OBJECT DESIGN

Group 3

Tran Thanh Minh – 103809048

Cao Phuc Khai Hoan - 103804739

Bani-Hashemi Le Minh – 103846074

SWE30003 – Software Architectures and Design

Instructor Tuan Tran

Swinburne University of Technology

17 March 2024

Table of Contents

Executive Summary

All *Your Healthy Foods* is a store that currently provides healthy foods in the local area. The store wants to broadly increase its operation, by having an online appearance and providing necessary features for the users who access shopping website, which relates to the aim is to reach the nationwide customers in Australia. ￼

Problem Analysis

The provided situation is:

- All Your Healthy Foods is a physical store with limited reach.
- They want to expand their business online and offer home delivery.
- Their current system lacks the functionalities to support online operations.

Goals

The goals are to increase sales which will provide a convenient platform for customers to browse and purchase products. Moreover, expand the customer base beyond the local area by offering online shopping nationwide.

Assumptions

1. It is assumed that detailed information about healthy food products, including descriptions, prices, and availability, will be provided by All Your Healthy Foods for integration into the online store.
2. Customers must register and provide necessary details like name, address, and contact information to create online shopping accounts.
3. The online store assumes the implementation of secure payment processing mechanisms to handle transactions securely, protecting customers' financial data.
4. The system assumes that inventory management processes are in place to track product availability accurately and update stock levels in real-time.

5. It is assumed that arrangements for product packaging and delivery logistics are established to ensure timely and efficient delivery of orders to customers' homes.
6. The online store assumes compliance with relevant regulations and standards governing online retail operations, particularly regarding food safety and labeling requirements.
7. It is assumed that the customers have reliable internet access and basic computer literacy to navigate through the online store.

Simplifications

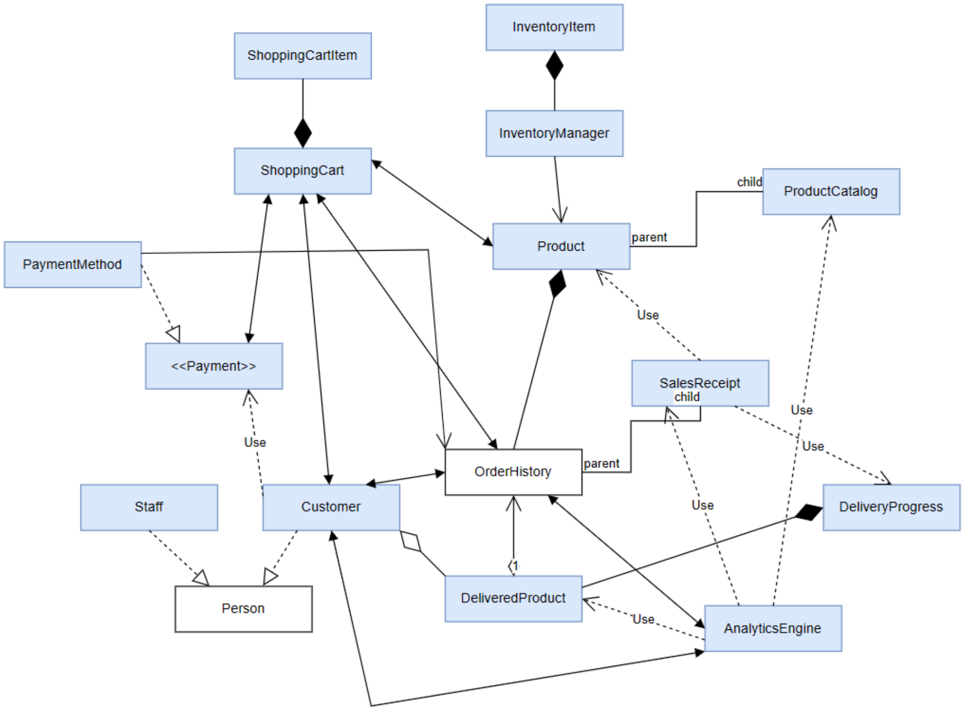
- Since offline and online information about product details and quantities provided by inventory management section are the same, it can be stored as one in the database.
- Customer credentials and their financial data can be saved on their profile page for convenient transaction conducting and protected with authentication and data encryption.
- Discount codes or loyalty programs can be integrated at a later phase.
- Implement a basic search function by product name or category. Complex search features with multiple filters can be added based on future needs.
- Provide basic reports on sales data by day, week, month, year, or year-to-date. More granular reporting with product breakdowns or customer segmentation can be addressed later.

Candidate Classes

Candidate Class List

- OrderHistory
- SalesReceipt
- DeliveryProgress
- DeliveryProduct
- ShoppingCart
- Product
    - ProductPrice
    - ProductDescription
    - CategoryID
- Person
    - Staff
    - Customer
- Payment
- PaymentMethod
    - BankDetails
    - COD (Cash on Delivery)

UML Diagram



CRC cards

Product

Class name: Product Super class: -	
Product is a dish/meal item that is produced for sale both online and offline. In this case business organization, food item for sale contains its unique ID, name, category group, stock quantity, originality, product description, and price tag.	
Responsibilities	Collaborators
Store product details such as manufacturer, production date, expiration date, and product information, etc.	-
Has the price tag/Calculate price	-
Knows its category and brand section	-
Presents remaining stock quantity	-

Apply discounts	

ProductCatalog

Class name: ProductCatalog Super class: Product	
ProductCatalog is responsible for maintaining and providing access to the product listings, including details and availability.	
Responsibilities	Collaborators
Maintain a list of products in separate category	Product
Provide product details	InventoryManager
Offer search and filter functions	ShoppingCart
Update product entries	AnalyticsEngine
Interface with InventoryManager	InventoryManager

Person

Class name: Person Super class: -	
A common person with information shared by other classes inheriting from it.	
Responsibilities	Collaborators
Knows name, address, phone number	Staff

Staff

Class name: Staff Super class: Person
A staff of the sales system, control and manage the product states and inherits from the Person class. They are responsible for managing product counts, orders and deliveries,

ensuring product quality, and communicating any changes in product availability or specifications.	
Responsibilities	Collaborators
Check product counts and quality to the store	InventoryManager, AnalyticsEngine
Print the invoice	SalesReceipt
Manage orders and deliveries	ProductCatalog
Communicate availability and changes	

Customer

Class name: Customer Super class: Person	
A customer of the sales system, inherits from the Person class	
Responsibilities	Collaborators
Can find a list of orders placed	OrderHistory
Knows what product is being delivered	DeliveredProduct

DeliveryProgress

Class name: DeliveryProgress Super class: -	
Tracking the order delivering through the progress, from the storage to the given destination.	
Responsibilities	Collaborators
Track the delivery status	OrderHistory
Update status in real-time	Customer
Notify customers of delivery updates	DeliveredProduct
Coordinate with delivery personnel/shippers	-

DeliveredProduct

Class name: DeliveredProduct Super class: -	
The order arrives at the given destination and is delivered by the shipper, to the customer who orders the product.	
Responsibilities	**Collaborators**
Represent products that have been	Product
Maintain delivery details	OrderHistory
Update inventory upon delivery	InventoryManager
Confirm products delivered to customers	DeliveryTracker

ShoppingCart

Class name: ShoppingCart Super class: -	
ShoppingCart class is responsible for managing the customer's selection of products, calculating the total cost, handling the checkout process, applying any discounts or promotions, and updating the quantities of items in the cart.	
Responsibilities	**Collaborators**
Hold selected products for purchase	ShoppingCartItem
Calculate total price	Customer
Manage checkout process	PaymentMethod
Apply discounts and promotions	ProductCatalog
Update item quantities	-
Remove items from cart	-

ShoppingCartItem

Class name: ShoppingCartItem Super class: ShoppingCart

ShoppingCartItem class is responsible for storing information about a specific product and its quantity within the shopping cart. It calculates the price of the item based on the quantity and product details, updates the quantity as needed, and can remove itself from the cart if necessary.	
Responsibilities	Collaborators
Store product and quantity	ShoppingCart
Calculate item price	Product
Update quantity	-
Remove itself from cart	-

<<payment>>

Class name: <<payment>> Super class: -	
Payment interface outlines the necessary structure and functions for processing payments, ensuring that any class implementing this interface, such as PaymentMethod, adheres to the required standards and provides necessary payment functionalities.	
Responsibilities	Collaborators
Define payment process structure	PaymentMethod
Ensure compliance with payment	Order
standards and security	Customer
Provide transaction status	-

PaymentMethod

Class name: PaymentMethod Super class: <<payment>>	
PaymentMethod class is an implementation of the Payment interface and is responsible for the actual processing of payment transactions. It handles the validation,	

<table>
<tr><td colspan="2">confirmation, and security of payments, and can be extended to include various types of payment methods like credit cards or digital wallets.</td></tr>
<tr><th>Responsibilities</th><th>Collaborators</th></tr>
<tr><td>Process payment transactions</td><td>Payment</td></tr>
<tr><td>Validate payment details</td><td>ShoppingCart</td></tr>
<tr><td>Handle payment confirmations</td><td>Customer</td></tr>
<tr><td>Manage payment security/Adhere to payment receipts</td><td>SalesReceipt</td></tr>
<tr><td>Implement specific payment types (COD, credit card, PayPal)</td><td>Order</td></tr>
</table>

OrderHistory

<table>
<tr><td colspan="2">Class name: OrderHistory
Super class: -</td></tr>
<tr><td colspan="2">OrderHistory class is responsible for managing all aspects of customer orders, from recording the initial order details to tracking the order's progress, managing payments, and coordinating delivery.</td></tr>
<tr><th>Responsibilities</th><th>Collaborators</th></tr>
<tr><td>Record details of customer orders</td><td>Customer</td></tr>
<tr><td>Track order status</td><td>ShoppingCart</td></tr>
<tr><td>Manage payment transactions</td><td>PaymentMethod</td></tr>
<tr><td>Generate sales receipts</td><td>SalesReceipt</td></tr>
<tr><td>Update inventory</td><td>InventoryManager</td></tr>
<tr><td>Coordinate delivery process</td><td>DeliveryProgress</td></tr>
<tr><td>Handle order modifications</td><td>ShoppingCartItem</td></tr>
</table>

InventoryManager

<table>
<tr><td colspan="2">Class name: InventoryManager
Super class: -</td></tr>
<tr><td colspan="2">InventoryManager class is responsible for maintaining optimal inventory levels, ordering new stock, updating records, and liaising with suppliers.</td></tr>
<tr><th>Responsibilities</th><th>Collaborators</th></tr>
<tr><td>Oversee inventory levels</td><td>InventoryItem</td></tr>
<tr><td>Reorder stock as necessary</td><td>ProductCatalog</td></tr>
<tr><td>Update inventory records</td><td>Staffs</td></tr>
<tr><td>Communicate with suppliers</td><td>AnalyticsEngine</td></tr>
<tr><td>Provide inventory data for analysis</td><td>-</td></tr>
</table>

InventoryItem

<table>
<tr><td colspan="2">Class name: InventoryItem
Super class: InventoryManager</td></tr>
<tr><td colspan="2">InventoryItem class is tasked with monitoring the stock level of each item, alerting when supplies are low, adjusting stock based on sales, and aiding in inventory audits. It works closely with InventoryManager to ensure stock levels are adequate, Product for item details, SalesReceipt for sales data, and Order for processing sales and returns.</td></tr>
<tr><th>Responsibilities</th><th>Collaborators</th></tr>
<tr><td>Keep track of item stock level</td><td>InventoryManager</td></tr>
<tr><td>Notify when stock is low</td><td>Product</td></tr>
<tr><td>Update stock based on sales</td><td>SalesReceipt</td></tr>
<tr><td>Assist in inventory audits</td><td>OrderHistory</td></tr>
</table>

SalesReceipt

<table>
<tr><td>Class name: SalesReceipt
Super class: -</td></tr>
</table>

SalesReceipt class is responsible for documenting the specifics of a sale, including the items sold, their prices, and the total sale amount. It serves as evidence of the transaction for the customer and is used to track the sale's date and time.	
Responsibilities	**Collaborators**
Record the details of a sale	Customer
Include sold products and price	Order
Calculate total amount of sale	PaymentMethod
Provide proof of purchase	Product
Track date and time of transaction	ShoppingCart

AnalysisEngine

Class name: AnalysisEngine Super class: -	
AnalysisEngine is meant to provide further insights in all collected data from online customers that can efficiently provide good generalization and feedback for the store's owner to better understand the eating habit trends and effectively manage their service.	
Responsibilities	**Collaborators**
Analyze sales and customer data	ProductCatalog, Order, Customer, SalesReceipt, DeliveredProduct
Generate reports on product performance and sales trends	-
Provide insights for inventory management and marketing strategies	-

Design Quality

Design Heuristics

1. Single Responsibility Principle: Most classes seem to adhere to this principle. For example, Product stores information about a product, SalesReceipt provides a receipt for a purchase, and Customer handles customer information.
2. Information Hiding: It's difficult to say for certain without seeing the code, but CRC cards suggest potential data hiding. Classes likely have private attributes and public methods to access/modify them.
3. Base Class Data Privacy: Unclear from CRC cards. If you have base classes like Person, their data members should ideally be private.
4. Consistency: Review your code, class names, variable names, and comments for consistency.
5. Error Handling: Implement informative error messages that suggest solutions whenever possible.
6. Overriding Base Class Methods: Ensure derived classes don't override base class methods meant to be empty (return null).
7. Avoiding God Classes: Avoid creating a class that handles everything. Strive for focused and well-defined classes.
8. Abstract Classes: Determine if any classes can be abstract (cannot be directly instantiated) to enforce specific behavior in subclasses.

Design Patterns

Creation Patterns

Consider using the Factory Method pattern if you anticipate adding new types of Products, Customers, or other entities in the future. This allows for creating different objects based on a common interface.

We also want to implement the Singleton pattern in to ensure that a class has only one instance and provides a global point of access to it. A real-life example could be a print spooler or a logging service in an application. These services should ideally have only one instance running in the system.

Behavioral Patterns

The Strategy pattern might be a good fit for handling different payment methods (assuming there are subclasses of PaymentMethod) through a common Payment interface. This allows adding new payment methods without affecting existing code.

Structural Patterns

The Model-View-Controller (MVC) pattern can be a good fit for separating data (models like Product), presentation (views for displaying information), and user interaction (controllers for handling user input). However, for smaller projects, a simpler architecture might suffice.

Bootstrap Process

1. User interaction: A salesperson interacts with the system, likely though a menu-driven interface.
2. System Initialization: The main program creates an instance of the Menu class, responsible for starting the application and presenting user options.
3. Sale Initiation: The Menu class might trigger the creation of a SalesReceipt object, indicating the start of a new sales transaction.
4. Customer Information: The system prompts the salesperson to ask for customer details, which are stored in the Customer object and potentially saved on the database.
5. Payment information: The SalesReceipt object creates a Payment object. The salesperson enters payment details, which are processed by the Payment object and potentially saved to the database.
6. Product Selection: The SalesReceipt object creates a Product object. The system retrieves product information from the database and populates the Product object.
7. Purchase Record Creation: The SalesReceipt object creates a OrderHistory object, recording the sale of the product to the customer. This record is likely saved to the database.

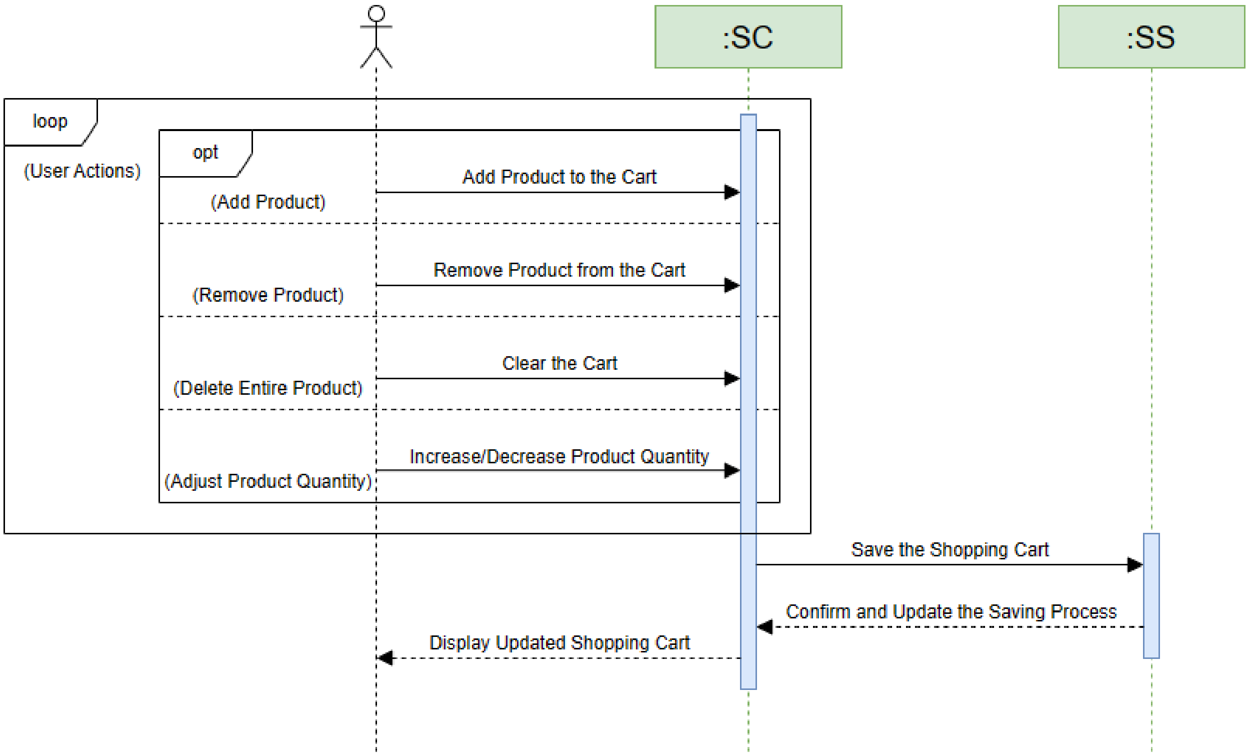
Verification

Definitions, acronyms, and abbreviations

The table below illustrates the acronyms that are converted from some of the used terms, and use those acronyms in UML sequennce diagrams and in diagram explanations that are also shown below.

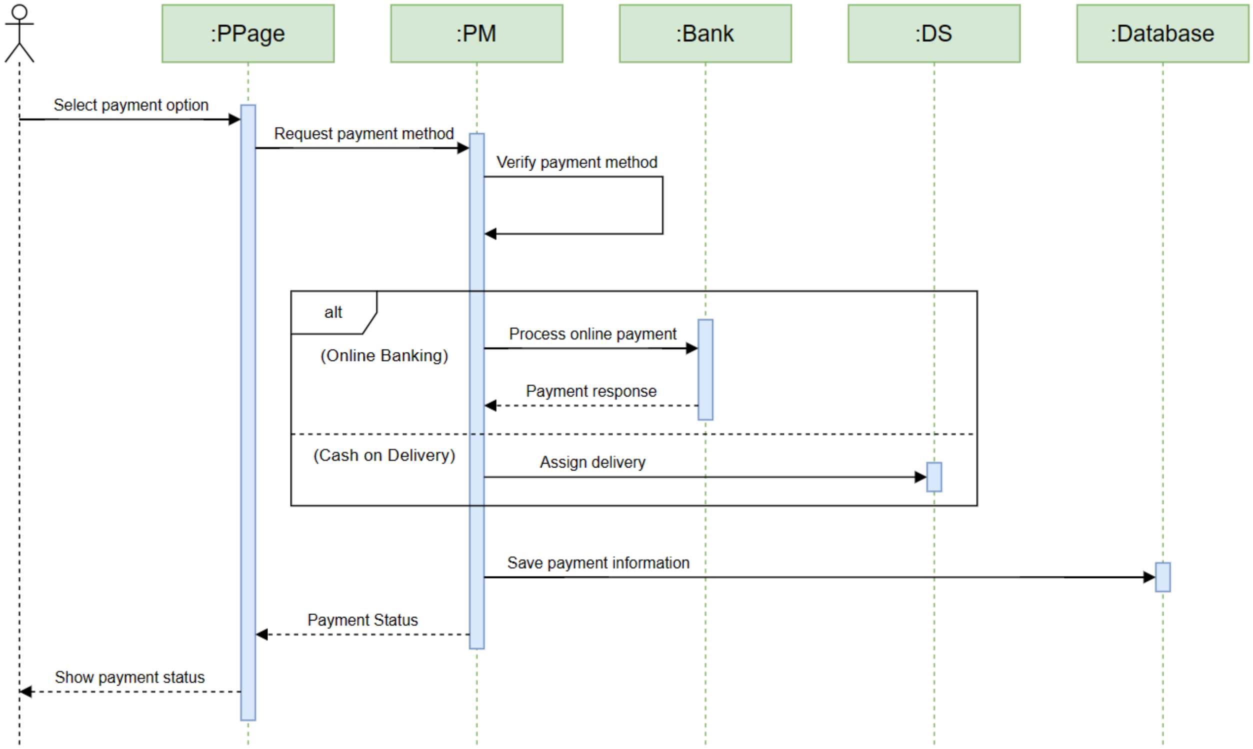
Used Terms	Acronyms
ShoppingCart	SC
ShoppingService	SS
PaymentPage	PPage
PaymentManagement	PM
DeliveryService	DS
OrderTrackingPage	OTPage
OrderTrackingManagement	OTM
CustomerSupportPage	CSPage
CustomerSupportManagement	CSM
CustomerService	CS
Cash on delivery	COD

Add the product to the cart


:SC
:SS
loop
(User Actions)
opt
Add Product to the Cart
(Add Product)
Remove Product from the Cart
(Remove Product)
Clear the Cart
(Delete Entire Product)
Increase/Decrease Product Quantity
(Adjust Product Quantity)
Save the Shopping Cart
Confirm and Update the Saving Process
Display Updated Shopping Cart

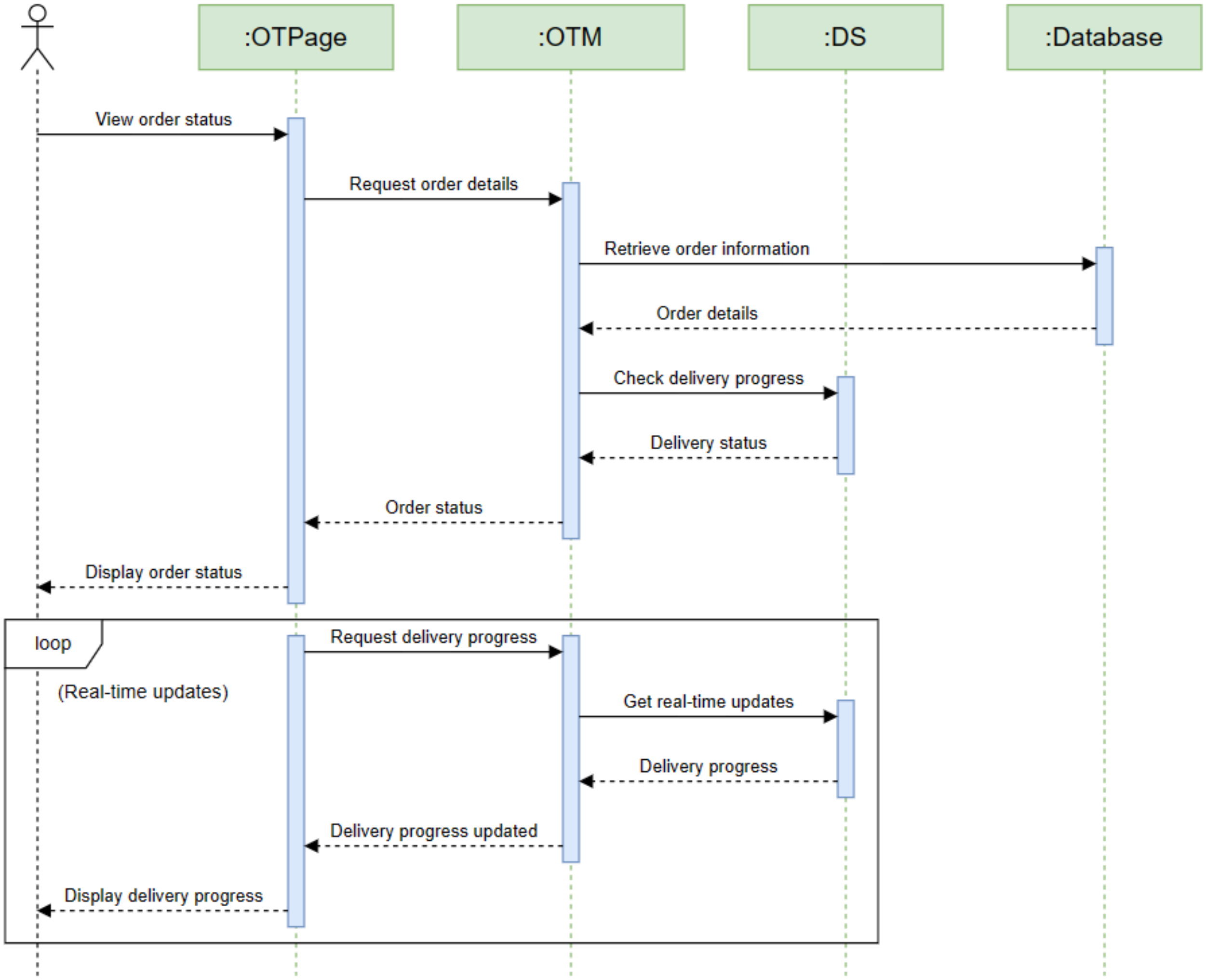
When a customer access to the online store to conduct shopping activities, customer has some abilities to interact with the SC, repetitively. Some of the options are: adding, removing, clearing entirely, and adjusting quantity, of a product or item that customer wishes to buy. Each of the above action is saved by the SS, by confirming, and updating the saving process, and displayed the updated SC to customer.

Make the payment



The process of selecting a payment option, either online banking, or COD, and processing the payment accordingly, is illustrated through the above UML diagram. Customer can make a decision of choosing payment method on the PPage, which sends a request to the PM, and it double-checks the customer selection. If online banking is chosen, the bank is contacted in order to process the online payment, and responses back to the PM. For the COD, it is assigned to DS directly, as customer completes the payment after the order is delivered to the given destination. After settling the transaction option, the information of the invoice is saved in the database, and PM returns payment status back to the PPage, and shows it to the Customer.

Order Tracking and Delivery Progress

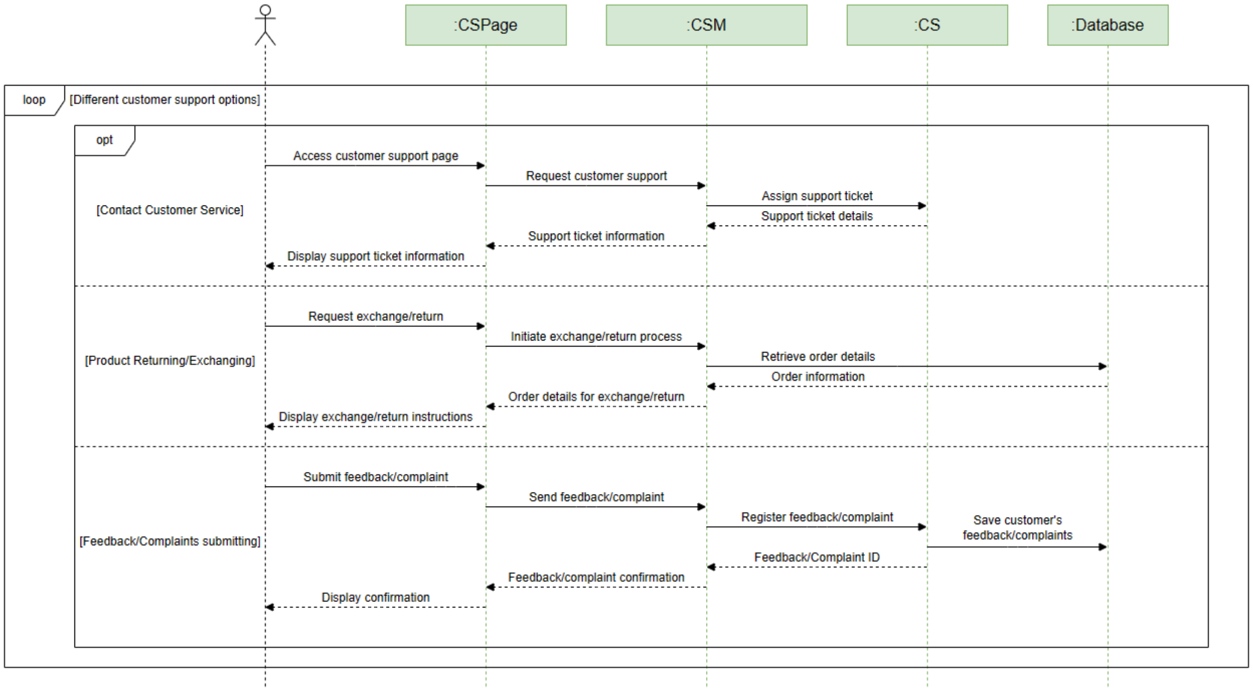


This UML represents the order tracking process with real-time updates on delivery progress. Customer can decide to initiate the order status viewing on the OTPage, which send a request to the OTM. The OTM retrieves information of the order from the database, receives back the order details, and also check the progress of the delivery with the DS. DS responds back with the current delivery status (like the order is processing, shipped, out of delivery, or delivered, etc.), sending it back to OTM and OTPage, respectively, and finally displays it to the Customer.

Meanwhile, it also provides to the Customer the feature to check up the real-time updates. The diagram illustrates a loop that requests the delivery progress from the OTPage to the

OTM, as a constant update. OTM contacts to the DS in order to get the latest updates of the delivery progress, and OTPage can show the delivery status, such as the order’s estimated delivery time, its current location, etc. to the Customer.

Customer Support


:CSPage
:CSM
:CS
:Database
loop
[Different customer support options]
opt
[Contact Customer Service]
Access customer support page
Request customer support
Assign support ticket
Support ticket details
Support ticket information
Display support ticket information
[Product Returning/Exchanging]
Request exchange/return
Initiate exchange/return process
Retrieve order details
Order information
Order details for exchange/return
Display exchange/return instructions
[Feedback/Complaints submitting]
Submit feedback/complaint
Send feedback/complaint
Register feedback/complaint
Save customer's feedback/complaints
Feedback/Complaint ID
Feedback/complaint confirmation
Display confirmation

The Customer Support process demonstrates the available options for customers to utilize the services, including:

- Contacting Customer Service:
  - Customer accesses the CSPage to request customer support.
  - CSM receives the request and assigns a support ticket to the CS.
  - CS provides the details of the support ticket return to the CSM.
  - CSM sends the information of the support ticket to the CSP and displays it to the Customer.
- Exchange/Return Process:
  - Customer requests an exchange or return of the product to the CSP.
  - CSP initiates the request process to the CSM.
  - CSM retrieves required order details from its Database, and returned the order information.
  - CSM returns the order details for exchange/return to CSP, and the instructions are displayed to the Customer.

- Submitting Feedback/Complaint:
    - Customer submits a feedback/complaint on the CSPage.
    - The request is sent, and registered by CSPage, CSM correspondingly, and CS receives the request at the end.
    - The feedback/complaint copy is saved in Database for future product/service developing reference.
    - CS provides a unique ID back to the CSM.
    - The CSM sends the request confirmation back to the CSPage, and the confirmation is displayed to the Customer by CSPage.

These are the options that the customer can optionally select to perform a Customer Support process since they have different purposes, and they can be repeated many times.

References

Lucidchart (2018). *How to Make a UML Sequence Diagram*. *YouTube*. Available at: https://www.youtube.com/watch?v=pCK6prSq8aw.

www.visual-paradigm.com. (n.d.). *SysML: Modeling Scenarios with Sequence Diagram*. [online] Available at: https://www.visual-paradigm.com/guide/sysml/modeling-scenarios-with-sequence-diagram/?fbclid=IwAR2QQlDqK2Xc9b9cjkCakck2pn10LxDleewYxfaKOfP9G0hnvN8DeQHNbIk [Accessed 24 Mar. 2024].

www.visual-paradigm.com. (n.d.). *Sequence Diagram - UML Diagrams - Unified Modeling Language Tool*. [online] Available at: https://www.visual-paradigm.com/VPGallery/diagrams/Sequence.html?fbclid=IwAR1ETeVjLnI5ZzkqFDjyDQY4oY2Vv32b3Xp4UALLKwNLOu5t1LZdyl_Wrxo [Accessed 24 Mar. 2024].