COMPT5047: Applied Software Engineering (Semester 1, 2024-2025)
Coursework Specification Software Engineering of a Modern Computer Application

Name : Mohamed Awaleh
Student Number :19211819

My Subsystem is CloudTables-Customer: a mobile app for running on smart phones for restaurant customers to make table booking, food order, bill payment, and service review and ranking, etc

Task 2: Analysis and Specify Software Quality Requirements (20 Marks)

1. Security and Privacy Protection

Security and Privacy Protections is very important to make sure that customer information is secured especially given the sensitive nature of data such as payment information, personal information such as name and date of birth and in some cases can be utilized to commit crimes such as identity fraud.

Requirements

Separation of Data between Occupants: This means that that the system must make sure data is inaccessible between different restaurant tenants to prevent acts such as unauthorized access from occurring to another restaurants information and data. This can be achieved through multi-tenancy design.

Protection from Security Attacks:

- Implement Encryption for Data in Transit. An Example is TLS 1.3 and at rest AES-256.
- Constant Security Audits and Vulnerability Assessments should be performed.
- Défense Mechanisms against such common attacks should be included as SQL Injections, Malware and DOS (Denial of Service Attacks.

- Multi-Factor Authentications should be implemented for customers accounts to make sure such only users with authorizations can be able to access sensitive features like order payments and personal information.

- Data Backups should be encrypted and constantly stored in a secure environment

Confidentiality, Integrity and Availability

Makes sure that customer data including its booking details and payment details remain confidential and intact and as well as accessible under all operational conditions.

Accountability and Authentication:

Users should have distinct roles such as Admin, Restaurant Staff and Admin with suitable access controls and actions should be logged for auditing purposes

---

2. Performance

Performance ensures that the app responds to user interactions swiftly and efficiently, particularly since it deals with real-time operations such as table booking and order placement.

Requirements:

Low Latency

- The App should maintain a response time of under 2 seconds for most user interactions especially during food order placement and processes.

- Real-Time data updates such as available table status or order tracking should not exceed 1-second latency

Scalable and Efficient Resource Usage

The App Should Consume Minimal System Resources to make sure it performs well across a variety of devices particularly lower-end smartphones.

- Server-Side Optimizations should be utilized to make sure quick data retrieval for frequently accessed features like restaurant menu or order histories.

3. Reliability

Reliability is a measure of how well the system executes its required functions constantly without let-down over time.

Requirements:

Continuous Operations:

- The App Should be operational 24/7 with minimal downtime, aiming for 99.9% uptime excluding scheduled maintenance periods
- A Robust Monitoring system should be in place to detect issues (e.g., server failure) early and trigger automatic recovery procedures

Isolation of Failures

Any Failure in One Part of The System such as Payment Gateway should not affect other parts e.g., Table Booking making sure that customers can still place orders or submit reviews even if one service fails.

The System should implement redundancy e.g. Load Balancers and failover mechanisms to isolate failures and reduce the risk to overall service quality.

Error Handling and Recovery

The App should gracefully handle failures providing users with clear error messages and alternative actions such as retry and customer support.

A Recovery Plan should be in place to make sure that service resumes within an acceptable time frame after failure.

---

4. Scalability

Scalability makes sure that a system can handle increasing numbers of users, transactions and data without degradation in performance.

Requirements:

Handling Big Data

The System must be able to process and store large volumes of data efficiently, particularly customer data, order histories, reviews and menu information's

The Architecture should support distributed databases and scalable cloud storage to accommodate growth

Great Number of Interactive Users

The System should be able to support huge amounts of concurrent users without performance degradation. This includes managing numerous concurrent bookings, orders and payments

Load Balancing should be implemented across servers to make sure even distribution of traffic and prevent overloading any single server

Geographic Scalability

The System should be designed to expand across several geographic locations without issues enabling new restaurants to join the platforms seamlessly

Latency should remain low even as the app is utilized in several regions around the globe. This can be achieved through the use of Content delivery networks (CDNs) and edge computing

Generational Scalability

The System architecture should enable for easy upgrades t incorporate newer technologies making sure their continued performances as hardware and software components evolve

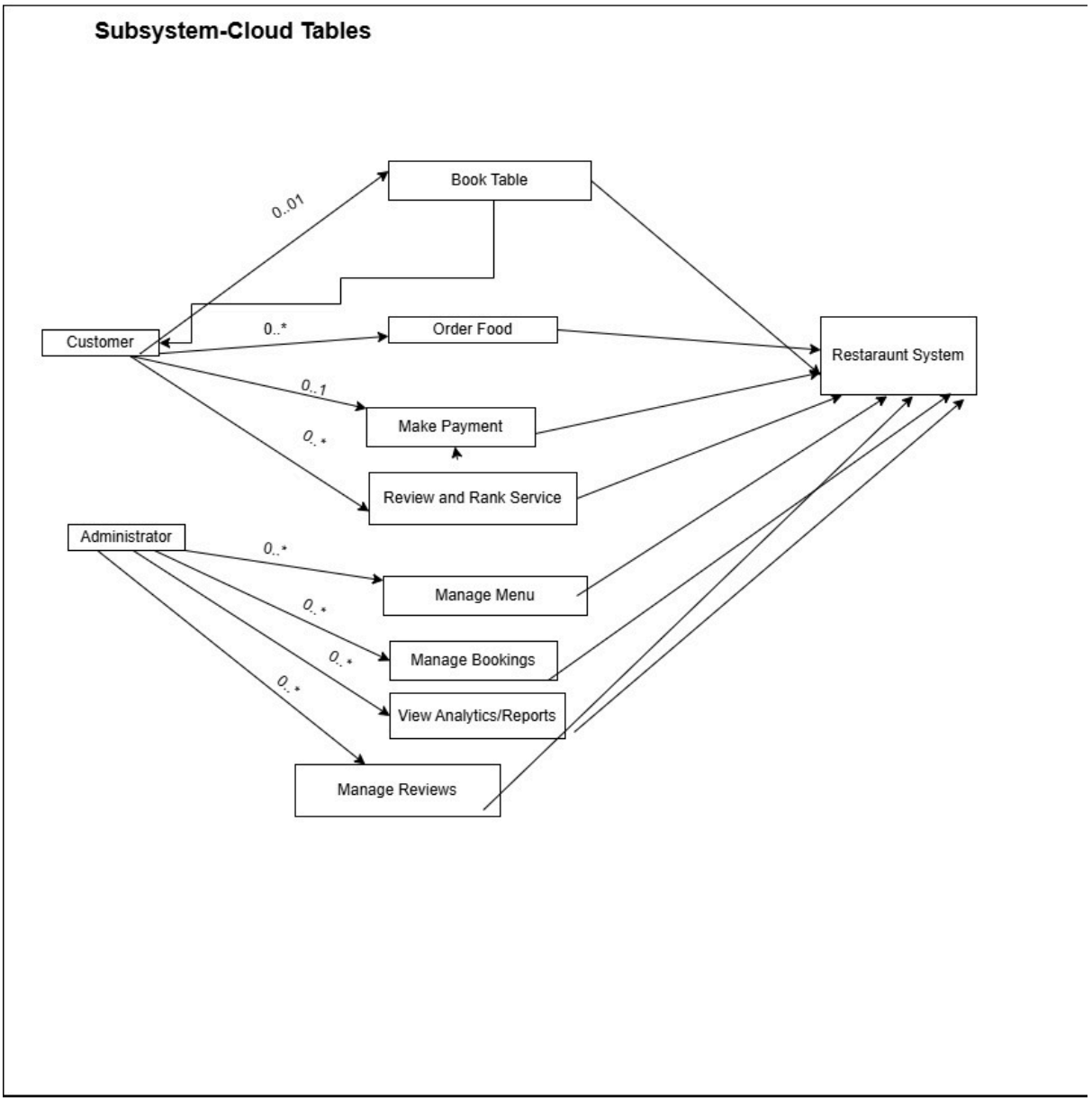
Conclusion

By Focusing on these quality attributes, Cloud Tables-Customer will be able to deliver a secure and reliable with high-performance and a scalable solution that meets the needs of its users and restaurant clients. The Security measures will safeguard sensitive data whilst the performance and reliability requirements will make sure there is a seamless user experience. Scalability guarantees that the system can grow with increasing demand which will make sure of long-term success.

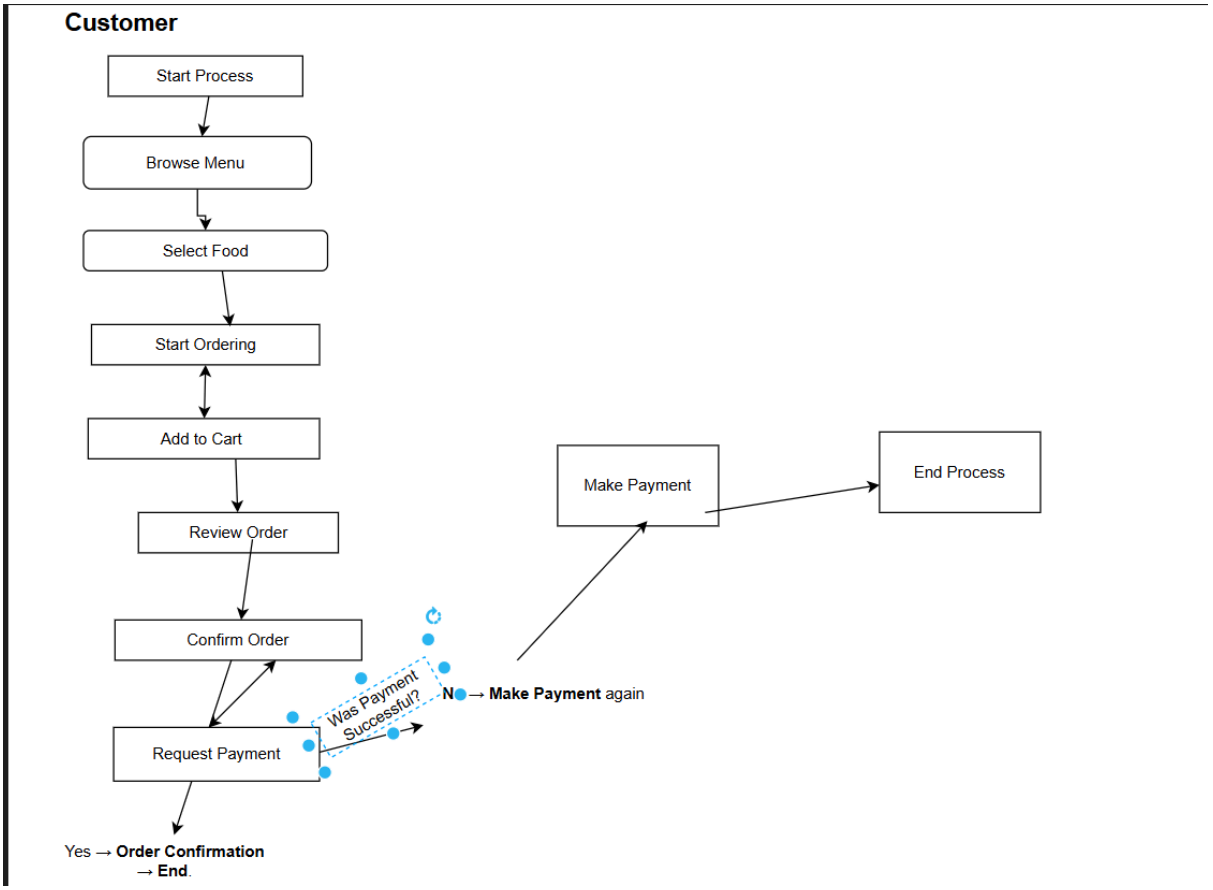
Task 3: Specification and Modelling Software Functional Requirements (20 Marks) In this task, you will work as a requirements analyst to produce a UML model of the software system to be developed using a software modelling tool. The UML model should contain the following types of models.

(a) Use Case Model (10 Marks, Individual effort): Each member of the team should develop one Use Case Diagram to define the use cases of the subsystem to specify the scope of the software engineering project.

(b) Activity Model (10 Marks, Individual effort): Each team member should select one use case of your subsystem to produce one Activity Diagram for the selected use case to specify the interactions between a user and the subsystem



This is The Use Case Diagram for Task 3 A which defines the use cases of the CloudTable-Customer Subsystem

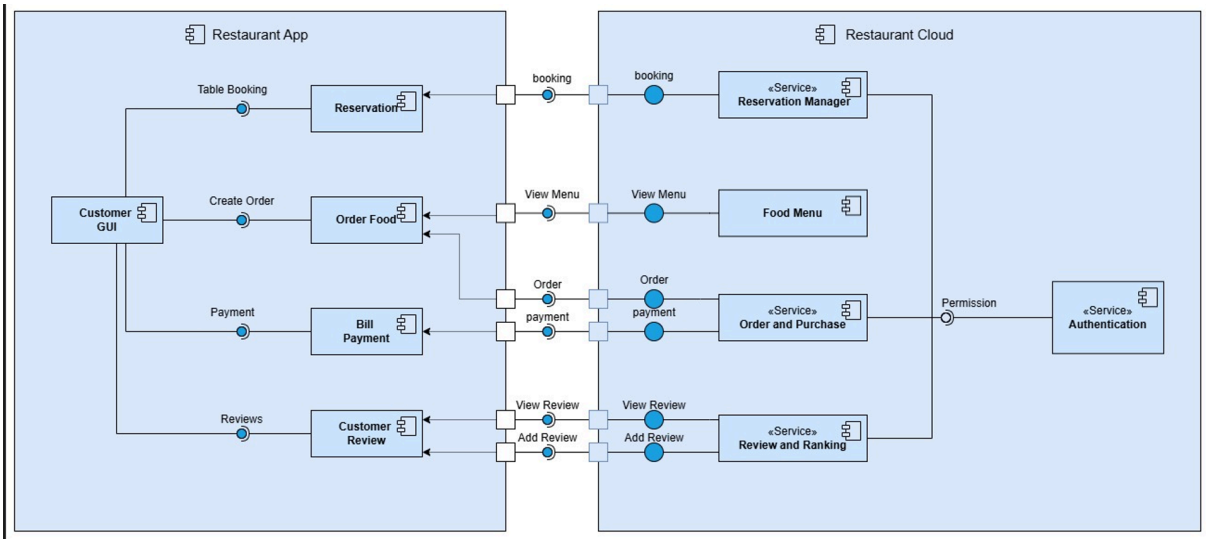


This is the Activity Diagram for the Customer Use-Case of the CloudTables-Customer SubSystem

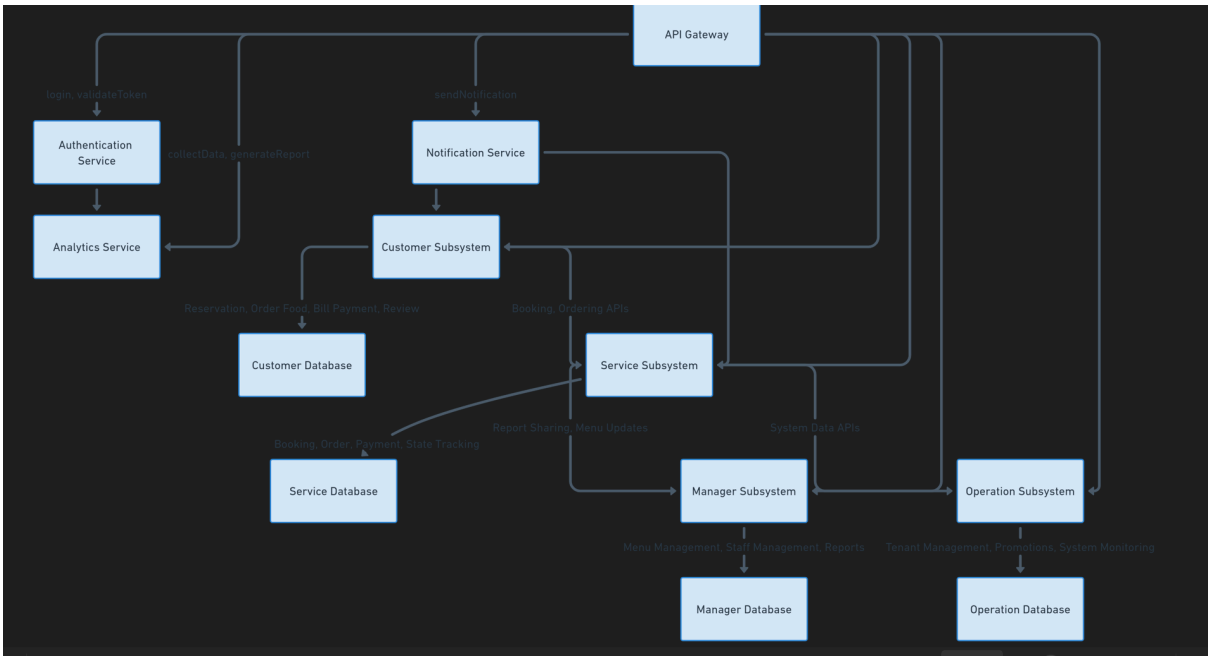
Task 4: Software Architectural Design (20 Marks) In this task, you will work as a software architect to produce an Architectural Design of the system in the microservices architectural style. The design should be at two different levels as follows.

(a) Architecture of the subsystem (10 Marks, Individual effort): Each member of the team should produce an architectural design of your subsystem with focus on the microservices your subsystem provides and the microservices that your subsystem requests and other subsystems provides.

(b) Architecture of the whole system (10 Marks, Team effort): The team should produce an architectural design of the whole system through integrating the subsystems together.



This is Task 4A which is the architecture of my the CloudTables-Customer subsystem that focuses on the microservices the CloudTables-Customer Subsystem and the microservices it requests.

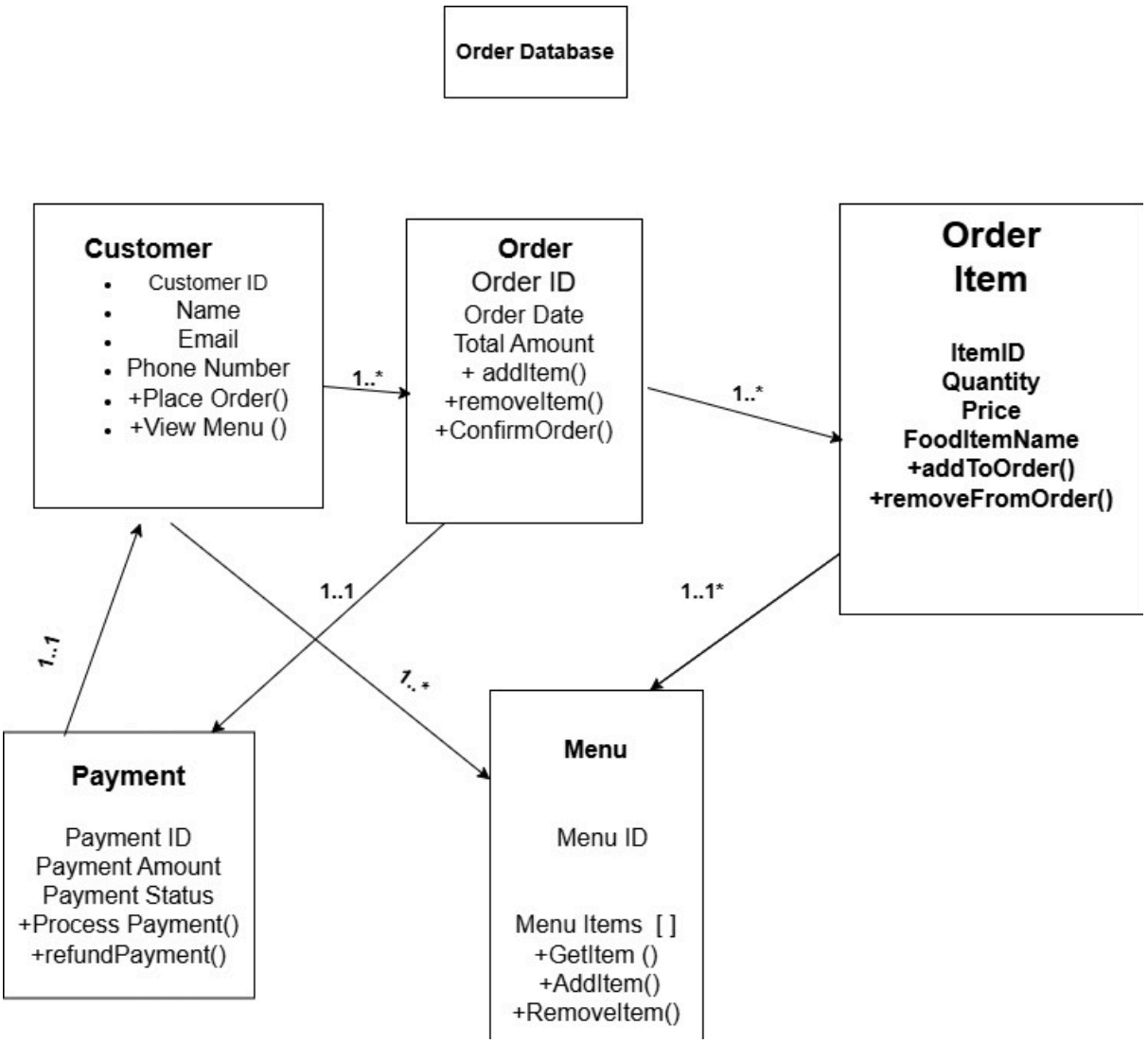


This is Task 4B , an Architectural design of the whole system integrating my subsystem and other subsystems together in my group

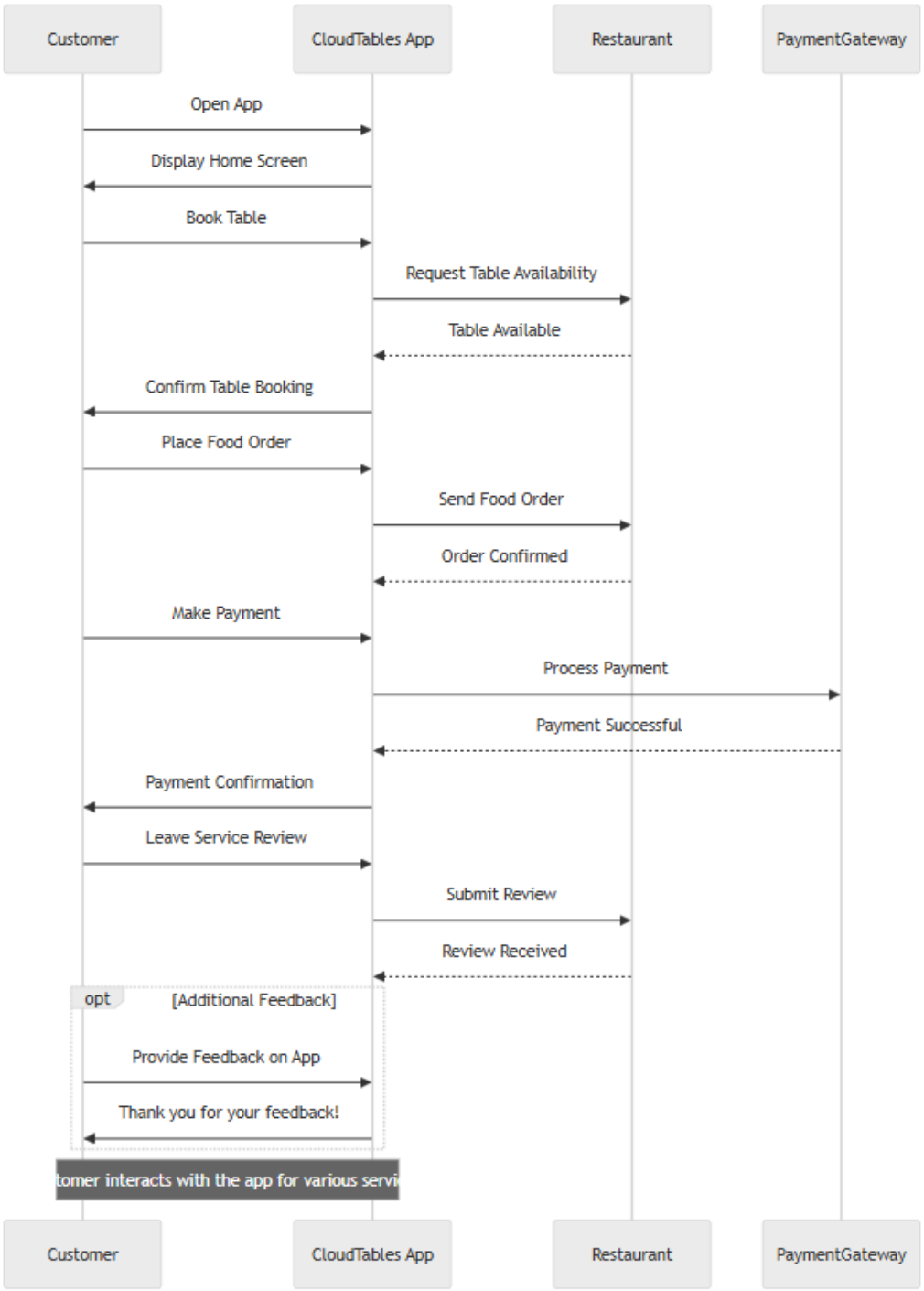
Task 5: Software Detailed Design (20 Marks) In this task, you will work as a software designer to produce a Detailed Design of the system for object-oriented implementation of the components (i.e. the microservices) in your architectural design of the subsystem. The design should specify the components from both structural and behavioural aspects in UML.

(a) Structural Model (10 Marks, Individual effort): Each student should select one component in the architectural design of your subsystem and develop a Class Diagram to define the structure of the component. Note: a. The selected component must be in the architectural model of your subsystem, but not in any other subsystems. b. The UML class diagram must contain the classes and relationships between them. You must give the attributes and methods of the classes.

(b) Behaviour Model (10 Marks, Individual effort): Each student should produce a Sequence Diagram for the same component selected in Task 5(a) to define the dynamic behaviour of the component. Note: a. The sequence diagram should specify the internal process of the interactions between the objects inside the component. It must be consistent with the structural model that you produced in Task 3(a). b. The behaviour model must cover all possible scenarios of the operations of the component using advanced modelling facilities such as frames.



This is Task 5A which is the structural model and the class diagram of the subsystem CloudTables-Customer.



This is Task 5B , the behavioural model and the sequence diagram for the CloudTables-Customer which is my subsystem

Conclusion

I was able to complete all tasks required for my subsystem which is CloudTables-Customer - a mobile app for running on smart phones for restaurant customers to make table booking,

food order, bill payment, and service review and ranking, etc in an effective manner without encountering any issues and was able to address the quality requirements of my software subsystem which is essential for its integration.