A food order and delivery system should offer a convenient and efficient way for users to explore, order, and receive meals from a wide range of local and international cuisines. It should provide an accessible and easy-to-navigate interface, available on both web and mobile platforms, where users can browse menus, read reviews, and customize orders to their preferences. The system should feature real-time order tracking, estimated delivery times, and notifications to keep users updated from the moment an order is placed until it is delivered. Integration with various payment gateways should ensure secure and flexible payment options. Additionally, the system should support promotional features, loyalty programs, and personalized recommendations based on user preferences and order history, enhancing the overall user experience and encouraging repeat business.

P1: Identify functional and non-functional requirements from the system description.

- Notational: 1-0/14 = 1
  - Functional, Non-Functional
  - User Interface non-functional
- Contextual: 1-2/14 = 0.857
  - 9 Functional: User registration, Menu browsing, User reviews, Order customization, Real-time tracking, Order notifications, Secure payments, Promotions and loyalty, Personalized recommendations,
  - 5 Non-functional: Accessibility, Usability, Performance, Security, Scalability.
  - 2 not explicitly covered: User Registration and Login, Scalability

P2: Create a use case model for the system.

- Notational: 1-0.5/5 = 0.9
  - System Boundary (0.5), Actors (1.5), Use Cases (2), Relationships (1)
  - No system boundary
- Contextual: 1-2/13 = 0.846
  - 13 use cases: Browse Menus, Search Restaurants, View Restaurant Details, Place Order, Customize Order, View Order Status, Make Payment, View Order History, Rate and Review Restaurants, Manage Restaurant Menus and Orders, Manage Delivery Personnel, Track Orders, Manage System Settings
  - 2 not explicitly covered: Manage Restaurant Menus and Orders, Manage Delivery Personnel and Manage System Settings

P3: Create use case specifications for identified use cases.

- Notational: 1-(1*0.3)/6.5 = 0.953
  - Use Case Name (0.5), Actor (1), Precondition (1), Postcondition (1), Main Scenario (2), Alternative Scenario (1)
  - Alternative scenario having no reference to basic flow.
- Contextual: 1-0/4 = 1
  - 4 UC Specs: Browse Menus, Place Order, View Order Status, Make Payment (partial)

P4: Create a domain model based on use case specifications.

- Notational: 1-0/5 = 1
  - Classes (2), Attributes (1), Relationships (1), Multiplicities (1)
  - Must not have Data Types (-0.5), Operations (-0.5), Navigabilities (-0.5), Visibilities (-0.5)
- Contextual: 1-1/7 = 0.857
  - 7 classes: User, Restaurant, Menu, Order, OrderItem, MenuItem, Payment, DeliveryPersonnel
  - 7 relationships: User-Order, Restaurant-Order, Order-User, Order-Restaurant, Menu-Restaurant, OrderItem-Order, OrderItem-MenuItem, Payment-Order, DeliveryPersonnel-Orders
  - DeliveryPersonnel not covered in UC specs.

P5: Identify system operations from use case specifications.

- Notational: 1-0.6/1.6 = 0.625
  - Base Use Case (0.2), Operation Name (1), Parameters (0.2), Return (0.2)
  - No base use case, no parameters, no return.
- Contextual: 1-12/23 = 0.478
  - 23 system operations: login, browseMenus, searchRestaurants, viewRestaurantDetails, placeOrder, customizeOrder, viewOrderStatus, makePayment, viewOrderHistory, rateAndReviewRestaurant, addMenuItems, updateMenu, receiveOrder, updateOrderStatus, confirmOrder, manageRestaurants, manageMenuItems, manageOrders, managePayments, manageDeliveryPersonnel, acceptOrder, updateOrderStatus (Delivery), deliverOrder
  - 12 not covered in UC specs: viewRestaurantDetails, viewOrderHistory, rateAndReviewRestaurant, addMenuItems, updateMenu, manageRestaurants, manageMenuItems, manageOrders, managePayments, manageDeliveryPersonnel, acceptOrder, deliverOrder

P6: Create design sequence diagrams for system operations.

- Notational: 1-(0.2)/4.6 = 0.956
  - Base System Operation (0.2), Participants (1), Operations (1), Parameters (0.2), Return Type (0.2), Sequence of Operation Calls (2)
  - No return type (0.2)
- Contextual: 1-1/6 = 0.833
  - 6 DSDs: Place Order, View Order Status, Make Payment, Update Order Status, Assign Delivery Personnel, Deliver Order
  - Assign Delivery Personnel not covered in system operations.

P7: Create design class diagrams based on the domain model and sequence diagrams.

- Notational: 1-(0.5*0.2+0.2+0.2+1+0.5+0.5)/5.2 = 0.519
  - Classes(1), Attributes with Types(0.5), Operations (1), Parameters with Types(0.2), Return Type (0.2), Relationship(1), Multiplicities(0.5), Navigabilities(0.5), Visibilities(0.3)
  - No data types, no parameters, no return types, no relationships, no multiplicities, no navigabilities
- Contextual: 1-(3*0.4)/5 = 0.76
  - 5 classes: User, Restaurant, Order, Payment, DeliveryPersonnel
  - 0 Relationships
  - placeOrder(), viewOrderStatus(), makePayment() in User, updateOrderStatus() in Restaurant, acceptOrder() in DeliveryPersonnel in consistent with sequence diagrams. These operations defined in System in SDs.

P8: Develop a Java implementation for the system as specified in the class diagram and sequence diagrams.

- Notational: 1-(0.2+0.2+3)/5 = 0.32
  - Classes (0.5), Data types (0.5), Visibilities (0.5), Constructor (0.1), Getters (0.2), Setters (0.2), Methods (3)
  - Missing getters, setters.
  - No method implementation
- Contextual: 1-0/5 =1
  - 5 classes: User, Restaurant, Order, Payment, DeliveryPersonnel

P9: Develop tests including unit tests, integration tests, and system tests for the implementation.

- Notational: 1-(1*0.8+1*0.8+1*0.8)/3 = 0.2
  - Unit tests (1), integration tests (1), system tests (1)
  - Only method declaration. No implementation
  - Unclear classes involved in integration tests and system tests.
- Contextual: 1-0/26 = 1
  - 13 Unit Tests: testLogin, testPlaceOrder, testViewOrderStatus, testMakePayment, testAddMenuItems, testUpdateMenu, testReceiveOrder, testUpdateOrderStatus (Restaurant), testUpdateOrderStatus (Order), testAddOrderItem, testProcessPayment, testAcceptOrder, testUpdateOrderStatus (DeliveryPersonnel)
  - 9 Integration Tests: testPlaceOrder, testUpdateOrderStatus (Order Integration), testMakePayment, testReceiveOrder, testUpdateOrderStatus (Restaurant Integration), testProcessPayment, testAcceptOrder, testUpdateOrderStatus (Delivery Integration)
  - 4 System Tests: testPlaceOrder, testViewOrderStatus, testMakePayment, testOrderDelivery