1. **Introduction**

**1.1 Goals and objectives**

The overall goals of the software are split between being essential, desirable, and a future requirement. The essential goals are to have a status of the customers per table, and have the ability to display and give a way to process checks for its designated table. The desirable goals are to flag the waiters if the table requires one, and to order food through the system per table. The future goals of the software are to have the ability to make reservations through a website and to reserve specific seating based on customer choice.

**1.2 Statement of scope**

The essentials of the system consist of Table Status and Payment functionalities. These features include the ability for staff to monitor which tables are seated/eating or not, as well as allow customers to pay for their food without the need of a waiter/waitress to bring them their check. These are the priority one functionalities, and we are confident we can implement these by the given deadline. Other functionalities that might be achieved within the deadline include the ability for customers to flag waiters for assistance and order food through the system. These are lower priority, but may be implemented if enough time remains. For future functionalities, a reservation/seating feature is planned. This is the ability for customers to reserve specific tables ahead of time through a website application, as well as be able to see which specific tables are available at which times.

**1.3**  **Software context**

The software will be developed with the optimization of the business in mind. The software will be meant to cater to the needs of the restaurant and nothing else. As it is, there exists inefficiency in the business’ day to day operation. When the software is implemented, the hope is that those inefficiencies will be alleviated.

**1.4 Major constraints**

* Reservation must be only allowed to be made 1 hour prior.
* Restaurants do not accept checks.
* When it comes to accessibility of the software it must be implemented as a website that can be accessed on a tablet that would be on the table .

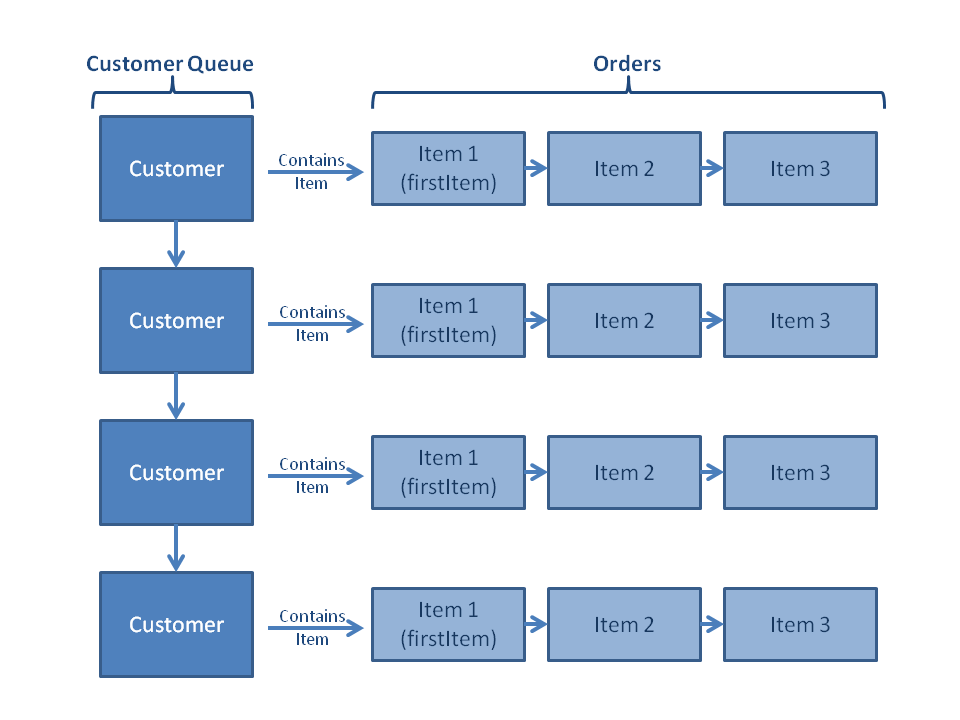
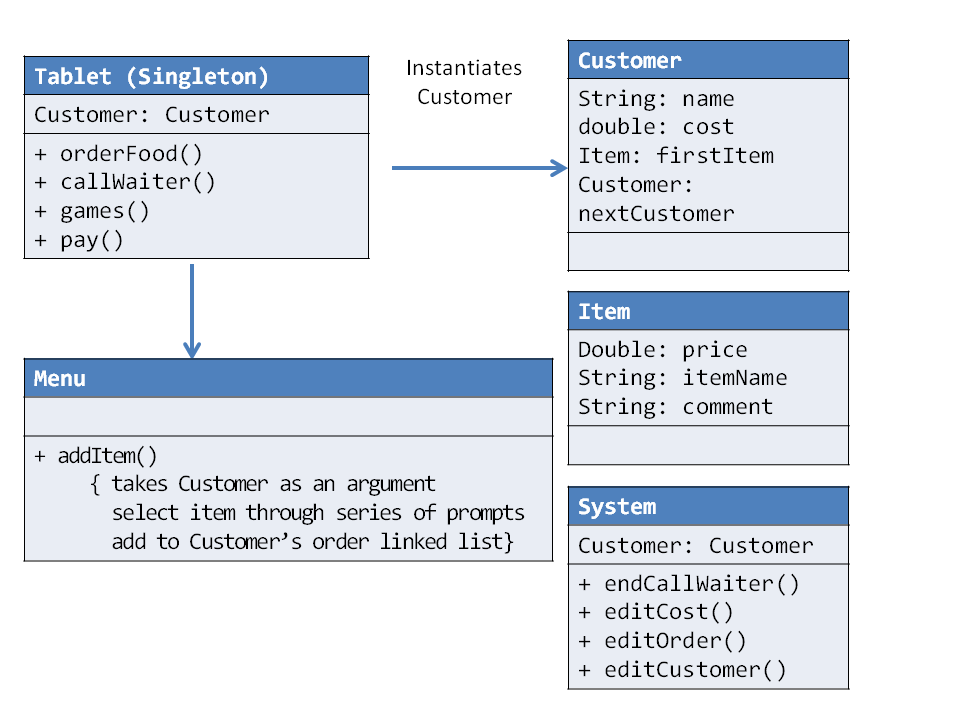
1. **Architecture Overview**
   1. **Architectural Boundaries**
      1. **Hardware design**

The hardware for this system consists of a network of tablets, each connected to the internet in order to communicate with the staff tablet, which oversees the system of tablets. These tablets also place orders, which are shown on screens in the kitchen. A database will also be needed to store all relevant information regarding customers’ orders and table status

* + 1. **Data Design**

The orders of customers will be stored as linked lists with nodes consisting of items, each with its own price name, and optional comment on preparation for the kitchen staff. These linked lists will be displayed on the kitchen screens, and later used to compile an overall check. The status of each table will be updated on each table’s individual tablet and displayed on the staff tablet. Both the linked lists and table statuses will be stored on a seperate database which will be updated through wireless internet communication.

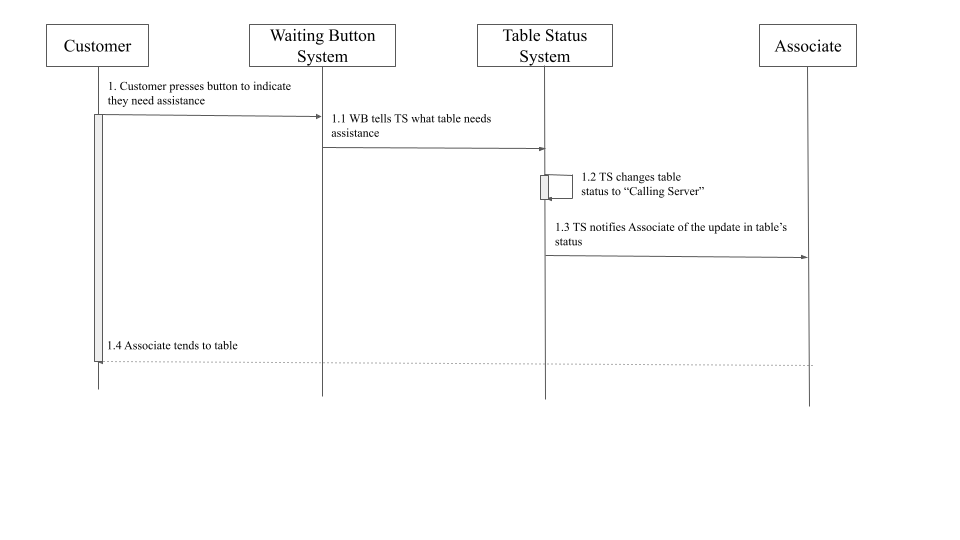
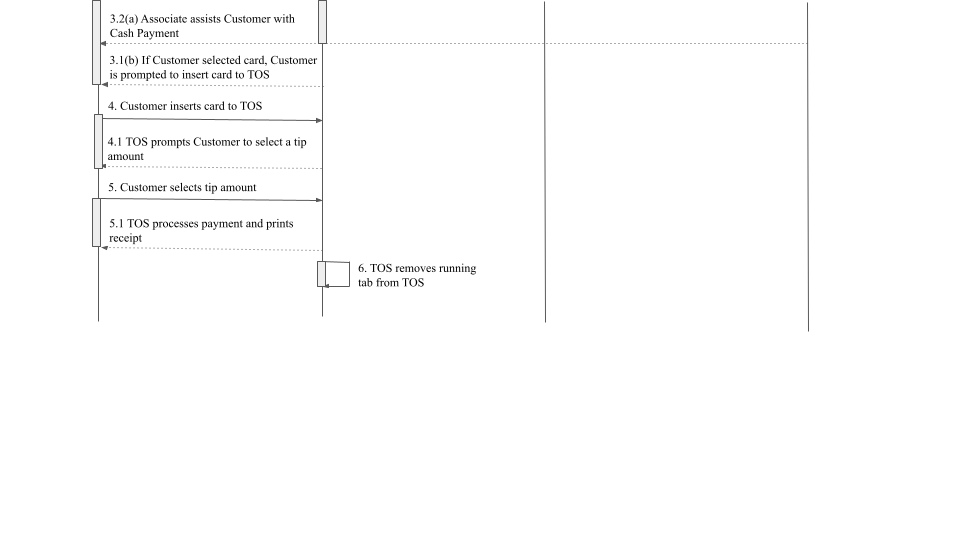
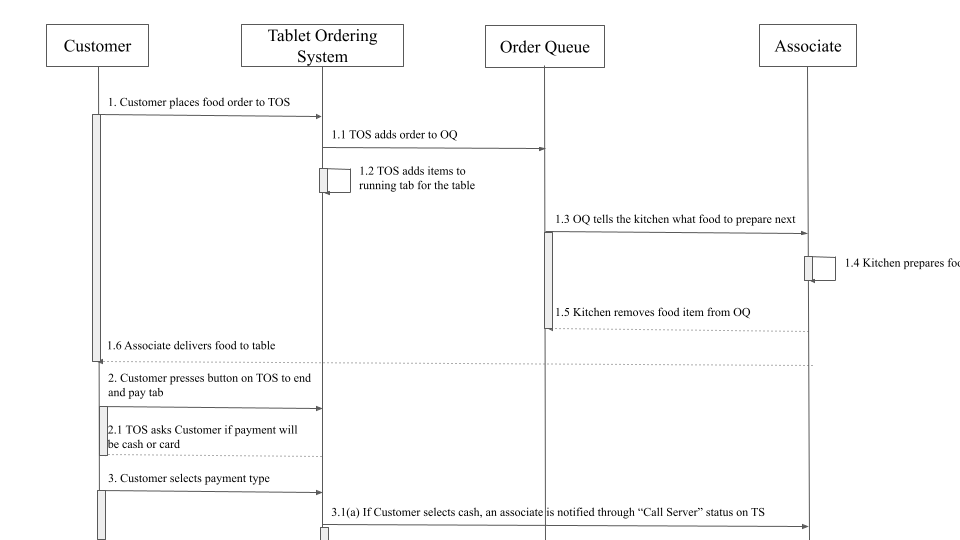
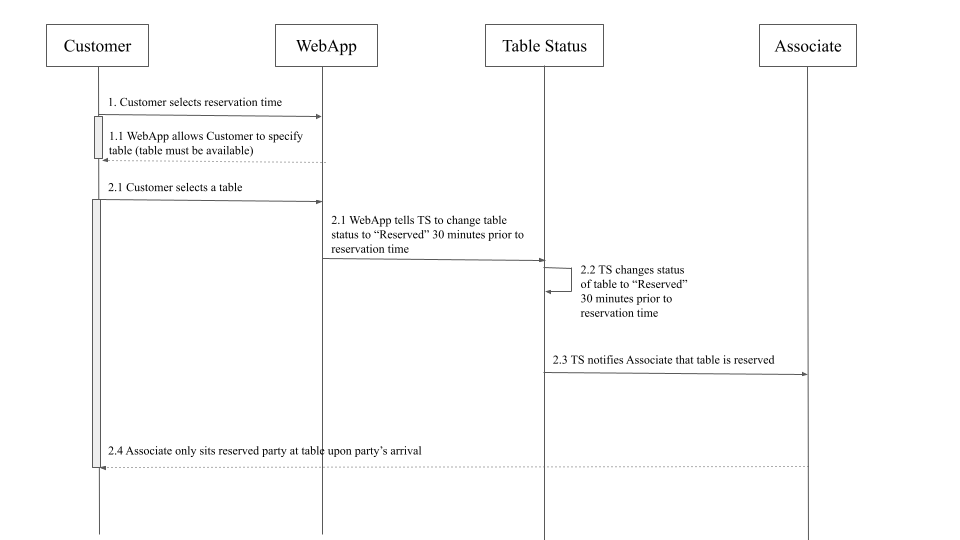
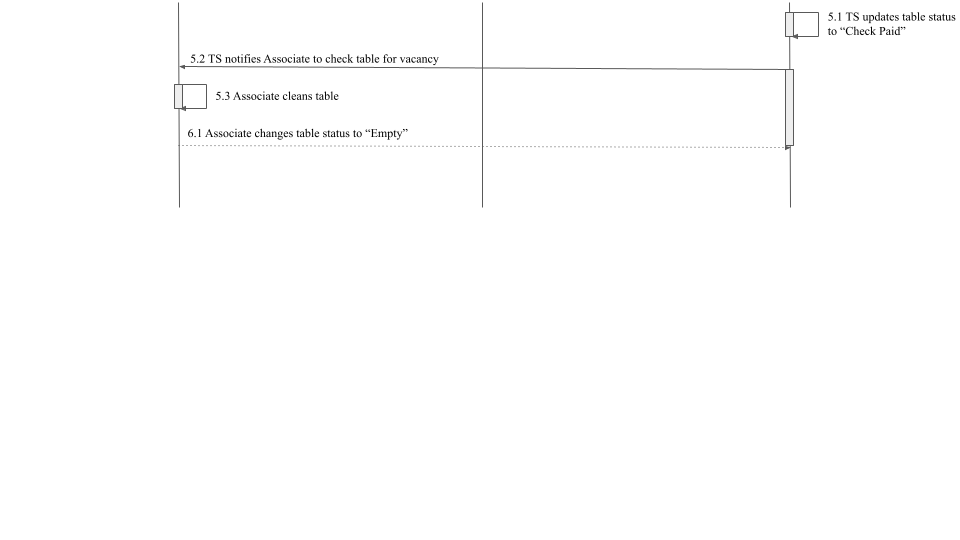
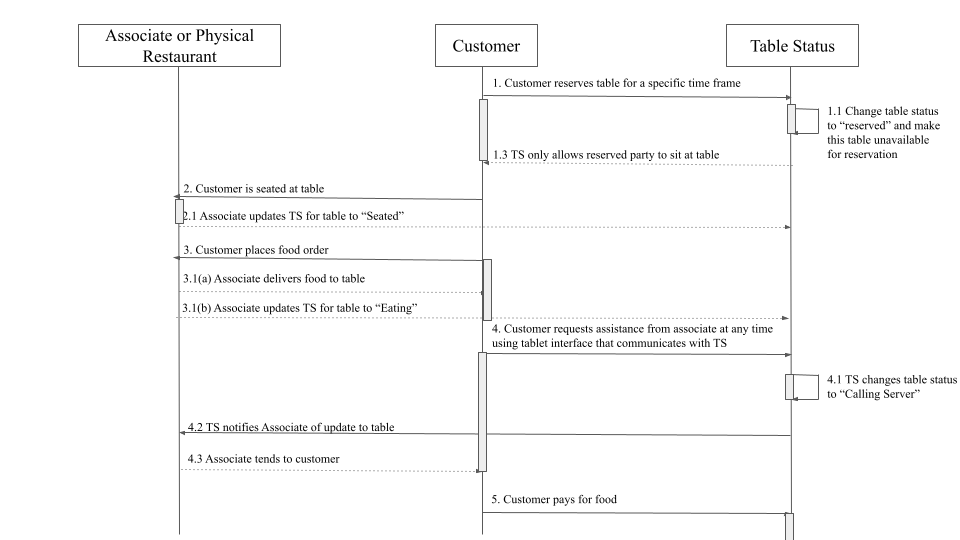
* + 1. **Class Diagram**

****

* 1. **Use Case specification achievement**
     1. **Rationale of how the system meets the use cases from the requirements.**

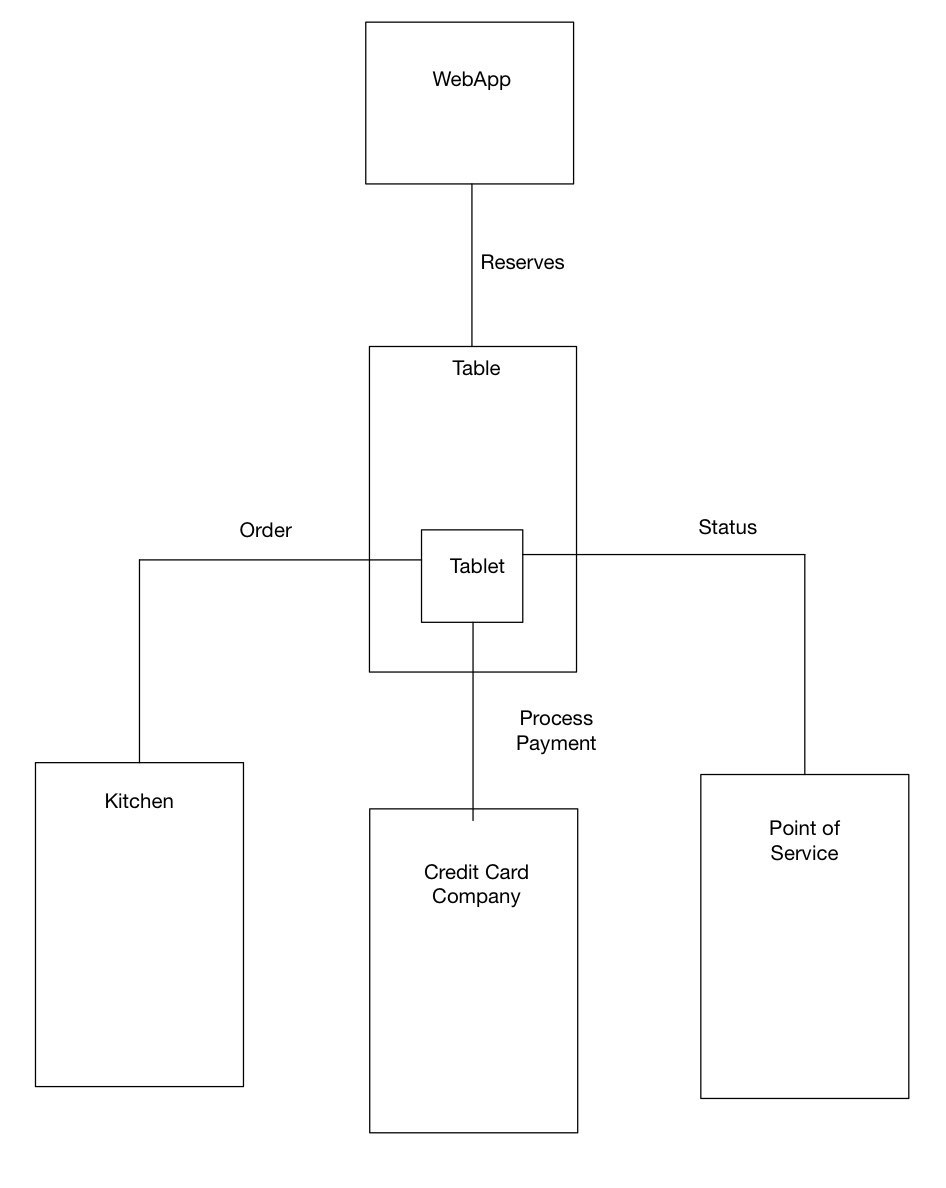
From the diagram below we see that in order to run the restaurant there is a physical restaurant that seats the customer and runs everything as the base. The customer reserves a table and places food orders when they are seated at the table. A table status is changed based on the availability of the table and when the customer is read to order. Which will call the associate/server that works at the restaurant that will tend to the customers. Once a customer has completed eating and is ready to pay the software will have a status called “check paid” allowing the customers to leave afterwards making the table accessible again.

* + 1. **Sequence Diagrams.**



* 1. **Outside system requirements**
     1. For the checking system, we will need to integrate a payment system to the tablets in order to receive payments through the tablet such as bank cards.
     2. For the website, we will need a web server to host our website and its pages. In order to access the website we will need a domain to be able to access it using plain text for ease of use.

1. **Component Design**
   1. **Internal Software Structure**



* 1. **Component Rationale**

Table Status Feature: provides workers with the layout of the restaurant, and a color code status that is automatic but can be overridden by an employee. This will allow for employees to monitor the tables.

Payment: Small tablets will be accessible for patrons on each table. Once the patron is satisfied, they will press a button in order to end their running tab. Patrons will be able to choose their method of payment. The system would present a slider allowing for the user to tip as fit. If the payment is through a card, the data would be processed through a financial service. Once the transaction is complete a receipt would be printed upon the patrons request.

Order Feature: Patrons will order through a tablet. The tablet would list out the menu and Patrons would choose what to order. The order would be sent to the kitchen queue, and the price would be added towards the running tab.

Flag Waiter: The waiting feature provides a platform for patrons to interact with their waiter. The patron would call for the waiter through a notification system within the tablet. The table status is then updated to “calling waiter” until a waiter tends to the patrons request.

Reservation and Seating Feature: The reservation and seating feature is a way to remotely reserve tables. The patron would pick a reservation time via a WebApp which would display all the available tables. Once the table is selected the table’s status would be updated to “reserved” 15 min ahead of the selected time. An associate then prepares for the customers’ arrival.

* 1. **Design Patterns**

Adapter Pattern: For the Flag Waiter and reservation and seating feature will be using this pattern because it deals with status. When there is a table status “Calling waiter” the waiter will come to the table. When a table is open for reservation the web app will show that it is available to reserve out, by calling the interface that would be running the software that deals with the waiter and reservation feature.

Command Pattern: When the customer clicks on “Order Now!” the webapp will display the menu items, the call waiter and kitchen queue. This command will activate all the features that will be one the webapp.

Proxy Design pattern: In order to grab the menu items, the software will implement a database that will contain all the items on the menu with the price. The software will then call the database which will display the items to the customer. The webApp would also be hosted on a server that would be called using a proxy using https which would also be implemented in the software.

Module: have different methods that each have unique jobs to complete. We will have different methods to accomplish certain tasks in order to have easier readability of the program.

Front Controller: a controller that handles requests for a website. Reservations are going to be made on a website. This will handle multiple people accessing the website and the website being able to handle all the clients.

1. **Restrictions, Limitations, and Constraints**

As of right now, no specific design choice restricts the design of the software.