Scenario: **Food Delivery System**

**Problem Statement**

* Today's food delivery market demands fast, reliable service with a wide variety of restaurants.
* Managing orders across restaurants, coordinating delivery partners, and ensuring food quality during transit creates significant operational complexity.

**Objective**

* Allow customers to order food easily from nearby restaurants.
* Enable real-time delivery tracking.
* Match delivery partners efficiently to orders.
* Ensure order accuracy and food quality.

# Strategic Design Principles

1. Collaboration between business experts and technical team
2. Domain Scope
3. Domain description
4. Ubiquitous Language
   * + Order – A customer food request placed
     + Cart – Temporary container for items before checkout
     + MenuItem – An item offered by a restaurant
     + Restaurant – A place where food is prepared and packed
     + Payment – Transaction for the order
     + Deliver Address – Location where food is delivered
     + Deliver Partner – Person delivering the order
     + Order status – current stage of the order
5. Domain Analysis

5.1 Core subdomain

* Order Management
* Delivery Management

5.2 Supporting subdomain

* Restaurant management
* Customer management
* Deliver partner management
* Payment handling
* Notification system

5.3 Generic subdomain

* Authentication and Authorization
* Location Services Integration (Eg. Google Maps)
* Audit Logging
* Documentation Mangement

1. Bounded Context

|  |  |  |
| --- | --- | --- |
| Bounded context | Responsibility | Interface with |
| Order Management context | Handling order creation, validation and lifecycle | Restaurant, Delivery, Payment |
| Restaurant context | Menu management, availability of food, preparation time | Order management, deliver |
| Delivery assignment context | Assign order to delivery partner, optimize routes, track delivery status | Order management |
| Payment context | Handle payment, tips, and refunds | Order management |
| Notification context | Order status updates to customer, restaurants, and delivery partners | All context |
| Promotion context | Manages discount coupons and promotions | Order management |

1. Context Mapping
   * Order -> Restaurant, Delivery, Payment
   * Restaurant -> Delivery
   * Notification <- Other contexts

Tactical Design Principles

1. Key Entities
   1. Order -> OrderId, CustomerId, RestaurantId, OrderStatu, TotalAmount
   2. MenuItem -> ItemId, name, description, picture, price
   3. Restaurant -> restaurantId, name, address, operatingHours
   4. Payment -> paymentId, amount, paymentMethod, status
2. Key Value Objects
   1. Address -> Street, City, postcode, Location coordinates.
   2. Money -> amount, currency
   3. PaymentDetail -> cardNumber, expiry, cvv (encrypted)
3. Key Aggregates
   1. Order aggregate -> Order, OrderItem, payment, deliverystatus
   2. Restaurant Aggregate -> Restaurant, menuItem, Address
   3. Delivery Aggregate -> DeliveryPartner, address, Restaurant, Customer
4. Key Repository
   1. Order Repository -> Manage all order records
   2. Restaurant Repository -> Manage menu items and restaurant data.
   3. Delivery Repository -> Manage delivery partners, location updates
   4. Payment Repository -> Track payments, tips and refunds
5. Domain Events
   1. OrderPlaced -> Triggered when a customer places a new food order.
   2. OrderCancelled -> Triggered when a customer or restaurant cancels an order.
   3. OrderPrepared -> Triggered when the restaurant finished preparing the order.
   4. DeliveryPartnerAssigned -> Triggered when a delivery partner is assigned to pick up the order
   5. OrderPickedUp -> Triggered when the deliver partner picks up the order from the restaurant.
   6. OrderDelivered -> Triggered when the delivery partner successfully delivers the food to the customer.
   7. DeliveryFailed -> Triggered if the delivery could not be completed (customer unavailable, wrong address)
   8. PaymentReceived -> Triggered when payment from the customer is successfully completed.
   9. TipReceived -> Triggered when a customer gives a tip after delivery.
   10. RatingSubmitted -> Triggered when the customer submitted a rating or feedback for the order, restaurant or delivery.
6. Application Services
   1. Order Application Service -> Manage order placement, tracking and cancelation
   2. Restaurant application service -> Update menu items, oprating hours, manage preparation time.
   3. Delivery Application service -> Delivery partner assignment, tracking live delivery.
   4. Payment Application service -> Payment processing, handle refunds and tips
   5. Notification application service -> Manage and send realtime notifications.