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

1. **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.

1. **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.

1. **Collaboration between business experts and technical team**

* **Supply chain experts** – Inventory, Warehouse, etc.
* **Accounts experts** – Tax, Billing, etc
* **Logistics experts** - Shipping, e-way bill, delivery partners integrate, etc.
* **Technical team** – Feasibility, tech stack, resources, infrastructure, etc.

1. **Domain Scope**

(Define the area / end users for the application)

1. **Domain description**

(Describing the complete domain briefly including code business domains and functionalities.)

1. **Ubiquitous Language**

|  |  |
| --- | --- |
| **Term** | **Meaning** |
| Order | A customer’s 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 |
| Return | Customer returning purchased items |
| Warehouse | Storage location for inventory |
| Packing Slip | Document listing order contents. |

1. **Domain Analysis**

(Identify code sub-domain, supporting sub-domain, and generic sub-domain)

* 1. **Core Domains /Sub-domains**
* **Order Fulfilment Management** – Processing, managing and tracking customer order from creation to delivery
  1. **Supporting Sub-domains**
* **Inventory Management** – Track stock levels, warehouse allocation.
* **Payment Processing** – Handle payment authorization and settlement.
* **Shipping and Logistics** – Manage shipment creation, tracking and delivery.
* **Customer Notification** – Inform customers about order status updates.
  1. **Generic Sub-domains**
* **Authentication and Authorization** – Secure user identity and access control.
* **Audit Logging** – Track 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 contexts |
| Security Context | Login, authorization, access control | All contexts |
| Reporting Context | Sales and order analytics | Order Management, Payment, Shipping |

1. Context Mapping

Order -> Payment, Inventory, Shipping

Inventory <-> Shipping

Notification <- Other Contexts

Payment -> Payment Gateway Integration

# Tactical Design Principles

1. Key Entities

|  |  |
| --- | --- |
| **Entity** | **Attributes** |
| Order | orderId, customerId, orderStatus, orderDate, totalAmount |
| OrderItem | orderItemId, orderId, productId, quantity, pricePerUnit |
| Payment | paymentId, orderId, amount, status, method |
| InventoryItem | productId, productName, availableQuantity, reservedQuantity |
| Shipment | shipmentId, trackingNumber, carrier, deliveryStatus |

1. Key Value Objects

|  |  |
| --- | --- |
| **Value Object** | **Attributes** |
| Address | Street, city, postalCode, country |
| Money | Amount, currency |
| PaymentDetails | cardNumber, expirationDate, paymentMethod |
| TrackingInfo | trackingNumber, carrier, status |

1. Key Aggregates

|  |  |  |
| --- | --- | --- |
| **Aggregate** | **Aggregate Root** | **Included Entities / Value Objects** |
| OrderAggregate | Order | OrderItems, Address, PaymentStatus, TrackingInfo |
| Payment | Payment | Order, PaymentDetails, Money |
| InventoryAggregate | InventoryItem |  |
| ShipmentAggregate | Shipment | TrackingNumber, TrackingInfo, DeliveryStatus |
| ReturnAggregate | Return | ReturnedItems, RefundAmount |

1. Key Repositories

* OrderRepository – Manage orders and their lifecycle
* PaymentRepository – Store and retrieve payment records
* InventoryRepository – Manage stock levels and reservations
* ShipmentRepository – Manage shipments and delivery status

1. Key Domain Events - Handle domain events ( Workflow for order fulfilment)

* Customer paces an order -> “OrderPlaced” event is emitted.
* Inventory Service received “OrderPlaced” -> Checks stock and emits “InventoryChecked”
* Payment service receives “OrderPlaces” -> Process payment and emits “PaymentProcessed”
* Order Service updates order status -> emits “OrderConfirmed”
* Shipping service received “OrderConfirmed” -> Ships order and emits “OrderShipped”
* Order delivered to customer -> emits “Orderfulfilled”

1. Key Factories

* OrderFactory – Build complete order aggregates from cart data.
* PaymentFactory – Create payment object after checkout.
* ShipmentFactory – Initiate shipment based on stock confirmation.

1. Application Services

* OrderApplicationService – Place new orders, cancel, update
* PaymentApplicationService – Capture, refund, verify payments
* InventoryApplicationService – Reserve stock, release stock, Verify Stock
* ShipppingApplicationService – Create and track shipments.
* NotificationApplicationService – Notify customers of status changes.