Online Food Ordering System

Diagram Explanations

Team 04

December 2024

Abstract

This document provides comprehensive documentation for the UML diagrams of the Online Food Ordering System. It covers use case diagrams, sequence diagrams, class diagrams, activity diagrams, and system architecture models.

Contents

1 Online Food Ordering System - UML Diagram Documentation

1.1 Use Case Diagrams

1.1.1 Customer Use Case Diagram

File: `usecase.puml`

This diagram shows customer interactions with the system. It covers authentication (register, login, manage profile), browsing (search menus, customize items, manage cart), ordering (checkout with cart validation and payment, place order), and post-order activities (track delivery, view history, cancel, request refunds, rate orders). Include relationships show mandatory flows while extend relationships show optional features.

1.1.2 Place Order Use Case Diagram

File: `place_order_usecase.puml`

This diagram breaks down order placement into four packages: Menu & Cart Management, Order Configuration, Payment & Pricing, and Order Processing. Mandatory functions include item selection, cart management, address selection, validation, payment, and notifications. Optional extensions allow customization, scheduling, coupons, tips, favorites, and cancellation. Actors include Customer, Payment Gateway, Notification Service, and Restaurant.

1.1.3 Track Order and Delivery Use Case Diagram

File: `track_order_and_delivery_usecase.puml`

This diagram shows tracking functionality in two packages: Customer Tracking (retrieve details, view status timeline, view ETA, receive notifications) and Operational Updates (update preparation/delivery status, share location). Optional features include contacting delivery agent/support, viewing proof of delivery, and rating. Actors include Customer, Restaurant, Delivery Agent, Mapping/GPS API, and Notification Service.

1.2 Sequence Diagrams

1.2.1 Place Order - High-Level Sequence Diagram

File: `place_order_sequence_high_level.puml`

This stakeholder view shows three phases: Order Configuration (select items, choose address, review summary), Payment (process through gateway with 30s timeout), and Order Fulfillment (create order, notify restaurant with 60s timeout, send confirmations). Payment failure triggers retry option; success triggers complete fulfillment workflow.

1.2.2 Place Order - Detailed Sequence Diagram

File: `place_order_sequence_detailed.puml`

This developer-focused diagram details the complete flow: Order Configuration with availability validation, cart management, coupon application, address serviceability checks, scheduling, and total calculation. Payment phase includes gateway processing with timeout and branching logic. Order Fulfillment shows order creation, restaurant notification with three response scenarios (accept, no response with auto-escalation, reject with refund), and parallel customer/restaurant notifications.

1.2.3 Track Order & Delivery - High-Level Sequence Diagram

File: `track_order_&_delivery_sequence_high_level.puml`

This stakeholder view shows three phases: Preparation (view status, restaurant marks ready, assign driver), In Transit (driver picks up, continuous location updates with live map and ETA), and Delivery (mark delivered, send confirmation, request feedback). Focuses on key milestone events.

1.2.4 Track Order & Delivery - Detailed Sequence Diagram

File: `track_order_&_delivery_sequence_detailed.puml`

This comprehensive diagram covers Order Status Inquiry (authenticate, retrieve status, handle errors), Real-Time Updates (preparation queries, driver assignment with location/rating/availability checks), Live Tracking (10-30s polling loop with GPS coordinates, ETA calculation), Status Notifications (pickup, arriving soon), and Delivery Completion (timestamp recording, confirmations, rating submission with updates to driver/restaurant ratings).

1.3 Activity Diagram

1.3.1 Order Processing Workflow Activity Diagram

File: `activity_diagram.puml`

This workflow spans five swimlanes (Customer, System, Payment Gateway, Restaurant, Delivery Agent). Flow: Customer browses and adds items, reviews cart, proceeds to checkout. System validates address (stops if not serviceable), calculates total, validates order. Payment Gateway processes transaction (stops if failed, returns transaction ID if successful). System generates Order ID, stores details, sends parallel notifications. Restaurant reviews and either rejects (triggers refund and stops) or accepts (prepares food, marks ready). System assigns delivery agent (reassigns if rejected). Agent collects order, navigates to customer, delivers. For COD, customer pays agent. System updates status, sends confirmations, requests rating. Customer optionally submits feedback.

1.4 Class Diagrams

1.4.1 Main System Class Diagram

File: `class.puml`

This diagram models the complete system with abstract User base class (Customer, DeliveryAgent inherit). Composition shows Restaurant owns Menus, Menu contains MenuItems, Order composed of OrderItems and Payment. Aggregation shows Restaurant partners with DeliveryAgents, users have Addresses. Associations link Customer places Orders, Restaurant receives Orders, OrderItem refers to MenuItem, Order fulfilled by Delivery, rating system connects entities. Enumerations define valid states. Demonstrates OOP principles: encapsulation, inheritance, polymorphism, proper relationships.

1.4.2 User Hierarchy - Generalization/Inheritance

File: `class_user_hierarchy_generalization.puml`

This diagram shows abstract User class with common attributes and abstract methods (getRole(), getPermissions()). Four concrete classes inherit: Customer (loyalty points, orders, tracking), DeliveryAgent (vehicle info, delivery methods), RestaurantManager (restaurant management, reports), Admin (system-level operations). All have aggregation with Address. Demonstrates Template Method pattern and Open/Closed Principle.

1.4.3 Payment Method Hierarchy - Generalization/Inheritance

File: `class_payment_method_hierarchy.puml`

This diagram shows abstract PaymentMethod base class with authorize(), charge(), refund() methods. Five concrete classes: CreditCardPayment, DebitCardPayment, DigitalWalletPayment, CashOnDelivery, BankTransfer, each with specific attributes and methods. Payment uses PaymentMethod (aggregation), Order has Payment (composition). Demonstrates Strategy pattern for runtime payment method selection.

1.4.4 Restaurant-Menu Structure - Composition and Aggregation

File: `class_restaurant_menu_structure.puml`

This diagram contrasts composition and aggregation. Composition: Restaurant owns Menu (1 to 1.*.), Menu contains MenuItem (1 to 0.*.) - strong lifecycle dependency. Aggregation: Restaurant has Address (independent existence), MenuItem uses Ingredients (shared resources). Shows how relationship types capture real-world business rules.

1.4.5 Order Management Structure - Composition Relationships

File: `class_order_management_composition.puml`

This diagram focuses on composition within order management. Order owns OrderItem, Payment, Delivery, Invoice, OrderTracking - all share order's lifecycle and are deleted with it. Contrast: OrderItem refers to MenuItem (association) - MenuItem exists independently in menu catalog. Demonstrates strong ownership vs. reference relationships.

1.5 System Architecture Diagrams

1.5.1 System Context Diagram

File: `context.puml`

This C4 Context view shows the core Online Food Ordering System with three client apps (Customer Web App, Restaurant Portal, Delivery Agent App) and three external services (Payment

Gateway, Mapping/GPS API, Notification Service). Defines system boundary, external actors, primary responsibilities, and integration points at the highest level.

1.5.2 Data Flow Diagram

File: `DFD.puml`

This DFD decomposes the system into eight processes with data stores and flows. Processes: User Management (D1 Users), Browse Menu (D2 Restaurants & Menu), Cart Management (D3 Shopping Carts), Order Processing (D4 Orders), Payments (D5 Payments), Restaurant Orders, Delivery Tracking (D6 Delivery Data), Notifications. Shows data movement between external entities (Customer, Restaurant, Agent, Payment Gateway, Notification Service) through processes and storage.

1.6 State Machine Diagram

1.6.1 Order Status State Diagram

File: `state_machine_diagram.puml`

This diagram models complete order lifecycle with composite states. Flow: Shopping Cart (ItemsInCart, CartModified, CartReviewed) → OrderPlaced (AwaitingPayment, PaymentProcessing, PaymentVerified) → PendingConfirmation → Confirmed → Preparing (InKitchen, QualityCheck, Packaging) → ReadyForPickup → OutForDelivery (AssignedToAgent, EnRouteToRestaurant, PickedUp, EnRouteToCustomer, ArrivedAtLocation) → Delivered → Completed. Includes transitions, guards (timeouts, conditions), actions, and terminal states (Cancelled, Rejected, DeliveryFailed).

1.6.2 State-Stimulus Table

File: `state_stimulus_table.md`

This table documents all state transitions with current state, stimulus/event, guard conditions, actions, and next state. Organized into sections: Main Transitions, Shopping Cart States, Payment States, Confirmation States, Preparation States, Delivery States. Provides complete tabular specification of state machine behavior.