Online Pizza Delivery System

**Assignment - 1**

**Prepared by:**

**Dabhi Vadehi**

**Dobariya Hirak**

**Nirmal Deesha**

**Kalathiya Jauthik**

**Patel Persis**

**Shukla Ronak**

**Srivastava Prakhar**

# Contents

1. About Pizza Delivery System.
2. Use Case scenarios with major functionalities.
3. Use Case diagram.
4. UML System Sequence Diagram(SSD)
5. UML Domain Model
6. GRASP Patterns
7. Class Diagram

# 1. In a document write the name and a description of your system (one or two paragraphs)

**Pizza Delivery System:**

As Internet users are increasing, we urge to introduce an online pizza delivery system to improve customer experience and ease the workload of the staff., Transforming a traditional pizza delivery into an modern application which is more reliable, portable and flexible to customers as well as to employees. The purpose of this system is to serve as an online ordering and management system for a pizza parlor. An app that can track, analyze, and observe an order of pizza from each corner of its preparation to get hot fresh ready. It comprises with a customer friendly interface which will be accessible at all times in the business hours. The customers will be allowed to log in, place orders, view orders, update profile, etc. It will also connect users with an interactive menu. The system also serves to ensure speedy and correct order fulfillment and inventory.

Customers should be able to:

* Create/Update their account and contact information.
* Search the Menu and add items in the cart.
* Place an order.
* Track the status of the order on the app.
* Cancel the order.
* Pay for the order.

# 2. Create a github repository name cecs575\_group# and share the link in the document from step1.

<https://github.com/deeshanirmal/cecs575_group9.git>

# 3. Identify four main use case scenarios for your application. Write two of them in a detailed (fully dressed) use case scenario format. The other two can be brief/casual. Also mention if the system has any major non-functional requirements (FURPS+ model)

Following are the main use cases of the Pizza Delivery Application –

* Register/Login
* Menu
* Place Order
* Payment
* Order Progress

|  |  |
| --- | --- |
| **ID** | **UC1** |
| **Name** | **Register** |
| **Actor** | **Customer** |
| **Description** | **Customer creates an account** |
| **Pre-condition** | **Pizza’s app should be installed** |
| **Event flow** | **1.  Customer downloads the app.**  **2.  Customer registration.** |
| **Extension Points** | **None** |
| **Triggers** | **Permission required for GPS.** |
| **Post-condition** | **Customer moves to log in to the portal** |

|  |  |
| --- | --- |
| **ID** | **UC1** |
| **Name** | **Login** |
| **Actor** | **Customer** |
| **Description** | **Login to the portal** |
| **Pre-condition** | **Customer must be registered** |
| **Event flow** | **1.  Customer using credentials for log in**  **2.  Customer sign in** |
| **Extension Points** | **None** |
| **Triggers** | **Permission required for GPS.** |
| **Post-condition** | **Customer moves to order the pizza. Or edit the profile** |

|  |  |
| --- | --- |
| **ID** | **UC1** |
| **Name** | **Customer Profile** |
| **Actor** | **Customer** |
| **Description** | **Update personal details on profile section** |
| **Pre-condition** | **Customer must be registered and signed in** |
| **Event flow** | **1. Customer can edit and update personal details including contact information and address** |
| **Extension Points** | **Adding additional details to the profile like order preferences** |
| **Triggers** | **Permission required for GPS for address** |
| **Post-condition** | **Customer moves menu section to order the pizza.** |

|  |  |
| --- | --- |
| **ID** | **UC2** |
| **Name** | **Menu** |
| **Actor** | **Customer** |
| **Description** | **Customer chooses dishes to order from the menu** |
| **Pre-condition** | **Customer must be registered and signed in and have valid profile to order** |
| **Event flow** | **1. Customers select and add their items to the cart from the menu** |
| **Extension Points** | **None** |
| **Triggers** | **Items must be available at restaurant for the order** |
| **Post-condition** | **Customer moves to payment section for checkout** |

|  |  |
| --- | --- |
| **ID** | **UC3** |
| **Name** | **Place Order** |
| **Actor** | **Customer, Shop owner** |
| **Description** | **Customer creates an order of pizza** |
| **Pre-condition** | **Customer have selected the items** |
| **Event flow** | **1.  Shop Owner monitors inventory from desktop**  **2.  Shop Owner orders necessary stock when there are 5 or lesser items of a given model in inventory**  **3.  Items will be filled in inventory after delivery of new goods and ShopOwner updates the database in the desktop app** |
| **Extension Points** | **Use case 2 “Replenish Stock Item”** |
| **Triggers** | **Items of the specific model go down by 5 in total in both warehouses** |
| **Post-condition** | **Inventory updated with the newly available items** |

|  |  |
| --- | --- |
| **ID** | **UC4** |
| **Name** | **Payment** |
| **Actor** | **Customer, Shop owner** |
| **Description** | **Customer pays for the order** |
| **Pre-condition** | **Customer must have selected items from the menu and restaurant owner has those items available** |
| **Event flow** | **1. Customers applies for offers if available**  **2. Customers selects the payment method**  **3. Customer pays**  **4. Restaurant owner acknowledge the payment** |
| **Extension Points** | **Customer can claim coupons**  **Customer can add additional items like beverages directly from the payment section** |
| **Triggers** | **Customer must have enough balance to place the order** |
| **Post-condition** | **Customer waits for the order to come and check status on order progress page** |

|  |  |
| --- | --- |
| **ID** | **UC5** |
| **Name** | **Order Progress** |
| **Actor** | **Customer, Shop owner** |
| **Description** | **Customer check status of the order** |
| **Pre-condition** | **Order must be placed and acknowledged** |
| **Event flow** | **1. Restaurant owner updates the cooking and delivery progress to the portal**  **2. Customers check updates of the delivery** |
| **Extension Points** | **Approximate delivery time can be seen on the portal** |
| **Triggers** | **None** |
| **Post-condition** | **Ratings for the food and delivery** |

# 4. Draw a UML use case diagram depicting the use cases from step 1 and their relationship.

Diagram

Description automatically generated

# 5. Draw a UML system sequence diagram (SSD) for one use case (the main use case in your application)

Reference: - used below tool for diagram creation

<https://online.visual-paradigm.com/diagrams/solutions/free-sequence-diagram-tool/>

Diagram

Description automatically generated

# 6. Identify the domain objects in your system and draw a UML domain model

Domain Model Objects:

* Account
* Customer
* Event
* Order
* Media Requirements
* Payment
* Bill
* Menu

Domain Model Diagram

Diagram

Description automatically generated

# 7. Using the 9 GRASP patterns, assign responsibilities to your class. Create the corresponding collaboration (three) or sequence diagrams (three). Annotate which GRASP pattern you applied in each scenario by UML comments (note symbol) in the diagrams. You need to apply at least four different GRASP patterns.

GRASP PATTERN –

* INFORMATION EXPERT

The below diagram represents cart as information expert for class beverages since it shows information about how many beverages are used !!

Diagram

Description automatically generated

* LOW COUPLING

Here, Low coupling means objects are more isolated and independent. If something is isolated, we can change it without worrying that we have to change something else.

Here, Cash is dependent on payment but it is independent from order.

Adds Items

Order

CalculateTotal()

makePayment()

Customer

Low Coupling

Payment

makePayment()

Cash

* HIGH COHESION

Class order has very low dependency on class payment as compare to class cart since cart fetches details of total items and sends to payment class !!

Diagram

Description automatically generated

* CREATOR

Here, in this online pizza delivery system, cart aggregates the order attribute as cart is a mandatory step for further initiating all other succeeding processes including order, payments and delivery.

Customer -> Adds -> Pizzas ->

CART

ORDER

* getItemdetails()
* calculate()
* PlaceOrder()

# 8. Create a Design class diagram for your system

Class Diagram :

Diagram, schematic

Description automatically generated