Software Engineering (ECE 452) - Spring 2019
Group #3

Restaurant Automation Codename Adam

URL: https://github.com/sa2423/SE-2019

Date Submitted: March 3, 2019

Team Members:
Seerat Aziz
Lieyang Chen
Christopher Gordon
Alex Gu
Yuwei Jin
Christopher Lombardi
Arushi Tandon
Taras Tysovskyi
Hongpeng Zhang

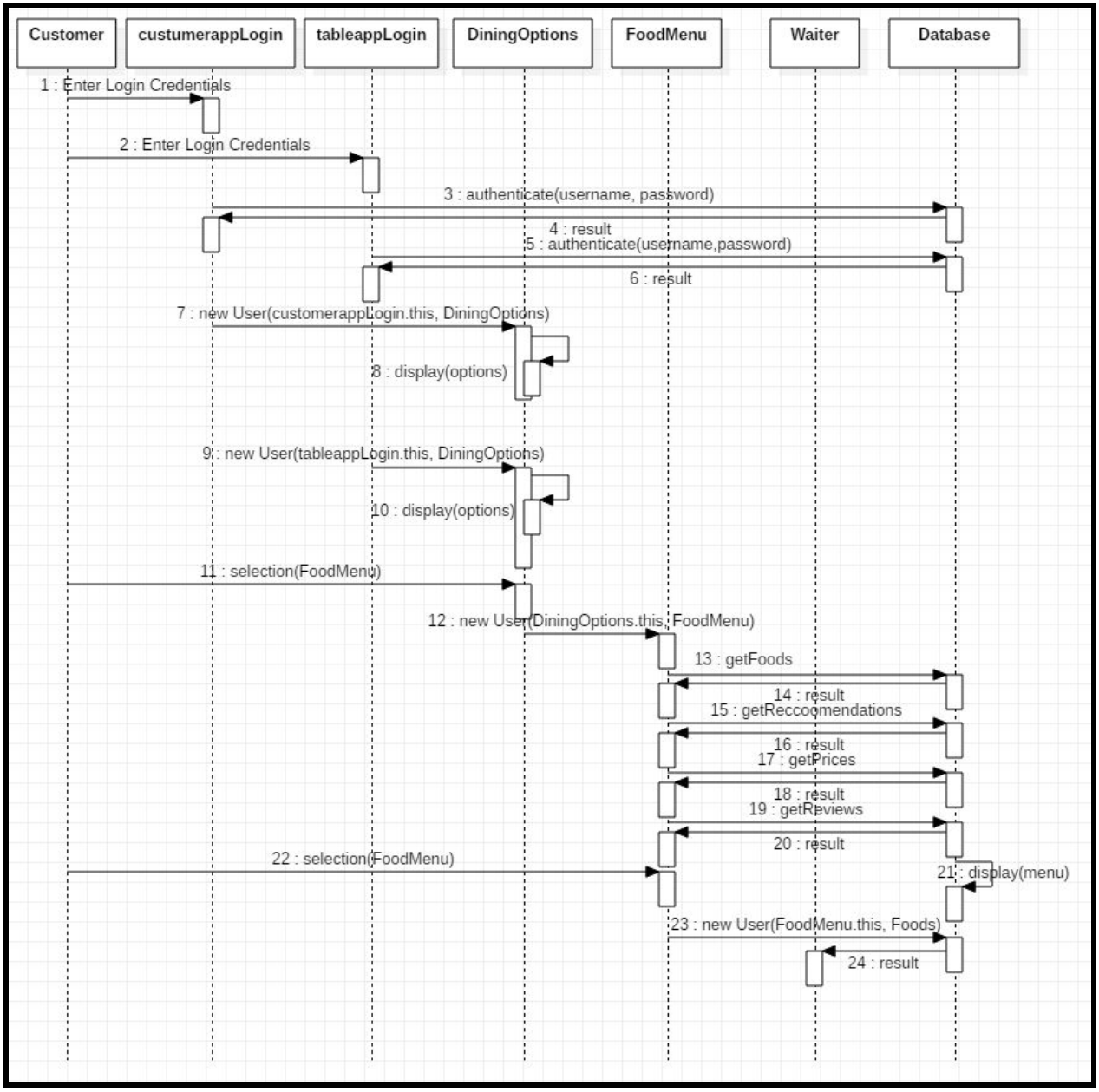
Table of Contents

Contributions Breakdown

Everyone contributed equally to this part of Report #2.

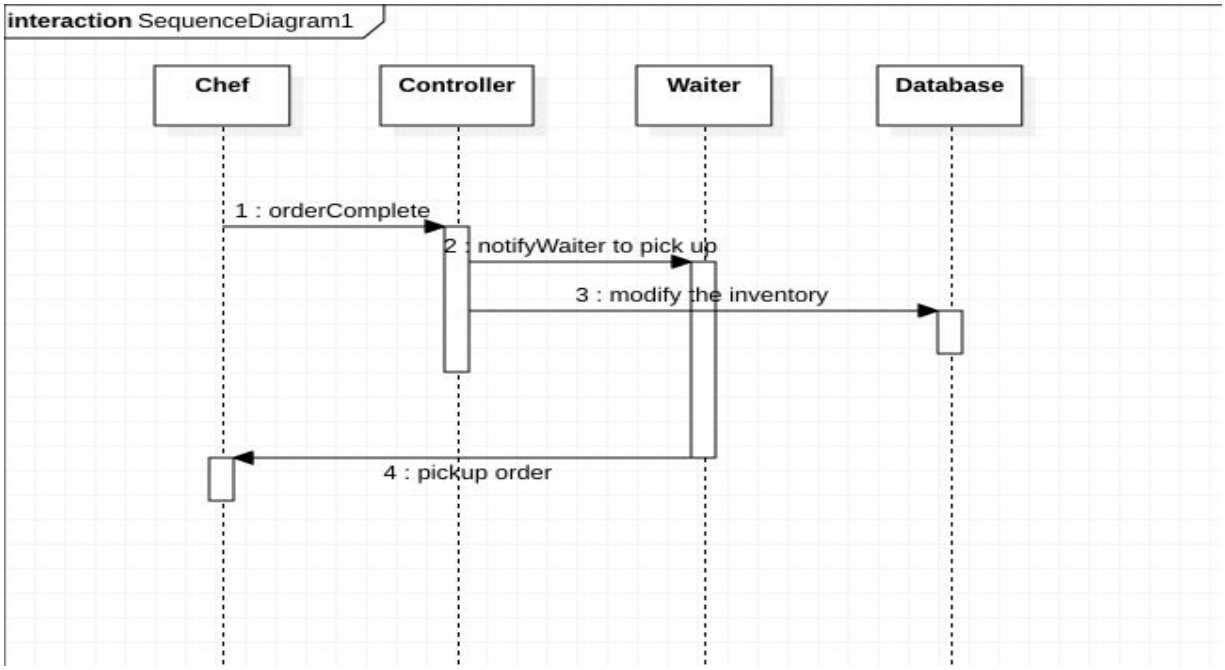
Section 1: Interactions Diagrams

1.1.1: UC-1 - Order Food



Design Principles Used: This uses the Interfaces Segregation Principle (ISP) because the customer and table app login interfaces are separated into two classes. This also uses the Liskov Substitution Principle (LSP) because all information is displayed using a result() method, which can be added to a higher order abstract class that displays information.

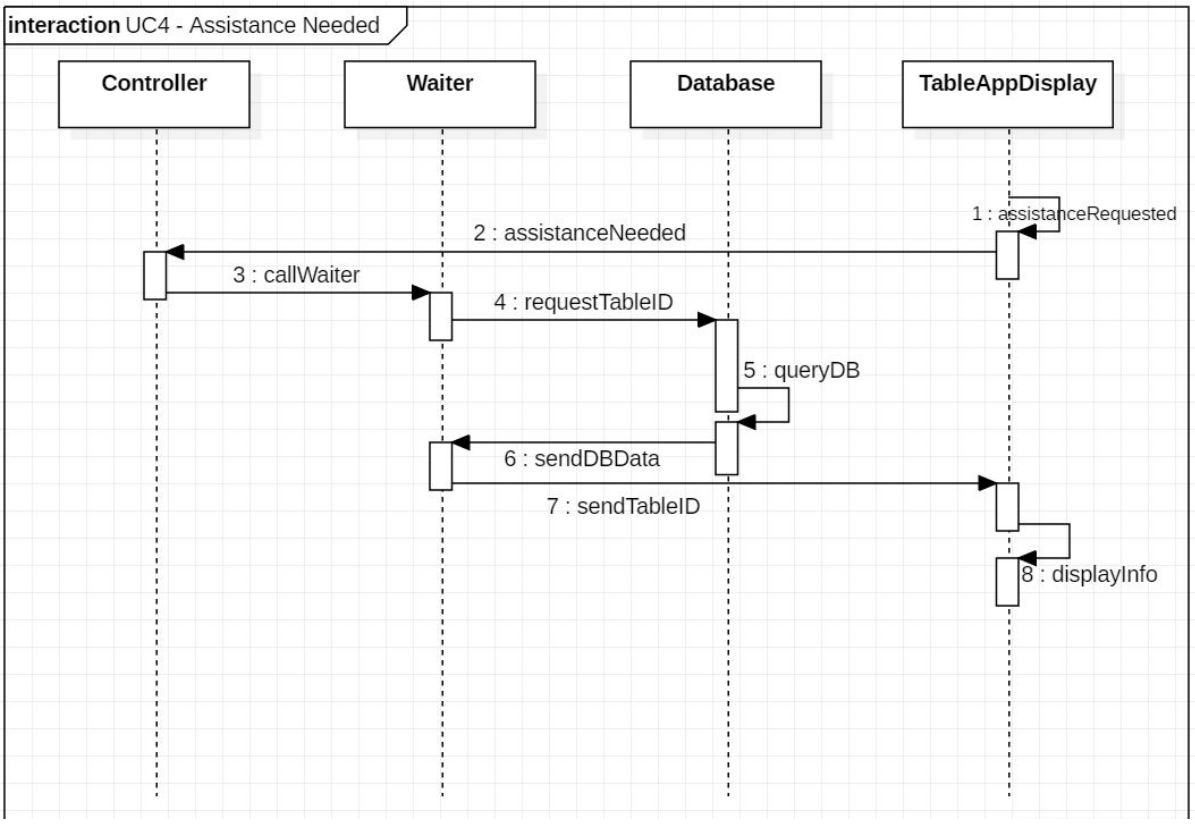
1.1.2: UC-3 - Order Complete



Design Principles Used:

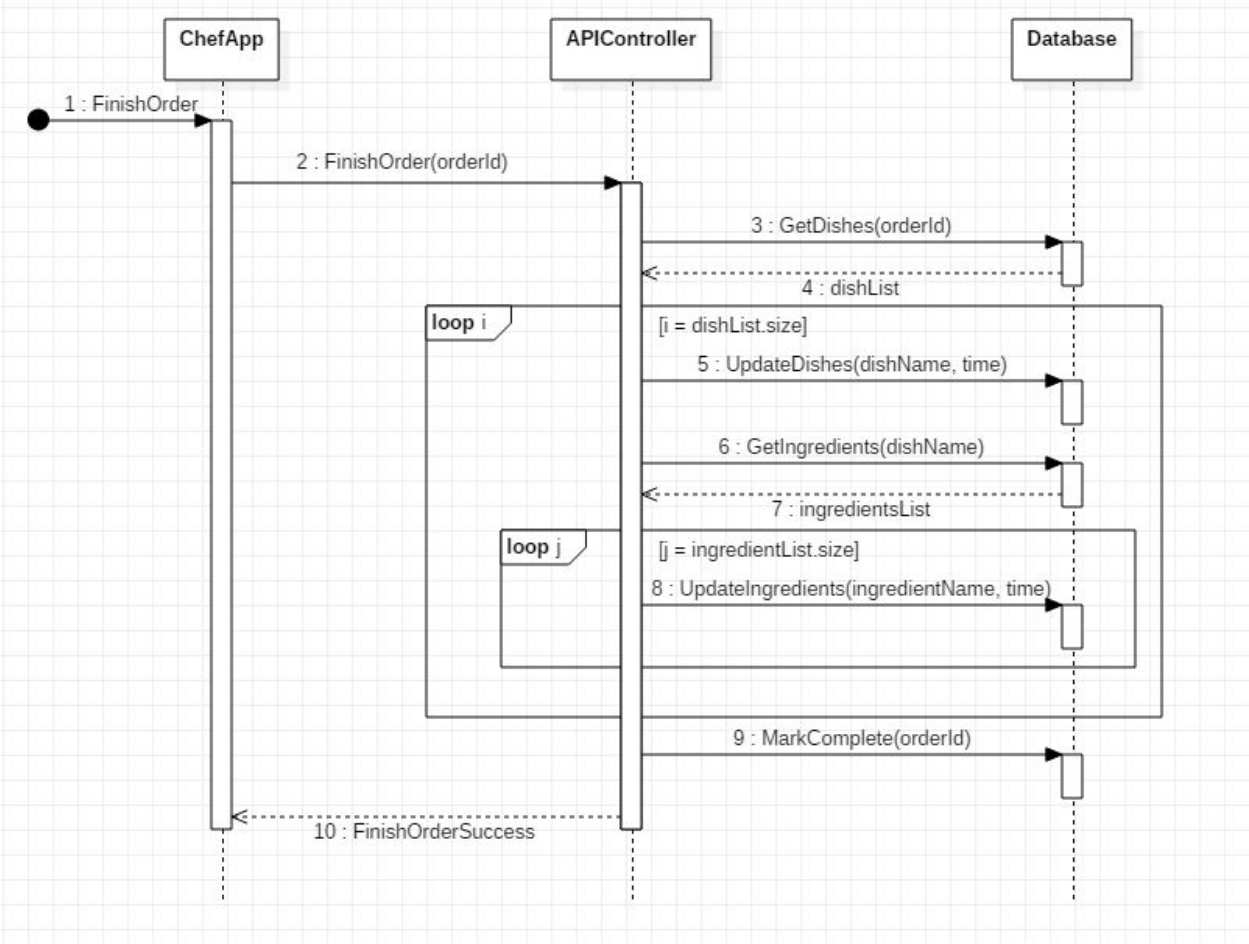
In line with the Dependency Inversion Principle, the Chef and the Waiter Apps do not communicate directly with the database, and instead have their requests routed through the Controller class.

1.1.3: UC-4 - Assistance Needed



Design Principles Used: This follows the Dependency Inversion Principle (DIP) because the app has to go through the controller class in order get the correct information from the database. This also follows the Liskov Substitution Principle (LSP) because the user interfaces will only contain information that customers and waiting staff needs (buttons and table id queues for assistance).

1.1.4: UC-6 - Finish Order



Design Principles Used:

Following the Dependency Inversion Principle, the Chef App sends its request to the API controller, which in turn handles the individual requests to the Database.

Following the Principle of Least Knowledge, or the Law of Demeter, the APIController has limited knowledge and must fetch all information about the orders, dishes, and ingredients from the database.

Section 2: Project Management

#	Feature Subteams	Names
1	Mobile Applications	• Taras Tysovskyi • Lieyang Chen • Hongpeng Zhang
2	Restaurant Website	• Arushi Tandon • Yuwei Jin

		• Seerat Aziz • Alex Gu
3	Database and API Design	• Yuwei Jin • Chris Lombardi • Taras Tysovskyi
4	Recommendation and Review Systems	• Chris Gordon • Seerat Aziz
5	Ingredient Prediction and Reservation Systems	• Alex Gu • Chris Lombardi

**Note that some developers are listed twice, we believe having one "expert" on two groups will help integrate the groups together. For example, Taras will integrate the Mobile app and the Database/Accounts together.

Weekly Meetings

We plan to meet at least once a week, either in person or online (through Google Hangouts) in order to keep everyone updated with each sub-team's progress.

Methods of Communication

We have a GroupMe and a Slack workspace environment for communicating with each other. The Slack workspace environment has channels for each subteam.

Shared Resources

Our group is using a Google Drive folder for keeping track of report work, since Google Drive facilitates collaboration. GitHub and Git will be used to keep track of progress of our software application.

Disaster Plan

We will enforce internal deadlines for project milestones as listed in the plan of work. If those deadlines are not met, then other group members will have to pitch in to ensure a milestone has been achieved.

Section 3: References

1. Professor Marsic's Lecture 10 Notes (Object Oriented Design II)
2. UML tool: https://staruml.io