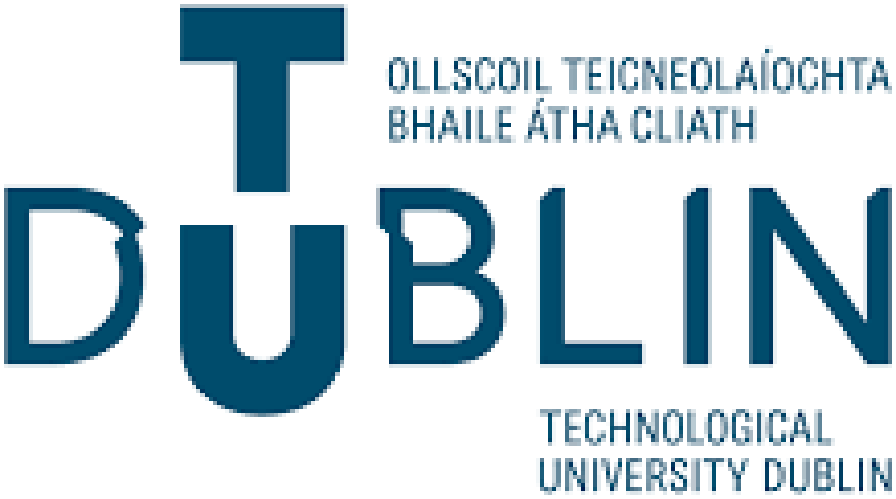
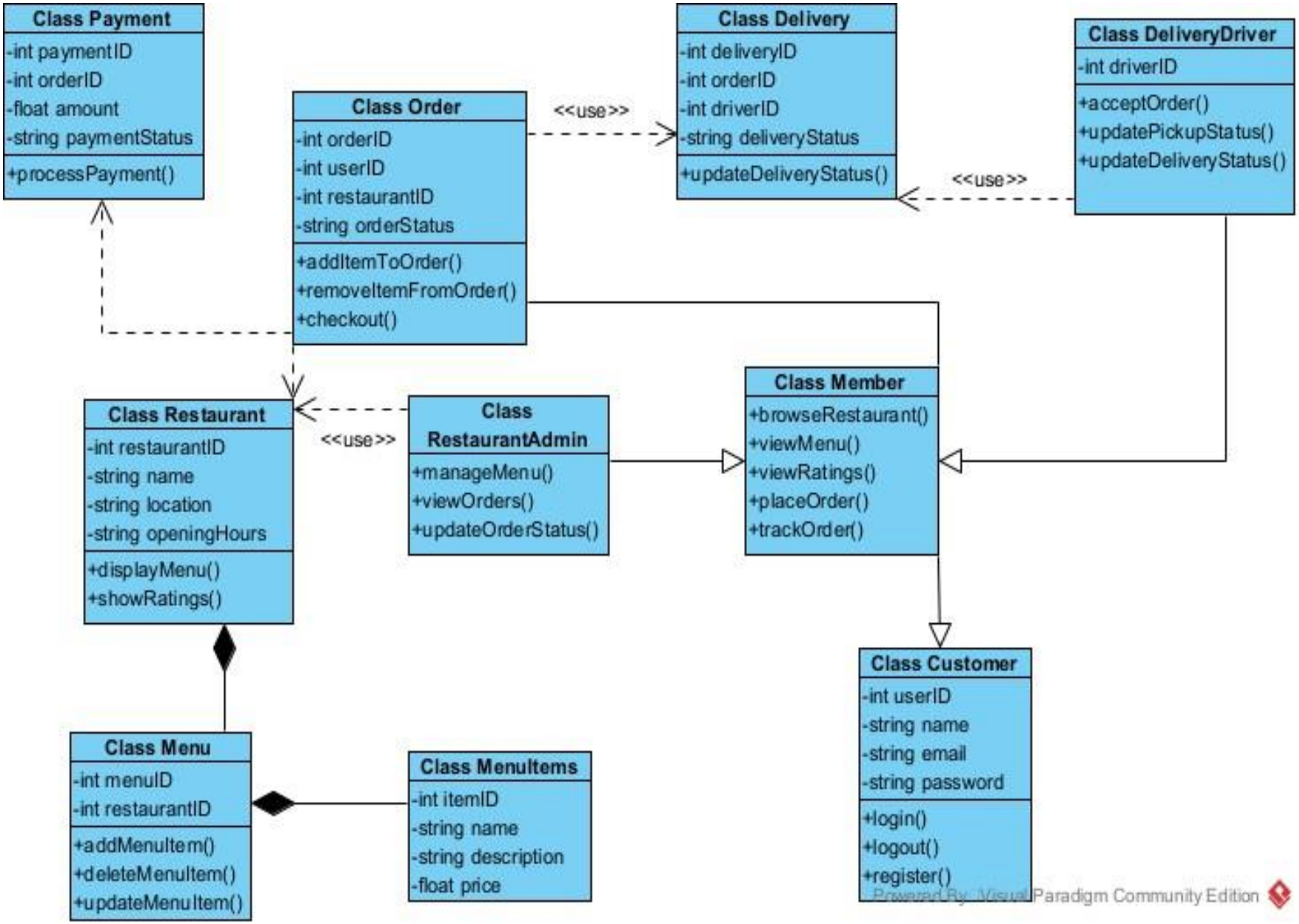


Systems Analysis and Design

Phase 2 Class Diagram

Dieketseng Lebuli
D12238294


Class Payment
-int paymentID
-int orderID
-float amount
-string paymentStatus
+processPayment()
Class Order
-int orderID
-int userID
-int restaurantID
-string orderStatus
+addItemToOrder()
+removeItemFromOrder()
+checkout()
<<use>>
Class Delivery
-int deliveryID
-int orderID
-int driverID
-string deliveryStatus
+updateDeliveryStatus()
<<use>>
Class DeliveryDriver
-int driverID
+acceptOrder()
+updatePickupStatus()
+updateDeliveryStatus()
Class Restaurant
-int restaurantID
-string name
-string location
-string openingHours
+displayMenu()
+showRatings()
<<use>>
Class RestaurantAdmin
+manageMenu()
+viewOrders()
+updateOrderStatus()
Class Member
+browseRestaurant()
+viewMenu()
+viewRatings()
+placeOrder()
+trackOrder()
Class Menu
-int menuID
-int restaurantID
+addMenuItem()
+deleteMenuItem()
+updateMenuItem()
Class MenuItems
-int itemID
-string name
-string description
-float price
Class Customer
-int userID
-string name
-string email
-string password
+login()
+logout()
+register()
Created By Visual Paradigm Community Edition

The Relationship Between Classes

Base Class: Customer class is the abstract represents the base function of the application with common attributes and methods. It is inherited by Member, which is further inherited by DeliveryDriver, and RestaurantAdmin. The application member all to have log in details to use the application (customer, restaurant and delivery driver).

Inheritance classes: Member, DeliveryDriver, and RestaurantAdmin inherit from the Customer Class, they share common attributes and methods such as userID, name, email, password, login(), logout(), and register().

Associated class: Member and Order have an association. A customer can place multiple orders. This is a one-to-many association relationship.

Dependency: An Order and Payment are dependent to each other. An order depends on a payment to be processed. Restaurant and Order are dependent to each other. The order can not be made without a restaurant.

Composition: The Restaurant has composition relationship with Menu. A restaurant has one or more menus. And Menu with MenuItems, a menu contains multiple menu items.

Usage: Order is inclusive of Delivery. An order includes delivery, and one cannot be made without the other. DeliveryDriver is inclusive of Delivery. A delivery driver handles multiple deliveries, and a delivery cannot be made without a delivery driver. Restaurant Admin is inclusive of Restaurant. The restaurant orders cannot be managed without a person managing the orders on the system.

Class Descriptions

Customer Class

Attributes:

- userID: unique identifier.
- name: Name.
- email: email address.
- password: password for the user's account.

Methods:

- login(): method for logging into the application.
- logout(): method for logging out of the application.
- register(): method for registering a new user.

Member Class (inherits from Customer)

Methods:

- browseRestaurants(): method to browse nearby restaurants.
- viewMenu(): method to view the menu of a selected restaurant.
- viewRatings(): method to read ratings and reviews of a restaurant.
- placeOrder(): method the customer to place an order.
- trackOrder(): method the customer to track the status of their order.

Restaurant Class

Attributes:

- restaurantID: unique identifier for the restaurant.
- name: name of the restaurant.
- location: location of the restaurant.
- openingHours: opening hours of the restaurant.

Methods:

- displayMenu(): displays the menu of the restaurant.
- showRatings(): shows the ratings and reviews of the restaurant.

Menu Class

Attributes:

- menuID: unique identifier for the menu.
- restaurantID: unique identifier of the restaurant that owns the menu.

Methods:

addMenuItem(): Adds a new item to the menu.
modifyMenuItem(): Modifies an existing item in the menu.
deleteMenuItem(): Deletes an item from the menu.

MenuItem Class

Attributes:

- itemID: unique identifier for the menu item.
- name: name of the menu item.
- description: description of the menu item.
- price: price of the menu item.

Order Class

Attributes:

- orderID: unique identifier for the order.
- userID: unique identifier of the user placing the order.
- restaurantID: unique identifier of the restaurant from which the order is placed.
- orderStatus: status of the order.

Methods:

- addItemToOrder(): adds an item to the order.
- removeItemFromOrder(): removes an item from the order.
- checkout(): finalizes the order and proceeds to payment.

Payment Class

Attributes:

- paymentID: unique identifier for the payment.
- orderID: unique identifier of the order being paid for.
- amount: amount to be paid.
- paymentStatus: status of the payment.

Methods:

- processPayment(): Processes the payment for the order.

Delivery Class

Attributes:

- deliveryID: unique identifier for the delivery.
- orderID: unique identifier of the order being delivered.

- driverID: unique identifier of the delivery driver.
- deliveryStatus: status of the delivery.

Methods:

- updateDeliveryStatus(): updates the status of the delivery.

DeliveryDriver Class (inherits from Member)

Attributes:

- driverID: unique identifier for the delivery driver.

Methods:

- acceptOrder(): accepts an order for delivery.
- updatePickupStatus(): updates the status when the order is picked up.
- updateDeliveryStatus(): updates the status when the order is delivered.

RestaurantAdmin Class (inherits from Member)

Methods:

- manageMenu(): manages the restaurant's menu items.
- viewOrders(): views the orders placed at the restaurant.
- updateOrderStatus(): updates the status of orders.