Scenario: **Order Fulfillment System in a Supply Chain**

**Problem Statement**

* Customers expect fast, accurate, and trackable order deliveries.
* Businesses face challenges in inventory management, warehouse operations, order processing, and customer satisfaction, especially with scaling e-commerce demands.

**Objective**

* Automate order processing from placement to delivery.
* Optimize inventory and warehouse operations.
* Provide real-time shipment tracking for customers.
* Handle returns and refunds efficiently.

# Strategic Design Principles

## Collaboration between business experts and technical team

* Supply chain experts – Inventory, Warehouse
* Accounts experts – Tax, Billing
* Logistics experts – Shipping, delivery partners integration
* Technical team – Feasibility, Tech stack, resources, infrastructure

## Domain Scope

(Define the area / end users of the application)

## Domain description

(Describing the complete domain briefly including core business domain and functionalities.

## Ubiquitous Language

* Order – A customer purchase request
* Fulfilment – The process of picking, packing and shipping an order
* Payment – Money received for an order
* Inventory – Stock available for fulfil orders
* Shipment – Delivery of the order to the customer

## Domain Analysis

(Identify core subdomain, supporting subdomain and generic subdomain)

* 1. Core subdomain
     + **Order fulfilment** – Processing, managing, and tracking customer order from creation to delivery.
  2. Supporting subdomain
     + **Inventory management** – Track stock level, warehouse allocation
     + **Payment Processing** – Handle payment authorization and settlement.
     + **Shipping and Logistics** – Manage shipment creation, tracking and delivery.
     + **Customer Notification** – Inform customer about order status updates.
  3. Generic subdomain
     + **Authentication and Authorization** – Secure user identity and access control
     + **Audit Logging** – Tract all critical changes and events
     + **Document management** – Invoice and receipt generation.

1. Bounded Contexts

|  |  |  |
| --- | --- | --- |
| Bounded context | Responsibility | Interfaces with |
| Order Management context | Handle creation, validation and lifecycle of orders | Inventory, Payment, Shipping, Notifications |
| Inventory Context | Manage stock, reservations, restocking | Order management, Shipping |
| Payment Context | Payment authorization, Fraud detection, refunds | Order Management |
| Shipping Context | Create shipments, manage couriers, track deliveries | Order Management, Inventory |
| Notification Context | Customer Communication  (email, sms, app alerts) | All other context |
| Security Context | Login, authorization, access control | All context |
| Reporting context | Sales and order analytics | Order Management, Payment, Shipping. |

1. Context Mapping

Order -> Payment, Inventory, Shipping

Payment -> Payment Gateway Integration

Notification <- Other contexts

# Tactical Design Principles

1. Key Entities
   * + Order – orderId, customerId, orderStatus, orderDate, totalAmount
     + OrderItem – orderItemId, orderId, productId, quantity, price
     + Payment – PaymentId, orderId, amount, method, status
     + InventoryItem – productId, productName, avaibleQuantity, reservedQuantity
     + Shipment – shipmentId, trackingNumber, deliveryStatus
2. Key Value Objects
   * + Address – streat, city, postalCode, country
     + Money – Amount, currency
     + TrackingInfo – TrackingNumber, Location, Status
3. Key Aggregates

|  |  |  |
| --- | --- | --- |
| Aggregate | Aggregate Root Entity | Included Entities / Value object |
| OrderAggregate | Order | OrderItems, Address, TrackingInfo |
| PaymentAggregate | Payment | Order, PaymentDetail, Money |
| ShipmentAggregate | Shipment | TrackingInfo, DeliveryStatus |
| InventoryAggregate | InventoryItem | Measurement |

1. Key Repositories
   * + OrderRepository – Manage orders and their lifecycle
     + PaymentRepository – Store and retrieve payment records
     + Inventory Repository – Manage stock levels and reservations
     + ShipmentRepository – Manage shipments and delivery status
2. Key Domain Events
   * + Customer places an order -> “OrderPlaced” event is emitted
     + Inventory Service received “OrderPlaced” -> Check stock and emits “InventoryChecked”
     + Payment Service received “OrderPlaced” -> Process payment and emits “PaymentProcessed”
     + Order Service update order Status -> emits “OrderConfirmed”
     + Shipping Service receive “OrderConfirmed” -> ship order and emits “OrderShipped”
     + Order delivered to customer -> emits “Orderfulfilled”
3. Key Factories
   * + OrderFactory – Build complete order aggregates from cart data.
     + PaymentFactory – create payment object after checkout.
     + ShipmentFactory – initiate shipment based on stock confirmation.
4. Key Application Services
   * + OrderApplicationService – Place new order, cancel, update
     + PaymentApplicaitonService – Capture, refund, verify payments
     + InventoryApplicationService – Reserve Stock, release stock, verify stock
     + ShippingAPplicaitonService – Create and track shipments
     + NotificationApplicationService – Notify customer on status changes.