IFSC 7310: Information Systems Analysis

Final Project Report

Online Food Order Management System

Reetu Vageesh Babu Gujjar
T00682456

1. Introduction

1.1 Objectives

As it is getting easier to access internet anywhere, anytime these days, ecommerce is gaining more importance and people are more inclined towards getting their job done online as it is making the lives easier of a busy person.

My project is related to ecommerce. I have built a system to showcase how an online food order is managed by the system. Online food order management system is about ordering the food online, getting it delivered to the preferred location without having to make an effort to move out to get the food. Especially, people with busy life prefer to get the food delivered to their place instead of taking out time to drive to the restaurant. It is also more convenient way of trying new food any day, anytime. As more people start using mobile applications and browser to order food, continuous updating the available dishes and managing payment becomes necessary.

Main objective of this project is to build a system that shows how an online food order is managed, who all are involved in the process and how data flows within the system.

System is triggered each time a customer or client orders the food from a restaurant. Order can be placed either from website or through mobile application. In this system, while users are picking a dish, they also have an option to check the reviews of other customers on a particular dish. If the customer likes the review, they can continue with the food order, else they can choose a new dish. Once dish is chosen, customer then provides the delivery address, makes payment and places the order. Order confirmation will be sent to the client in return.

Once payment is successful, received order will be sent to the restaurant staff by the system for food preparation and to deliver the food to the customer.

There is also inventory database consisting of all the ingredients which will be updated post every order. List of available food items is also updated after every order to display only the available food items to the customer.

1.2 Existing or New System?

This is an existing system with some modifications. Two modification of my system are as follows:

- There are plenty of existing applications for management of online food order. These existing systems have an option to leave a review and to check the reviews of other customers of the restaurant. However, they do not have an option to check out the reviews of an individual dish. In this system, customer instead of just having the restaurant review, can check out the reviews of a particular dish before ordering it and can leave a review of a particular dish by selecting that dish.

- Customer needs to notify the system when food is delivered. This reduces the risk of delivering the food to the wrong address.

1.3 Methodology

This system is developed using Agile methodology.

- As we know, mobile applications in comparison with desktop applications are more volatile. Based on the current trends, they need to be updated often to meet the customer requirements. Agile methodology suits best for this system as it is more flexible compared to traditional development methodologies.
- I divided my work into sprints and completed those sprints as per the schedule resulting in a prototype after each sprint.
- Also, as agile gives a prototype after each sprint, it is easy to get the system tested at early stages and fix issues sooner than traditional methodologies.
- I experienced project creep in sprint 1 as I was going back and forth between the context and DFD due to the scope change which was avoided in sprint 2.

2. System Requirement Analysis

Requirements of the system are:

- Customer places the order successfully and gets order confirmation.
- System updates the database and displays updated list of available dishes.
- Restaurant delivers food to the customer and notifies the same to the customer.

2.1 DFD Diagrams

Context Diagram

Context diagram gives high level overview of the system boundary and shows how the data flows in the system.

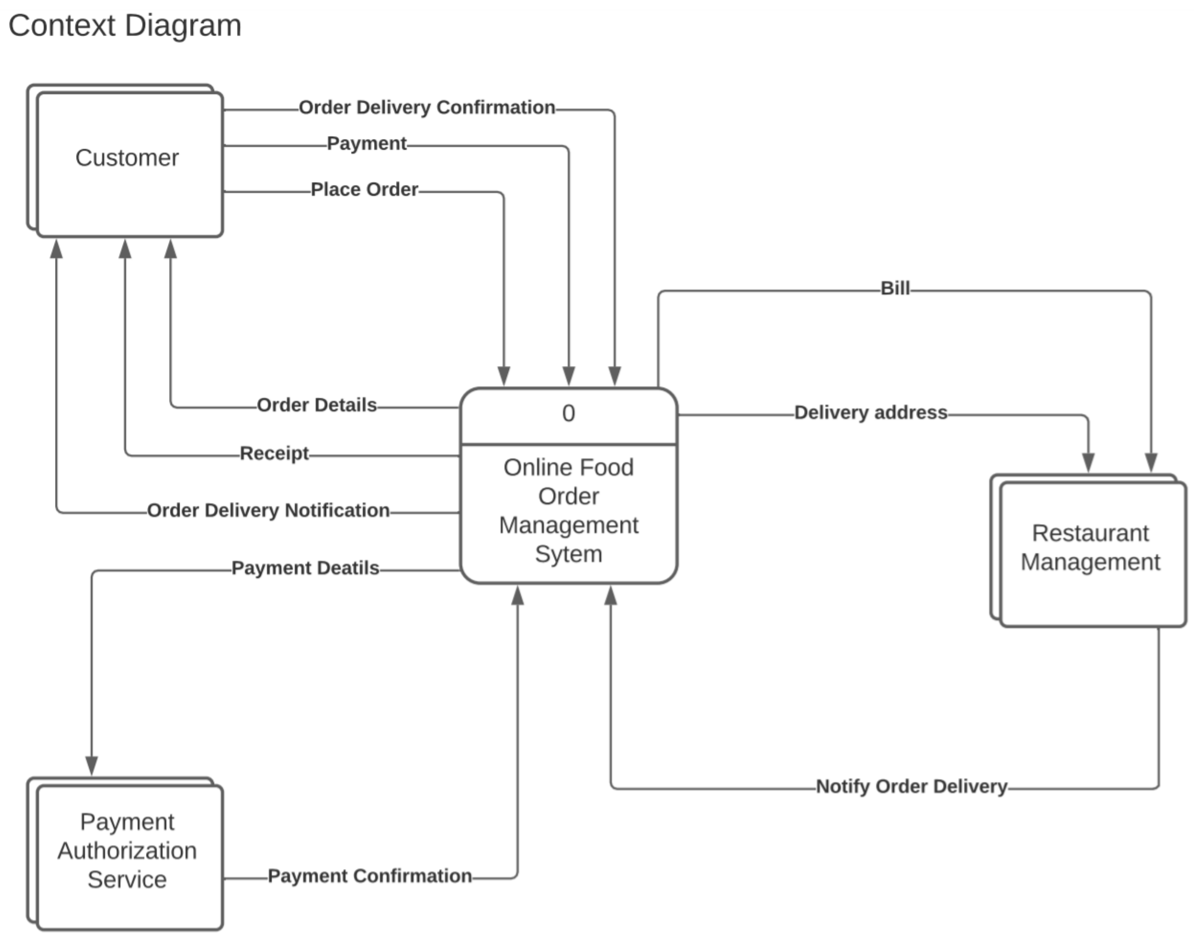


Fig 1: Context Diagram – Online Food Order Management

- Online food order management system is the main process.
  - Takes food order from the customer and sends back the order details.
  - When customer makes payment, system gets it authorized by the payment authorization service and sends back the receipt to customer.
  - System then sends delivery address to the Restaurant management.
  - Restaurant staff delivers the food and notifies the system. System notifies the same to customer and gets back food delivery notification.
  - It also updates the inventory database with the items used for preparing food plus updates the list of available food items.

External Entities involved in this system are: Customer, Restaurant Management, Payment authorization service.

- Customer
  - Opens mobile application or browser of the food order system. Home page displays all the available dishes.

– Customer chooses a dish and if required, checks out the rating/reviews of the dish and if reviews are satisfactory, adds the dish to the cart. Else goes back to home page and chooses different dish.

    – Makes payment and places the order.

– Once food is delivered, customer confirms the food delivery.

    • Payment authorization service/Bank

    – Validates the payment made by the customer.

    • Restaurant management

– Food is delivered by the staff to the provided address and notifies the system.

Diagram 0 DFD

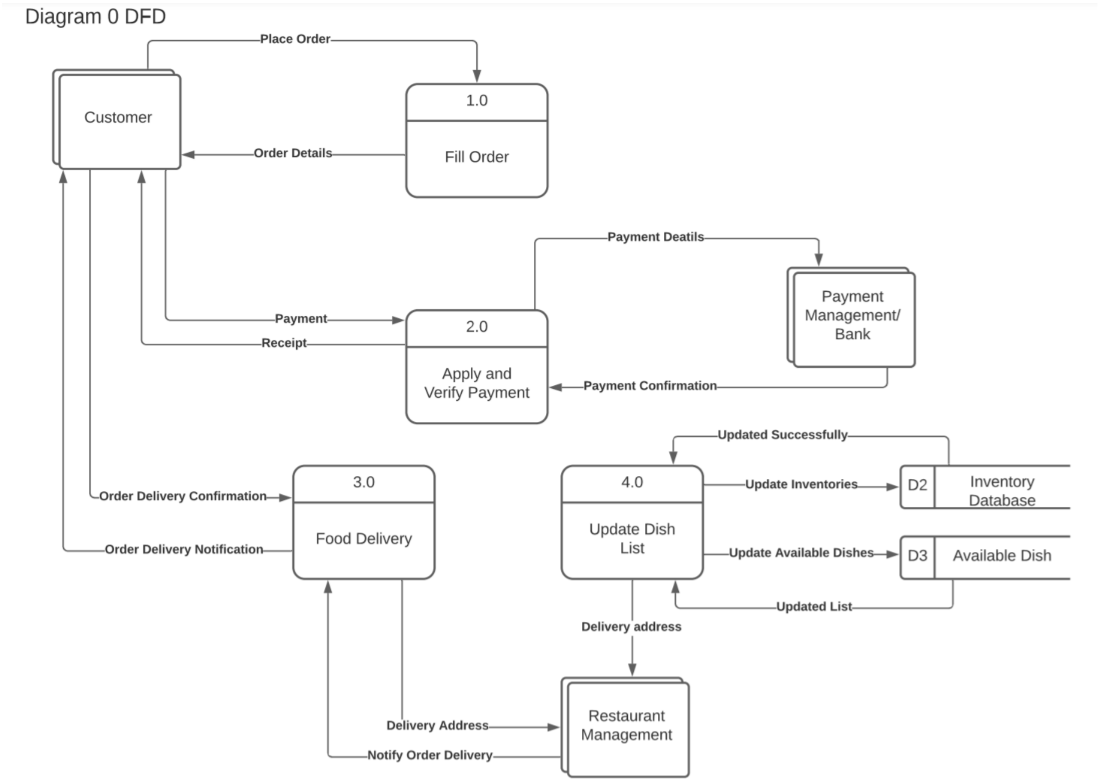


Fig 2: Diagram 0 DFD – Online Food Order Management

In DFD, the process or the system in context diagram is zoomed in to show subprocesses, external entities, data stores and how data flow within the system.

Online food order management system is partitioned into 4 subprocesses namely fill order, apply and verify payment, Food delivery and update dish list.

- Fill Order: Customer places the food order and receives the confirmation.
- Apply & Verify Payment: Once payment is done by customer, payment authorization service authorizes it.
- Food delivery: Delivery address is sent to staff. Once food is delivered, staff notifies the system.
- Update dish list: Once payment is successful, inventory database and available dishes lists are updated.

Datastores involved are inventory database and available dishes.

2.2 Use Case Diagram

Use case model is a type of static modelling and consists of primary actors, supporting actors, use cases, success, failure or alternate scenarios.

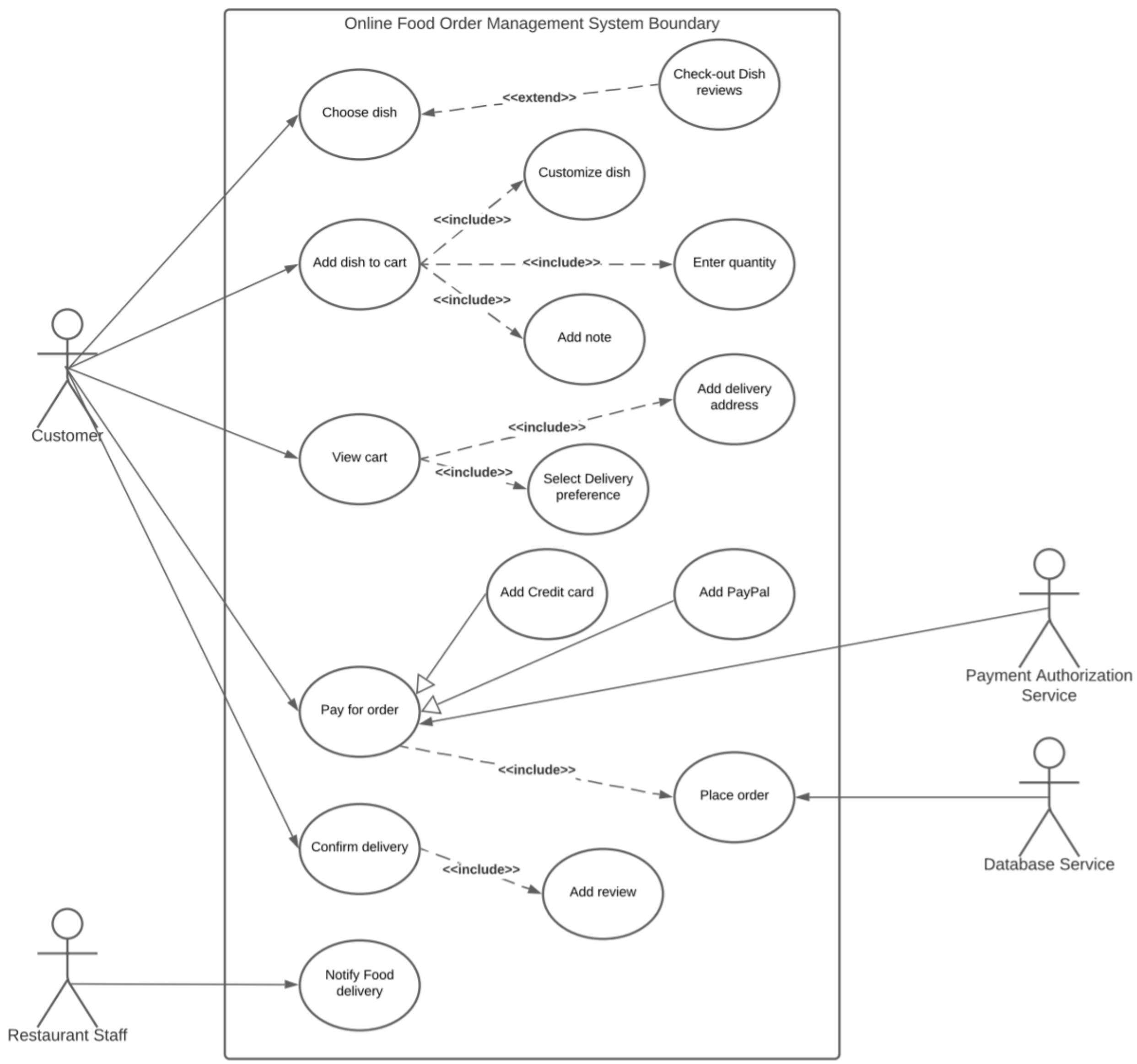
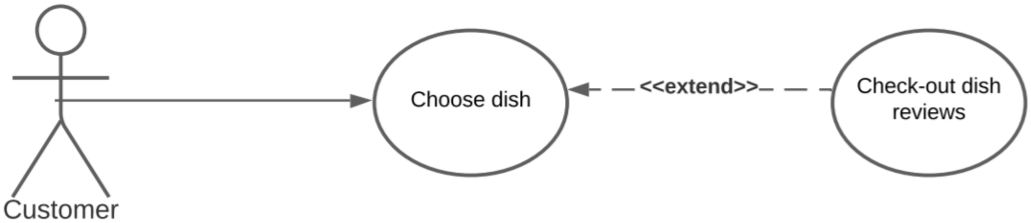


Fig 3: Use Case Model – Online Food Order Management

In global view of the use case diagram, customer and restaurant staff are primary actors, payment authorization service and database service are supporting actors. Choose dish, add dish to cart, view cart, pay for order and confirm delivery are the use cases of a customer. When customer pays for the order, payment authorization service and database service support the system to flow further towards the goal of the system. Notify food delivery is a use case of a restaurant staff who delivers the food to the customer once he receives the address from the system and notifies the system after delivering the food.

Use Case: Choose Dish



USE CASE	Description
Use case name	Choose Dish
Level	User-goal
Primary Actor	Customer
Stakeholders & Interests	When a restaurant's application or website is launched, home page should display all the available dishes. Customer should be able to look for a dish and find the reviews/rating on that dish
Precondition	▪ Customer could launch the application/website successfully. ▪ Customer could login to the system successfully ▪ Home page displays all the available dishes
Postcondition/Success Guarantee	Customer finds the dish and reviews/rating of that dish
Main Success Scenario	1. Choose a dish in the home page of the system. 2. Dish along with its description is displayed
Alternate Scenarios	Check out the rating/reviews of the dish selected. Go back to home page and choose a different dish if dish reviews are not satisfactory.

Table 3: Local View of the Use Case Choose Dish

This is the local view of the use case Choose dish. As customer is a primary actor in this system and without choosing a dish we cannot continue with the other processes, this particular use case becomes a user-goal level.
Stakeholders & Interests: When customer opens the restaurant page either in browser or mobile application, home page should display all the available

items from which customer can choose a dish. Also, when customer chooses a particular dish, dish review should be made visible. There are no other stakeholders involved in this use case.

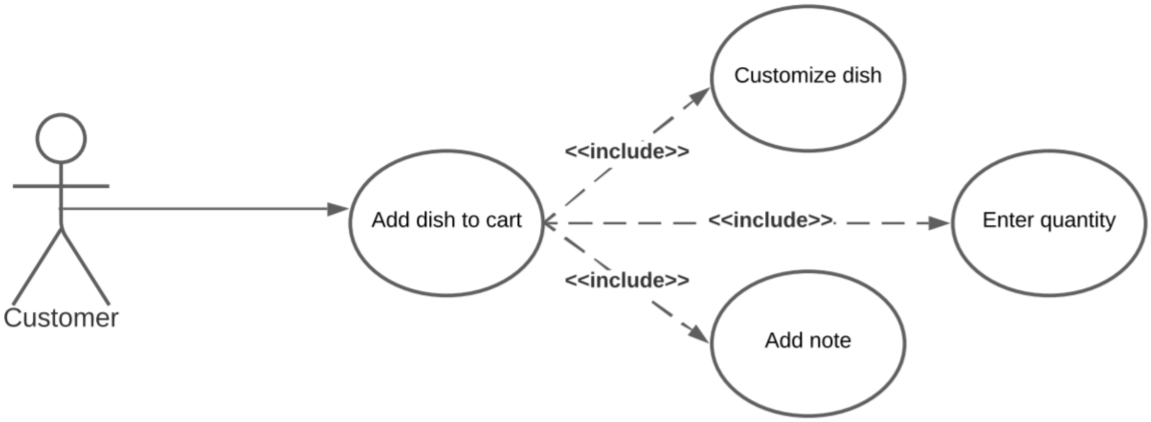
Pre-conditions: Pre-condition for this use case is that Customer should be able to launch the application without any errors. He should be able to login to the system without any errors. Once logged in, only available dish list should be visible to the customer.

Post-Conditions: Post-condition is that when customer chooses a dish, he should be able to view the review, rating of that dish along with the pictures, videos of the dish if present.

Main Success Scenario: Customer can make a choice from the dishes available and is able to view the ratings.

Alternate Scenarios: If ratings/reviews of a dish are not satisfactory, customer can go back to the home page and make a different choice. If customer is not able to login to the system, he can report an error to the restaurant. After making a choice among the dish displayed in the home page, if the system shows dish not available, again customer can report it as an issue to the restaurant.

Use Case: Add Dish to Cart



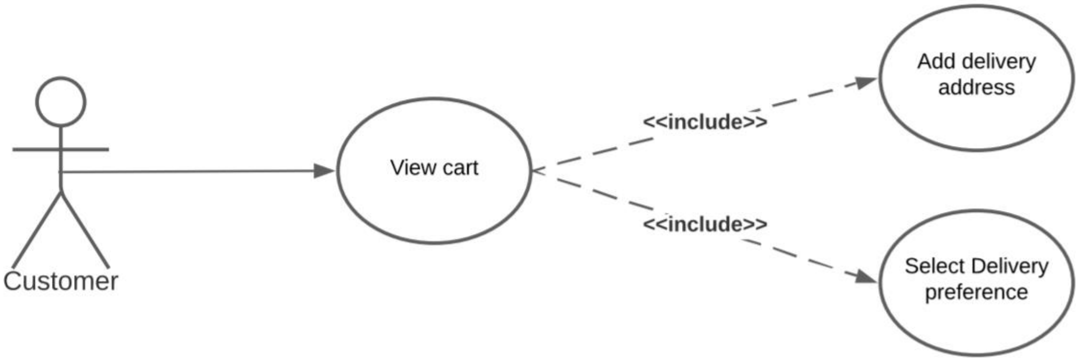
USE CASE	Description
Use case name	Add Dish to Cart
Level	User-goal
Primary Actor	Customer

Stakeholders & Interests	Customer should be able to add the dish to the cart. Customer can also customize the dish (choose spice level) and select quantity. Optionally, customer can add a note like to add or not to add a specific ingredient to the dish.
Precondition	▪ Dish was displayed to the customer. ▪ Quantity by default is displayed 0
Postcondition/Success Guarantee	Customer was able to successfully add the dish to the cart
Main Success Scenario	1. Click on move the dish to the cart. 2. Successfully customizes the dish, selects quantity and adds a note

Table 3: Local View of the Use Case Add Dish to Cart

This is the local view of the use case Add dish to Cart. As customer is a primary actor in this system and without adding a dish we cannot continue with the further processes, this particular use case becomes a user-goal level.
Stakeholders & Interests: After customer has made a choice of the dish after viewing the reviews, he should be able to add the dish to the cart. When user clicks on add dish to the cart, he should be able to customize the dish where he can select spice level and make other adjustments. Customer should also be able to select the quantity and add a note which should be optional. There are no other stakeholders involved in this use case.
Pre-conditions: Only available dishes are shown to the customer and after adding a dish to the cart, quantity by default is displayed 0.
Post-Conditions: Post-condition is that when customer adds a dish to the cart, added dish should be displayed to the customer along with the dish amount.
Main Success Scenario: Customer can add a dish to the cart successfully by selecting the quantity, customizing the dish and adding a note if required. Dish amount will be displayed to the customer in the cart page.
Alternate Scenarios: If customer gets error in any of the steps of adding dish to cart, selecting quantity or customizing the dish, he can report the error to the restaurant and try adding the dish again. Even if customer does not add the note, he should be available to continue with the order.

Use Case: View Cart



USE CASE	Description
Use case name	View Cart
Level	User-goal
Primary Actor	Customer
Stakeholders & Interests	Customer can view the cart, validate all the items added, add delivery address and delivery preference i.e., if the order needs to be placed near the door (contactless delivery) or handed over to the customer. and delivery note if needed
Precondition	Customer has added at least 1 item to the cart
Postcondition/Success Guarantee	Customer was able to successfully view the cart
Main Success Scenario	1. Go to view cart page. 2. Select the delivery preference. 3. Add delivery address
Extensions/Alternate Scenarios	If delivery address is added previously, just select the delivery address.

Table 3: Local View of the Use Case View Cart

This is the local view of the use case View Cart. As customer is a primary actor in this system and without viewing the cart and total, we cannot continue with the further processes, this particular use case becomes a user-goal level.

Stakeholders & Interests: Interests of this particular use case are that customer should be able to view the cart once all the dishes have been added. In view cart page, customer should also be able to add delivery address, select delivery preference like if he wants the delivery staff to hand it over the parcel to the customer directly or place it near the door. Customer should also be able to add delivery note which is optional.

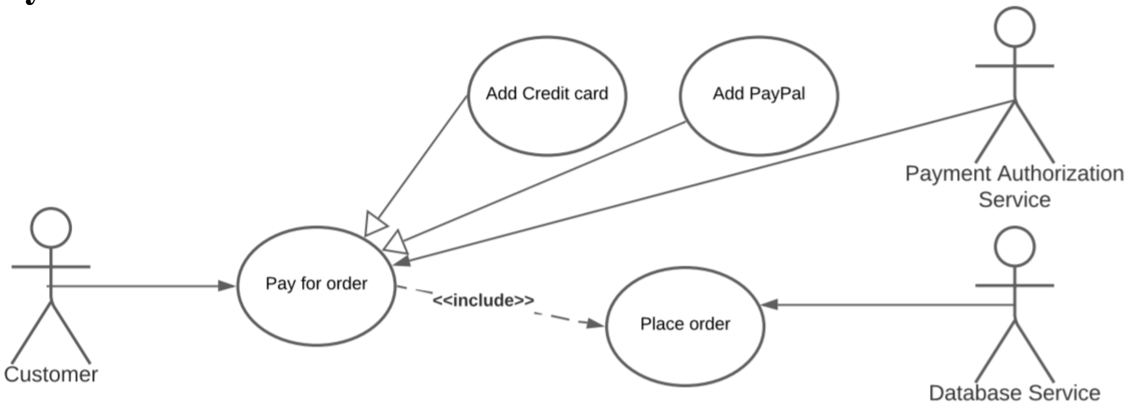
Pre-conditions: Pre-conditions are that customer has added at least one item to the cart before viewing the cart.

Post-Conditions: Post-condition is that when customer views the cart, he can see all the dishes added, their individual price and cart total including delivery fee, service tax.

Main Success Scenario: Customer clicks on view cart, sees the dishes added, quantity selected, individual price of the dishes and cart total. Customer also adds the delivery address and selects the delivery preference and has an option to add delivery note. Added delivery address should be saved to the database to avoid customer re-entering the same address the next time he orders.

Alternate Scenarios: If customer has ordered previously from the same restaurant, he would have already added the delivery address. Customer can select the previously added address and move on next step successfully.

Use Case: Pay for Order



USE CASE	Description
Use case name	Pay for order
Level	User-goal
Primary Actor	Customer

Stakeholders & Interests	Customer can select the payment method i.e., credit card or PayPal. Can add credit card information if credit card is selected or PayPal details if PayPal is selected. Bank should then verify the payment. Once payment is successful, customer can place the order. After every order, inventory database and the list of available dishes should be updated.
Precondition	▪ Customer has added at least 1 item to the cart. ▪ Customer has the delivery address in place
Postcondition/Success Guarantee	Customer could successfully place the order
Main Success Scenario	1. Customer selects either credit card or PayPal. 2. Add Credit card details/PayPal details. 3. Successfully finish the payment with authorization from the payment service 4. Successfully place the order 5. Database service successfully updates the inventory database and list of available dishes
Alternate Scenarios	▪ If authorization fails, select different payment method. ▪ If Database service does not update the list of available dishes, when a customer selects that dish, appropriate message needs to be displayed.

Table 3: Local View of the Use Case Pay for Order

This is the local view of the use case Pay for Order. As customer is a primary actor in this system and if payment is not made, system fails to continue with the further processes, this particular use case becomes a user-goal level.

Stakeholders & Interests: Interests of this particular use case are that customer should be able to add a payment method successfully, either credit card or PayPal. Once payment method is selected, customer can make payment which will be authorized by the payment authorization service and once it is authorized, customer must receive the receipt in return.

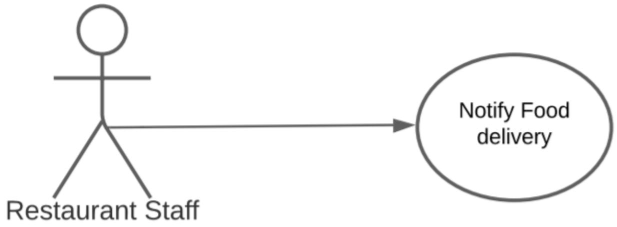
Pre-conditions: Pre-conditions are that customer has added at least one item to the cart and has added delivery address and delivery preference successfully.

Post-Conditions: Post-condition is that customer should be able to make the payment successfully and once payment is successful, order must be placed.

Main Success Scenario: In the cart page, customer clicks on pay for order, selects either PayPal or credit card, add the credit card/PayPal details and makes the payment. Payment authorization service validates the payment. If

payment is successful, system sends the delivery address to the staff and updates the inventory database with the ingredients used and displays the updated list of dishes that are available.
Alternate Scenarios: When customer makes payment, if authorization of payment fails, customer selects a different payment method. If database service fails to update the dish list, when customer selects the dish that is no longer available in the restaurant, appropriate message is to be displayed.

Use Case: Notify Food Delivery

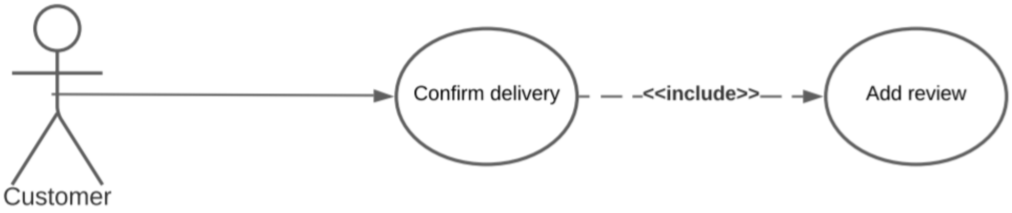


USE CASE	Description
Use case name	Notify Food Delivery
Level	User-goal
Primary Actor	Restaurant staff
Stakeholders & Interests	Restaurant will process the order and when food is prepared, delivery staff picks it up and delivers it to the customer and notifies the system about the delivery.
Precondition	Delivery address is provided.
Postcondition/Success Guarantee	Restaurant staff delivers the food to the customer.
Main Success Scenario	Once food is delivered, notify the system about food delivery
Alternate Scenarios	If delivery address is not found, contact the customer to get the correct address.

Table 3: Local View of the Use Case Notify Delivery

This is the local view of the use case Notify Food Delivery. Restaurant staff is a primary actor for this use case. Since, if payment is not made, system fails to reach the goal, this particular use case becomes a user-goal level.
Stakeholders & Interests: Interests of this particular use case are that staff when receives the delivery address and the order details from the system, order is processed. Once food is prepared, restaurant staff picks it up and delivers it to the customer and notifies the same to the system.
Pre-conditions: Pre-condition is that system has sent the delivery address to the staff.
Post-Conditions: Post-condition is that staff delivers the food to the customer successfully as per the preference selected.
Main Success Scenario: Staff delivers the food to the right address as per the instructions provided by the customer. Once food is delivered, notifies the system about the same.
Alternate Scenarios: If delivery address is not found, system contacts the customer to get the confirmation of the address. System sends this updated address to the restaurant staff. Staff then delivers the food to the updated address and notifies the same to the system.

Use Case: Confirm Delivery



USE CASE	Description
Use case name	Confirm Delivery
Level	User-goal
Primary Actor	Customer
Stakeholders & Interests	Once the order has been delivered, customer can confirm the delivery. Customer can leave a review for the restaurant and rate the individual dish and optionally can add photos of the dishes.

Precondition	Order has been delivered to the customer.
Postcondition/Success Guarantee	Customer confirms the delivery.
Main Success Scenario	Customer confirms the delivery
Alternate Scenario	If food is delivered to the wrong address, customer complains the restaurant and gets refund.

Table 3: Local View of the Use Case Confirm Delivery

This is the local view of the use case Confirm Delivery. As customer is a primary actor in this system and if he does not confirm the delivery, system fails to achieve the goal. Hence this particular use case becomes a user-goal level.

Stakeholders & Interests: Interests of this particular use case are that customer should be able to confirm the delivery once food is delivered. Customer can also provide the review of the restaurant, able to add images, videos. Along with this, customer should also be able to add the review of the individual dish, image and videos of the dish. Also, customer should be able to comment on other customer's reviews (both restaurant and dish review).

Pre-conditions: Pre-conditions are that customer has received the food according to the preference selected.

Post-Conditions: Post-condition is that customer confirms the delivery and provides reviews, add images, videos, comments on the restaurant review. Also, provides feedback add images, videos, comments on the dishes.

Main Success Scenario: Once food is delivered to the customer, he confirms the delivery to the system by sending a notification. System then sends the customer closing message that contains a link requesting for restaurant and dish review. Customer might or might not provide the feedback.

Alternate Scenarios: If customer is unable to add reviews, he can log an error with the restaurant. If food is delivered to the wrong address and customer does not receive the food, customer can complain the same to the restaurant and request for refund of the amount paid.

2.3 System Sequence Diagram

Interaction Diagram is a type of dynamic object modeling as it shows the interaction between the users and various entities in the system. Interaction diagram has system sequence diagram and sequence diagram.

System sequence diagram gives high level overview of the sequence diagrams.

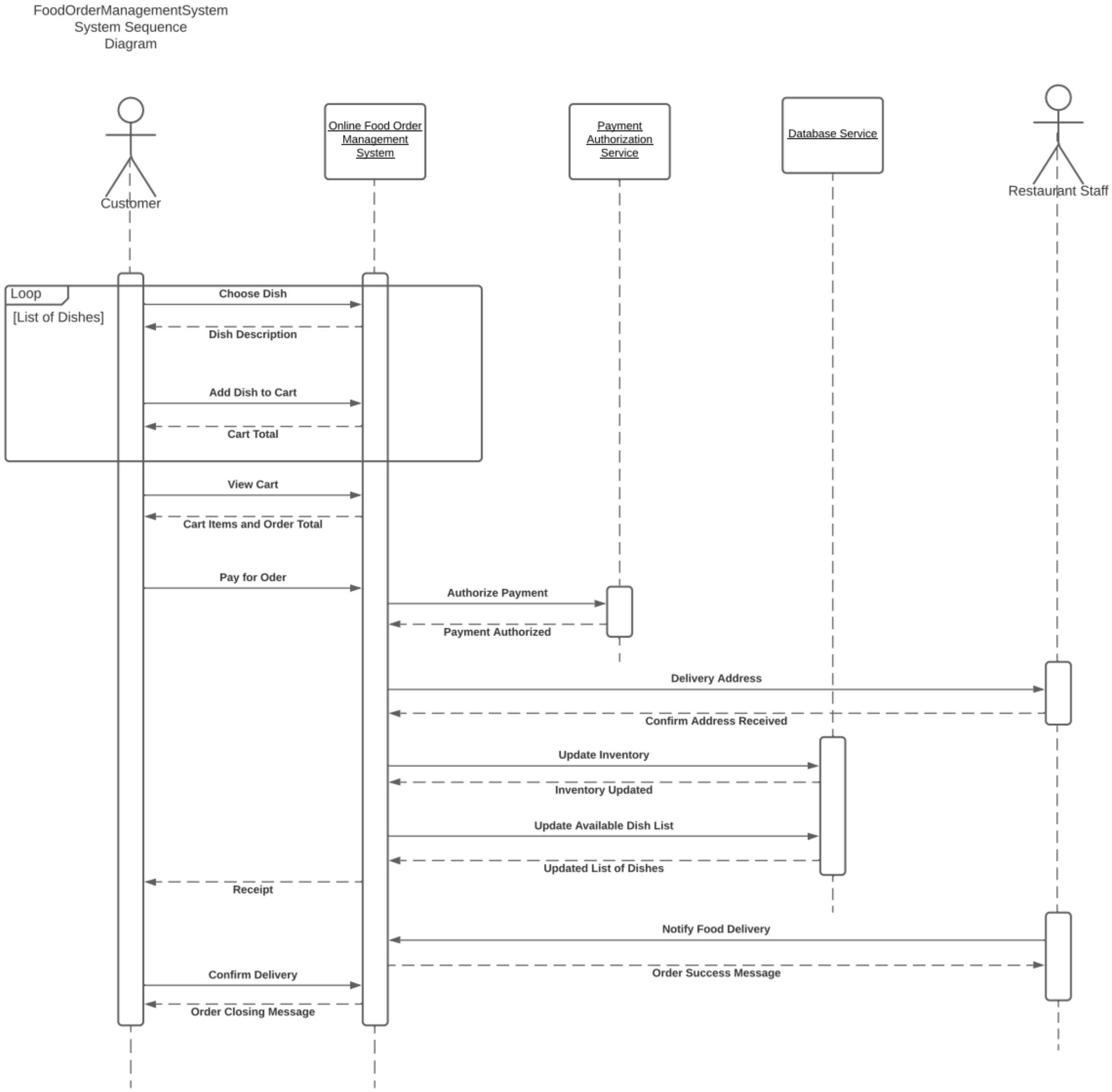
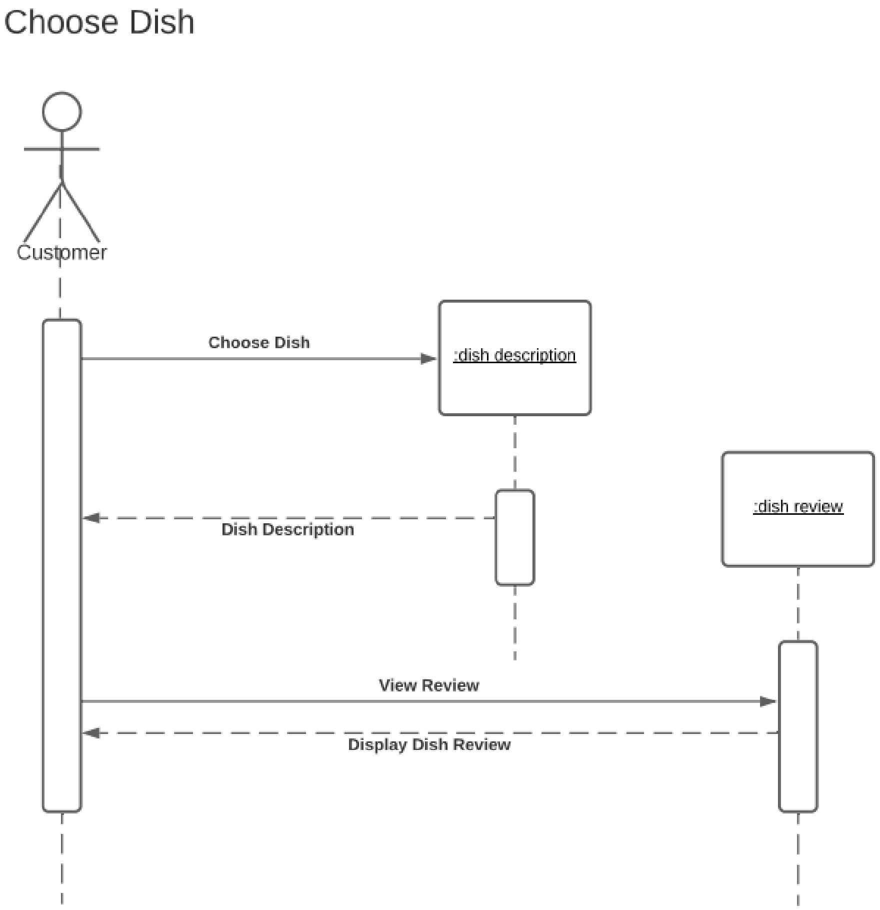


Fig 4: System Sequence Diagram

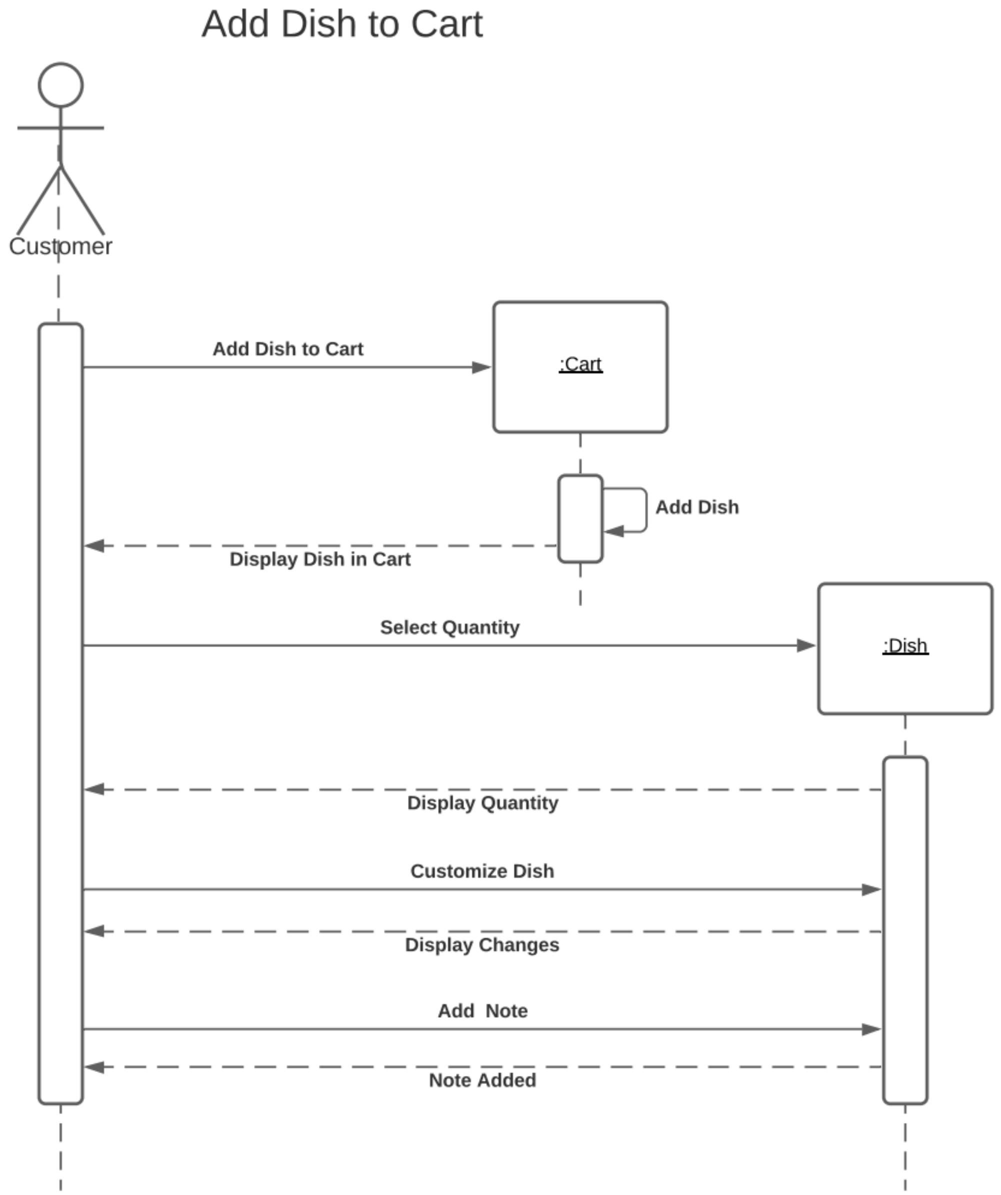
In the sequence diagram of online food order management, customer chooses a dish. Online food order management system responses by displaying the dish description. Customer then sends a message to the system to add dish to the cart. System responds back by displaying the cart total. These two interactions can take place any number of times whose output is a list of dishes that customer wants to order. Customer then views cart where he can see cart items and cart total. Customer then pays for order which is authorized by payment service. System then sends delivery address to staff and updates the database and finally sends back receipt to customer. Restaurant staff notifies food delivery to the system. Customer confirms the food delivery to the system. System sends the order closing message to customer.

Sequence Diagram: Choose Dish



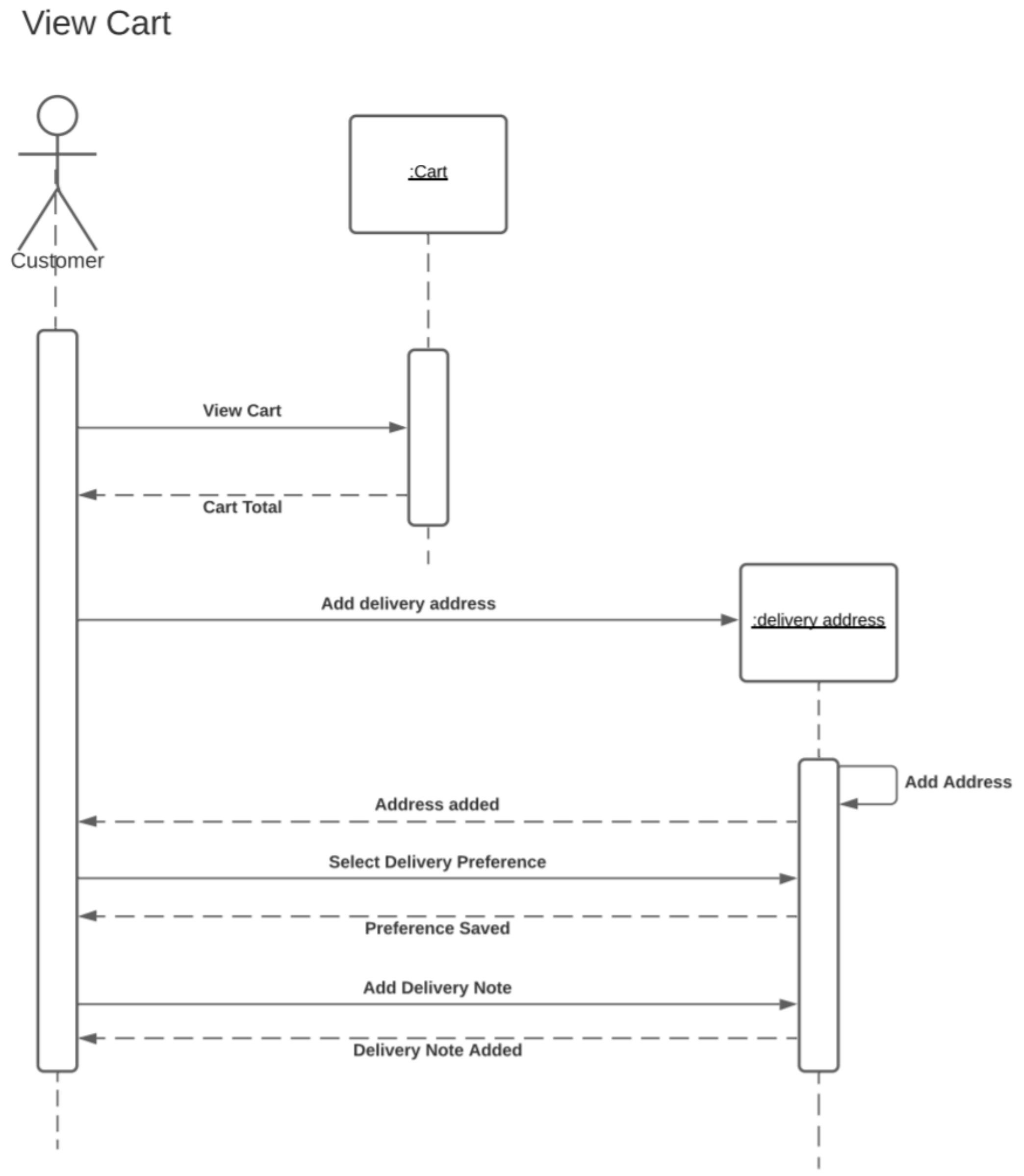
Sequence diagram consists of actors and objects, shows actions and responses between them. New objects are created horizontally to the right of the existing system and new actions or requests are added vertically down. Sequence diagram of choose dish shows the interactions between customer and system while choosing dish. Customer chooses a dish from the system which creates an instance dish description. System responds to customer with the dish description. Customer then views the review of that dish which creates an instance dish review. As the execution bar of the dish review shows, it responds back to the customer by displaying the dish review.

Sequence Diagram: Add Dish to Cart



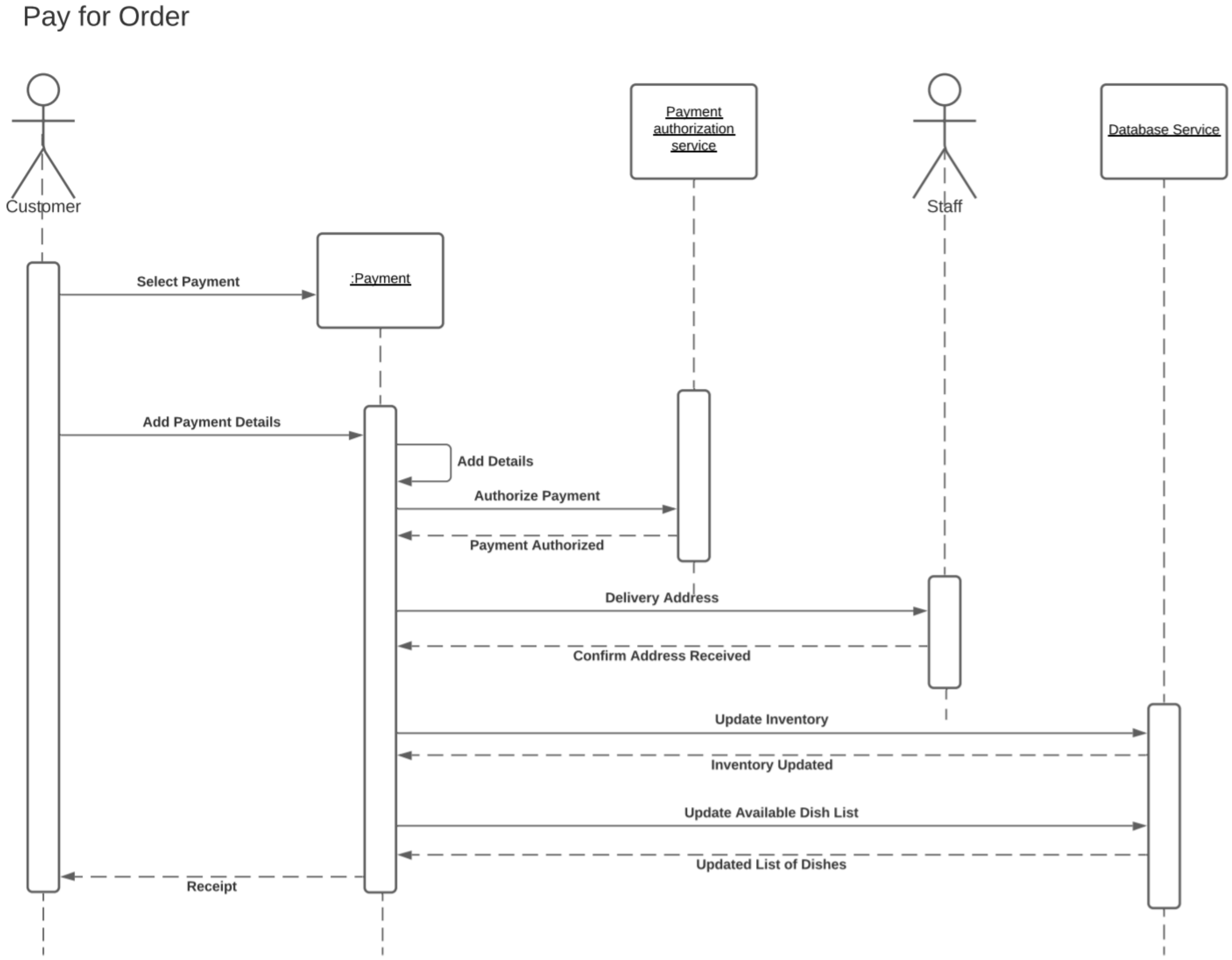
Sequence diagram of add dish to cart consists of customer and shows the interactions between customer and system during adding dish to the cart. Customer first adds the dish to the cart which creates an instance of the cart. Execution bar of the cart shows a self-message, add dish to the cart. Cart also responds to the customer by adding dish to the cart. Customer then selects quantity of the dish which creates an instance dish. Dish displays the quantity back to the customer. Customer then customizes the dish and adds a note which are done successfully as the dish responds.

Sequence Diagram: View Cart



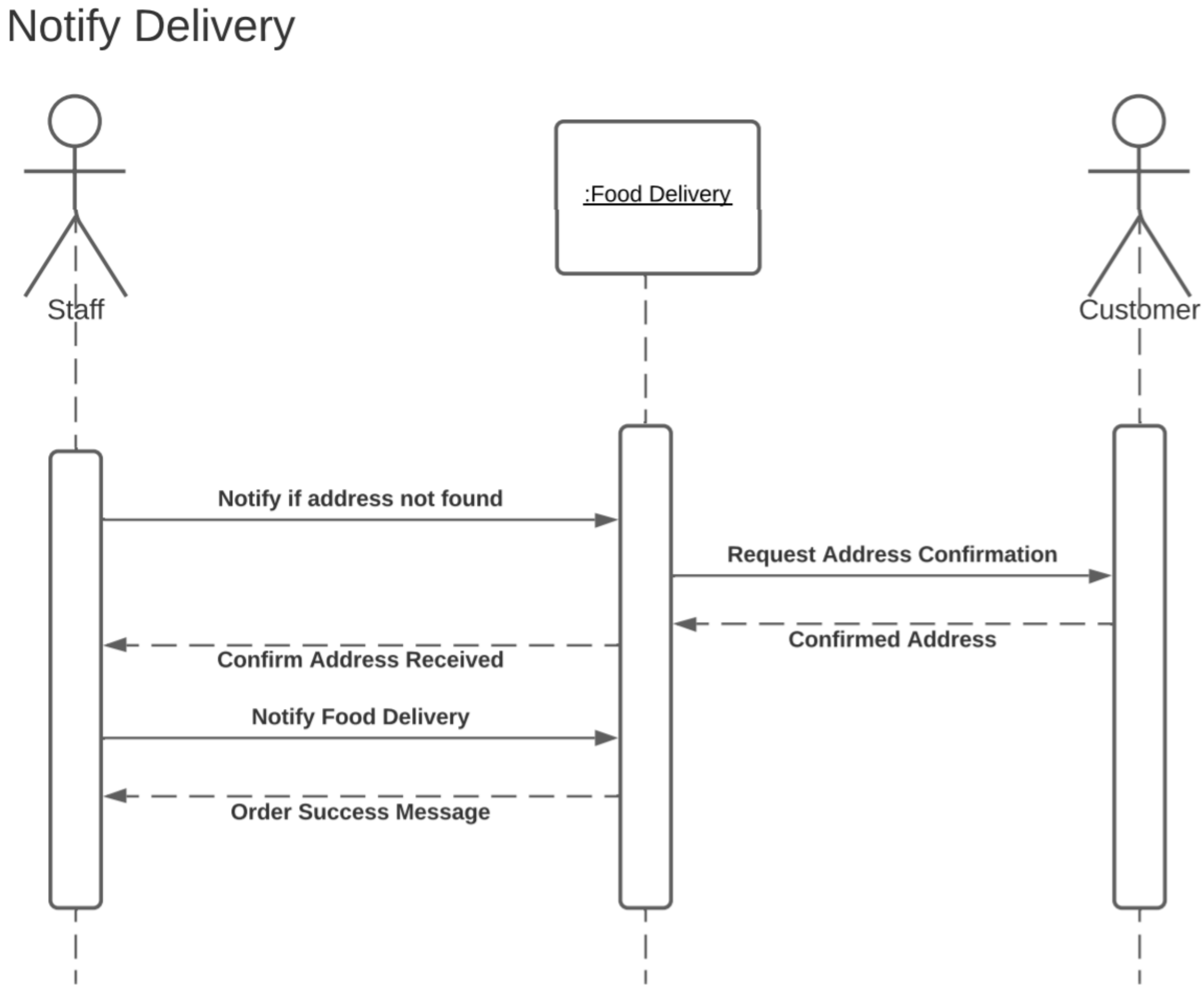
Sequence diagram of view cart consists of customer and shows the interactions between customer and system during viewing the cart. It starts with customer viewing the cart. Cart shows the cart total and all the dishes added to the cart along with individual prices. Customer then adds the delivery address which creates an instance of delivery address. Delivery address instance has a self-message which adds address to itself and saves the address to the database for future reference. System then responds with a successful message to customer. Customer then selects the delivery preference for which he receives a confirmation. Finally, customer adds delivery note. System responds back by sending a confirmation message.

Sequence Diagram: Pay for Order



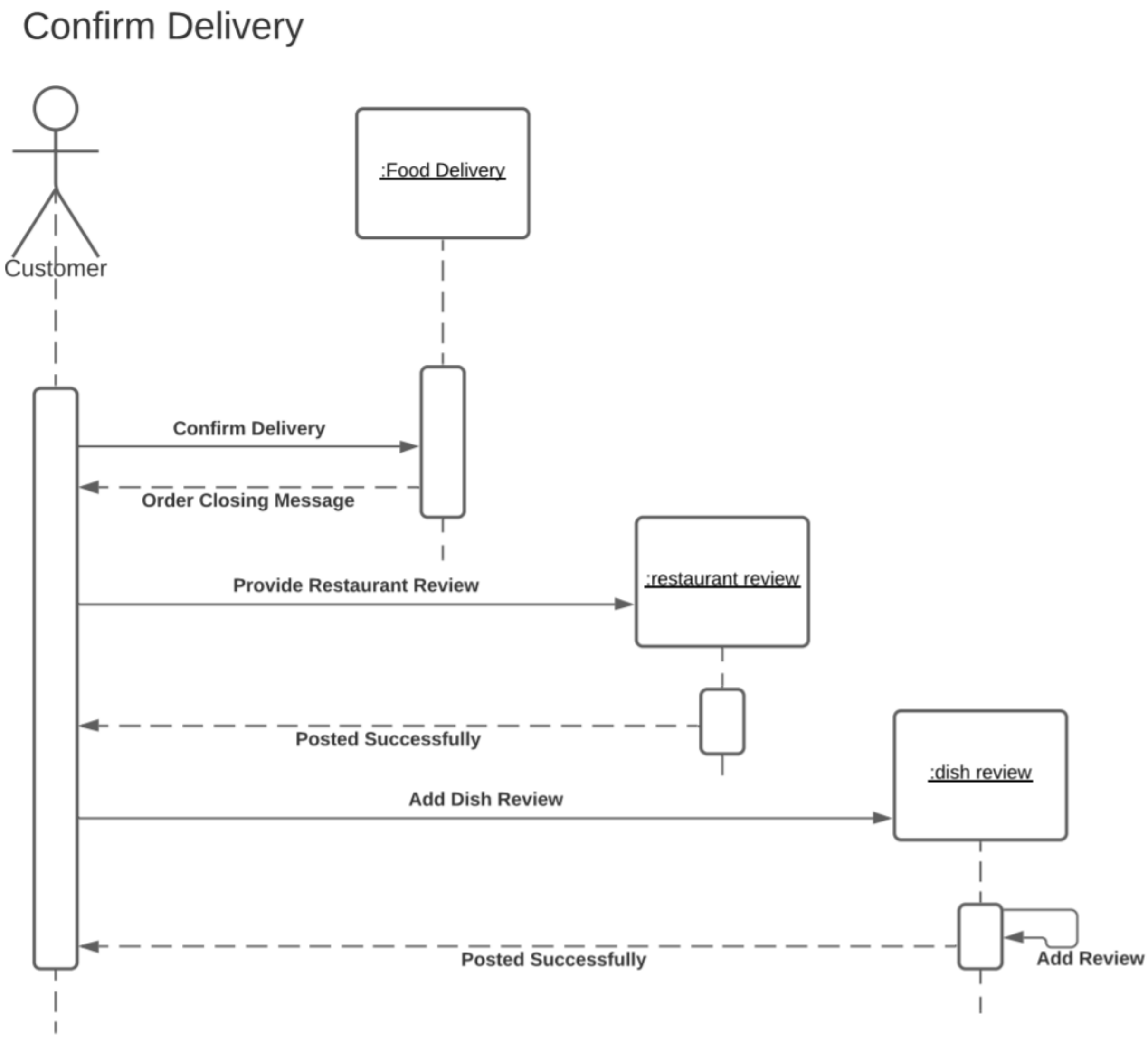
Sequence diagram of pay for order consists of customer, staff and shows the interactions between customer, staff and system while customer makes payment. Customer selects a payment method and adds payment details to the system. Payment authorization service authorizes the payment and sends back confirmation to the system. System then sends the delivery address to the staff for delivering the food who confirms the received address. System also updates the inventory database and the available dish list once payment is successful. System responds back by updating the inventory database and displaying the updated dish list. System then sends the customer receipt as confirmation of the order.

Sequence Diagram: Notify Delivery



Sequence diagram of notify delivery consists of staff, customer and shows the interactions during staff notifying the delivery to the system. Staff once receives the address, processes the order. Once food is prepared, staff delivers the food to the received address. If staff does not find the address, notifies the food delivery object which then requests the customer to confirm the added address. Customer responds by confirming the address. Delivery system then sends the confirmed delivery address to staff. Staff then delivers the food to the right address as per the preference set by customer and informs the system about the delivery. Delivery system responds back to the staff by sending the order success message.

Sequence Diagram: Confirm Delivery



Sequence diagram of confirm delivery consists of customer and shows the interactions between customer and system once customer receives the food delivery. Once customer receives the food, confirms the food delivery to the food delivery system. Food delivery system responds back to the customer by sending the order closing message which also requests for customer review. Customer then provides the restaurant feedback by adding comments, images, videos which will be posted successfully. Customer then provides feedback of the dish as well by adding separate comments, image/videos of the dish which will be posted on the restaurant page.

2.4 Class Diagram

Class diagram is a type of static object modelling. It shows all the classes, objects, their attributes and methods and the relationship between objects.

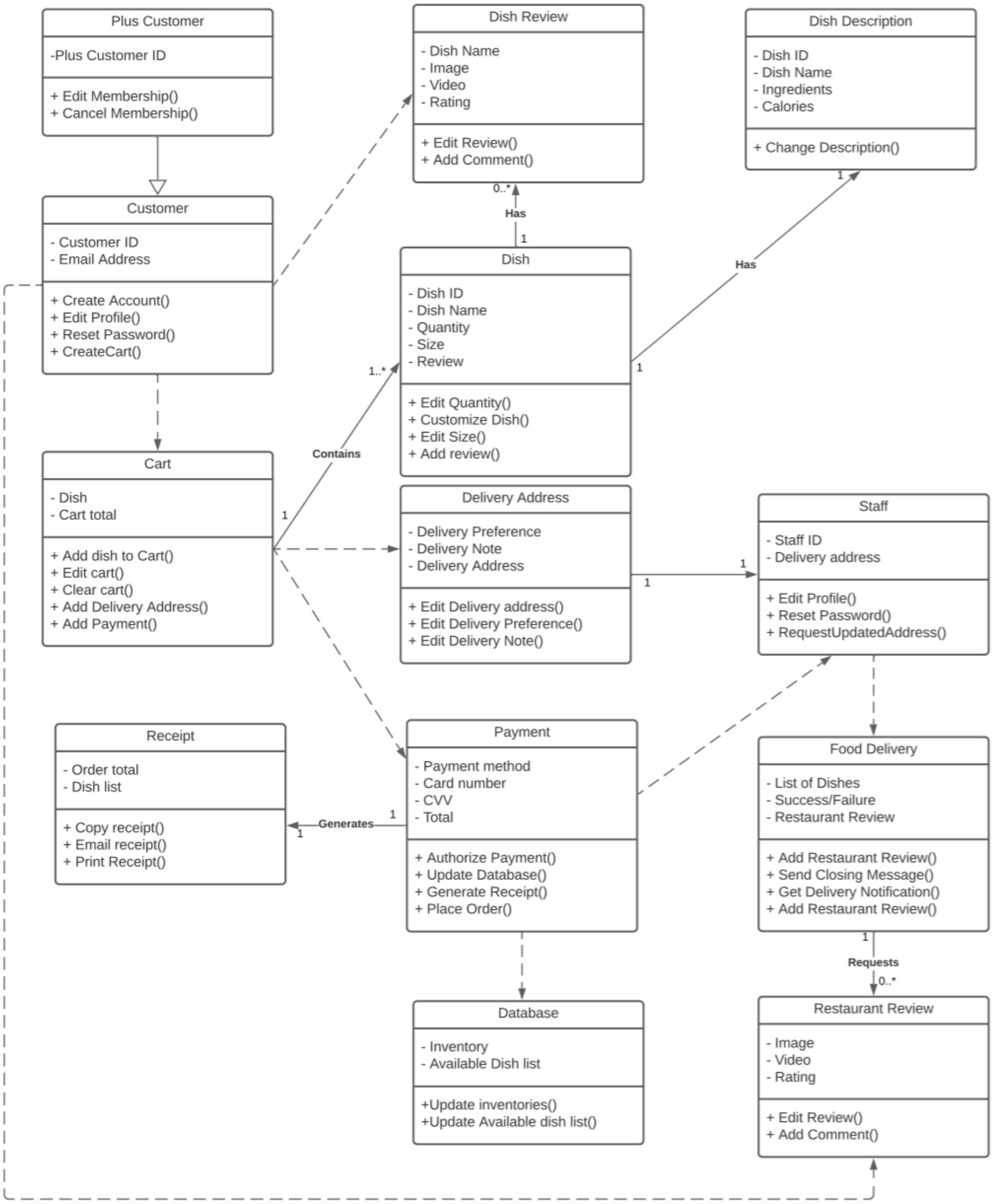


Fig 6: Class Diagram

We have three different relationships as shown below.

Inheritance:

- Customer to Plus Customer

Association:

- Cart to Dish
- Dish to Dish Review
- Dish to Dish Description
- Staff to Delivery Address
- Payment to Receipt
- Food Delivery to Restaurant review

Dependency:

- Customer to Cart
- Customer to Dish Review
- Customer to Restaurant review
- Cart to Delivery Address
- Cart to Payment
- Payment to Staff
- Staff to Food Delivery
- Payment to Database

Customer class creates a cart by adding dishes. Hence there is dependency between customer and cart. Cart is associated with dish object as it contains dish. Association between cart and dish is one to many as a cart can contain 1 or more items. Each dish has a description hence it is associated with dish description object. Each dish will have one description. Hence it is one to one association. Dish also can have zero or more dish review. Hence there is one to many association between dish and dish review. Since reviews are added by the customer, review object is dependent on customer. Cart is not complete without delivery address, so there is dependency between cart and delivery address. For cart to move further, payment method also needs to be added. Hence there is dependency between cart and payment method. Once payment is successful, a receipt is generated. Payment and receipt have one to one association. Once payment is validated, database is updated with inventories and list of available dishes. Along with it, once payment is successful, delivery address is sent to the staff member which creates the staff object. Staff object is also associated with the delivery address as staff needs delivery address to deliver food. Staff notifies delivery once food is delivered, which creates an object Food Delivery. Once food delivery is successful, a closing message is sent to the customer requesting for restaurant review.

3. Lessons Learnt

I learnt to make more thorough planning to reduce the scope changes made in the later stages of the project so that I do not have to revisit and modify my work. I also got to know the difficulties and complexities in managing the Online food order. It is while building this system, I got to know the importance of customer feedback.

If I had another chance at the project what I would do differently is that I would make other modifications to the system as mentioned below.

- Since feedback from a customer helps a lot in the growth of the restaurant as well as helps other customers to know more about a dish and the restaurant, customer should be encouraged to provide their feedback. Hence providing the reward points to the customers for their reviews which can be redeemed in their next order, encourages customers to provide feedback. By this, both restaurant and customers are benefitted.
- I would also like to break restaurant management into kitchen and delivery management. This gives a broader picture of how the data flows in the system. For example, restaurant staff receives the address and order details from the system and sends kitchen management the order details for food preparation. Kitchen management notifies delivery staff once food is prepared and when it is ready for pick up.
- To reduce the risk of delivering the food to the wrong address, once food is delivered, staff can take a picture of the parcel left at the door and send it to the customer.