6	changeContainerStatus	String	void	Get the ID which changes the position of the container and reflects the information in the Re-dish database.
		FoodOrderID, ContainerOrderID, PickupRequestID	-	

NO.	Function (method)	parameter	return	process
7	changeInventoryStatus	Dictionary	List	The changed container inventory is received as a parameter and reflected in the inventory list.
		Container	ContainerInventoryInfo	

NO.	Function (method)	parameter	return	process
8	createFoodOrder	Dictionary	String	

		MenuID, restaurantID, CustomerID	FoodOrderID	If the user completes a food order with a button, get MenuID, restaurantID, and CustomerID and return FoodOrderID which contains FoodOrder Information.

NO.	Function (method)	parameter	return	process
9	manageInfo	String	void	After identifying the store by the restaurant ID, store the information as entered by the user
		RestaurantID	-	

NO.	Function (method)	parameter	return	process
10	storeData	String,String, String, String, Int,Boolean	void	Information input by the user at the time of signup is received as a parameter and stored in database.
		Password, Name, PhoneNum, BizNum ,UserType, Authorization	-	

NO.	Function (method)	parameter	return	process
11	logOut	Boolean	void	When the user

		True or False	-	clicks the logout button, the boolean value will be changed. then this LogOut method will get the boolean value 0 or 1, and return logout state.

NO.	Function (method)	parameter	return	process
12	withdrawal	Boolean	void	When the user clicks the withdrawal button, the withdrawal variable's False is changed to True for checking whether the withdrawal button click or not. If the withdrawal variable is True, user withdrawal is done.
		True or False	-	

NO.	Function (method)	parameter	return	process
13	modifyBasicInfo	String	void	Identifies the user by ID and stores the changed information in the database as entered by the user
		ID	-	

NO.	Function (method)	parameter	return	process
14	accumulatePoint	String	void	Identifies the user by ID and stores the changed Point in the database
		ID	-	

NO.	Function (method)	parameter	return	process
15	manageReview	String	void	Distinguish orders by food order ID and write, modify, or delete reviews about them
		FoodOrderID	-	

NO.	Function (method)	parameter	return	process
16	goToMyPage	String	List<String>	Identifies the user by user ID and displays my information in a list format
		CutomerID or RestaurantID	BasicInfo	

NO.	Function (method)	parameter	return	process
17	goToPickupPage	String	Dictionary	Get the FoodOrederID when the user clicks such a food order history's pick-up request button. Then such food order information(restaurant name, deliver address, container type & number) will return.
		FoodOrderID	FoodOrder Info	

NO.	Function (method)	parameter	return	process
18	requestPickup	String	String	If the user clicks the 'request' button, get FoodOrderID and CustomerID. And the user have to upload a container image for pick-up requests. When clicking the
		FoodOrderID, CustomerID, Image	PickupRequestID	

				button, Image URL with String is also one of the parameters. With these parameters, PickUpRequestID will return from this function.

NO.	Function (method)	parameter	return	process
19	ShowPickupRequest	-	void	Returns pick-up requests in real-time in list form
		-	ListString	

NO.	Function (method)	parameter	return	process
20	createContainerOrder	String, Int	String	When a restaurant completes a container order with a button, get RestaurantID, ContainerType, and ContainerNum from the system and return ContainerOrderID that represents container order information.
		RestaurantID(String), ContainerType(String), ContainerNum(Int)	ContainerOrderID	

NO.	Function (method)	parameter	return	process
21	notify	String	void	

		UserID	-	In some cases like pickup request, food order, container order, etc. done and so on, it sends the notification with the UserID which is inputted as a parameter. That notification goes to such a user according to UserID.

NO.	Function (method)	parameter	return	process
22	storeOrderData	String	void	After loading the order data related to the order ID, save it in the redish database
		FoodOrderID, ContainerOrderID	-	

NO.	Function (method)	parameter	return	process
23	cancelFoodOrder	String	void	After loading the Food order data related to the order ID, delete it, and save in the redish database
		FoodOrderID	-	

NO.	Function (method)	parameter	return	process

24	cancelContainerOrder	String	void	After loading the container order data related to the order ID, delete it, and save in the redish database
		ContainerOrder ID	-	

12. Detailed Design

Class name	Signup		
Function	signUp	Process	Putting necessary data
Input	PhoneNum, (BizNum), Password, Name, UserType	Output	void
Logic	Get the value for PhoneNum, (BizNum), Password, Name, UserType from the user. var PhoneNum = readLine() var BizNum = readLine() - this is optional var Name = readLine() var Password = readLine() When (Userbtn) { Customer -> UserType == 1, Restaurant -> UserType ==2, Employee -> UserType ==3 } Store the putted data from each user in the user database, USERS respectively using UserType. INSERT INTO USERS(Uid, Name, PhoneNum,BizNum,Password,UserType) VALUES (#{Uid}, #{Name}, #{PhoneNum}, #{BizNum}, #{Password}, #{UserType})		

Class name	Login

Function	logIn	Process	Login as user
Input	PhoneNum or BizNum, Password	Output	void
Logic	User puts the phone as a parameter 'PhoneNum' or 'BizNum' and 'Password' for login. Check if 'PhoneNum' is in the user database and it matches correctly to user password. (SELECT PhoneNum, BizNum, Password FROM USERS) user state will be changed to login state and will move to the main page according to the user type.		

Class name	RestaurantList or SpecificRestaurant or EmployeeMainMenu		
Function	goToMainPage	Process	Move to main page
Input	Button	Output	-
Logic	When the login button is clicked, and user can login successfully, user state moves to main page. SELECT * FROM USERS WHERE (Usertype==2) SELECT * FROM RESTAURANT SELECT * FROM REDISHCENTER		

Class name	SpecificRestaurant		
Function	goToSpecificRestaurant	Process	Move to specific page
Input	RestaurantID	Output	List
Logic	When the specific restaurant is clicked, user state moves to the specific restaurant's page. SELECT RestaurantID == clickedRestaurantID FROM RESTAURANT		

Class name	RestaurantList		
Function	searchRestaurant	Process	Search store and

			show the product restaurant list.
Input	Keyword, Button	Output	List
Logic	When the user searches the restaurant using restaurant name or food name on the search bar, the list of restaurants which include the keyword(restaurant name or food name) will be displayed. keyword = readline() SELECT * FROM USERS WHERE (Usertype == 2) and FOODNAME,STORENAME LIKE '%Keyword%'.		

Class name	RedishCenter		
Function	changeContainerStatus	Process	Store container status in Re-dish DB.
Input	FoodOrderID, ContainerOrderID,PickupRequestID	Output	-
Logic	Getting the current information of containers through orderID, reflect the status to the RedishCenter. UPDATE ContainerInfo SET Current == "FoodOrderID" OR "ContainerOrderID" OR "PickupRequestID".		

Class name	RedishCenter		
Function	changeInventoryStatus	Process	Reflect container add, delete, modification in Re-dish DB.
Input	Container Info	Output	-
Logic	Getting the addition or deletion information and reflecting the current inventory. INSERT CONTAINER(containerID, containerType, containerAmount)		

	VALUES (#{containerID}, #{containerType}, #{containerAmount}) or DELETE CONTAINER(containerID, containerType, containerAmount) VALUES (#{containerID}, #{containerType}, #{containerAmount}) FROM REDISHCENTER

Class name	FoodOrder		
Function	createFoodOrder	Process	Create and complete the food order
Input	MenuID, restaurantID, customerID	Output	void
Logic	Create specific food order and do payment process. User input the address for delivery. If user wants to save address information, the address is added to USER. After completing the food order, the order is saved in RedishCenter and send to Restaurant. INSERT INTO FoodOrder(CustomerID, Name, PhoneNum, Address ,MenuID) VALUES (#{CustomerID }, #{Name}, #{PhoneNum}, #{Address}, #{ MenuID }) INSERT Address(Uid, Address,OrderID) values (#{Uid}, #{Address}, #{OrderID})		

Class name	ContainerOrder		
Function	createContainerOrder	Process	Create and complete the container order
Input	RestaurantID, RestaurantAddress, ContainerType, ContainerNum	Output	void
Logic	Create specific container order and do payment process. User checks the address for delivery. If user wants to save address information, the address is added to USER. After completing the container order, the		

	order is saved in RedishCenter and send to the Redish employee. INSERT INTO ContainerOrder(RestaurantID, ContainerType, ContainerNum, TotalPrice, RestaurantInventory, RestaurantAddress) VALUES (#{RestaurantID }, #{ ContainerType }, #{ ContainerNum }, #{ TotalPrice }, #{ RestaurantInventory }, #{ RestaurantAddress }) INSERT Address(Uid, Address,OrderID) values (#{Uid}, #{Address}, #{OrderID})

Class name	SpecificRestaurant		
Function	manageInfo	Process	Storing the restaurant's information
Input	RestaurantID	Output	void
Logic	Info object is created and the corresponding data is sent to the database server. And the corresponding data is inserted into the RInto table with the below query. SELECT ID FROM RestaurantOwner INSERT INTO Rinfo(foodname, price, description, materials, containertype, deliverytips, briefsentences, businesstimes, holidays, deliveryareas) VALUES (#{foodname}, #{price}, #{description}, #{materials}, #{containertype}, #{deliverytips}, #{briefsentences}, #{businesstimes}, #{holidays}, #{deliveryareas}) RInfo is revised and the corresponding data is sent to the database server. And the corresponding data is inserted into the Post table with the below query. If description is changed by store owner, UPDATE RInfo SET description = #{description}; is executed.		

Class name	User		
Function	storeData	Process	Information input by the user during

			signup is received a s a parameter and stored in DB.
Input	Password, Name, PhoneNum, BizNum, UserType, Authorization	Output	void
Logic	Store the putted data from each user in the each user's database, CUSTOMER and RESTAURANTOWNER, respectively. var PASSWORD = readLine() var NAME= readLine() var PHONENUM = readLine() var BIZNUM= readLine() var USERTYPE= readLine() var AUTHORIZATION= readLine() INSERT INTO CUSTOMER(PASSWORD,NAME,PHONENUM, USERTYPE,AUTHORIZATION) VALUES (#{PASSWORD}, #{NAME}, #{PHONENUM}, #{USERTYPE}, #{AUTHORIZATION}) INSERT INTO RESTAURANTOWNER (PASSWORD, NAME, PHONENUM, BIZNUM, USERTYPE, AUTHORIZATION) VALUES (#{PASSWORD}, #{NAME}, #{PHONENUM}, #{BIZNUM}, #{USERTYPE}, #{AUTHORIZATION})		

Class name	User		
Function	logOut	Process	Logout from user
Input	Button	Output	void
Logic	When the user clicks the Logout button, the user state will be changed to logout state and will move to the login page.		

Class name	User		
Function	withdrawal	Process	Withdraw accounts
Input	Button	Output	void

Logic	When the user clicks the withdrawal button, the user state will be changed to withdrawal state and will move to the login page with “withdrawal success !” message. DELETE Uid From Customer

Class name	MyPage		
Function	modifyBasicInfo	Process	Modify the basicinformation of the user.
Input	ID	Output	void
Logic	BasicInfo is revised and the corresponding data is sent to the database server. And the corresponding data is inserted into the BasicInfo table with the below query. If password is changed by the user, SELECT ID FROM User UPDATE BasicInfo SET password = #{password}; is executed.		

Class name	MyPage		
Function	accumulatePoint	Process	Update user’s point according to the order information.
Input	ID	Output	void
Logic	User’s point is revised and the corresponding data is sent to the database server. And the corresponding data is inserted into the point in MyPage table with the below query. UPDATE Mypage SET Point = #{point};		

Class name	SpecificRestaurant		
Function	manageReview	Process	Distinguish orders by food order ID and write, modify reviews about them
Input	FoodOrderID	Output	void
Logic	SELECT FoodOrderID FROM FoodOrder INSERT INTO FoodOrder(Review) VALUES (#{Review}) Or UPDATE FoodOrder SET Review = #{Review};		

Class name	MyPage		
Function	goToMyPage	Process	Go to mypage.
Input	CustomerID or RestaurantID	Output	List
Logic	When user clicks the myPage button, user moves to the mypage which shows the basic info. SELECT * From Customer or Restaurant WHERE {Uid = CustomerID or RestaurantID}		

Class name	Pickup		
Function	goToPickupPage	Process	Move to pickup request page
Input	Button (FoodOrderHistory)	Output	-
Logic	When the food order history is clicked, the user state moves to the pick-up request page.		

	SELECT RESTAURANTID, CONTAINERNUM, DELIVERADDRESS FROM FOOD ORDER WHERE FoodOrderID = #{FoodOrderID};

Class name	EmployeeMainMenu		
Function	requestPickup	Process	Pick-up request is completed and pick-up request information will deliver to employee
Input	FoodOrderID, CustomerID, Image	Output	PickupRequestID
Logic	When the user(customer) completes the pick-up request by clicking the 'request' button, requestPickup function is called with FoodOrderID, CustomerID, Image that customer is uploaded. With that information about the pick-up request, PickupRequestID will generate which includes pick-up request information like customer address, container number, and etc. if (image!=null) : SELECT FOODORDERID FROM FOODORDER; SELECT ID FROM CUSTOMER; SELECT IMAGE FROM PICKUP;		

Class name	EmployeeMainMenu		
Function	ShowPickupRequest	Process	Employees see the created pick-up request.
Input		Output	List
Logic	When the user(customer) completes the pick-up request by clicking the 'request' button, it is send to Re-dish employees.		

Class name	Notification

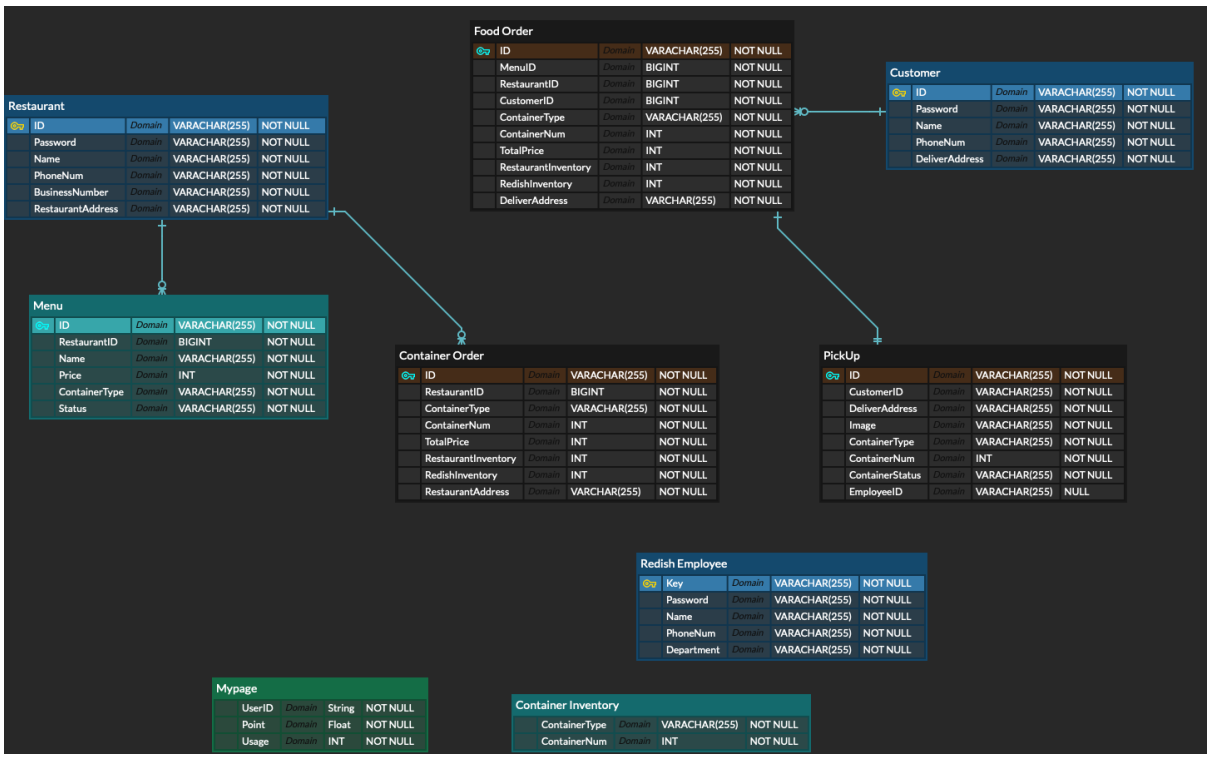
Function	notify	Process	Notifying the user with some events (pickup request, food order, container order, etc.)
Input	UserID	Output	-
Logic	When the event like pickup request, food order, container order, etc., then the notification function is called from that event. By getting UserID, send the notification to that user. SELECT ID FROM CUSTOMER; SELECT ID FROM RESTAURANT; SELECT ID FROM REDISH EMPLOYEE;		

Class name	RedishCenter		
Function	cancelFoodOrder	Process	Restaurant cancel the specific food order with some reasons.
Input	FoodOrderID	Output	-
Logic	When restaurant wants to delete the specific food order, restaurant owner clicks the delete button for specific food order. And customer can get a refund and container status is back to Restaurant. DELETE FoodOrderID FROM FoodOrder UPDATE RestaurantInventory		

Class name	RedishCenter		
Function	createContainerOrder	Process	Restaurant request the cancellation or exchange

			the specific container order with some reasons.
Input	ContainerOrderID	Output	-
Logic	When restaurant wants to change or cancel the specific container order, restaurant owner write the cancel request for specific container order. And customer can get a refund or other containers. This container status is reflected to the RedishCenter.\ UPDATE RestaurantInventory		

13. Database Design



14. Reference

https://www.hani.co.kr/arti/opinion/because/1034204.html

https://noorim.org/section/565?section_idx=426

http://www.me.go.kr/home/web/board/read.do;jsessionid=D9W2sa9cYZPIr41m2PyMG9+v.mehome1?pagerOffset=0&maxPageItems=10&maxIndexPages=10&searchKey=&searchValue=&menuId=286&orgCd=&boardId=1485830&boardMasterId=1&boardCategoryId=39&decorator=

https://www.hani.co.kr/arti/opinion/editorial/965522.html

https://play.google.com/store/apps/details?id=com.woowahan.vn.baemin&hl=ko&gl=US