

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

Restaurant Automation Codename Adam

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

Date Submitted: February 17, 2019

Team Members:

Seerat Aziz
Lieyang Chen
Christopher Gordon
Alex Gu
Yuwei Jin
Christopher Lombardi
Arushi Tandon
Taras Tysovskiy
Hongpeng Zhang

Table of Contents

Contributions Breakdown.....	1
Section 1: Customer Problem Statement.....	2
1.1: Problem Statement.....	2
1.2: Glossary of Terms.....	5
Section 2: System Requirements.....	6
2.1: Customer/Table Apps.....	6
2.2: Server App.....	7
2.3: Chef App.....	7
2.4: Restaurant Website.....	8
Section 3: Functional Requirements Specification.....	8
3.1: Stakeholders.....	8
3.2: Actors & Goals.....	8
3.3: Use Cases.....	9
3.3.1: Customer/Table Apps.....	9
3.3.2: Server App.....	9
3.3.3: Chef App.....	10
3.3.4: Restaurant Website.....	10
3.4: Use Case Diagram.....	11
3.5: Traceability Matrix.....	11
3.6: Fully Dressed Use Case Descriptions.....	12
3.7: System Sequence Diagrams.....	15
Section 4: User Interface Specification.....	18
4.1: Preliminary Design.....	18
4.1.1: UC-1: Order Food.....	18
4.1.2: UC-3: Order Complete.....	19
4.1.3: UC-4: Assistance Needed.....	20
4.1.4: UC-6: Finish Order.....	21
4.2: User Effort Estimation.....	21
4.2.1: UC-1: Order Food.....	21
4.2.2: UC-3: Order Complete.....	22
4.2.3: UC-4: Assistance Needed.....	23
4.2.4: UC-6: Finish Order.....	23
Section 5: Project Management.....	24
Section 6: References.....	26

Contributions Breakdown

Everyone contributed equally to this part of report #1.

Section 1: Customer Problem Statement

1.1: Problem Statement

Restaurant automation is a daunting task, as there are many variables to consider in the automation process. Moreover, it is important to maintain a balance between efficiency and hospitality as most customers associate restaurants with their ambience as opposed to mechanized efficiency. Many restaurants nowadays have started to automate their tasks, such as Panera Bread and TGIF, by streamlining the ordering and payment processes by employing tablets with specialized apps. Spyce, a restaurant in Boston, has automated its cooking process by replacing human chefs with a robot [4]. However, even the owners of Spyce seek to maintain the human touch by attempting to tailor the experience for each customer.

For this reason our project aims to automate basic tasks that customers, waiting staff and chefs have to perform in restaurants, without outright replacing them, in order to maintain the human touch that the restaurant experience depends on. This, in turn, will decrease the time it takes from customers coming in and placing their order to actually being served their food, resulting into better dining experience. Additionally, it will help the restaurant owners manage their business better through detailed statistics and analysis of trends. It will provide a way to easily see all the placed orders as well as a way to predict the amount of ingredients that will be needed for any given day, (inferred from historical data). This allows the restaurant owners to dramatically cut down the leftovers and shape their menu accordingly to what sells best. Moreover, our project will improve customer experience by recommending meals.

While the proposed solution requires additional investment in the form of electronic devices for customers and servers, as of 2018 lower-end tablets running Android operating system can be purchased for under \$50 dollars, especially in larger quantities. As such, this is well within the realm of financial feasibility for medium and even small, family owned restaurants. Although our prototype is tailored towards Android operating systems, it can be implemented on the iOS operating system if restaurants do not mind paying more.

Problem: Helping customers decide on a meal to eat

A common problem that customers face when going to eat a restaurants, whether they have visited the restaurant before or not, is deciding on what food to eat. This problem is often worsened by menus with food items with very “foreign” sounding names or unfamiliar menus in general. Such problems can discourage customers from visiting the same restaurant, so we find that assisting customers with choosing meals will cause them to maintain their relationship with our establishment. As a result, this will cause our customer base to increase over time.

The Codename Adam Solution:

We seek to solve this problem by implementing a meal recommendation system, which will be enhanced by using responses from the review system. In order to implement the recommendation system, the user will be given the choice to take a short quiz before ordering their meal. This quiz will ask questions about their meal preferences (if they want a meal with cheese, rice, meat, etc.) to recommend a list of appropriate meals. This quiz component will incorporate active learning by taking the user's choices (what meal they select from the given list) into account for future recommendations. Likewise, the user will be allowed to "favorite" meals on the application, which will also be taken into account for future meal recommendations. After a user finishes their meal, they will be given a short questionnaire to rate the meal and restaurant service. This responses from this short questionnaire (the review system) will also be taken into account when recommending meals to future customers. If the customer returns, we can further streamline the ordering process, using an account system with facial recognition. During this procedure, all the customer needs to do is take a seat and look at the camera, avoiding any inconveniences such as user login. Of course, to avoid any concerns about data collection and invasion of privacy users are informed of their information being stored and can opt out of this feature at any time.

Problem: Providing customers with efficient and careful service

With the advancement of internet technologies, customers want information delivered to their screens instantly. For restaurants in particular, they would like to see today's menu, reserve a table or even order food in a matter of seconds, without ever leaving their homes or talking to people. Along with customer service, restaurants have consider their customers' dietary concerns. This includes taking religious dietary laws (such as Halal and Kosher) and fad diets (Keto and Vegan) into account. Moreover, as food allergies have become more prevalent in today's era, it has become more important for restaurants to be vigilant with their food preparation efforts.

The Codename Adam Solution:

Codename Adam provides an ecosystem of websites and mobile apps that allow customers to interact with restaurants remotely and effortlessly. Checking today's specials, getting food recommendations, reserving a table and making an order can now be done in a matter of seconds. In order for the restaurant staff to be able to keep up with the features offered online, we will also offer apps for servers and chefs to increase their productivity by being able to receive orders instantly and be notified immediately when an order is done cooking and is ready to be served. Furthermore, the app specialized for the waiting staff will provide information about the meals, such as

ingredients used, calorie information, and any possible allergy warnings. Likewise, the restaurant website (with admin privileges and the ingredient prediction system) will help restaurant owners decrease operating costs and increase profits.

Problem: Make server spend their time more efficiently

A server has a lot of things to do when they are working. For example, they need to serve for new-coming customers and stand by them to listen to their decisions on meals, they need to bring a finished meal from chef to a customer who is waiting, they need to go helping those customers who is calling a server for some specific questions. They have a lot things to do in their work, so an efficient way to use their time is really desired so that all customers can enjoy their time by being fully served.

The Codename Adam Solution:

Actually, there are some unnecessary actions that can be avoid by making our application more advanced. For example, if we can allow the customers to order their food on the table app, the time server used to spend on record customers' decisions can be saved. If we can allow customers to order food online instead of making phone calls, then the time servers use to talk to customers on phone can be saved. If we can allow customers to use the application to write down their specifications on each meal, then the overhead servers spend in transmitting the information from customers to chefs can be saved. With time saved, servers can spend more time on those customers who requests a server calling so they can be helped at once.

Problem: Maximizing profits and streamlining process for the Chef

The kitchen is the engine of a restaurant. Every customer's order must pass through the kitchen, which means that the efficiency and quality at which the kitchen operates heavily impacts the performance of the restaurant itself. As a result, it is in the restaurant's best interest to promote the efficiency of its chefs and the quality of its ingredients and to reduce wastage in its kitchen. If this can be accomplished, chefs will have more time and fresher ingredients, allowing them to produce a larger quantity of meals of higher quality, not only improving the customer's dining experience, but also increasing the restaurant's profit potential.

The Codename Adam Solution:

Codename Adam's ingredient prediction system helps to solve many of these problems that restaurants face. Based off of records of which menu items were ordered on the same day of the week in previous weeks, the ingredient prediction system will calculate an estimated amount of each ingredient that will be required on that specific day. This will allow chefs to determine how much of each ingredient to prepare for the

the day's dishes, reducing food wastage, preventing chefs from wasting time preparing unnecessary ingredients, and ensuring that dishes use only the freshest of ingredients. In the beginning, the chef may want to prepare a little more than what is estimated, but over time, the estimate should become more refined and more accurate, as outliers are identified and pruned from the estimation. This will be accomplished by performing the RANSAC algorithm on the amount of ingredients needed for the k most recent weeks. The estimate may also be refined by trends in dish reviews and favorites and pending reservations. An extension of this feature may further divide the ingredient prediction by time of day, to maximize the freshness of ingredients and efficient use of the chefs' time.

What makes Codename Adam a better solution for restaurant automation?

Codename Adam is a better solution for restaurant automation because unlike applications created in previous years, it provides a way for restaurants to plan their future spending through the ingredient prediction system and enhances the customer experience through the meal recommendation system. Our project is an improvement over Why W8 [1], the Fall 2018 solution to restaurant automation, because it utilizes their innovations (rating and favorites system) to create a meal recommendation system. Along with providing a list of meals frequently eaten at a restaurant, registered customers will benefit from receiving recommendations based on their personal preferences even if the menu changes. Moreover, our project is an improvement over FoodEZ [2], the Spring 2015 solution to restaurant automation, because it streamlines the meal payment process even more by providing different payment options through the customer application and table application. This reduces the time waiting staff spends on accounting for meals paid using different methods.

1.2: Glossary of Terms

Customer app: a software that customers can download on their mobile devices and use to order food from the restaurant

Table app: a software that is built inside every table devices, so dine-in customers can use it to order food, request service and pay the bill

Server app: a software that every server needs to download on their mobile devices and it will help alert everything associated with the demand of their service

Chef app: a software that every chef needs to see the list of ordered meals they need to cook

Section 2: System Requirements

2.1: Customer/Table Apps

Functional Requirements:

Identifier	Priority (Higher number indicates higher priority)	Requirement
REQ-01	5	For customer and table app, users should be able to browse the most up to date menu
REQ-02	1	For customer and table app users should be able take a quiz to get meal recommendations
REQ-03	5	For customer and table app unregistered users should be able to register
REQ-04	1	For customer and table app all registered users should be able to see meal suggestions based on their previous dining habits
REQ-05	3	For customer app, registered users should be able to order food online, either to be delivered or to be picked up.
REQ-06	3	For customer app, registered users should be able to reserve a table at the restaurant.
REQ-07	3	The table app should automatically detect when someone sits down at the table and attempt to log them in. If login fails, give the user the opportunity to login or register.
REQ-08	5	For table app, all users should be able to make an order and pay for it by entering their credit card information on the device, or in person.
REQ-09	5	For table app, customer should be able to request assistance

2.2: Server App

Functional Requirements:

Identifier	Priority (Higher number indicates higher priority)	Requirement
REQ-10	5	For server app, all users have to login before using the app
REQ-11	5	For server app, all registered users should be notified when a customer requires assistance
REQ-12	5	For server app, all registered users should be able to manually place an order for a customer
REQ-13	5	For server app, all registered users should be notified when an order has been completed and should be served to the customer

2.3: Chef App

Functional Requirements:

Identifier	Priority (Higher number indicates higher priority)	Requirement
REQ-14	5	All users should be able to view a dynamically updated queue of orders.
REQ-15	4	All users should be able to mark an order as fulfilled after it has been prepared, which would send a notification to the server app.
REQ-16	2	For the chef app, when an order is completed, the required ingredients will automatically be deducted from the ingredient inventory.

2.4: Restaurant Website

Functional Requirements:

Identifier	Priority (Higher number indicates higher priority)	Requirement
REQ-17	5	The website should contain information about the menu and most popular food items.
REQ-18	4	Customers should be able to make reservations or take out/delivery orders (if applicable) from a page on the website.
REQ-19	3	The website should allow users to sign in to their accounts in order to use the meal recommendation feature.
REQ-20	4	The website admin page should display relevant statistics, such as earnings, most popular foods and ingredient usage.
REQ-21	4	The website admin page should predict future dish popularity and ingredient consumption based on order history of the past few weeks.
REQ-22	1	The website admin page should allow the owner to modify the menu, change prices, and enable/disable features.

Section 3: Functional Requirements Specification

3.1: Stakeholders

The following are the stakeholders who will have the most interest to design and run the system.

- Restaurant owner
- Employees i.e Manager, Chef, Waiter or Busboy
- Investors who want to explore new business mode

3.2: Actors & Goals

Table 3.2.1 Initiating Actors

Actor	Role	Goal
Customer	The customer who pays the money for delicious food and good service. They would use the application to take quizzes and get	The goal of the customer is to enjoy the service and food provided by the restaurant. They can choose the food that best suits their appetite.

	recommendations from the system.	
Owner	The owner uses the application to monitor the restaurant running status to make sure the restaurant is running efficiently.	The goal of the owner is to maximize the profit of running the restaurant by using the application and to increase customer satisfaction .
Chef	The chef decides the taste of food they wait for the orders from the customers and cooks the food in customers' specific request.	The goal of the chef is to prepare meals on the order queue and make the food that suit customer's appetite.
Waiter	The waiter is the person who assists dine-in customers in taking the quiz, making the order and bringing the food to their table.	The goal of the waiter is to provide the wonderful service to customers, such as help customers make orders, bring food to their table and pay the bills.

Table 3.2.2 Participating Actors

Actor	Goal
Database	The database record the quizzes result, table selection, customer's order, and inventory

3.3: Use Cases

3.3.1: Customer/Table Apps

Actor	Actor's Goal	Use Case Name
Customer	To order food, either to be delivered, served or picked up	OrderFood (UC-1)
Customer	To be offered meal suggestions.	Suggest (UC-2)
Customer	To log in via facial recognition software	Recognize (UC-11)

3.3.2: Server App

Actor	Actor's Goal	Use Case Name
Waiter	To be notified when an order has been prepared by the chef and should be delivered.	OrderComplete (UC-3)
Waiter	To be notified when a customer needs assistance.	AssistanceNeeded (UC-4)
Waiter	To be able to place customer's order manually	OrderFood (UC-1)

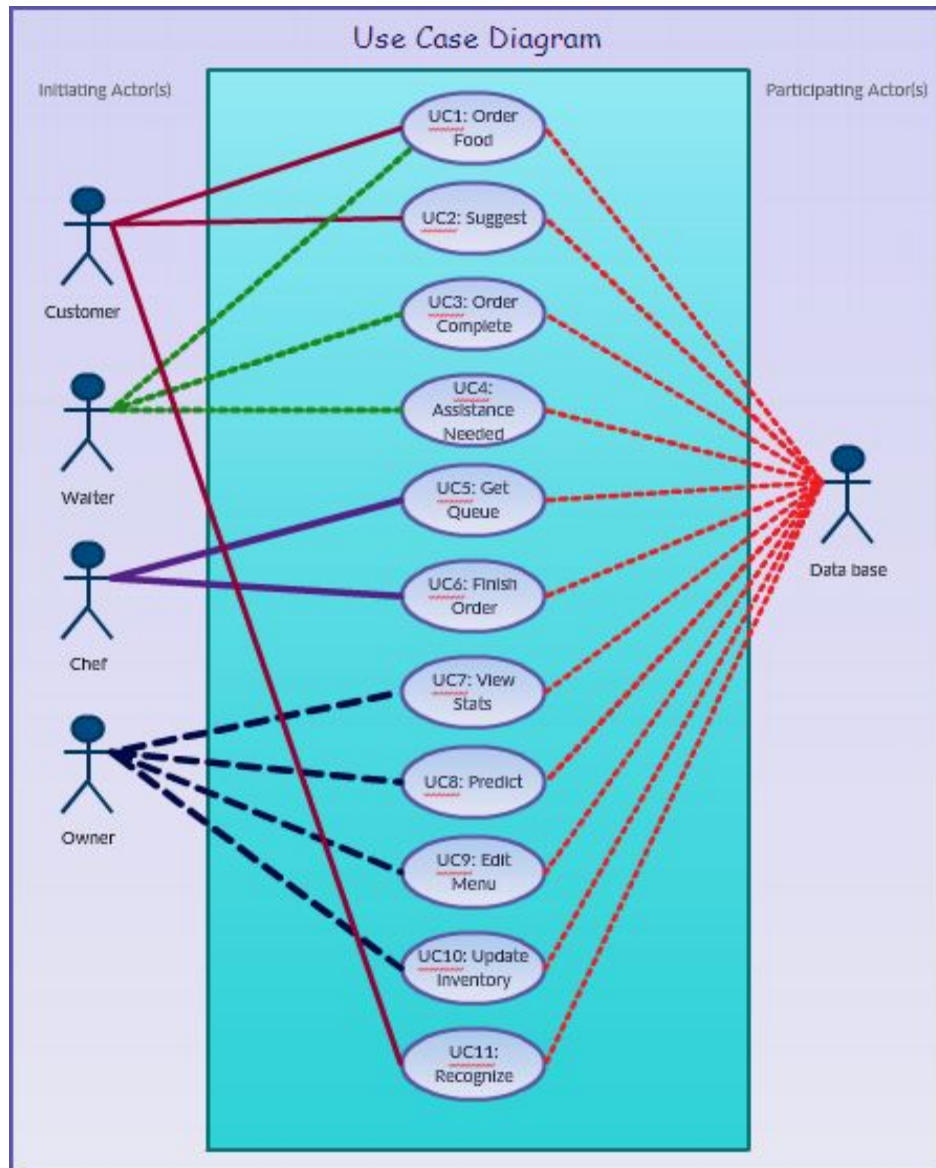
3.3.3: Chef App

Actor	Actor's Goal	Use Case Name
Chef	To be able to see the list of current orders	GetQueue (UC-5)
Chef	To mark a dish as complete and to update ingredient inventory automatically	FinishOrder (UC-6)

3.3.4: Restaurant Website

Actor	Actor's Goal	Use Case Name
Customer	To order order food, either to be delivered, served or picked up	OrderFood (UC-1)
Owner	To view statistics about restaurant performance	ViewStats (UC-7)
Owner	To predict trends in dish popularity and dish consumption	Predict (UC-8)
Owner	To modify the menu and prices	EditMenu (UC-9)
Owner	To update ingredient inventory	UpdateInventory (UC-10)

3.4: Use Case Diagram



3.5: Traceability Matrix

Req	P W	UC-1	UC-2	UC-3	UC-4	UC-5	UC-6	UC-7	UC-8	UC-9	UC-10	UC-11
REQ-01	5	X										
REQ-02	1		X									
REQ-03	5											
REQ-04	1		X									
REQ-05	3	X										

REQ-06	3	X										
REQ-07	3											X
REQ-08	5	X										
REQ-09	5				X							
REQ-10	5											
REQ-11	5				X							
REQ-12	5				X							
REQ-13	5			X								
REQ-14	5					X						
REQ-15	4			X			X					
REQ-16	3						X					
REQ-17	5	X										
REQ-18	4	X										
REQ-19	3		X									
REQ-20	4							X				
REQ-21	4								X			
REQ-22	1									X	X	
Max PW		5	3	5	5	5	4	4	4	1	1	3
Total PW		25	5	9	15	5	7	4	4	1	1	3

3.6: Fully-Dressed Descriptions

UC-1: Order Food	
Related Requirements: REQ-01, REQ-05, REQ-06, REQ-08, REQ-17, REQ-18	
Initiating Actor: Customer	
Actor's goal: To order food, either to be delivered, served or picked	
Participating Actors: Waiter, Database	
Preconditions: <ol style="list-style-type: none"> Customer successfully sets up the application on the table or his/her own mobile device and logs in 	

2. The system displays a menu that users can look up
Postconditions: <ol style="list-style-type: none"> 1. The order has been stored into database 2. One of waiters should have the order displayed on their application
Flow of Events for Main Success Scenario: <ul style="list-style-type: none"> → 1. Customers open the application by clicking the icon on their phone and select “menu” → 2. Customers add the food they like into orders by clicking the “+” icon appear under each food description → 3. Customers click the “confirm” button ← 4. The system automatically brings customers to the payment page → 5. Customers enter their payment information and click “confirm” button to verify the payment ← 6. The system signals database to store this order and notify one waiter to add the order ← 7. The system brings customers to the ending page and signals “Order Completed!”

UC-3: OrderComplete
Related Requirements: REQ-13, REQ-15
Initiating Actor: Chef
Actor’s goal: To notify the waiter that order is completed and should be delivered
Participating Actors: Waiter, Database
Preconditions: <ol style="list-style-type: none"> 1. Food of the order is fully prepared 2. The chef app displays the button “Order Complete”
Postconditions: The waiter is coming to get the food and going to deliver it
Flow of Events for Main Success Scenario: <ul style="list-style-type: none"> → 1. Chef click the button “Order Complete” on his chef app ← 2. The system displays a notification on the waiter’s app → 3. Waiter gets ready to come to pick up the food

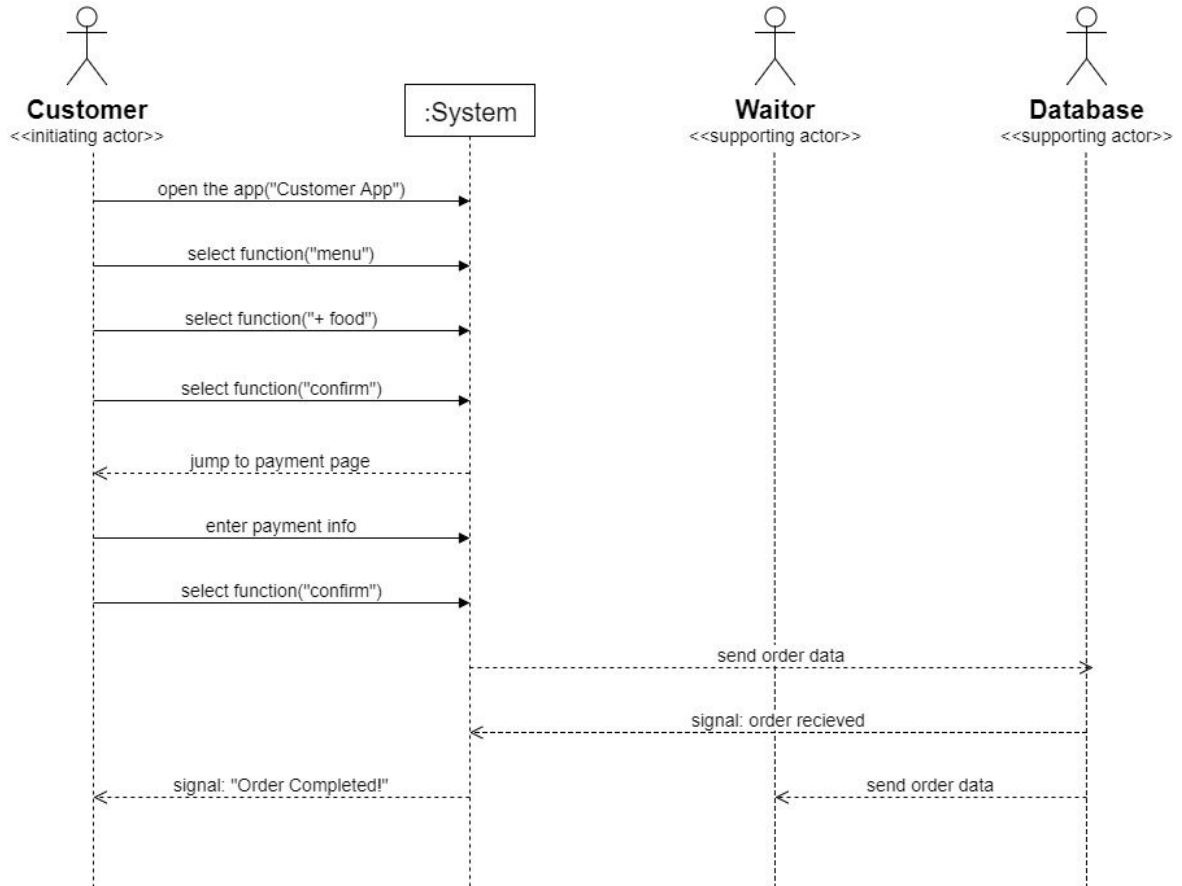
UC-4: Assistance Needed
Related Requirements: REQ-9, REQ-11,REQ-12

Initiating Actor: Customers
Actor's goal: To notify the waiter that assistance is needed
Participating Actors: Waiter, Database
Preconditions: Customers open the table app and see the “call server” button
Postconditions: The waiter is coming to the table whose table app has called him/her
Flow of Events for Main Success Scenario: → 1. Customers click the button “call waiter” on the table app ← 2. The system sends the table id from table app to the database ← 3. The system takes the table id from database to server app → 4. The waiter sees customers of that table need help → 5. The waiter gets ready to go to the table

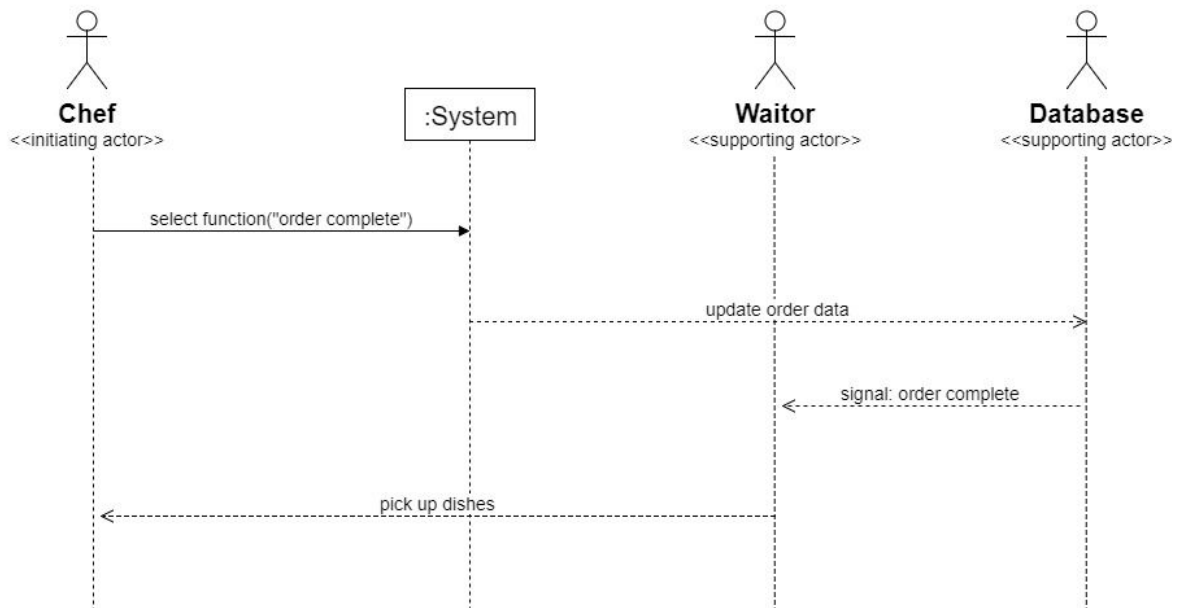
UC-6: FinishOrder
Related Requirements: REQ-15,REQ-16
Initiating Actor: Chef
Actor's goal: To mark a dish as complete and to update ingredient inventory automatically
Participating Actors: Database
Preconditions: Chef should be notified that customers have successfully receive their food after the “order complete” UC
Postconditions: 1. The order should be fully deleted from the database and remove from both the server and chef app 2. The information of ingredient inventory should be updated in the database
Flow of Events for Main Success Scenario: → 1. Chef click the “FinishOrder” button appearing below the order on the chef app ← 2. The system sends the order id and request from chef app to the database to delete the order information ← 3. The system updates the information of ingredient inventory in the database by referring to the order information ← 4. The system displays a removal in both server and chef app

3.7: System Sequence Diagrams

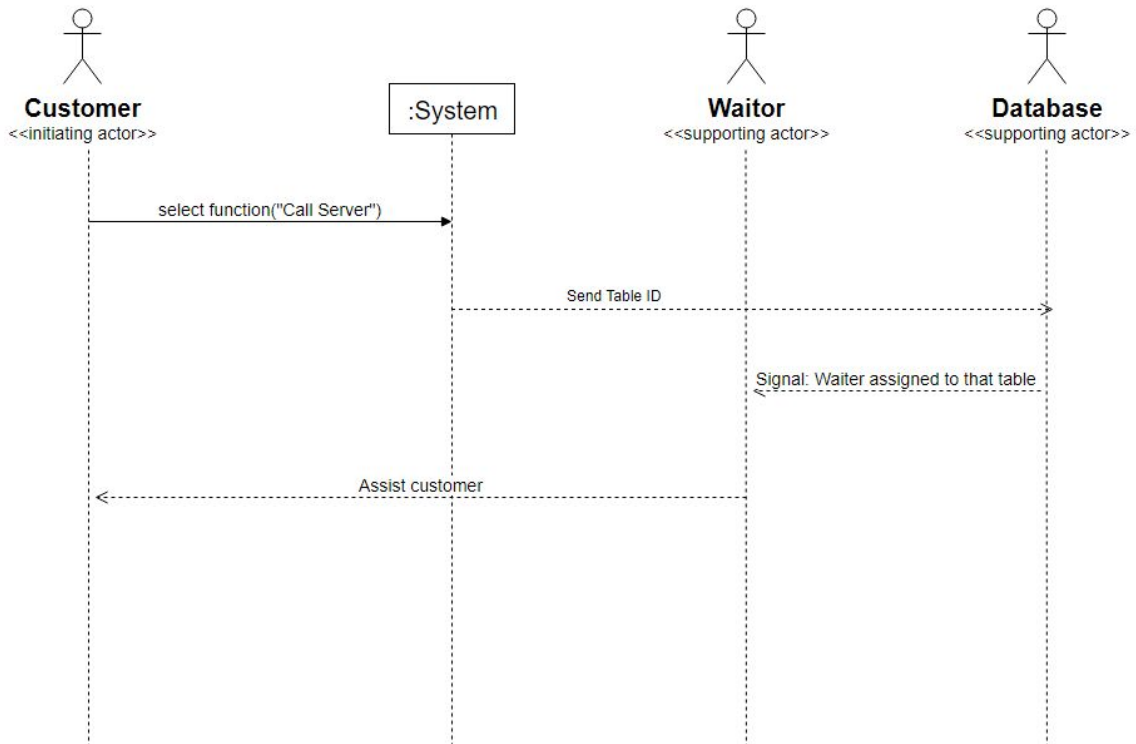
User Case 1: Order Food



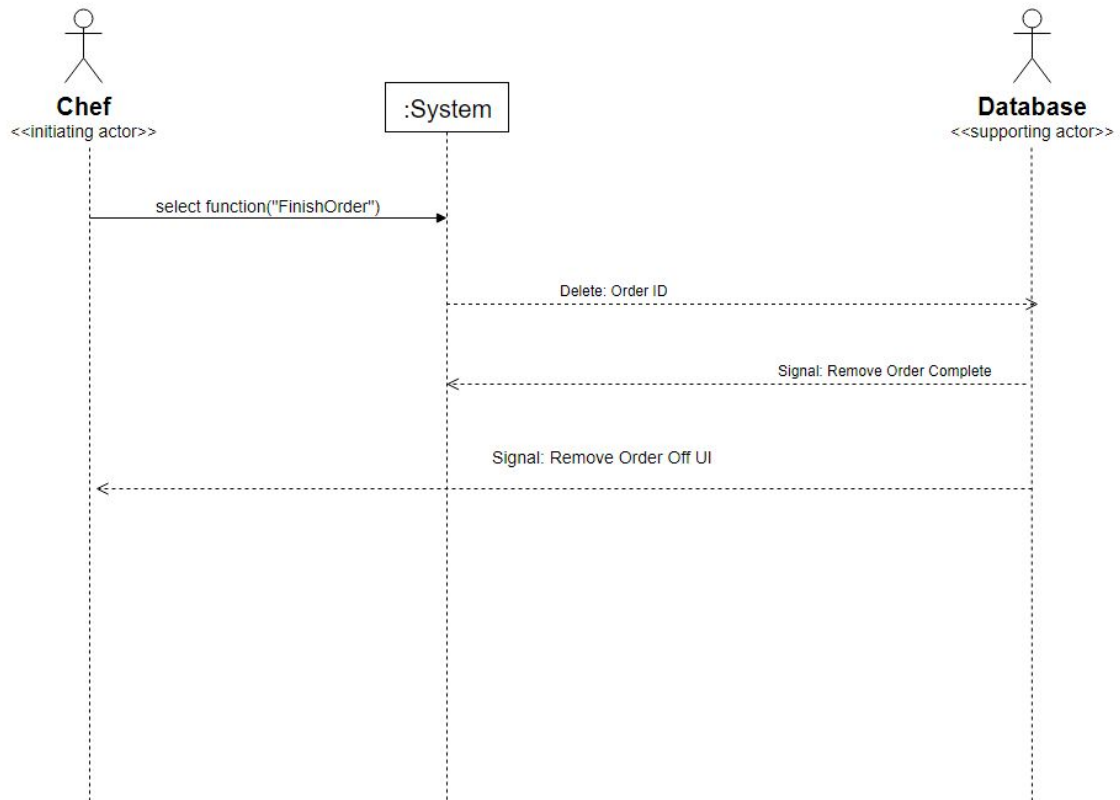
User Case 3: Order Complete



User Case 4: Assistance Needed



× × ×
× **User Case 6:** × × ×
× **Finish Order** × × ×
× × ×



Section 4: User Interface Specification

4.1: Preliminary Design


4.1.1: UC-1: Order Food

Customer App Screens

Menu

Help

Cart




Pizza

\$3.99

Pizza description long

+




Hamburger

\$8.99

Burger description long

+




Pasta

\$7.99

Pasta description long

+



Steak

\$9.99

Steak description long

+

Confirm

Payment Page

Help

Cart(3)

Items in Cart:

Food Item	Quantity	Subtotal
Pizza	2	6.99
Hamburger	1	8.99

Taxes:

3.00

Gratuuity:

0.00

Total:

18.98

Select Payment Option:

☒ Credit

Enter Credit Card #

Enter Credit Card Pin

☐ Debit

Submit Payment

Payment Page

Help

Cart

Order Complete!

Thank you for submitting your order!

A confirmation email has been sent to your inbox.

Upon arrival at the restaurant, a waiter will be ready to give you your order.

Please let us know if you have any questions in the meantime at 1-800-555-5555!

Steps to Take:

1. Customer goes to menu screen on the customer app.
2. Customer selects the '+' button located in each menu item on the list to add to their cart.
3. After customer selects all of the food items they want, they press the 'confirm' button on the bottom of the menu page.
4. Customer is directed to the payment page, where they can view their order information. On the bottom of the screen, customer selects their desired payment option and enters their information.
5. After entering all of their information, customer presses 'submit payment' button.
6. Customer is directed to order completion page on the app once payment is submitted and processed. In the meantime, the restaurant database is updated and a waiter is notified of the order.

4.1.2: UC-3: Order Complete

Chef App Screen

Order Screen		
Order Details:		
Food Item	Quantity	Complete?
Pizza	2	<input checked="" type="checkbox"/>
Hamburger	1	<input checked="" type="checkbox"/>
Pasta	4	<input checked="" type="checkbox"/>
Appetizer	5	<input checked="" type="checkbox"/>

Order Complete

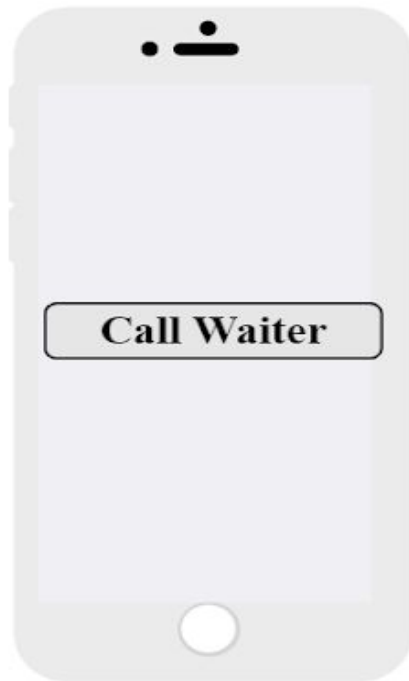
Waiter App Screen

Dashboard	
New Notification!	
Order #34 for Table#12 is ready to be served to the customers.	
Other information about Waiter App goes here	

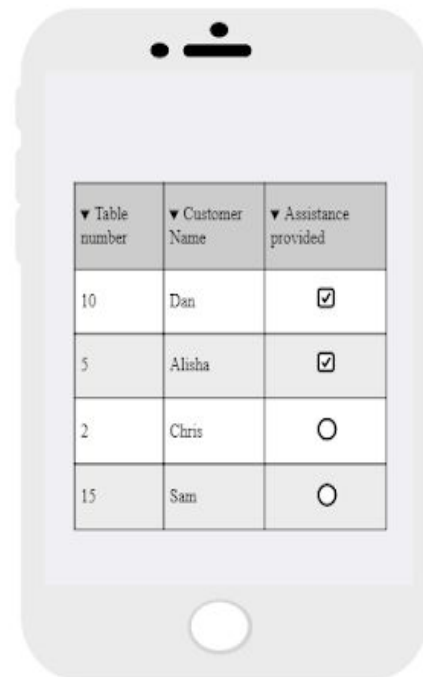
Steps to Take:

1. The chef checks off all order items that were prepared.
2. After all items are checked off, the chef presses the 'order complete' button.
3. Once an order is completed, a waiter receives a notification on the waiter app that informs them of the order that was completed and to what table it should be taken to.

4.1.3: UC-4: Assistance Needed
Customer app



Waiter Screen

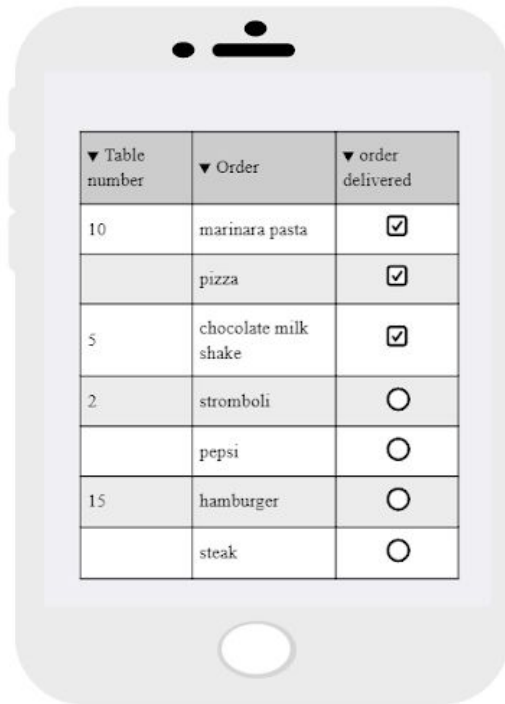


Steps to Take:

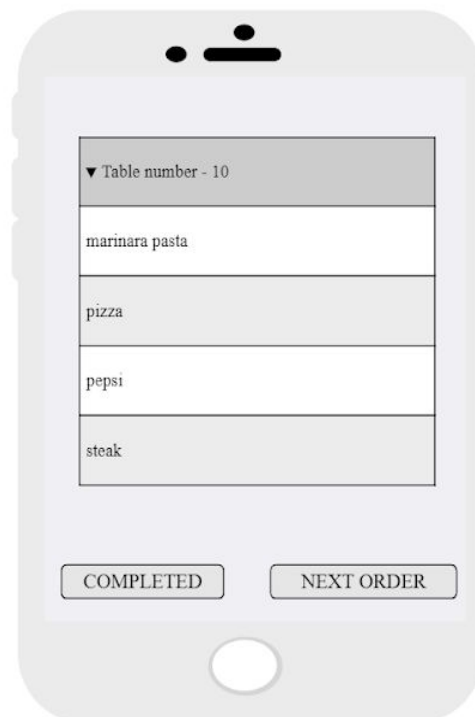
1. The customer presses the 'CALL WAITER' option when they need assistance on their application
2. The person's details who needs assistance pops up on waiters app
3. The waiter checks of the name of the person who has already been assisted and the waiter moves on to the next person.

4.1.4: UC-6: Finish Order

Waiter App



Chef App



Steps to Take:

1. Chef's screen shows the ongoing order and once the chef completes the order he presses the 'COMPLETE' button and then the 'NEXT' button for the next order he wants to prepare
2. The waiter takes over the order and tick marks the orders he has delivered
3. The orders completed at both ends are removed from the queue.

4.2: User Effort Estimation

4.2.1: UC-1: Order Food

Scenario 1: Customer knows exactly what to order

In this scenario, the customer would press the '+' button n times for all n items that they want order. Depending on the size of the menu, the customer would have to use at least 1 keystrokes in order to press the '+' button next to the menu items they want to order. Then the customer will have to press the 'confirm' button. Once directed to the payment page on the customer app, the customer will have to select the appropriate radio button for their desired payment choice. Then they will have to manually enter their payment information. Once all information is entered, the

customer will press the 'submit payment' button. This will direct them to the order completion screen.

Number of button presses: n (menu items) + 3 (other buttons)

Number of keystrokes: $\geq 1 + 19$ (for entering in credit card number)

Fraction of user interface navigation: $4/(n + 23)$

Fraction of clerical data entry: $19/(n + 23)$

Scenario 2: Customer wants to edit their order before paying

In this scenario, the customer will press a 'edit order' button on the first payment page (this button will be added to the app although it's not on the mock-up). Once directed back to the menu screen, the user will press the '+' button m items or the '-' button (this will appear next to items already added to cart) k times. Then they will press the 'confirm' button once more. Depending on the length of menu, they customer will have to use at least 1 keystrokes while adding and deleting menu items. These are the only additional steps the customer will have to take in this scenario.

Number of button presses: $n + m + k + 4$

Number of keystrokes: $\geq 2 + 19$

Fraction of user interface navigation: $6/(n + m + k + 25)$

Fraction of clerical data entry: $19/(n + m + k + 25)$

4.2.2: UC-3: Order Complete

Scenario 1: Chef marks all food items as completed and confirms order completion

In this scenario, the chef will use n keystrokes to mark all n menu items as complete. Afterwards, the chef will press the 'order complete' button. Then the waiter will receive a notification on their app and will have to use 1 keystroke to view the notification.

Number of button presses: 1

Number of keystrokes: $n + 1$

Fraction of user interface navigation: $2/(n + 2)$

Fraction of clerical data entry: $n/(n + 2)$

Scenario 2: Chef confirms order completion before marking all food items as complete

In this scenario, the chef forgot to mark all menu items as complete. So, after pressing the 'order complete' button, the chef will receive a notification with an error message about why the order completion was not processed. Then, the chef will have use 1 keystroke to open the notification. After reading the notification, the chef will mark the remaining menu items as complete and press the 'order complete' button again. Then the waiter will receive a notification on their app and will have to use 1 keystroke to view the notification.

Number of button presses: 2
Number of keystrokes: $n + 2$
Fraction of user interface navigation: $4/(n + 4)$
Fraction of clerical user data entry: $n/(n + 4)$

4.2.3: UC-3: Assistance needed

*Customer clicks the 'call waiter' button to address the waiter for assistance**

In this scenario, the customer will press the 'call waiter' button. Then the waiter will receive a notification and will use 1 keystroke to view the notification. After providing assistance, the waiter will check 'assistance provided' checkbox next to the table number to mark that he has offered assistance to the given table. Then the customer's assistant request is waved off automatically.

Number of button presses: 1
Number of keystrokes: 2
Fraction of user interface navigation: $1/3$
Fraction of clerical user data entry: $2/3$

*Due to the nature of this use case, it only has one possible scenario

4.2.4: UC-6: Finish Order

Scenario 1: Chef marks all food items as completed and confirms order completion

In this scenario, the chef will use n keystrokes to mark all n items ordered as complete. Afterwards, the chef will press the 'complete' button and then the 'next' button to move on to the next order. Then the waiter will receive a notification on their app and will have to use 2 keystroke one to view the notification and one to mark the order completed.

Number of button presses: 2
Number of keystrokes: $n + 2$
Fraction of user interface navigation: $2/(n + 4)$
Fraction of clerical user data entry: $(n + 2)/(n + 4)$

Scenario 2: Chef confirms order completion before marking all food items as complete

In this scenario, the chef forgot to mark all menu items as complete. So, after pressing the 'order complete' button, the chef will receive a notification with an error message about why the order completion was not processed. Then, the chef will have use 1 keystroke to open the notification. After reading the notification, the chef will mark the remaining n menu items as complete and press the 'order complete' button again. Then the waiter will receive a notification on their app and will have to use 1 keystroke to view the notification

Number of button presses: 2

Number of keystrokes: $n + 2$

Fraction of user interface navigation: $2/(n + 4)$

Fraction of clerical user data entry: $(n + 2)/(n + 4)$

Section 5: Project Management

Functionality

Our team will be split up into subdivisions of groups of 1, 2, or 3 people. The subdivisions that we believe will require more work will have 3 people, while simpler tasks may have only one. Here is the breakdown of the project tasks and who is assigned to each one:

#	Feature Subteams	Names
1	Mobile Applications	<ul style="list-style-type: none">• Taras Tysovskyi• Lieyang Chen• Hongpeng Zhang
2	Restaurant Website	<ul style="list-style-type: none">• Arushi Tandon• Yuwei Jin
3	Database and API Design	<ul style="list-style-type: none">• Yuwei Jin• Chris Lombardi• Taras Tysovskyi
4	Recommendation and Review Systems	<ul style="list-style-type: none">• Chris Gordon• Seerat Aziz
5	Ingredient Prediction and Reservation Systems	<ul style="list-style-type: none">• 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.

Timeframe

2/25 - 3/3	3/4 - 3/10	3/11 - 3/17	3/18 - 3/24	3/25 - 3/31	4/1 - 4/7	4/8 - 4/14	4/15 - 4/21	4/22 - 4/28	4/29 - 5/5	5/6 - 5/12
Report 2.1	Report 2.2	Final Report 2	Spring Break	First Demo		Report 3.1		Second Demo	Final Report 3 Reflective Essay	E-Archive Due
API Design										
Mobile Application Front End Dev										
Testing Facial Recognition API										
Website Front End Dev										
		Website		Admin Dev						
					Adding features to Website Admin					
Plan Menu										
Plan Rating System										
		Rec/Review Backend Dev								
Define Database Structure										
		Ingredient Pred.		Backend Dev						
		Reservation		System Backend Dev						
								Debugging/Tuning		

API: before working on everything else we need to agree on the API endpoints our server will expose. We plan for this to be done in the first 2 weeks. At this point front-end teams can start developing the apps and websites, while the back end team can work on actually writing the code that implements the API.

Mobile Applications: we are hoping to have basic customer, table, chef and serve apps done within the first 5 weeks. Since all the heavy lifting is done by the server, the apps will be just a front end that makes HTTP calls to the server and displays the data it receives from it. Chefs app is the simplest one to make, since it only has one view that displays the current orders. Table and Customer apps share a lot of the same views (menu, order, recommendations) which would allow us to reuse a lot of the same code.

Facial Recognition System: We first need to test the accuracy of a third-party software that can be integrated with our Android app (2-3 weeks). If the accuracy rate of the third-party software is low, then we will use OpenCV to implement this (2-3 weeks).

Recommendation and Review Systems: We first need to define our menu items and identify attributes that can be used in the quiz (1 week). Then we will develop an algorithm that takes the user's selected meals, "favorited" meals, and meal reviews to generate more accurate suggestions for the future (2-3 weeks). Moreover, we also have to design a review system that would work best for this subproblem (1 week).

Ingredient Prediction System: The bulk of the logic that the ingredient prediction system will use is based off of trends in order history, so it will make heavy use of the central database. As a starting point, the database structure will have to be well defined (~1 week). Beyond that, implementation and refinement of the prediction algorithm will be the next step (2 weeks). Any extensions to this system should be relatively simple, with only some minor tuning and adjustments needed (1 week).

Customer Reservation Systems: The customer reservation system is relatively simple, only requiring some database entries to be made on the back end. (1 week) The majority

of work involving the reservation system lies in interfacing with other aspects of the overall restaurant automation system (~1-2 weeks).

Restaurant website: The overall structure of the restaurant website needs to be established first. The static elements of the restaurant website, such as the menu of the restaurant and the ordering and reservation pages will be completed first (2 weeks). The basic functionalities of the admin console, such as the statistics and ingredient prediction pages will be the next step (2 weeks). Since the website will provide the front-end UI for many of the component systems, interfacing with the various systems of the Restaurant Automation system will be completed throughout the semester as they achieve various degrees of completion. Additional features on the admin console, such as editing the menu will be stretch goals (2+ weeks).

Section 6: References

1. Why W8 - Fall 2018 Restaurant Automation Project
2. FoodEZ - Spring 2015 Restaurant Automation Project
3. Professor Marsic's Restaurant Automation Project Description
(<https://www.ece.rutgers.edu/~marsic/books/SE/projects/Restaurant/>)
4. Spyce (Boston restaurant with a robotic chef) -
<https://www.greenbiz.com/article/full-service-automation-restaurants-changing-food-industry>