San Francisco State University

CSC648/848 - Software Engineering Section #1

LAST CALL

Team 07

Team Members: Luis Acuna Mendez, Jooho Chang, Gwangwoo Lee, Leslie Mora Ponce, Dorrie Shen, Vedang Sakxena

Milestone 2

October 11, 2023

Table of Contents

1. **Data Definitions V2**

Revision from Milestone 1

Primary Data Name	Definition (& Examples)	Usage
Customer	A registered user located in one location eg: a customer intending to use application services.	Each user will be allowed to create an account for personalised experience while using the application using their email address.
Restaurant	A registered business or food vendor that has surplus unsold food items which they're looking to sell online.	Each Restaurant is allowed to create an account in the application, using the services, they can add available food items (left-over) which is to be displayed to the user account.
Owner	The application developer, monitoring and ensuring proper functioning of the web-application.	Owner is the only account that can add/remove a customer or restaurant account. It also controls the penalty system.
Food	This will be the extra food from the day that was not sold, but it is still too good to go.	This will be sold on the website for a much cheaper price, food items will be placed in the menu as well as their quantity which will help the user's

		know if there are any more available items.
Base Point Payment	This poitns will work just like money, when the payment is done with them this will be subtracted from the user's account	The user can buy food items by points that will be rewarded by the restaurant every time an item is picked up according to the time frame according to the reservation.
Reservation	Each reservation has a unique identification number associated with one customer. Reserved orders can contain multiple food items from the same restaurant.	Reservation system is implied because surplus unsold food will always be in limited quantity. Therefore, customers are allowed to reserve any food items listed by the restaurant.
Penalty System	A minimum penalty implied on users on their next order.	Penalty implied in cases where customers fail to pick up their food items after reserving them through the application.
Location	A location represents a particular region where customers can see nearby listed	Often customers prefer to look for options when it comes to food. Therefore, the location feature allows

	restaurants and can order food items from the same.	the user to access different restaurants from the same location.
Review	Every restaurant menu will be followed by a brief feedback shared by previous customers.	It is important for users to order food from high rated restaurants and their food items. The review feature allows them to pick out highest rated food chains, based on their food quality inspected by another customer.
Open Status	This status is a symbol to notify whether the restaurant has stock of unsold food items.	In cases, where restaurants and food vendors fail to provide surplus unsold food, they can notify that their services will be closed for the day.

Data	**Sub-Data**
Restaurant	● Id ● Location ● Phone ● Cuisine ● Status ● Role ● email
Customer	● Id ● Firstname/Lastname ● Username ● Password ● Role ● Phone ● Email

	• Point base payment
Reservation	• Id • Restaurant_id • Food_id • Customer_id • date
Review	• Review_id • Customer_id • Restaurant_id • Description • Rate
Menu	• Menu_id • Name • Price • Quantity • Restaurant_id
Favorite	• Favorite_Id • Customer_id • Restaurant_id

2. Functional Requirements V2

Must Have
Desired
Opportunistic

Initial List of Functional Requirements

Search Bar

- The users shall search the restaurants by restaurants' name.
- The users shall search the restaurants by area.
- The users shall search the restaurants by food category.

Log In

- The users shall log in using their account.
- The users shall log in by their other platforms such as google, facebook, instagram.

Sign Up

- The users shall create an account.

Log Out

- The users shall log out their account.

List up Restaurant

- The users shall see all the restaurant lists.

Order History

- The Users shall check a record of their past orders.

Restaurant Food Stocks.

- The users shall add their food stock.
- The users shall check other restaurants' stocks.

Open Now

- The users shall check the restaurant's name that is open now.
- The users shall use this point to get a discount for their next purchase.

Favorite Restaurant List

- The users shall add restaurants as their favorite.
- The users shall see all the restaurants that they mark as their favorite.

Checkout / Reservation

- The users shall checkout or reserve the food.
- The users shall make an order after they checkout.

Code for Reservation order

- The users shall get 4-digits code after confirming their checkout.
- The users shall use this code to identify their specific order.

Show How much money we saved.

- The users shall check the amount of money that users have saved by using our platform.

Review

- The users shall create reviews about their order.
- The users shall rate the restaurants based on their order experience.
- The users shall check other users' reviews.

Point Promotion

- The users shall get 5% of point after they purchase any food

Log-In Authentication

- The users shall use their account without login while cookies exist.

Penalty System

- The users shall get a penalty 10 points off when canceling a reservation without notice.

Find Restaurants around the User

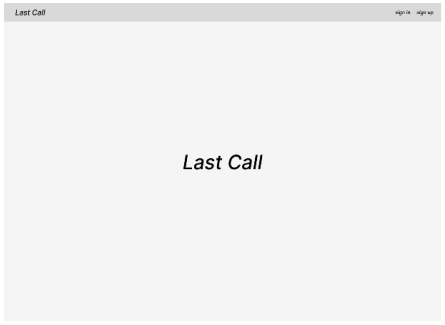
- This function displays a list of restaurants based on customers' current location on the restaurants page.

Suggest Restaurant

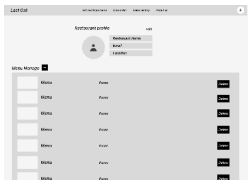
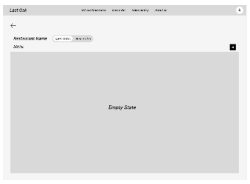
- This function displays a list of restaurants based on popularity.

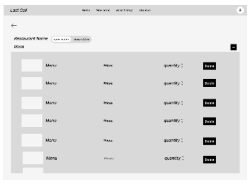
3. UI Mockups and UX Flows

3.1. Sign up, Sign in pages

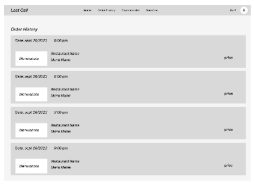
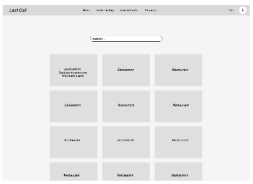


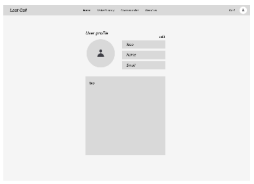
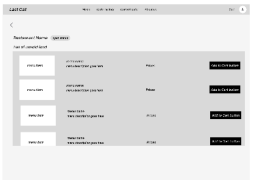
3.2. Restaurant - profile, add menu pages

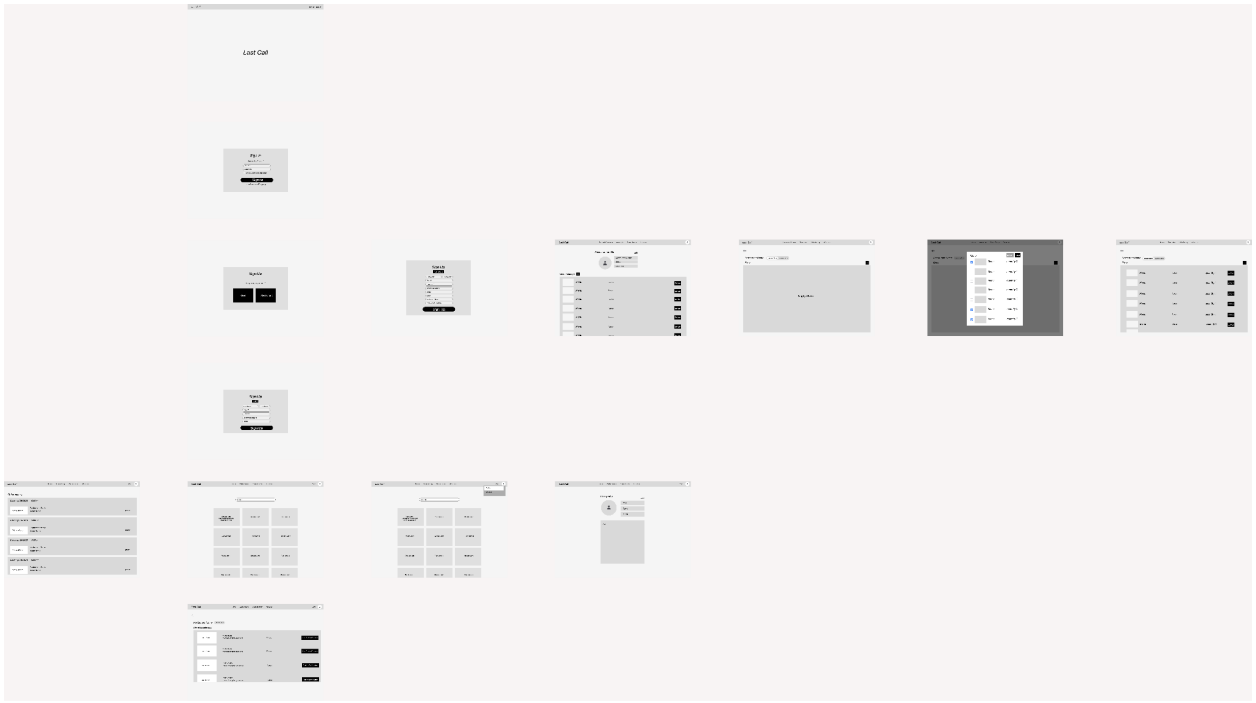


3.3. User - Order history, search, restaurant lists, profile, and current menu

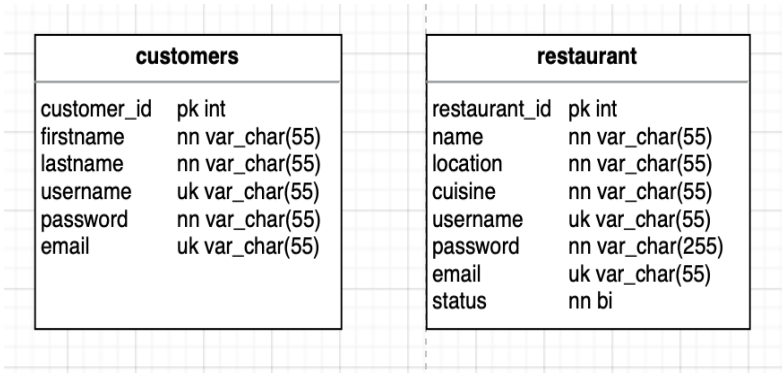
3.4. UX Flows

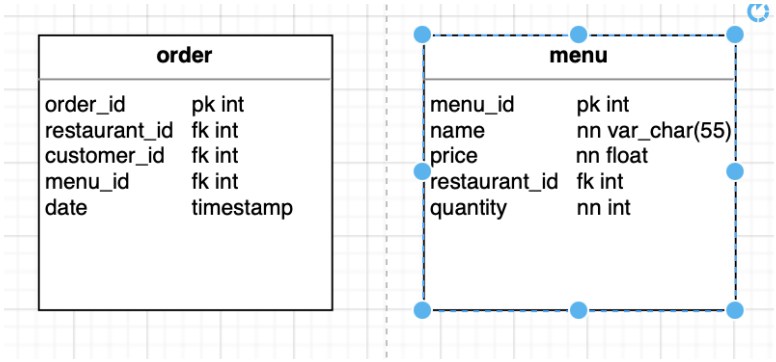


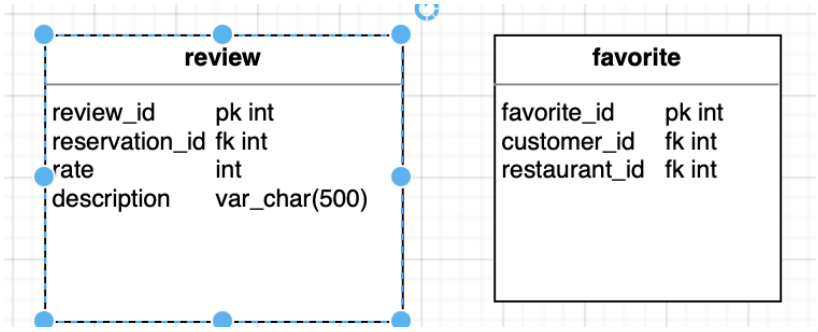
4. **High Level Architecture, Database Organization (back-end lead)**

Overall description of the database: main DB tables along with explanations including back-end API used.

Userr

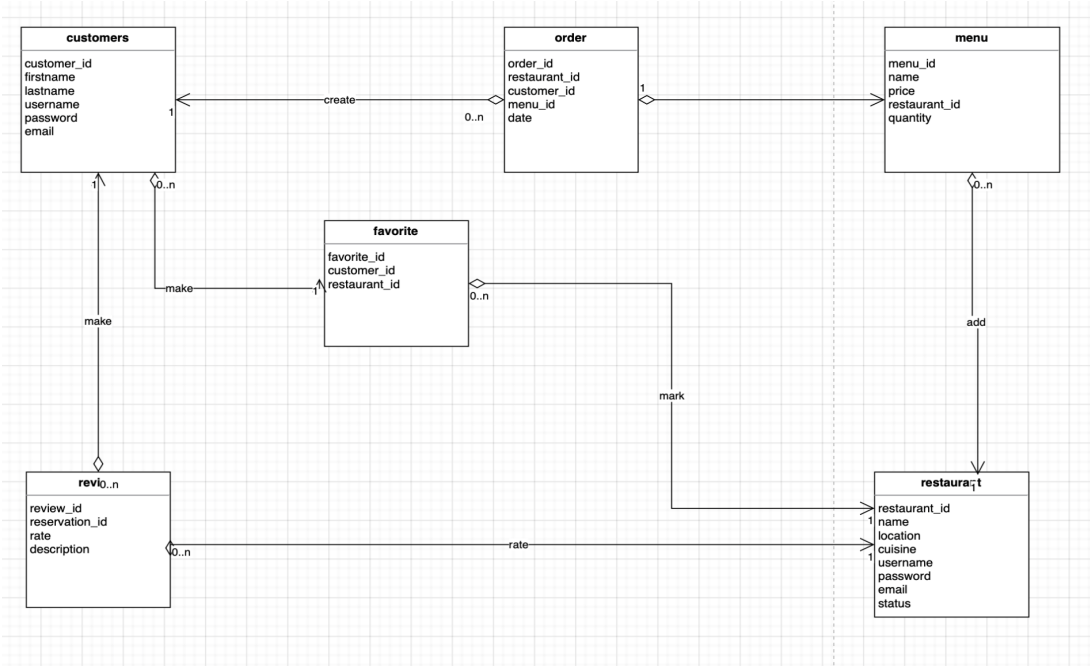






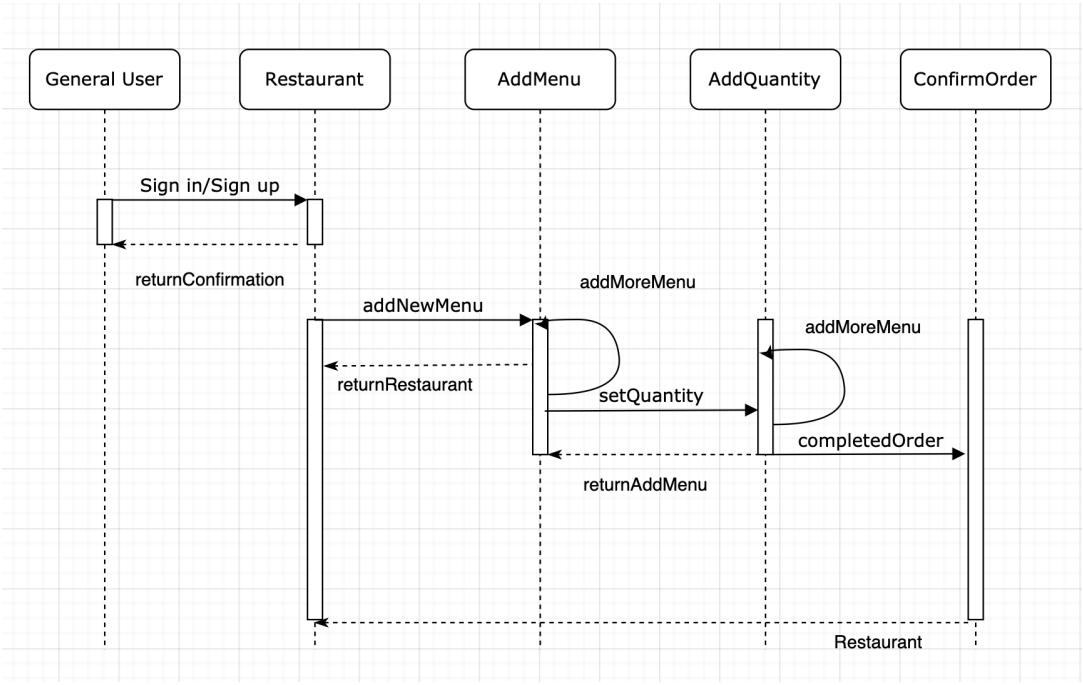
Add/Delete/Search architecture	**Functional Requirement**
Add/Delete/Search for Users	When users register
Search/Display for restaurants	When customer users click the search with restaurant name or type of food
Display for Menu lists	When customer users click the restaurant
Add/Delete Menu in Cart	When customer users click add or delete button for menu
Add/Delete Order	When Customer users click the checkout button
Add/Delete Menu for Restaurant	When restaurant users add more menu for restaurant
Add/Delete/Search setQuantity	When restaurant users change quantity for menu

5. High Level UML Diagrams

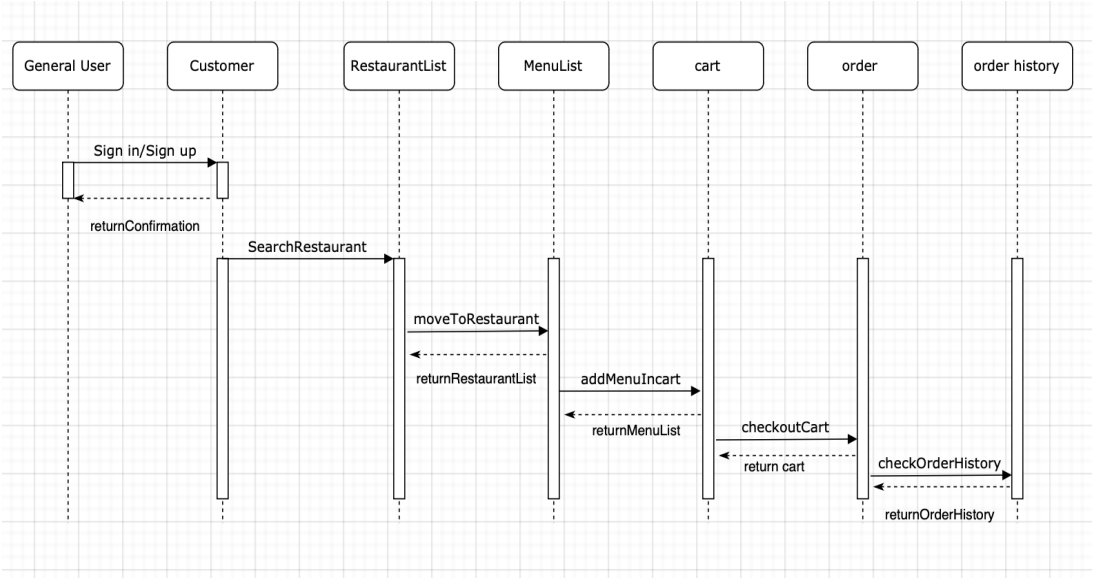


High Level Sequence Diagram

1. Restaurant



2. Customer


General User
Customer
RestaurantList
MenuList
cart
order
order history
Sign in/Sign up
returnConfirmation
SearchRestaurant
moveToRestaurant
returnRestaurantList
addMenuIncart
returnMenuList
checkoutCart
return cart
checkOrderHistory
returnOrderHistory

6. Identify *actual* Key Risks for your Project at this Time

Skills risks and mitigation plan

- Group members do not have expertise level skills in the chosen software stack which may significantly affect the application's performance.
- To avoid this risk, we've devised a proper study plan for the front-end as well as the back-end technologies. It ensures that every member gains sufficient familiarity when it comes to software used in the application: React, Express JS & MySQL

Schedule risks

- Members can be lagging behind due to not being able to give enough time in the given tasks, which will effectively delay the overall team progress.
- To avoid this risk, we've devised a proper schedule that tracks progress of every member in their given tasks. If a member faces difficulty with the tasks and seeks help from others, the issue is addressed immediately, so that it doesn't impact the team progress.
- Changes are updated transparently and regularly using project management tool: Trello.

Teamwork risks (any issues related to teamwork)

- While working in a team, members can have miscommunication among themselves, regarding different aspects of the project, which will cause a significant delay in the milestone.

- To avoid this risk, team meetings are scheduled twice a week (Tuesday: 1PM to 3PM & Thursday: 2PM to 4PM). Every single member is regularly attending the team meetings and it enables collectively tracking individual progress.
- The team is proceeding with a steady pace ensuring everyone's involvement in the project. It is ensured that we maintain proper communication through different channels, every member is familiar with what other members are working on which prevents overlapping of tasks.
- Team leader encourages the members to be more vocal about teamwork related issues faced at a personalized level, to avoid any conflict of interest. Individual opinions are valued as they should be.

legal/content risks

(can you obtain content/SW you need legally with proper licensing, copyright).

7. **Project Management (team leader)**

We have three front-end and three back-end developers on our cross-functional development team. We started our project last Thursday by carefully addressing the user interface (UI) flow in an effort to speed up the UI design process. Following that, we used Figma to develop a thorough UI prototype. We defined priority functions that are crucial for a flawless user experience after having a focused discussion. We placed special emphasis on sign-in, menu navigation, and order placement because we recognized their critical importance in meeting the needs of our users.

At the same time, we wisely decided that some features, like the customer review section, would be of lower priority. The review option was given lower priority because, in our opinion, its absence wouldn't affect our web application's essential features.

On the back end, our team carefully considered the necessary data entities and the sub-data structures that go with them for the database. These factors included the linkages between various entities. Our back-end developers immediately started building the appropriate database tables and configuring Express.js to enable easy interaction with the front-end once we had made sure that every component had been carefully thought out.

Every week, our team meets in two full group meetings on Tuesdays and Thursdays. We organize an additional weekly meeting in addition to these meetings to handle certain tasks and issues, ensuring that our project stays on track and that our communication channels are available for any necessary conversations.