Partial Design Document

Group 14
Members : Saurav Suresh, Nwaobi Victor Chibueze, Rajashekar Gudimetla, Venkatadri Kurapati, Milena Tomaszewska, Queen Chinonso Omangima

1. Class Responsibility Collaborator model

A Class Responsibility Collaborator (CRC) model is a collection of standard index cards that have been divided into three sections. A class represents a collection of similar objects, a responsibility is something that a class knows or does, and a collaborator is another class that a class interacts with to fulfill its responsibilities. (Agile Modeling, 2022)

Customer	
Author : Saurav Suresh	
Super Class : User	
Sub Classes : None	
Responsibilities	**Collaborations**
Create Booking	Booking
Create Order	Order
Book an Event	Event
Add Customer ID automatically when a new customer is created	
Add customer name	
Add customer address	

Staff	
Author : Venkatadri Kurapati	
Super Class : User	
Sub Classes : Manager, Waiter, Chef, Driver	
Responsibilities	**Collaborations**
Check Order and change the status of the order	Order
Check Booking and allocate staff based on the booking	Booking
Check the event and allocate the staff based on it	Event
Manage staff(add, delete and update)	
Track and update the working hours of the staff	
Add staff ID automatically when a new customer is created	
Add staff name	
Add staff contact details	

Report	
Author : Milena Tomaszewska	
Super Class : None	
Sub Classes : None	
Responsibilities	**Collaborations**
Generate report with various data based on the requirement	
Get data about most active customer	Customer
Get data on busiest period in the restaurant	Booking
Get data on staff with highest working hour	Staff
Get data on most popular item in the menu	Order

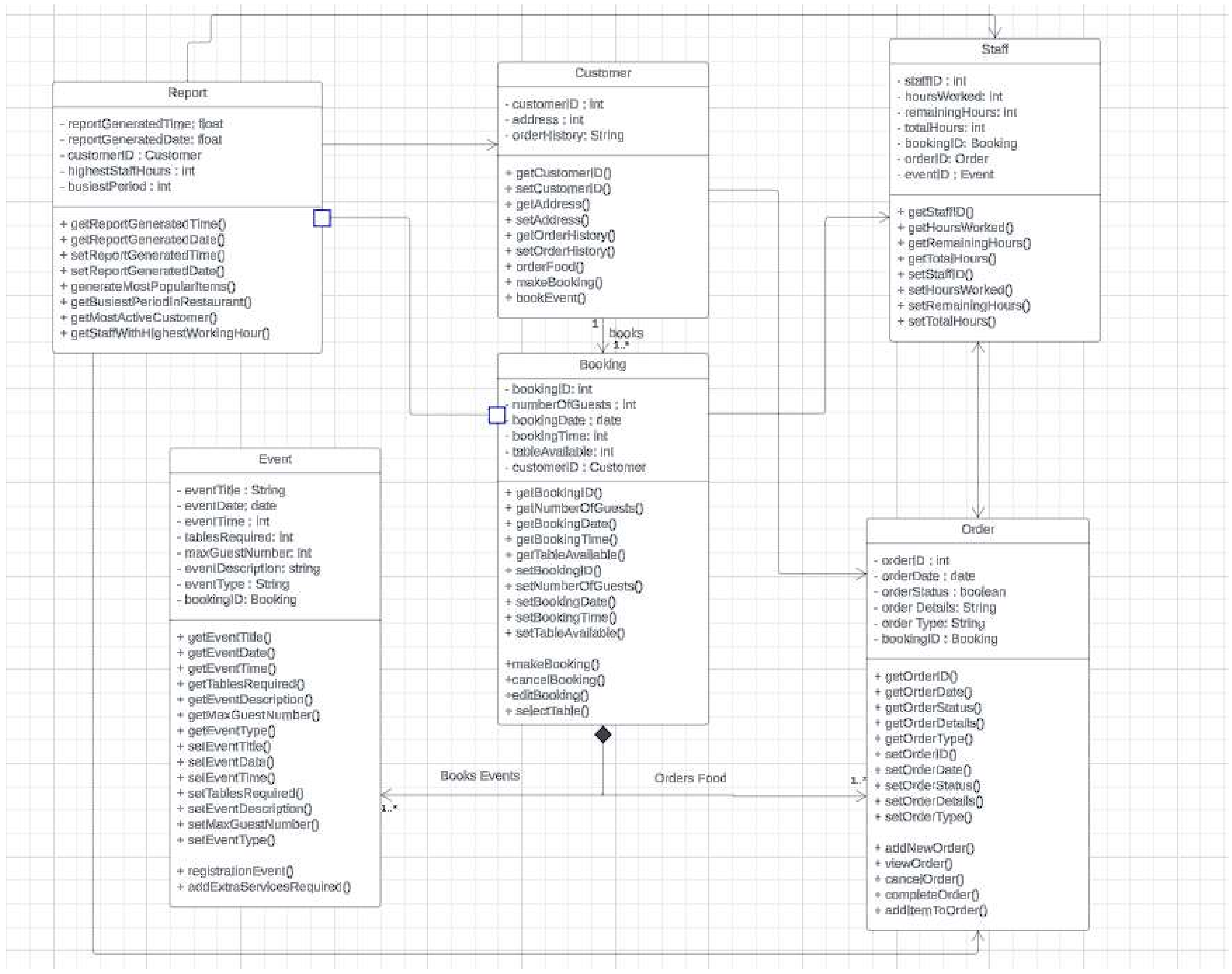
Booking
Author : Nwaobi Victor Chibueze

Super Class : None	
Sub Classes : None	
Responsibilities	**Collaborations**
Taking booking and reservation from customer	Customer
If customer chooses to book event it will collaborate with event class	Event
Allocate the staff based on the booking details	Staff
The customer should be allowed to order food using the booking system through the booking ID created	Order
Allow the customer to add the number of guests and choose the seating type.	
Edit and cancel bookings	

Event	
Author : Rajashekar Gudimetla	
Super Class : None	
Sub Classes : None	
Responsibilities	**Collaborations**
Allow user to register the event	Customer
Booking checks if there is any other bookings or event is present at the time	Booking
Allocates staff to the event based on the requirements	Staff
Add event title	
Add event time	
Add event description	
Add event type	
Allocate table	
Add extra services	

Order	
Author : Queen Chinonso Omangima	
Super Class : None	
Sub Classes : EatIn, TakeAway, Delivery	
Responsibilities	**Collaborations**
Allow customer to order food	Customer
Booking checks if there is any other bookings or event is present at the time	Booking
Send the order details to the staff.	Staff
Staff can alter order based on available items	
Based on the order type(eat in,delivery, takeaway) the different type of staff are selected for completing the order	
Cancel order	
Complete order	

2. UML Diagram



There are 6 main classes in the UML diagram and they are Customer, Staff, Booking, Order, Event, Report and all of the classes have attributes and also methods to get and set them.

Customer: This class represents a customer using the application. The customer class can access methods in the booking class which allows the customer object to create reservations and order food or register for events.The customer class also can directly access the Order class and order takeaway or the home delivery options.

Staff: The staff class can access the order class to get data about the orders made by the customer and various other details like the order type, seat number and also manage the status of the order. The data regarding staff required on a particular day depends on the data it gets from the booking class. It also gets data regarding the events through the booking class.

Booking: The booking class is like a central point and controls a lot of the workflow in the application. Booking receives data from the customer class and it sends the required data to all the classes that have to work. It directly controls the staff class, event class and order class. The booking class has functions to make, cancel and edit reservations and also allows users to select the number of guests and assigns staff based on it. The Booking class has a composition relationship with event and order as booking controls both of the classes.

Order: The order class has details on the orders and various parameters like items ordered, status of the order and the order type. It also sends data to the staff class where this data is used to complete the order. The association between the staff and order is two way as data transfer across the two is required in order to run properly.

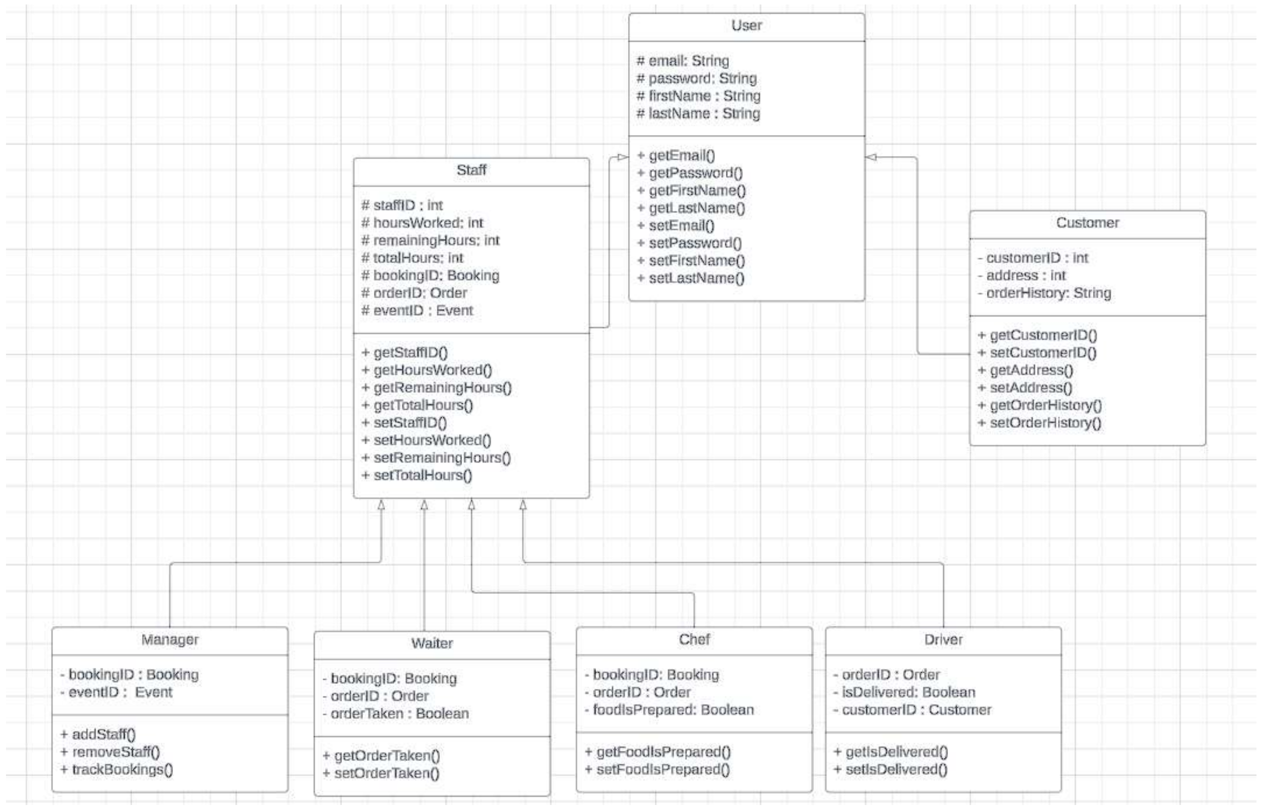
Event: The event class allows the user to book events through the booking class. There will be a check in place in the booking class that makes sure the date the customer is trying to create an event is a day or time without any other bookings. There is a composition relationship between booking and event as the event class is designed in a way that it cannot work without the booking class.

Report: The report class has methods that are used to create reports like most ordered food, most active customer, staff with most working hours and busiest period in the restaurant. It is associated with all the classes except event as it requires data from other classes to generate the report.

Multiplicities

1. 1 Customer can have 1 or more Bookings
2. Each booking can have multiple instances of Events and Orders
3. Each Booking can have multiple Staff instances

3. Hierarchy Description(Inheritance diagram)



In our design User is the superclass that has six subclasses under it. The User superclass only has four attributes being firstName,lastName,password and userName. It is designed like this because all users from customers to all types of staff members will have all these attributes in common. It also has methods to get and set all those attributes.

The User class has two direct subclasses Staff and Customer. The Customer subclass has details needed for the customer like address, customerID and order history which are only needed for the customer. The Customer class also does not have any subclasses. Meanwhile the Staff class has four subclasses based on the type of staff. All the different types of staff have a different functionality and hence in the Staff class the attributes present in the class are the generic ones and are required for all the types of staff.

The four subclasses of Staff class are Manager, Waiter, Chef and Driver. The Manager class will have the admin functionalities and can access all the methods in the whole User class and subsystems and even in the other classes in the application. Manager has the ability to create and delete staff and also update details about them. The remaining 3 subclasses Waiter, Chef and Driver have attributes and methods based on the requirements set on them

4. References

[1] Agile Modeling. (2022). Class Responsibility Collaborator (CRC) Models: An Agile Introduction – The Agile Modeling (AM) Method. https://agilemodeling.com/artifacts/crcmodel.htm