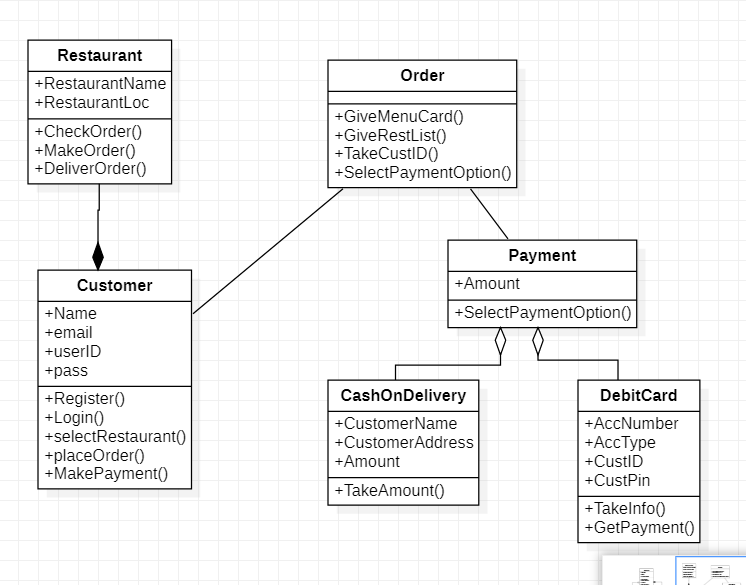
Aggregation and Composition are subsets of association that means they are specific cases of association. In both aggregation and composition object of one class "owns" object of another class. Aggregation means a relationship where the child can exist independently of the parent. Example: Class (parent) and Student (child). Delete the Class and the Students still exist. Composition means a relationship where the child cannot exist independent of the parent. Example: House (parent) and Room (child). Rooms don't exist separate to a House.



**Python Code:**

class CashOnDelivery:  
 def \_\_init\_\_(self):  
 self.CustomerName = None  
 self.CustomerAddress = None  
 self.Amount = None  
  
 def TakeAmount(self, ):  
 pass

class Customer:  
 def \_\_init\_\_(self):  
 self.Name = None  
 self.email = None  
 self.userID = None  
 self.pass = None  
  
 def Register(self, ):  
 pass  
  
 def Login(self, ):  
 pass  
  
 def selectRestaurant(self, ):  
 pass  
  
 def placeOrder(self, ):  
 pass  
  
 def MakePayment(self, ):  
 pass

class DebitCard:  
 def \_\_init\_\_(self):  
 self.AccNumber = None  
 self.AccType = None  
 self.CustID = None  
 self.CustPin = None  
  
 def TakeInfo(self, ):  
 pass  
  
 def GetPayment(self, ):  
 pass

class Order:  
 def \_\_init\_\_(self):  
  
 def GiveMenuCard(self, ):  
 pass  
  
 def GiveRestList(self, ):  
 pass  
  
 def TakeCustID(self, ):  
 pass  
  
 def SelectPaymentOption(self, ):  
 pass

class Payment:  
 def \_\_init\_\_(self):  
 self.Amount = None  
  
 def SelectPaymentOption(self, ):  
 pass

class Restaurant:  
 def \_\_init\_\_(self):  
 self.RestaurantName = None  
 self.RestaurantLoc = None  
  
 def CheckOrder(self, ):  
 pass  
  
 def MakeOrder(self, ):  
 pass  
  
 def DeliverOrder(self, ):  
 pass