

Assignment #1

Reem Alhamami - 202302133

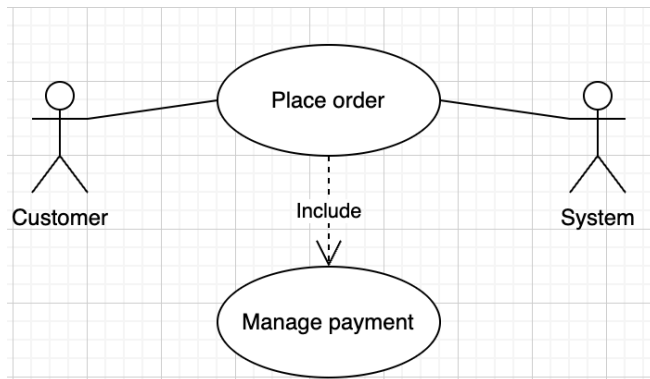
28 - FEB - 2025

1.a Identifying Use Cases:

- Place Order
- Process Delivery
- Generate Delivery Note
- Track Order
- Update Delivery Status

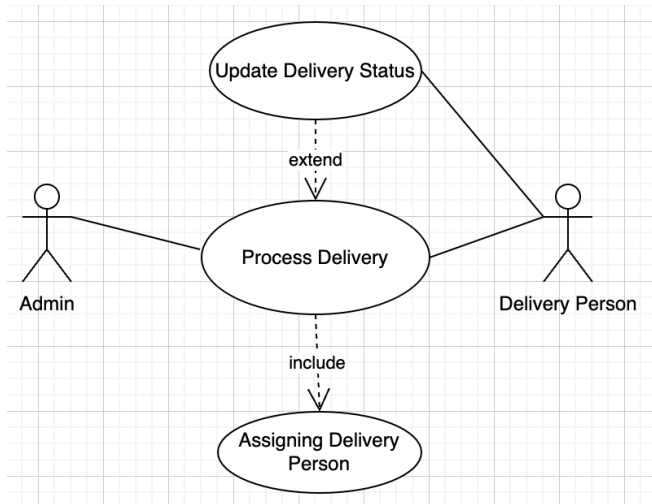
1.b and 1.c Use-Cases Diagram & Descriptions:

- Place Order



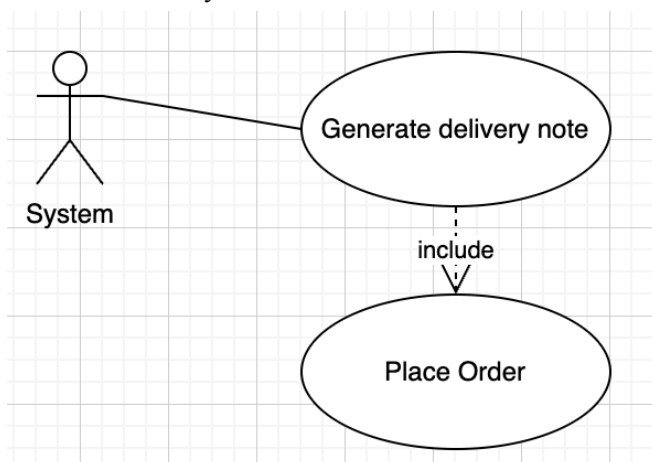
Use case:	Place Order
Actors:	Customer, system
Trigger:	Customer wants to place a new order.
Pre-conditions	The customer must be logged in and have a valid Items in the cart
Main steps	<ol style="list-style-type: none">1. Customer selects items and adds them to the cart.2. Customer proceeds to checkout.3. System prompts for delivery.4. Customer enters required details and confirmed payment.5. System validates payment and saves order details.
exception	<p>If payment fails, notify the customer to retry.</p> <p>If items are out of stock, notify the customer and suggest similar items.</p>

- Process Delivery



Use case:	Process Delivery
Actors:	Admin, delivery Person
Trigger:	A new Order is placed and ready for delivery.
Pre-conditions	An Order must exist and be paid for.
Main steps	<ol style="list-style-type: none"> 1. Admin views pending orders. 2. Admin assigns a delivery person. 3. System notifies the delivery person. 4. Delivery person accepts the assignment. (include) 5. System updates order Status to 'out for delivery' (extend)
exception	If no delivery person is available. If the delivery person declines.

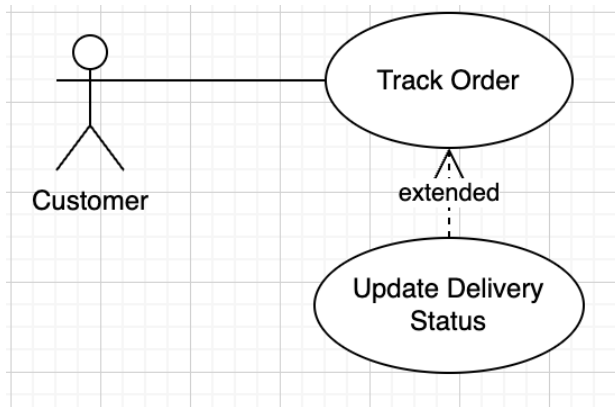
- Generate Delivery Note



Use case:	Generate Delivery Note
-----------	------------------------

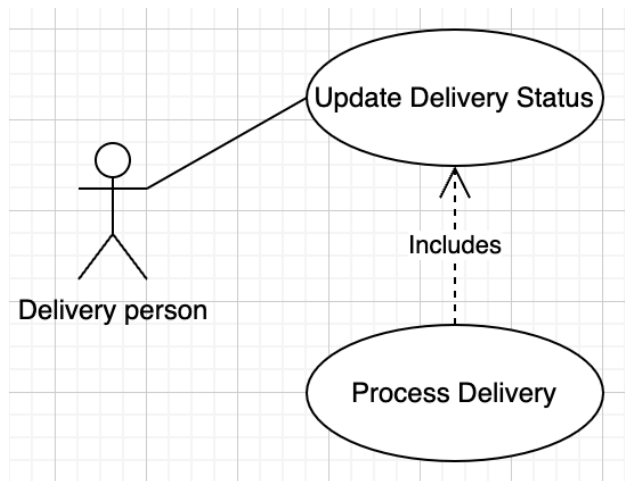
Actors:	System
Trigger:	Order is confirmed and ready for delivery.
Pre-conditions	The order must be placed and processed
Main steps	<ol style="list-style-type: none"> 1. System retrieves order details. 2. System formats a delivery note including order information.(include) 3. system attaches the delivery note to the order record.
exception	If order details are incomplete.

- Track Order



Use case:	Track Order
Actors:	Customer, System
Trigger:	Customer wants to check the status of an order
Pre-conditions	The order must be exist in the system.
Main steps	<ol style="list-style-type: none"> 1. Customer logs into the system. 2. Customer navigates to the “Track Order” section. 3. System retrieves and displays order status. (Extend)
exception	If customer enters an invalid order ID.

- Update Delivery Status



Use case:	Update delivery Status.
Actors:	Delivery Person, System
Trigger:	Delivery person updates order progress.
Pre-conditions	The order must be assigned to a delivery person.
Main steps	<ol style="list-style-type: none"> 1. Delivery person logs into the system. 2. Delivery person selects the assigned order. 3. Delivery person update status. (includes)
exception	If order is not found. If system fails to update.

2 Identify Objects and classes:

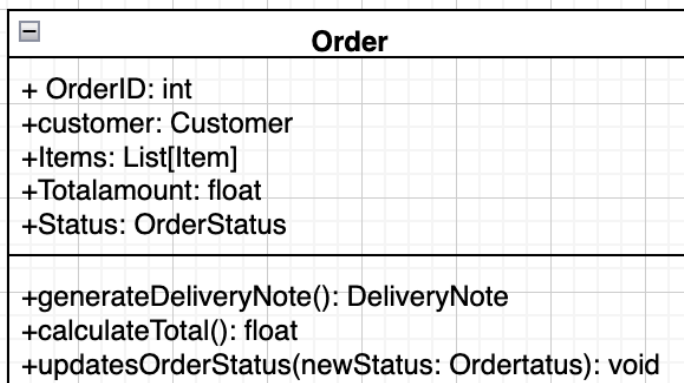
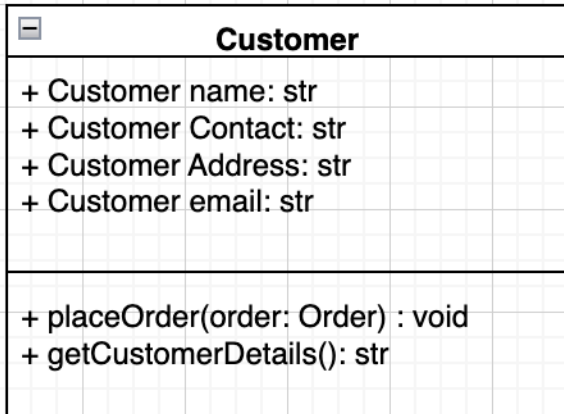
Objects	corresponding Class.	Description
Customer	Customer	Represent a user who places orders
Order	Order	Stores order details, items, and status.
Delivery person	DeliveryPerson	Represents a person assigned to deliver the order.
Delivery	Delivery	Track Delivery status and progress
Item	Item	Represent items that can be ordered.
Payment	Payment	Handles payment processing and validation
Delivery note	DeliveryNote	Generates and stores delivery details


Relationships


- Customer places one or more order objects.
- Order contains multiple item object.


- Order is linked to a payment object.
- Order generates a DeliveryNote.
- Order is assigned to DeliveryPerson.
- DeliveryPerson updates the Delivery status.


2.b UML Class Diagram:




 DeliveryPerson
+ personID: int + name: str +PhoneNumber:str +assignedDelivery:Delivery
+getAssignedDeliveries(): List[Delivery] +updatesOrderStatus(status: str): void

 Delivery
+ DeliveryID: int + order: Order +DeliveryPerson: DeliveryPerson +Status:str +estimatesArrival: str
+trackOrder(): str +updateDeliveryTime(newTime: str): void

 Item
+ ItemID: int +name: str +price: float +quantity: int
+ calculateItemTotal(): float +updateQuantity(newQuantity: int): void

 Payment
+paymentID: int + order: Order +paymentMethod: PaymentMethod +amount Paid: float
+processPayment(): Enum +getpaymentDetails(): str

 DeliveryNote
+noteID: int +order: Order deliveryDetails: str
+generateNote() : Enum +getDeliveryDetails(): str

Relationships:

- Customer places multiple Order objects.
- Order contains multiple Item Objects.
- Order is linked to payment
- DeliveryPerson is assigned to delivery
- Delivery is linked to Order.
- Order owns Deliverynote.