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: 1-0/18 = 1
  - Functional, Non-Functional
  - User Interface non-functional
- Contextual: 1-7/18 = 0.611
  - 9 Functional: User Registration and Authentication, Browse Menus, Customize Orders, Real-Time Order Tracking, Notifications, Payment Integration, Promotional Features, Loyalty Programs, Personalized Recommendations.
  - 9 Non-functional: Usability, Availability, Performance, Security, Scalability, Reliability, Health and Safety, Localization, Integration.
  - 7 not explicitly coverd: User Registration and Authentication, Availability, Scalability, Reliability, Health and Safety, Localization, Integration

P2: Create a use case model for the system.

- Notational: 1-0/5 = 1
  - System Boundary (0.5), Actors (1.5), Use Cases (2), Relationships (1)
- Contextual: 1-2/8 = 0.75
  - 8 use cases: Browse Menus, Customize Order, Place Order, Track Order, Receive Notifications, Manage Profile, Restaurant Management, Delivery Management
  - Restaurant Management, Delivery Management not covered by requirements.

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: 1-0/8 = 1
  - 8 UC Specs: Browse Menus, Customize Order, Place Order, Track Order, Receive Notifications, Manage Profile, Restaurant Management, Delivery Management

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

- Notational: 1-(0.5+1*0.8)/5 = 0.74
  - Classes (2), Attributes (1), Relationships (1), Multiplicities (1)
  - Must not have Data Types (-0.5), Operations (-0.5), Navigabilities (-0.5), Visibilities (-0.5)
  - Partial relationships, no multiplicities.
- Contextual: 1-0/9 = 1
  - 9 classes: User, Restaurant, Menu Item, Order, Customization Option, Notification, Delivery Personnel, Loyalty Program, System
  - 6 relationships: System connects users, restaurants, menu items, orders, notifications, and delivery personnel

P5: Identify system operations from use case specifications.

- Notational: 1-0/1.6 = 1
  - Base Use Case (0.2), Operation Name (1), Parameters (0.2), Return (0.2)
- Contextual: 1-0/8 = 1
  - 8 system operations: Display restaurant menus and reviews, Modify order details, Process user's order, Provide real-time order status, Send notifications to users, Allow users to update personal information, Manage menu items and process orders, Handle order deliveries

P6: Create design sequence diagrams for system operations.

- Notational: 1-(1*0.2+0.2+0.2)/4.6 = 0.869
  - Base System Operation (0.2), Participants (1), Operations (1), Parameters (0.2), Return Type (0.2), Sequence of Operation Calls (2)
  - No concrete operation name, no parameter, no return
- Contextual: 1-0/8 = 1
  - 8 DSDs: Browse Menus, Customize Order, Place Order, Track Order, Receive Notifications, Manage Profile, Restaurant Management, Delivery Management

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

- Notational: 1-(0.5*0.2+1+0.2+0.2+0.3+0.3*0.1)/5.2 = 0.648
  - 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 operations (Except System class), no parameters, no return types, no multiplicities, partial visibilities, incorrect visibilities ("public" used for IDs).
- Contextual: 1-1/9 = 0.888
  - 9 classes: User, Restaurant, Menu Item, Order, Customization Option, Notification, Delivery Personnel, Loyalty Program, System
  - 15 relationships: User to Order, User to Notification, User to Loyalty Program, Restaurant to MenuItem, Order to MenuItem, Order to Customization Option, Order to Notification, Delivery Personnel to Order, System to User, System to Restaurant, System to MenuItem, System to Order, System to Notification, System to Delivery Personnel, System connects users, restaurants, menu items, orders, notifications, and delivery personnel
  - "connects()" in System class is invalid. Misinterpreting the description of the sequence diagrams.

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

- Notational: 1-(0.1+0.2+0.2+2.9)/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 constructors, getters, setters.
  - No methods except System.
- Contextual: 1-1/9 = 0.888
  - 9 classes: User, Restaurant, MenuItem, Order, Customization Option, Notification, Delivery Personnel, Loyalty Program, System
  - ”placeOrder()” in System not defined in class diagram or sequence diagrams.

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)
  - Showing only examples. Missing concrete tests and implementation
- Contextual: 1-2/5 = 0.6
  - 2 Unit Tests: Increase Loyalty Points, Update Order Status.
  - 2 Integration Tests: Update Menu Availability, Send Order Confirmation Notifications.
  - 2 System Tests: Complete Order Process, Manage User Profile.
  - “Update Menu Availability” not covered in class diagram.
  - “Manage User Profile” not a system test involving no interactions.