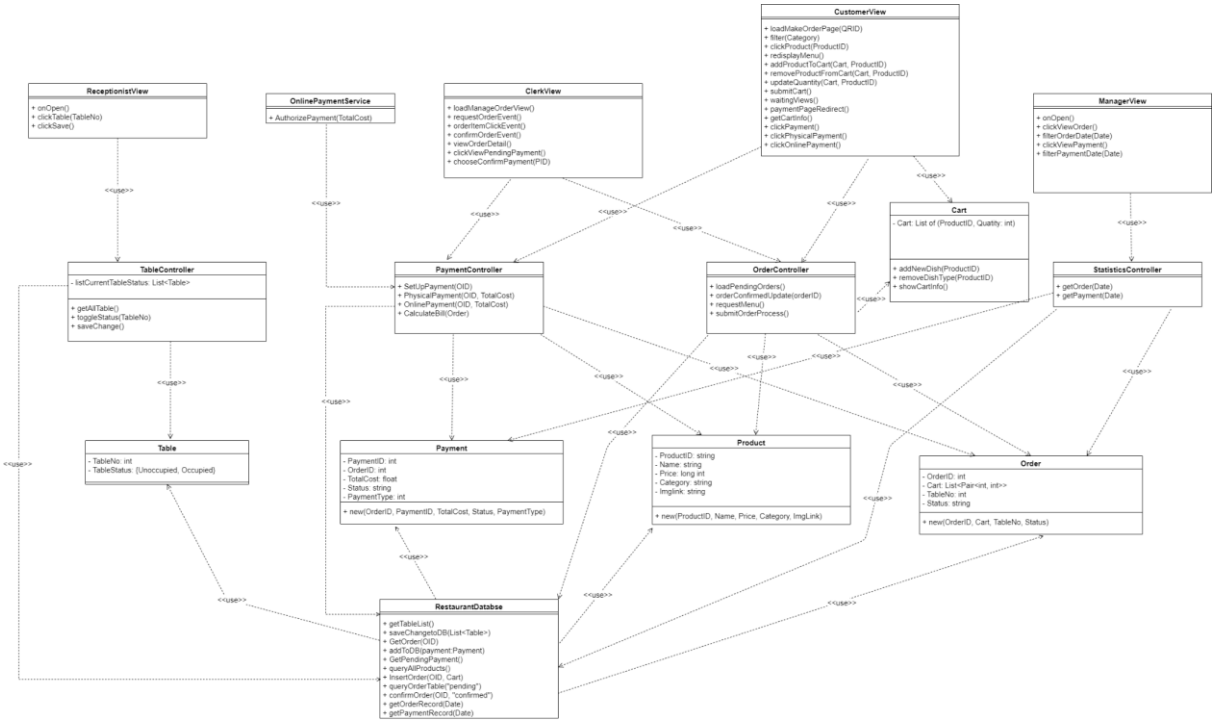
Class Diagram

We have drawn the activity diagram representing the flow of the work inside the system. On the other hand, the sequence was designed specifically for the use case "Food Ordering". Now we establish the class diagram in order to have a closer look at how the system is organized. Below is how the class diagram representing the system:



Due to the amount of <<use>> can cause confusion when reading, below is the summary of each class as well as their relationship to each other:

- ReceptionistView: Receptionist's system UI, in charge of provide receptionist interface to use the Table Management feature
  (Use class TableController)
- ClerkView: Clerk's system UI, in charge of provide clerk interface to view pending orders as well as pending payments
  (Use class OrderController, PaymentController)
- CustomerView: Customer's system UI, in charge of provide customers interface to view restaurant's menu, make order as well as make payment for the order

(Use class OrderController, PaymentController, Cart)

- ManagerView: Restaurant's system UI, in charge of provide manager interface to view restaurant statistic, like the restaurant's order and payment history
  (Use class StatisticController)
- OnlinePaymentService: An interface simulates the Online Payment Service, providing authorization and validation for online payment
  (Use class PaymentController)
- TableController: A class provides functionality that handle service for Table Management
  (Use class Table, Database)
- PaymentController: A class provides functionality that handle service for Payment Management
  (Use class Table, RestaurantDatabase)
- OrderController: A class provides functionality that handle service for Order System
  (Use class Order, Product, RestaurantDatabase, Cart)
- PaymentController: A class provides functionality that handle service for Making Payment
  (Use class Order, Product, Payment, RestaurantDatabase)
- StatisticController: A class provides functionality that handle service for Viewing Statistic
  (Use class Order, Payment, RestaurantDatabase)
- RestaurantDatabase: The class is in charge of query data from restaurant database, passing them as model class as arguments to other classes as well as functioning as an API for other classes to add, remove, update data in database
  (Use class Order, Payment, Table, Product)
- Cart: A model class create when Customers are making their order as a way to store orders before add into the database
- Model class such as Table, Payment, Product, Order

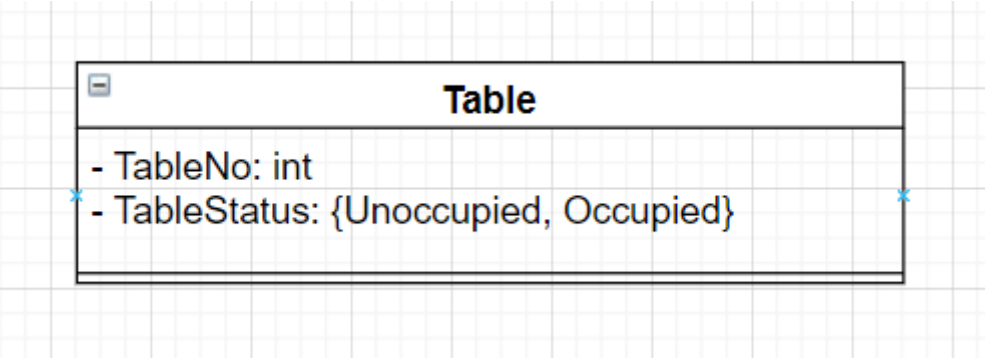
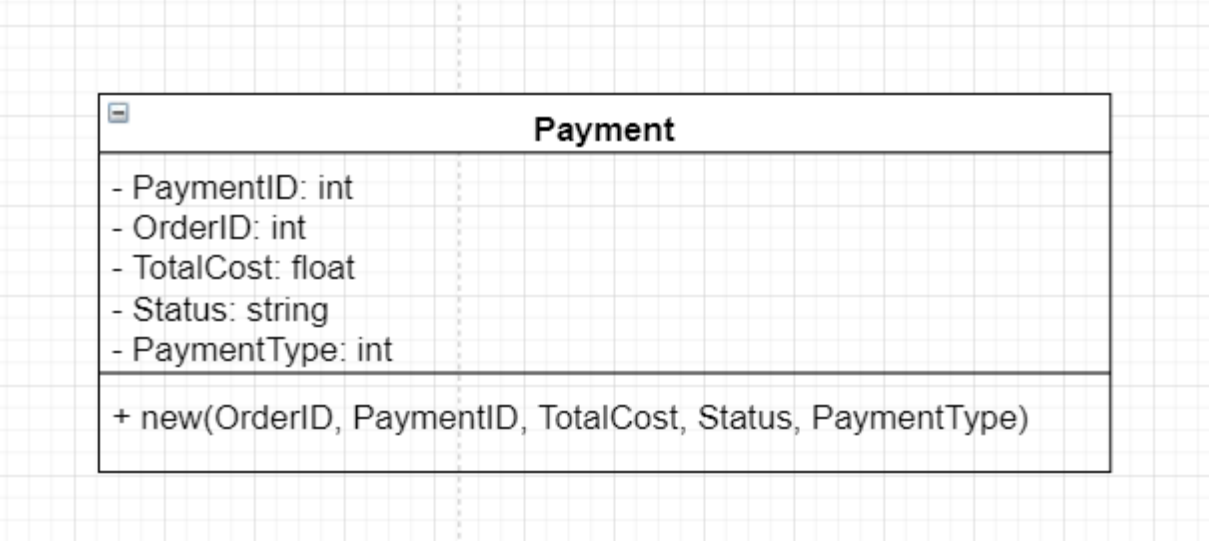
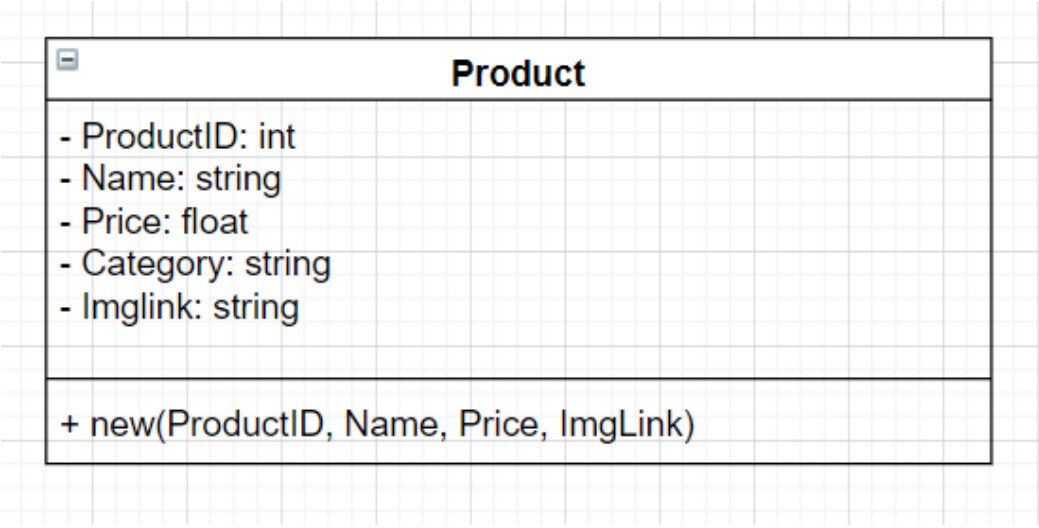


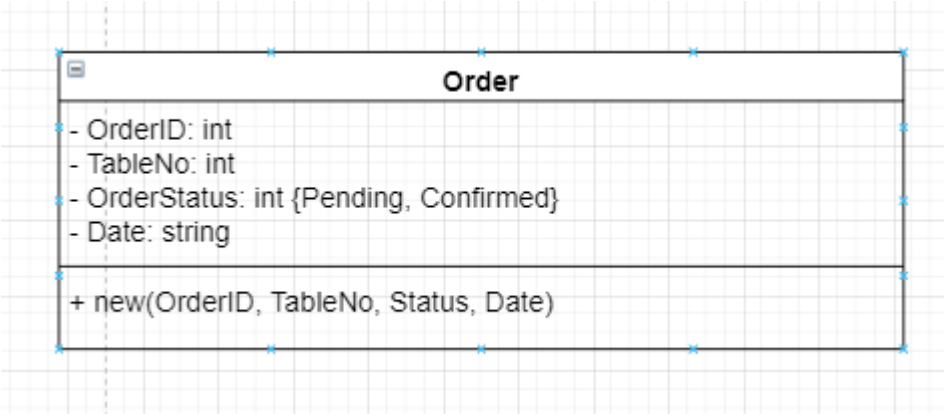
Table Class



Payment Class



Product Class



Order Class

Visit this url to see the diagram better:

https://drive.google.com/file/d/11tW7szjQDztx29I3hLsIQsEodpMTL2ha/view?usp=sharing