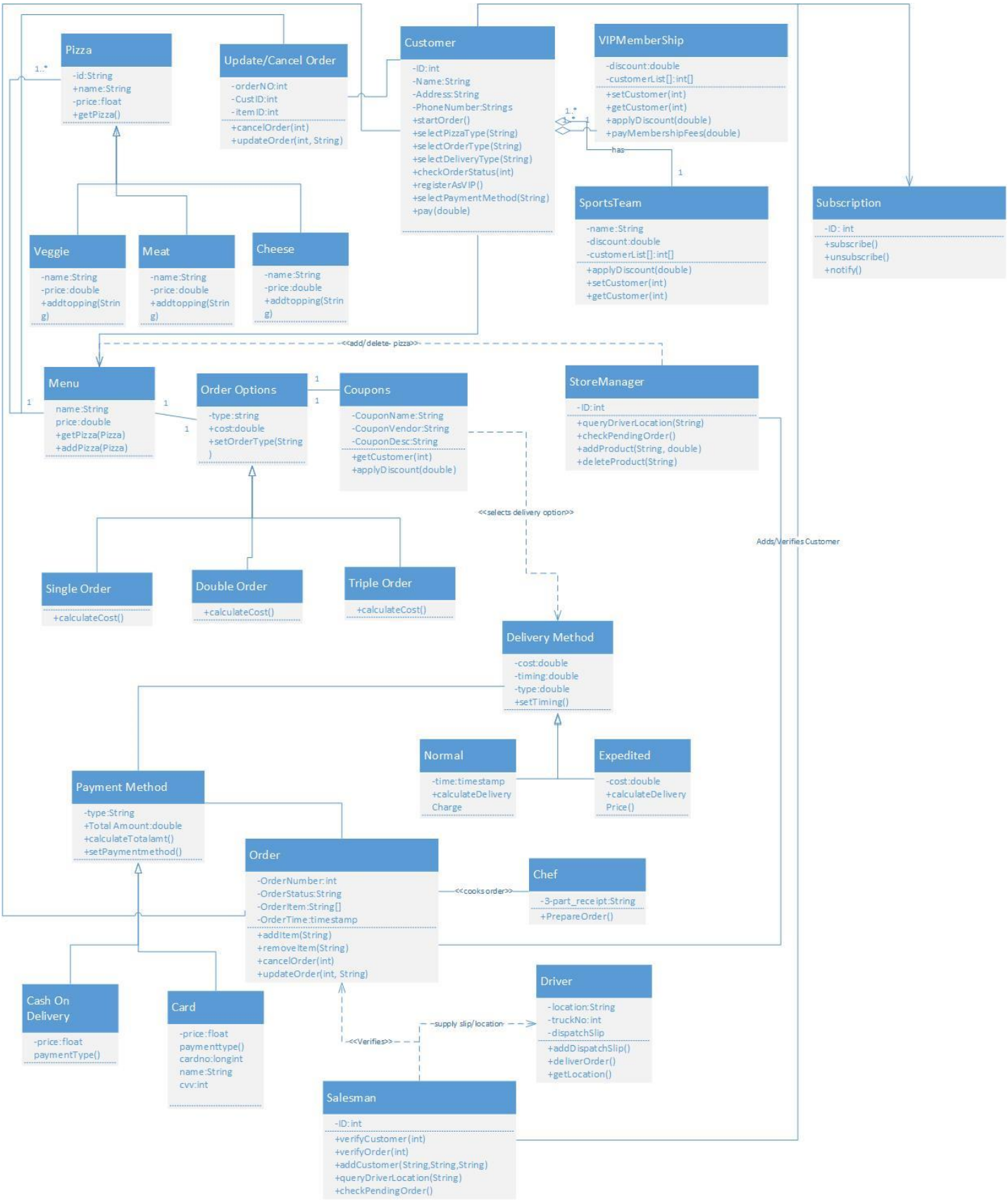
UML - NIKHIL ARORA – A20339503

ASSIGNMENT NO: 3

COMPLETE UML DESIGN MODEL WITHOUT EXPANDING ANY CLASSES

- **Customer**
- **Salesman**
- **Store Manager**
- **Menu**
- **Pizza**
  - **Meat**
  - **Cheese**
  - **Veggie**
- **Order Options**
  - **Single Order**
  - **Double Order**
  - **Triple Order**
- **Coupons**
- **Delivery Method**
  - **Normal**
  - **Expedited**
- **Payment Method**
- **Update / Cancel Order**
- **Order Confirmation**
- **Order Status**
- **Pizza Inventory**
- **Chef**
- **Driver**
- **Print**
- **Verification**
- **VIP Membership**
- **Subscription**

Complete UML Design Model / Class Diagram

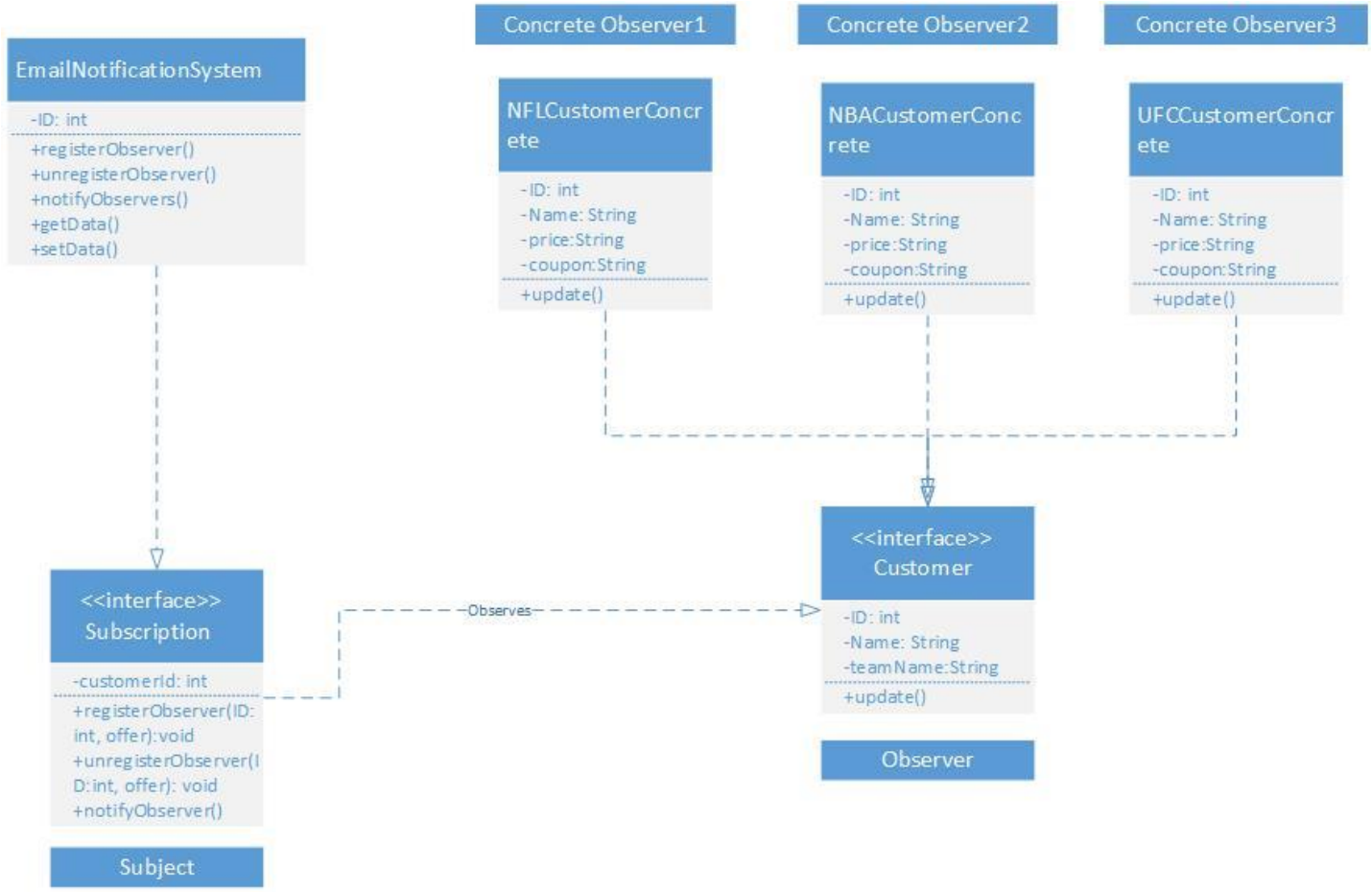


List of design patterns used.

- Observer Pattern: Used on E-mail Subscription module
- Strategy Pattern: Used on Order Option module.
- Factory Pattern: Used on Payment module.

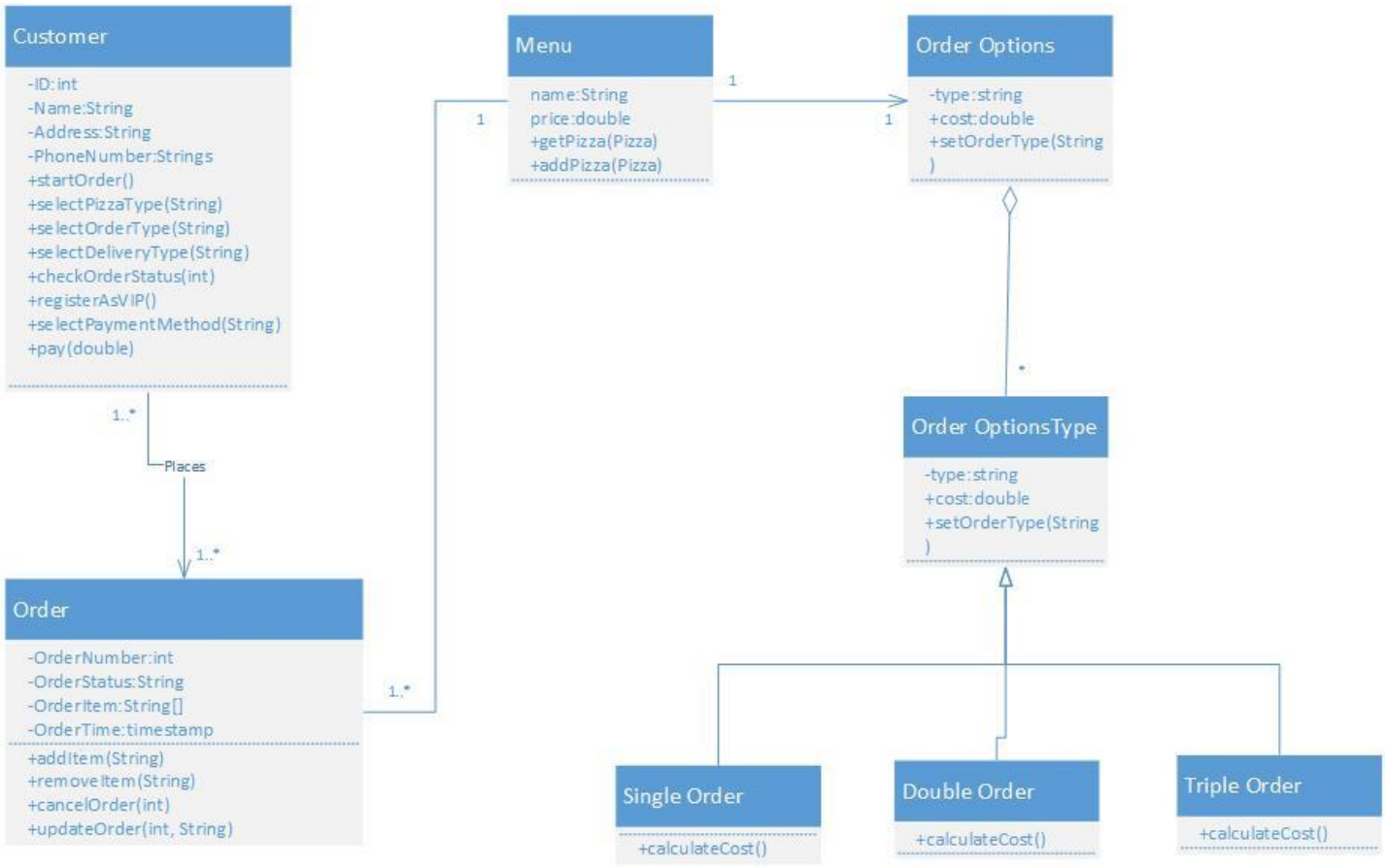
Design Pattern Usage defined below.

1. Observer Pattern:



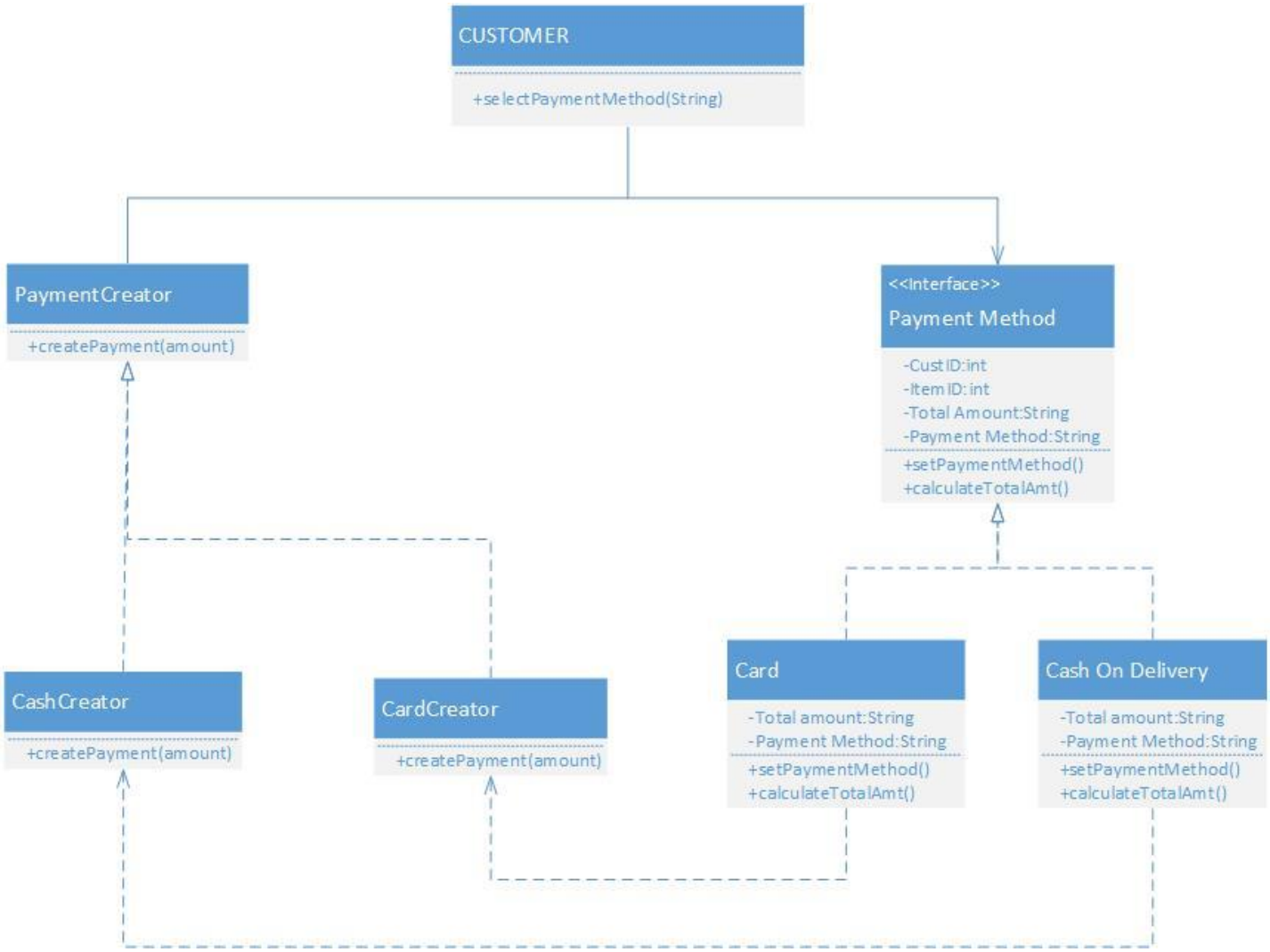
In subscription module observer pattern is used. Here above certain amount, user can subscribe / unsubscribe for email notifications for free pizza delivery. In our case Customer is the observer who use the interface subscription to use that offer. To subscribe / unsubscribe for email notification is being provided be the subscription interface.

2. Strategy Pattern:



Here Customer can choose any of the three kinds of pizza options which help customer to multiple the amount of order based on their choice. I have used strategy pattern to show the proper strategy for choosing the order options which includes single /double/triple the order as per customer need.

3. Factory Pattern:



On Payment module I have implemented factory pattern. Here we have different methods to pay for the order for example payment by card or cash on delivery etc. each payment type has its own creator class connected to them. All the methods are provided by the Payment method which is super class acting as an interface for them. Customer uses the desired method to pay for the order.

NOTE: Here I have expanded all relevant classes related to each design patterns.