11/6/2022

FoodForTech (FTT) Cafeteria Ordering System

Software Requirement

Specification

Prepared by:
David Archer
Tuan Ngo
Vorleak Yek
Guosheng Wei
Marissa Palos
Peter Tran

CALIFORNIA STATE UNIVERSITY, FULLERTON
CS 541
Professor Song-James Choi, Ph.D.

Table of Contents

1. Introduction

1.1 Purpose

The intended audience for the SRS is workers in the tech industry: software engineers, product managers, as well as those wherein every minute in their profession is crucial: they can’t afford to miss deadlines, nor can they be at their best if they are hungry because the distance from their workplace to their preferred meal location is long and leaves little room to eat a full meal. We will implement a software product that ensures great output for the organization whilst at the same time ensuring its employees are fed. After all, it will be beneficial as well as rewarding for organizations to implement a system that would allow their employees to buy meals online and have cafeteria staff either deliver the food to said employee’s workstation or hold the purchased meal for them at the cafeteria. A system like this would allow employees who utilize the services to cut down on distance traveled and enhance the likelihood that they will receive their desired choice of food. Both their productivity and the quality of their work lives would be enhanced, thereby increasing the likelihood of employees becoming happy which would, in turn, lead to a more productive output by the employees.

1.2 Scope

FoodForTech (FFT) is the web and mobile application software that will enable technology companies to provide an access to food that will enable their employees to cut down on time spent traveling (e.g., cycling, scooter, etc.) to have a meal. FFT will provide customers (end users) the ability to order the same food served in the company's cafeteria by way of the FFT app installed on their phone or by visiting the website foodfortech.com on their computer or mobile phone's web browser. FFT will permit end users to access the cafeteria menu, make food selections, and beverage selections, submit credit or debit card payments, and, lastly, the option of dining in or ordering for takeout.

The goal of FFT is twofold, firstly, it will allow companies to cut back on overtime hours caused by employees missing their scheduled lunch breaks and later deciding to take the lunch break during time-crunching moments of the day when deadlines are due. FFT’s second goal will be to save employees time whether the employees have a one-hour lunch break or thirty minutes lunch break, and those employees' workstations are located at a great distance from their company’s cafeteria, and as a result of that, it takes them anywhere between twelve to twenty minutes of traveling time to arrive at the cafeteria for their meal; then FFT will fill that role by reducing time traveled for a meal to five minutes, specifically for employees that prefer to eat in the cafeteria. FFT will serve the role of reducing time traveled by allowing employees to order their food or beverages any time of the day during the company operating hours.

What will FFT do?

- FFT will utilize third-party services such as DoorDash or UberEats to deliver food picked up from the company's cafeteria to the intended FFT end users' workstation
- If the end user's workstation is in an access restricted location, based on the end user's instruction, FFT will relay the message and instruct the third-party service DoorDash or UberEats delivery person to await the end user at the specified food drop-off location

What will FFT not do?

- FFT does not prepare the meals or beverages being purchased
- FFT does not regulate the associated technology company's timesheet or lunch break clock-out time
- FFT makes no promises to keep employees from skipping their lunch breaks
- FFT does not tip third-party food deliverers nor does it instruct its end users to do so. The decision is at the end user's discretion

1.3 Definitions, acronyms, and abbreviations

COS	Cafeteria Ordering System.
FFT	FoodForTech - is the name of the application software that performs the function of allowing end users to submit a food order and payments.
End users	An employee who uses FFT's mobile app or FFT's website to order food
App	Application - also known as application software.
UML	Unified Modeling Language - outlines a

	collection of common notations used, notably in object-oriented software development, to create diverse visual representations of systems.
CPU	Central processing unit - interprets and executes instructions in the form of machine code.
RAM	Random-access memory - stores operational data and machine code.
HDD/SSD	Hard Disk Drive/Solid State Drive - a device for storing and retrieving digital data.
DoorDash	An American company that operates an online food ordering and food delivery platform.
UberEats	An online food ordering and delivery platform launched by Uber in 2014.
Two-Factor Authentication (2FA)	An identity and access management security method that requires two forms of identification to access resources and data: the end user's device and the code generated by the 2FA. On the organizational side, 2FA gives businesses the ability to monitor and help safeguard their most vulnerable information and networks.
Amazon Web Services (AWS)	Provides on-demand cloud computing platforms and APIs to individuals, companies, and governments.

Hypertext Transfer Protocol Secure (HTTPS)	It is used for secure communication over a computer network and is widely used on the Internet.
JSON	An open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute-value pairs and arrays.
Megabyte (MB)	Is a measurement of binary data.
Megabits or megabits per second (Mbps)	Is a unit of measurement for data size, most often used in describing data transfer rate. It takes eight megabits to make a megabyte.

1.5 Overview

The Software Requirements Specification document contains six main sections with each main section having one to seven sub-sections. Each sub-section details and gives a clearer understanding of the main section.

The Software Requirements Specification is organized in numerical order and it deals with FoodForTech's goals, objectives, and purpose of the software system. Additionally, activity diagrams are used to highlight FoodForTech's food ordering process:

- browsing the cafeteria menu
- selecting a meal from the menu
- placing the meal in the shopping cart
- selecting the preferred time of pickup, delivery, or table reservation
- finalizing the order and making the payment via a debit card or a credit card

2. Overall Description

2.1 Product Perspective

The project is based on the COS through a web and mobile application FoodForTech has some unique features allowing users to order meals from some nearby restaurant contract with our system. Moreover, FoodForTech provides delivery to your company's designated location in a short time and at a reasonable price compared to other systems. This app will integrate with multiple application delivery, such as DoorDash, Uber, and e.t.c, to improve the user experience by increasing the option to choose from. In addition, users and deliverers can give feedback about the system, food, and restaurant to improve the system's quality and services. Our app can save more time for employees and give them more choices with their meals.FoodForTech can replace the traditional method, such as ordering manually or via telephone.

2.2 Product Features (FEs)

FE-1 - Order meals from the cafeteria menu to be picked up or delivered. End users can go through the FFT website to place a meal order, and choose two options which are pick up or delivery.

- End users may place a meal from the cafeteria menu
- End-users have options to pick up their meals at the local restaurant or the designated location at their company.
- The menu manager gets the requests from end-users and then creates the request for restaurants and deliverers.

FE-2: Order meals from local restaurants to be delivered. Suppose the cafeteria does not satisfy end customers. In that case, they can order from nearby local restaurants, which are contracted with the FFT company, and choose from two options: pick up or deliver.

- End users should be able to place meals from local restaurants contracted with FFT COS.
- End users should be able to choose to be delivered to a designated location.

FE-3: Create, view, modify and delete meal service subscriptions. End users can register a subscription or cancel it on their profile display. They can see the prices of subscriptions.

- Users can sign up, view, modify, and unsubscribe from the meal subscriptions.

FE-4: Register for meal payment options. If the end users choose their company cafeteria, the company will take the meal payment from their salary. Moreover, if the end user chooses to pay from a visa card or a mastercard, then they have to provide their card information.

- Users can add their credit card information so that they can quickly use it during the ordering process

FE-5: Request meal delivery. End users can pick up or deliver by their local restaurant if it is available or a third party such as DoorDash or Uber, e.t.c.

- Users can choose the delivery option and specify the location for the meal to be delivered

FE-6: Create, view, modify, and delete the cafeteria menu. This feature contains the display, which can support managers, or staff who have responsibility to change their menu, prices, for their cafeteria.

- Managers or staff who are responsible to manage the menu for their restaurant should be able to create, view, modify, and delete for their restaurant via the COS.

FE-7: Order custom meals that aren't on the cafeteria menu.

- Users can place an order for meals that are not listed in the cafeteria menu

FE-8: Rating :End users can rate their orders and comment on the food and the delivery experience. Rating will display from 0 to 5 stars and have a comment box on the bottom to gather customer feedback for their order.

- Users can rate how satisfied they are with the meal at the restaurant that they had placed the order.
- The rating scale is from 1 to 5 stars with 5 being the best and would definitely recommend it to other people

FE-9: Provide live chat with customer representatives. Live chat will ask customers common questions and try to answer those common questions by putting the answers on the website. However, if the answers cannot satisfy customers, they can choose to chat with our customer support.

- Users can click on the customer service icon to chat with a live customer service representation from 8:00 AM to 5PM

FE-10: Multiple payment methods. Users can pay with multiple platforms such as Apple pay, Samsung pay, Visa card, Master card, Cash (pick up only) or Paypal.

- Users can save multiple credit or debit cards so that they can use it quickly during the meal check out.
- The system will save and store it securely in the database.
- Only only the last four digits of the cards will be displayed.

FE-11: Access FFT outside the cafeteria intranet. End users can use the intranet to access our FFT, and they can still access FFT with the internet, but they cannot choose their company payment methods.

- Users can order the meal anywhere without having to login to the intranet
- Users have the option to login via Facebook or Google

2.3 User Classes and Characteristics

Customer: A customer is an employee of a company that signed a contract with our system. Customers can view a menu, select desired items, and order them. After they finish choosing items, they can decide to pick them up at a local restaurant or get them delivered to a designated location in their company. Later, they can pay the bill by multiple provided payment methods via payment gateways.

Deliverers: After a customer pays their bill, if they choose to deliver to the designated location of their company, then deliverers from a local restaurant have a priority to do it. After 3 minutes, if a local restaurant can't confirm delivery, our system will send a request to an external food delivery system like DoorDash, Uber, etc. Later, when deliverers accept a request, they must go to the restaurant to pick up and deliver to a designated location on time.

Cafeteria Staff/ Restaurant Staff: a staff member can be local restaurant staff or cafeteria staff, who is responsible for confirming orders, cooking meals, and packing them for delivery. They need to be trained to use some basic computer to use the app.

Menu Manager: A menu Manager is a local restaurant manager or cafeteria manager responsible for establishing and maintaining the whole week's menu of food items. They can choose which items cannot be delivered and create a special menu for some specific day. Moreover, they can also change the price of items and refuse online orders based on the inventory stocks.

2.4 Operating Environment

OE-1: Final product shall support multiple web browsers including Chrome, Firefox 90+, and Safari 13+.

OE-2: Mobile version shall support both IOS and Android.

OE-3: both web and mobile shall support multiple payment gateways such as Visa, MasterCard

OE-4: Mobile version shall support Android Pay and Apple Pay.

OE-5: COS shall permit user access from the corporate Intranet.

OE-6: The client-side browser must operate within common web-browser environments using Secure Sockets Layer (SSL) / Transport Layer Security (TLS) cryptographic protocols at a minimum encryption level of 128 bits.

Server Side:

- Operating System: Red Hat Enterprise Linux Server 8
- DMBS: PostgreSQL
- Web Server: Apache

Client Side:

- Operating System: Windows/Linux/macOS/ Android / IOS
- Browser: Firefox, Internet Explorer, Chrome, Safari, and Microsoft Edge

2.5 Design and Implementation Constraints

CO-01: All HTML code shall conform to the HTML 5 standard

CO-02: All cascading Style Sheets shall use the latest version of Bootstrap 5, Version 5

CO-03: The system shall use the current version of PostgreSQL while supporting up to two earlier version releases. Currently, those include Version 14, and support versions 13 and 12.

CO-04: Software must be multilingual, including the following languages: English, Spanish, French, German, Japanese, and Mandarin.

CO-05: The system shall comply with all Accessibility, Web Design, and Security policies applicable.

2.6 User Documentation

UD-01: A tutorial will provide a quick start and a walk-thru of major system features.

UD-02: The system shall provide the end-users with the privacy policy

UD-03: An online form will enable users to request help, and frequently asked questions will be screened for the FAQ pages.

UD-04: The user's guide will contain:

UD-04.1: Overview of the system feature, and architecture.

UD-04.2: Instruction for accessing the system

UD-04.3: Samples of screens.

2.7 Assumptions and Dependencies

AS-01: The system is open for breakfast, lunch, and early dinner on weekdays.

AS-02: The restaurant shall provide at least one order online app such as Uber, DoorDash, etc.

AS-03: Deliverers will be available to deliver the food to nearby customers.

AS-04: Cafeterias and restaurants must have internet.

DE-01: The customer and payment gateways are passed with every transaction.

DE-02: The system is dependent on the end-users and the payment gateways and transactions must be approved.

DE-03: The system would be dependent on rules and regulations by government authorities.

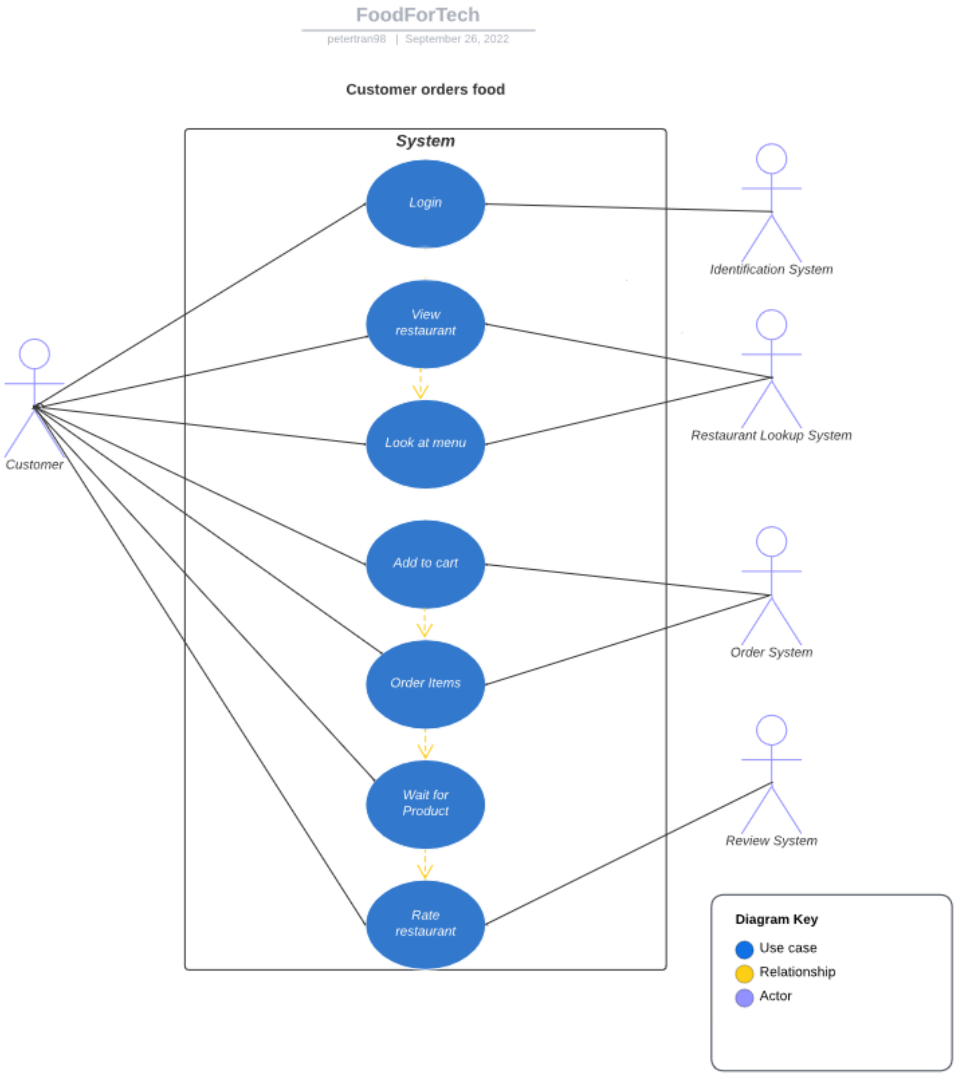
DE-04: The system available items depend on changes in the Cafeteria Inventory System which is another feature of FoodForTech.

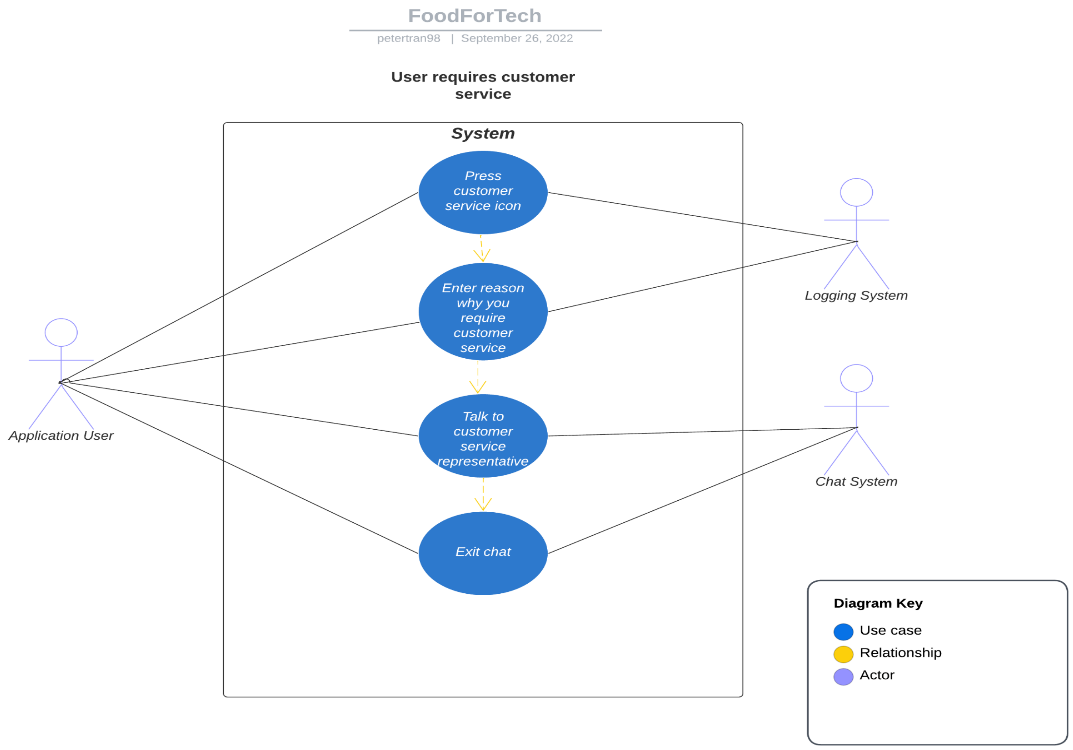
3. Use-Case Diagram with Use-case descriptions (UCs)

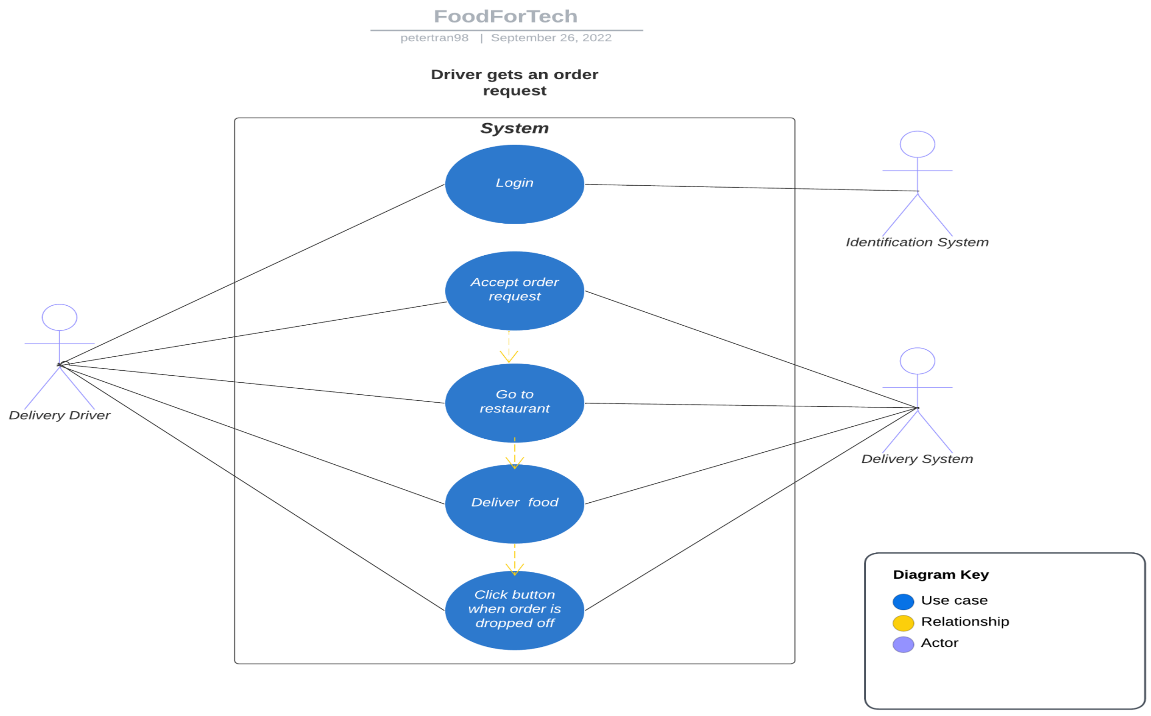
A use case is a sequence of interactions between a software application and an external actor or scenario in order to fulfill a business goal or requirement.

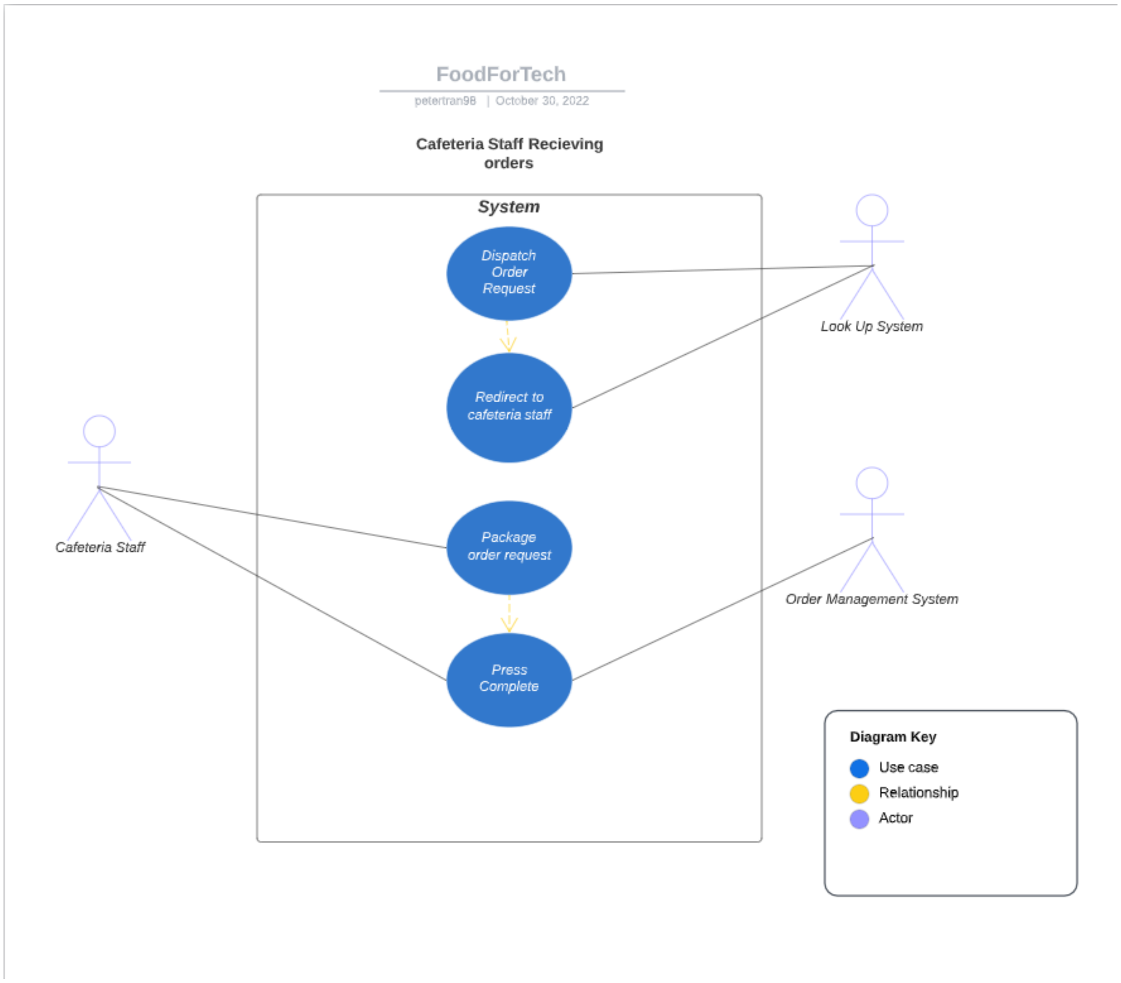
3.1 Use-case Diagram

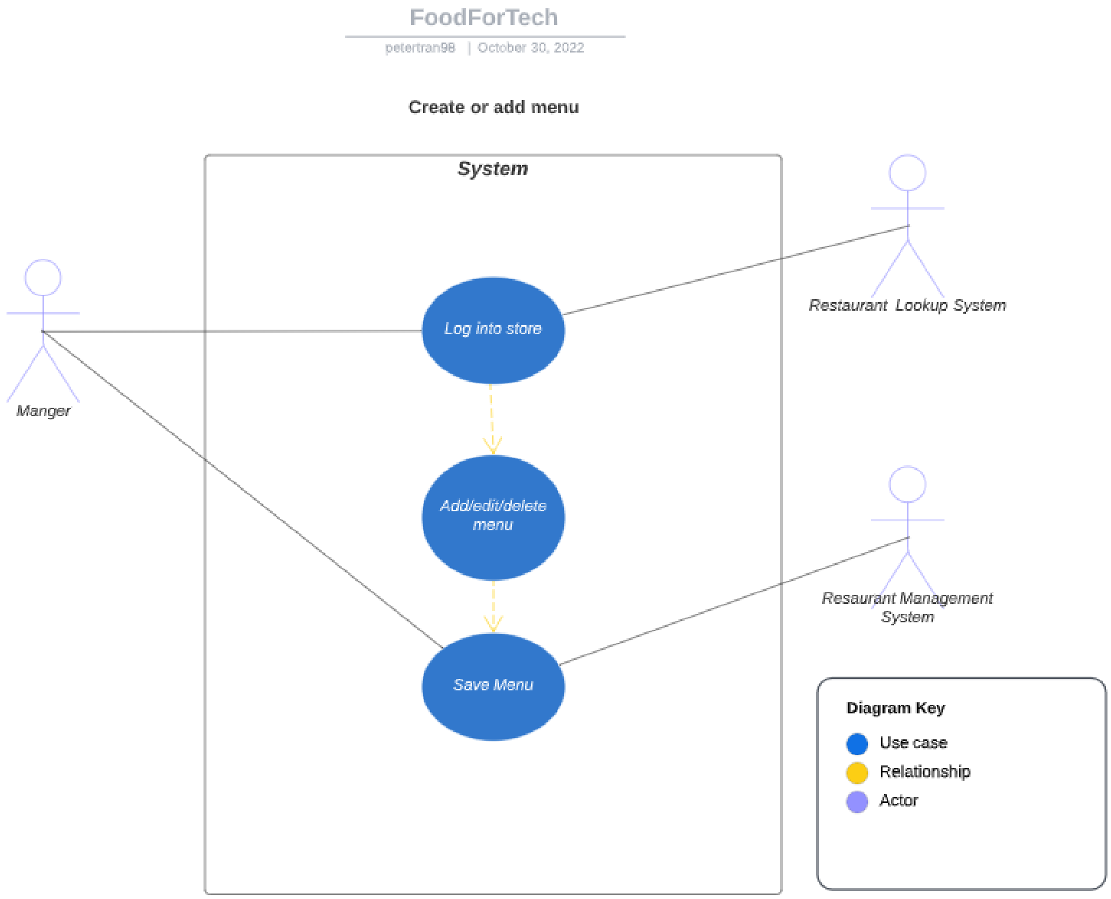
UML Use-case Diagrams: UML diagrams are a good candidate to show a high-level view of the main functional requirements for the FoodForTech application. It is an easy-to-understand graphical diagram made to represent use cases, actors, and system processes.



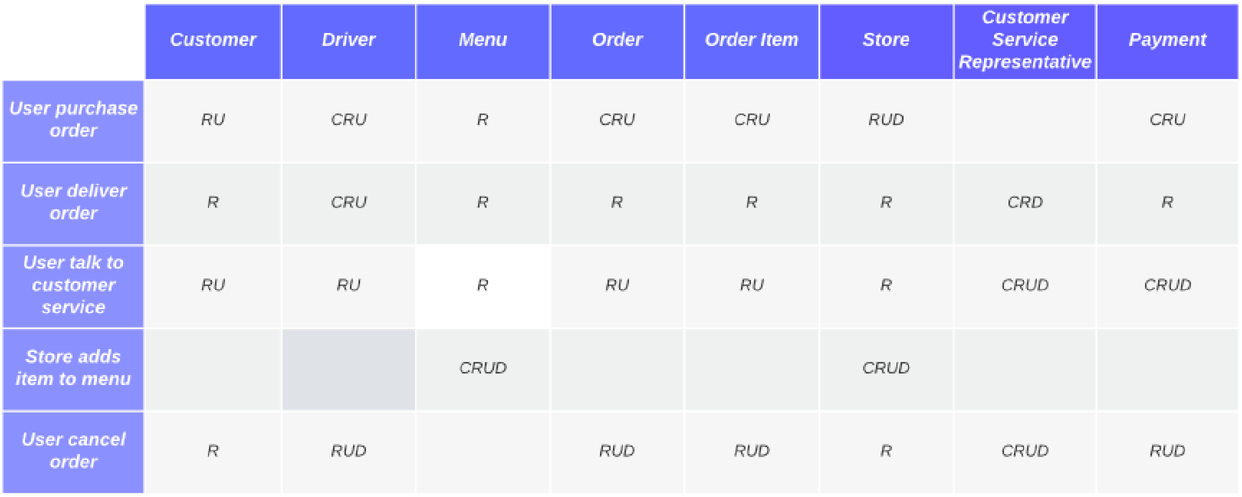

FoodForTech
petertran98 | September 26, 2022
User requires customer service
System
Press customer service icon
Enter reason why you require customer service
Talk to customer service representative
Exit chat
Application User
Logging System
Chat System
Diagram Key
Use case
Relationship
Actor


FoodForTech
petertran98 | September 26, 2022
Driver gets an order request
System
Login
Accept order request
Go to restaurant
Deliver food
Click button when order is dropped off
Delivery Driver
Identification System
Delivery System
Diagram Key
Use case
Relationship
Actor


FoodForTech
petertran98 | October 30, 2022
Cafeteria Staff Recieving orders
System
Dispatch Order Request
Redirect to cafeteria staff
Package order request
Press Complete
Look Up System
Cafeteria Staff
Order Management System
Diagram Key
Use case
Relationship
Actor



CRUD Matrix Diagram: A CRUD Matrix is a data analysis tool that works well for identifying missing requirements. The words “Create, Read, Update, and Delete" are abbreviated as "CRUD." These verbs are to describe the main fundamental system actions that represent how data entities are modified or transferred within your system. Use cases are placed on the left side of a CRUD matrix, and the entities are placed on top. As you add the CRUD acronyms to the diagram, keep an eye out for any vertical rows that are empty. These columns will be used to highlight any requirements that may have been forgotten or ignored. For example, if one of your entities is missing a delete action, you can ask questions such as does this entity have a delete feature?



	Customer	Driver	Menu	Order	Order Item	Store	Customer Service Representative	Payment
User purchase order	RU	CRU	R	CRU	CRU	RUD		CRU
User deliver order	R	CRU	R	R	R	R	CRD	R
User talk to customer service	RU	RU	R	RU	RU	R	CRUD	CRUD
Store adds item to menu			CRUD			CRUD		
User cancel order	R	RUD		RUD	RUD	R	CRUD	RUD

3.2 Use-Case Description(s)

ID and Name:	Enter name and unique identification for use case		
Created By:	Who was this use case written by	Date Created:	Date on which the use case description was written
Actors:	Internal/External system entity or people that interact with the use case		
Primary Actor (Initiate)	Primary entity that initiate the use case		
Description:	Description of the use case		

Trigger: (Optional)	Event or action that starts this use case
Preconditions:	A prerequisite that needs to be met before the use case can be implemented. These can include system process, environment, variables, and events.
Postconditions:	State the system is in after the use case has been implemented.
Normal Flow Path:	The basic flow of events for use cases.
Alternative Flow Path: (Optional)	Alternate path through the normal flow path.
Nonfunctional Requirements	Description of the use case's non-functional performance, security, quality of service, and reliability requirements
Priority:	The importance of the use case in comparison to the rest of the use cases.
Frequency of Use:	How often is the use case being used?
Assumptions:	Statements that are true about the system when the use case is activated?

ID and Name:	UC-1 Create an account		
Created By:	Peter Tran	Date Created:	10/29/22
Actors	● Customer ● Driver ● Registration service		
Primary Actor (Initiate)	● Customer ● Driver		
Description:	● Prospective clients who want to use the application must first fill out a registration form with username and password .		
Trigger: (Optional)	● User click on registration button		
Preconditions:	● Registration form must be filled out		
Postconditions:	● Success message stating that account has been created		
Normal Flow:	1. User open application 2. User navigate to navigation menu 3. User press register 4. New registration form is presented 5. User enter username 6. User enter password 7. User enter birthdate 8. User enter username		

	9. User press login 10. Success dialog is presented
Alternative Flow Path: (Optional)	1. Start at step 4 2. User enter information 3. User press register 4. Dialog prompt with error message 5. User fix errors 6. User press register 7. Success dialog is presented
Nonfunctional Requirements	● Registering service works in under .5 seconds per click event.
Priority:	● Low
Frequency of Use:	● Register service must support 100 request minimum per second
Assumptions:	● Database is fully functional ● Registration data is stored in database

ID and Name:	UC-2 Login to the system		
Created By:	Peter Tran	Date Created:	10/29/22
Actors:	● Customer		
Description:	● A registered user wants to log in to the system so they can use the application and all of its features.		
Trigger: (Optional)	● User click on login button		
Preconditions:	● The user must have completed the registration process and already have an account.		
Postconditions:	● Success message on upper corner that says “logged in successfully” ● Navigation menu has username listed ● User has access to application		
Normal Flow:	1. User open application 2. User navigate to navigation menu 3. User press login 4. User enter username 5. User enter password 6. User press login		
Alternative Flow Path: (Optional)			

ID and Name:	UC-2 Login to the system
	- The user enters the wrong password, prompt with incorrect password. Repeat step 5.
Priority:	- Low
Frequency of Use:	- Login service must support 200 request minimum per second
Assumptions:	- Database is fully functional - Login Service stores login data in database - Other system are full functional and connected

ID and Name:	UC-3 View and choose Restaurant		
Created By:	Peter Tran	Date Created:	10/29/22
Actors	- Customer - Tech Workers - Authentication service - Restaurant Lookup system		
Primary Actor (Initiate)	- Tech Customer		

Description:	● Give users a list of all nearby restaurants and the items on their menus.
Trigger: (Optional)	● The customer requests to view the food menus of all restaurants that are currently open.
Preconditions:	● User must be authenticated
Postconditions:	● The menu of the restaurant is presented after the trigger is activated.
Normal Flow:	1. User open application 2. User navigate to navigation menu 3. User press restaurants tab 4. User scroll through restaurants 5. User click on a restaurant 6. New page is presented with menu items 7. User scroll through menu items
Alternative Flow Path: (Optional)	1. Start at step 2 2. User go to recently ordered tab 3. User is presented a list of restaurants that order from recently 4. User click on restaurant 5. User scroll through menu items
Nonfunctional Requirements	● Gps service is updated once every time user is using application

Priority:	● High
Frequency of Use:	● Main functional requirement, must be operational every time the system is online. ● On average 100 requests per minute.
Assumptions:	● Gps service is working ● User is within 30 miles of the restaurant ● Menu items are presented with pictures and price ● Menu item data is retrieved from database

ID and Name:	UC-4 Order items		
Created By:	Peter Tran	Date Created:	11/03/22
Actors:	● Tech Customers ● Authentication service ● Restaurant Lookup system		
Primary Actor (Initiate)	● Tech Customers		
Description:	● User add menu items and has gone through the payment process to order items		
Trigger: (Optional)	● Click on “Make Payment” button		

Preconditions:	● User is logged into the system ● User has menu items > 0 in cart ● System has user’s address and credit card info
Postconditions:	● Success dialog saying, “Order has been placed”. ● Order receipt sent to user’s email ● Order management system is in order received state ● Delivery system is in order pending state
Normal Flow:	1. All steps in UC-3 normal flow 2. User add item to cart 3. User click on cart icon 4. Online payment form is presented 5. User fill out online payment form 6. User press next button 7. Selected Menu items are displayed for review 8. User press make payment 9. Redirected back to main page
Alternative Flow Path: (Optional)	
Nonfunctional Requirements	● Order Management System must support up to 200 requests per minute as long as the system is online.
Priority:	● Medium
Frequency of Use:	● Main functional requirement, must be operational every time the system is online. ● On average 100 requests per minute.

Assumptions:	● Order Management System is fully operational ● User is authenticated into application ● Payment is encrypted and secured ● Order items are saved to user's profile in database

ID and Name:	UC-5 Deliver food to Patron		
Created By:	Peter Tran	Date Created:	11/03/22
Actors:	● Tech Customers ● Deliver Driver ● Restaurant ● Delivery System ● Order Management System ● Restaurant Lookup system		
Primary Actor (Initiate)	● Deliver Driver ● Restaurant ● Tech Customer		
Description:	● Patron accepted order and is in the process of delivering it		
Trigger: (Optional)	● Click on "Accept order" button		
Preconditions:	● Tech Worker order food ● Deliver driver got matched with Tech Customer and accepted request		

	● Deliver driver is logged into system
Postconditions:	● Review service is activated ● User is logged on ● The database attributes for the tech customer, the delivery driver, and the associated restaurant are updated to reflect the delivered state.
Normal Flow:	1. All steps in UC-4 normal flow 2. Deliver driver accept deliver request 3. Deliver driver follow gps service to restaurant 4. Deliver driver accepts package from restaurant 5. Deliver driver press “On the way” button 6. Gps service shows route to tech customer’s address 7. Deliver driver follows Gps service 8. Deliver driver reach destination 9. Deliver driver drop off food 10. Deliver driver press “Order completed” button 11. Tech customer gets alert notifying order is delivered.
Alternative Flow Path: (Optional)	
Nonfunctional Requirements	● GPS service must be capable of handling 200 deliver drivers at a given time ● Software system capable of support real time location update
Priority:	● Medium
Frequency of Use:	● Main functional requirement, must be operational every time the system is online.

	● On average 10 requests per minute.
Assumptions:	● Restaurant received notice of the order ● Restaurant is in the process of making order ● The order was placed by an authenticated customer ● The payment system has received payment.

ID and Name:	UC-6 Rate Order		
Created By:	Peter Tran	Date Created:	11/03/22
Actors:	● Tech Customers ● Deliver Driver ● Restaurant ● Rating Service ● Order Management System ● Restaurant Lookup system		
Primary Actor (Initiate)	● Deliver Driver ● Tech Customer		
Description:	● After an order has been fulfilled, the driver, client, and restaurant are all given the opportunity to post reviews. This review will include ratings for the restaurant, the driver, and the clientele.		

Trigger: (Optional)	- After the driver clicks on the "completed order" button.
Preconditions:	- Delivery state change from false to true before presenting review form - The restaurant must submit a completed order request. - Payment from the tech customer must go through successfully - Software system's state has passed UC-4 normal flow - Software system's state has passed UC-5 normal flow
Postconditions:	- User is still authenticated - Review data sent and stored in the database - User interface updated to match the new review.
Normal Flow:	1. Deliver driver press "Completed Order" button 2. Customer, restaurant, and delivery driver gets notified and review form is prompted 3. Customer, restaurant, and delivery driver has the ability to fill out form a. Customer, restaurant, and delivery driver fill out form and press "Submit Review" button b. Customer, restaurant, and delivery driver press "Decline Review" button
Alternative Flow Path: (Optional)	1. Locate navigation menu 2. Click on previous orders 3. Pick a previous order 4. Select "Review" button 5. Fill out review form 6. Click "Submit Review" button

Nonfunctional Requirements	● Rating service is online 23 hours of the day ● Each rating response gets processed and updated in the database within 10 seconds of submission. ● Delivery driver must maintain a minimum of 80% satisfaction score
Priority:	● Low
Frequency of Use:	● Ideally every order place will be reviewed ● On average 1 request per minute, 24 hours a day .
Assumptions:	● Users can only review orders they placed or receive ● Deliver driver has completed order ● Tech customer has received order

ID and Name:	UC-7 Talk to customer service representative		
Created By:	Peter Tran	Date Created:	11/04/22
Actors:	● Chat moderator ● Application User ● Chat System		
Primary Actor (Initiate)	● Application User		

Description:	● Users must be authenticated in order to access customer support. Whenever a user has a question they have the option to talk to a live customer service representative.
Trigger: (Optional)	● Clicked on customer support icon located on navigation menu
Preconditions:	● User is authenticated ● Chat System is online ● Chat moderator are available
Postconditions:	● The status of the current rating service is updated to "Pending review." ● Database gets updated to reflect that
Normal Flow:	1. Locate navigation menu 2. Click on customer service icon 3. Fill out prompt 4. Press “Talk to Customer Service” button on form 5. Talk to customer service representative 6. When the session is over, press “End Session” button
Alternative Flow Path: (Optional)	
Nonfunctional Requirements	● User waits no more than 10 minutes to hear back from the chat moderator. ● User messages are secured and data is protected ● Compatible with both android and apple phones ● Chat system can be ramped up during peak hours

Priority:	● Low
Frequency of Use:	● The system receives an average of one request every half hour while it is online.
Assumptions:	● Chat system is online ● User submits question during operation hours ● User is authenticated

4. External Interface Requirements

4.1 User Interfaces

UI-1: *Splash Screen*

The splash screen is the introductory window that displays when the user starts the application. The “Food for Tech” splash screen displays the logo, a fork, and a knife, alongside the app name. In branding, orange conveys boldness and energy.

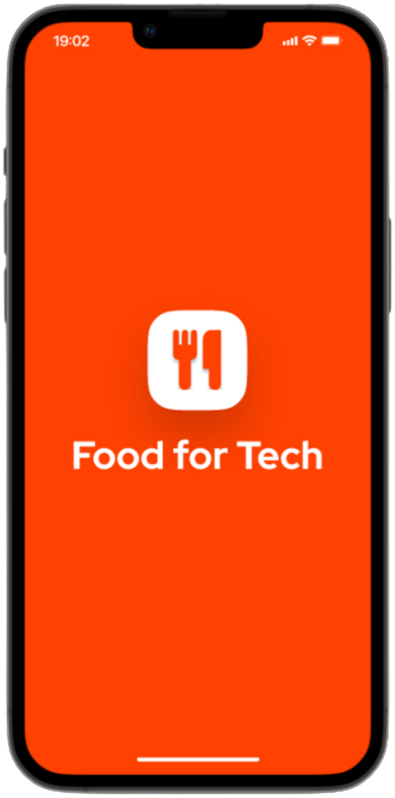
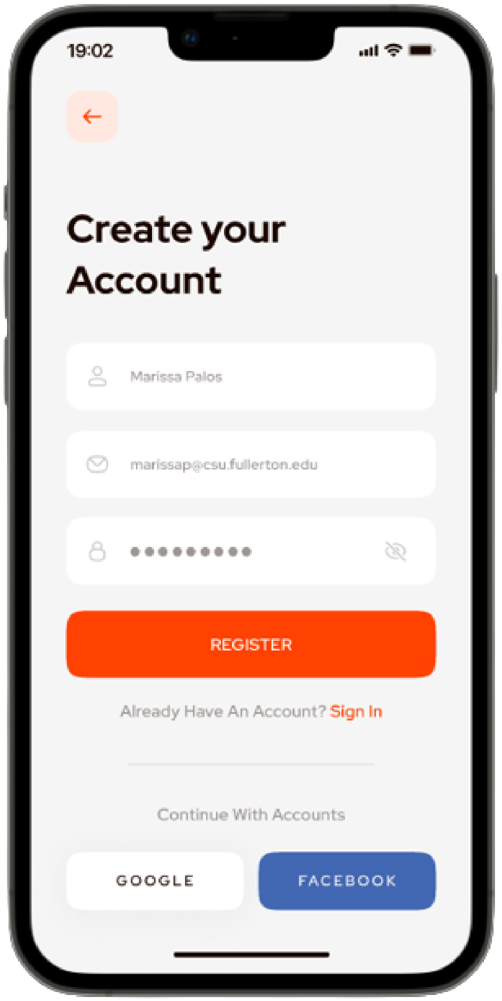


Figure 4.1

Splash Screen

UI-2: *Create your Account Screen*

First impressions are crucial - the “create account” screen is one of the first interactions a user will have with the app. Account creation must be user-friendly and easy. If the user encounters too many hurdles when simply creating an account - they will be more likely to discredit the app’s usefulness. In addition, Facebook and Google integration will expedite account creation. This is standard practice in modern applications.



Create your Account

Marissa Palos

marissap@csu.fullerton.edu

●●●●●●●●●

Already Have An Account? Sign In

Continue With Accounts

Figure 4.2

Create your Account Screen

UI-3: *Two-Factor Authentication*

The standard for security is two-factor authentication. When a user provides a phone number, an account is secured with two unrelated authentication methods. This prevents data breaches, an incident where information is taken from a system without authorization. FFT incorporates this additional layer of security.

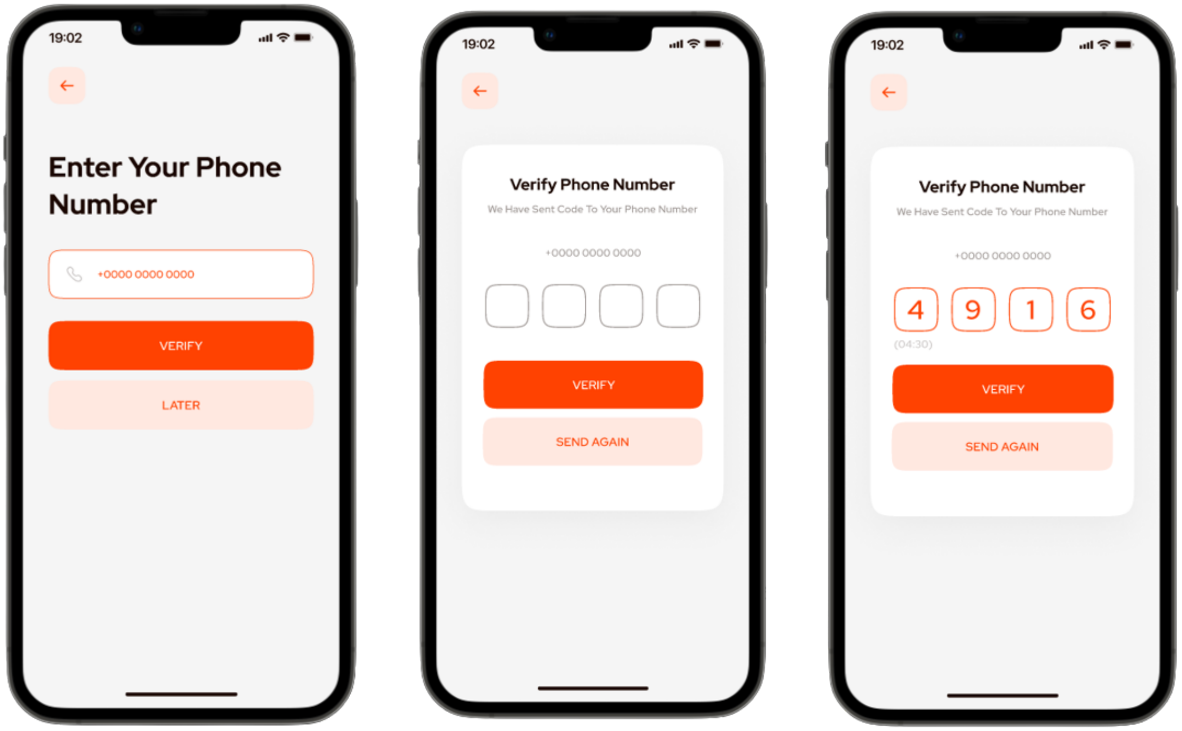


Figure 4.3

Two-Factor Authentication

4.2 Hardware Interfaces

HI-1:

FoodForTech uses Amazon Web Services (AWS) to host our application in a cloud environment. Rather than building and maintaining infrastructure for an on-premises application, FFT lowers costs by paying only for what we use. Owning and operating the infrastructure on which the app is hosted can be expensive. This cost can be mitigated by using Amazon Web Services.

HI-2: *Web Services*

For the Web and Application Server, Amazon EC2 (Amazon Elastic Compute Cloud) will be used. This allows for a virtual environment with the operating system, services, databases, and application platform stack required.

HI-3: *Database Server*

For the Database Server, Amazon S3 (Simple Storage Service) will be used. This allows for the application's data to be stored and retrieved securely.

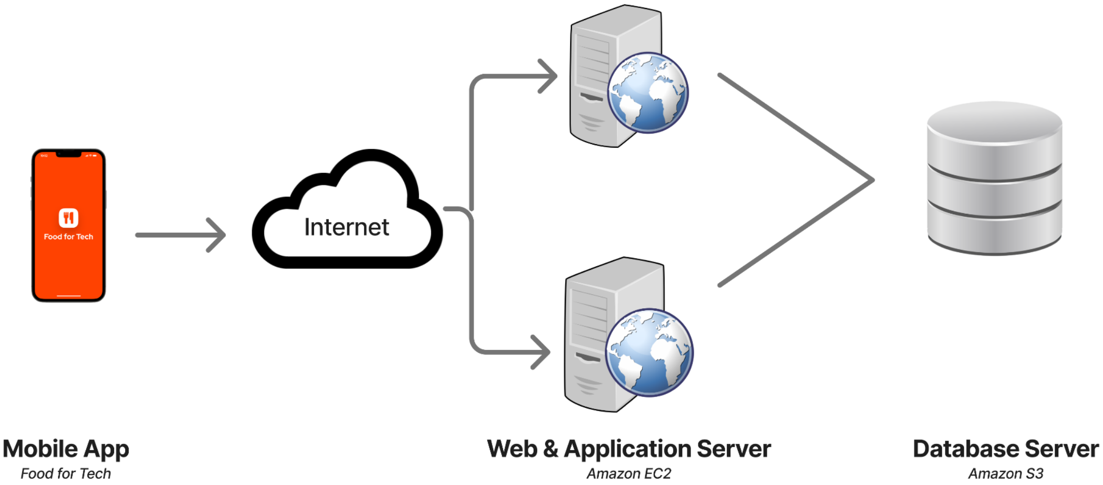


Figure 4.3

AWS Diagram

4.3 Software Interfaces

SI-1: *iOS requirements*

- The FoodForTech app requires iOS 13.0 or higher.

SI-2: It is compatible with the following iOS devices

- iPhone 5S or later
- iPad Air or iPad Air 2
- iPad Mini 2 or later
- iPad Pro (1st generation) or later
- iPad (5th generation) or later

- iPod touch (6th generation)

SI-3: *Android requirements*

- The FoodForTech mobile app requires Android 7.0 or higher.

4.4 Communications Interfaces

CI-1: Communication standards and Network server communications protocols:

For security purposes, HTTPS will be used - which will come with some additional cost of security certificates. JSON will be FoodForTech's data format.

CI-2: Message formatting:

Text and image messaging will be utilized for the restaurant to communicate with the customer, and vice versa.

CI-3: Data transfer rates:

The FoodForTech user can send a file with a maximum file size of 16 MB.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

PER-1: The web pages generated by the COS shall fully download in no longer than 5 seconds over 20 megabits/second Internet connection.

PER-2: The COS shall be able to request and authorize the credit card transactions within 3 seconds, and then display the confirmation message to users no longer than 5 seconds.

PER-3: The maximum number of records stored in the database of the COS shall not exceed 10,000 users. Any existing users that are not active within a year shall be removed from the data. Any transaction data that is created after 6 months shall be removed from the record.

PER-4: The system shall be able to handle a maximum number of 150 concurrent users during the peak hours of lunchtime from 11:30 A.M. to 1:00 A.M. local time.

PER-5: The COS shall be able to request and authorize the credit card transactions within 2 seconds, and then display the confirmation message to users no longer than 5 seconds after the users submit their orders.

PER-6: Each user shall be able to save their credit card information and use it during the checkout within a second.

PER-8: The system shall display a warning message for any older devices that are no longer supported on the homepage each time the user login.

PER-9: The system shall display any error message when users submit the order with a missing required field within 1 second.

PER-10: The system shall alert the user on their devices once the delivery has arrived at the ordering location within 3 seconds.

5.2 Safety Requirements

SEC-1: The system shall require all users to log in for any activities, except viewing the menu.

SEC-2: The system shall block a user's account for 10 minutes after five consecutive unsuccessful login attempts.

SEC-3: The system shall only allow users to log in to one device at a time, and automatically log the users out from other devices.

SEC-4: The system shall restrict patrons from viewing other users' orders.

SEC-5: The system shall store each credit card information and each personal identification securely using encryption.

5.3 Security Requirements

SAF-1: The system shall display a list of all ingredients for each menu item and emphasize the ingredients that are commonly known to cause allergic reactions to many people in the US.

5.4 Software Quality Attributes

AVL-1: The COS shall be available on weekdays from 8:00 A.M. to 5:00 P.M. The maintenance hours shall be scheduled outside of these hours.

ROB-1: The COS shall enable users to recover an incomplete order and continue working on it if the users somehow disconnected from the COS.

6. Other Requirements

6.1. Scalability

SCA-1: The COS shall be able to scale horizontally by adding more virtual machines to the server pool.

SCA-2: The COS shall be able to scale vertically by adding more system resources to the virtual machines, such as CPU, RAM, and disk storage (i.e. HDD or SSD).

6.2. Portability

POR-1: The COS must be able to launch within one environment or another without changing its behavior and performance.

6.3. Compatibility

COM-1: The COS must be able to maintain OS compatibility of two OS versions back.

COM-1: The COS must be compatible with an OS firewall or antivirus protection.

6.4. Maintainability

MAI-1: The COS shall be designed to have minimal downtime during system maintenance.

MAI-2: The COS shall have a modular design in which system components can be updated without taking the entire system offline.

MAI-3: The COS shall be designed in which new updates can be rolled back if there are any issues.

6.5. Internationalization

INT-1: The COS shall be designed to use UTF-8 encoding.

INT-2: The COS code shall be written in a way that software can be translated without changing a line of code.

INT-3: The COS code shall be designed with sufficient space to accommodate long strings in a different language.

INT-4: The COS shall be designed to use either imperial or metric systems in local standards.

INT-5: The COS shall be designed to put words in a different order in the destination language.

INT-6: The COS shall be designed to track and update changes.

6.6. Learnability

LEA-1: The COS shall be designed in which there is no learning or training required for users to use the software.

LEA-2: The COS’s use case shall be designed to minimize alternative flows.

7. Functional Requirements (FRs)

7.1 FR1: Order meals from the cafeteria menu to be picked up or delivered

a. **Introduction/Functionality**: This functionality will let users order meals from the cafeteria menu with the options to pick up or deliver to a specific location.

b. **Traced :** UC-3, UC-4

c. **Inputs**: User is logged in to the system, and then chooses the meal from the menu

d. **Processing**: Users can pay for the meal by entering the credit card information and the bank can process the payment transaction.

e. **Outputs**: The confirmation dialog will be displayed, and the order information will be stored in the order history.

f. **Error Handling**: Display an inline error message if there is an error with the credit card information and prompt the user to enter it again.

7.2 FR2: Order meals from local restaurants to be delivered

a. **Introduction/Functionality:** This functionality will let users order meals from local restaurants with the option to deliver.

b. **Traced:** UC-3, UC-4

c. **Inputs:** User is logged in to the COS from the FFT app, and then chooses the meal from the menu.

d. **Processing:** Users can pay for the meal by entering their credit card information and the bank can process the payment transaction.

e. **Outputs:** The confirmation dialog will be displayed, and the order information will be stored in the order history.

f. **Error Handling:** Display an inline error message if there is an error with the credit card information and prompt the user to enter it again.

7.3 FR3: Create, view, modify, and delete the meal service subscriptions

a. **Introduction/Functionality:** This functionality allows users to sign up, view, modify, and unsubscribe from the meal subscriptions.

b. **Traced:** UC-1, UC-2

c. **Inputs:** User is logged in to the COS from the FFT app, and then goes to the subscription to create, view, modify, or delete.

d. **Processing:** based on the user's choice, the COS will add, send, modify, or remove the user's information from the database.

e. **Outputs:** The confirmation dialog will be displayed, and the user's information should be updated accordingly in the subscription.

f. **Error Handling:** Display an error message if users did not provide all the required information.

7.4 FR4: Register for meal payment options

a. **Introduction/Functionality:** This functionality allows users to add their credit card information so that they can quickly use it during the ordering process.

b. **Traced:** UC-4

c. **Inputs:** User is logged in to the COS from the FFT app, and then goes to the credit card page.

d. **Processing:** The credit card information should be stored properly in the database.

e. **Outputs:** The confirmation dialog will be displayed, and the credit card information should be added properly on the credit card page.

f. **Error Handling:** Display an error message if the credit card information is invalid.

7.5 FR5: Request meal delivery

a. **Introduction/Functionality:** This functionality allows users to choose the delivery option and specify the location for the meal to be delivered.

b. **Traced:** UC-4

c. **Inputs:** Users will provide the location for the meal to be delivered.

d. **Processing:** The delivery location will be recorded in the system.

e. **Outputs:** The confirmation dialog will be displayed.

f. **Error Handling:** Display an error message if the location is out of the delivery distance or if there is any missing required information.

7.6 FR6: Create, view, modify, and delete cafeteria menus

a. **Introduction/Functionality:** This functionality allows the menu manager to create, view, modify, and delete cafeteria menus.

b. **Traced:** Add/edit/delete menu

c. **Inputs:** The menu manager is logged in to the FFT app to create, view, modify, and delete cafeteria menus.

d. **Processing:** based on the menu manager's choice, the COS will add, send, modify, or remove the cafeteria menus from the database.

e. **Outputs:** Display the confirm message, and the cafeteria menu should be updated accordingly.

f. **Error Handling:** Display an error message if managers did not provide all the required information.

7.7 FR7: Order custom meals that aren't on the cafeteria menu

a. **Introduction/Functionality:** This functionality allows users to place an order for meals that are not listed in the cafeteria menu.

b. **Traced:** UC-3, UC-4

c. **Inputs:** User is logged in to the FFT app, then select the type of food and the available ingredients to customize the meal.

d. **Processing:** the system should store the information and save it in the database.

e. **Outputs:** Display the confirm message.

f. **Error Handling:** Display any error if there any required field is missing.

7.8 FR8: Rating

a. **Introduction/Functionality:** This functionality allows users to rate how satisfied they are with the meal at the restaurant that they had placed the order. The rating scale is from 1 to 5 stars with 5 being the best and would definitely recommend it to other people.

b. **Traced:** UC-6

c. **Inputs:** Users can click on the stars for rating with 5 stars the most satisfied.

d. **Processing:** The system stores each user rating and calculates the average rate.

e. **Outputs:** Display the confirmation message "Thank you for your rating."

f. **Error Handling:** Prompt users to select 1-5 stars if they submit without any rating number.

7.9 FR9: Provide live chat with customer service representatives

a. **Introduction/Functionality:** This functionality allows users to live chat with customer service representatives during the available work hours if they have any questions or issues that they need to resolve with the system.

b. **Traced:** UC-7

c. **Inputs:** User is logged in to FFT

d. **Processing:** The system will alert the customer service representative when the users connect to live chat.

e. **Outputs:** Display a message that a customer service representative will be available as soon as possible.

f. **Error Handling:** Display an error message if there is a disconnection during the conversation.

7.10 FR 10: Multiple payment methods

a. **Introduction/Functionality:** This functionality allows users to save multiple credit or debit cards so that they can use it quickly during the meal check out.

b. **Traced:** UC-4

c. **Inputs:** Users provide the credit or debit cards information that they want to save.

d. **Processing:** The system stores each card information in a secure way.

e. **Outputs:** Display all the cards information, and only the last four digits of the cards.

f. **Error Handling:** Display an error message if the credit card information is invalid.

7.11 FR 11: Access FFT outside cafeteria intranet

a. **Introduction/Functionality:** This functionality allows users to order meals anywhere without having to login to the intranet. Users also have the option to login via Facebook or Google.

b. **Traced:** UC-2

c. **Inputs:** User is logged in to the system from the FFT user account or the Facebook or Google account.

d. **Processing:** The system will allow users to login if the login info is valid

e. **Outputs:** Display the FFT homepage after the user has logged in.

f. **Error Handling:** If the login information is invalid, prompt the use to enter the information again.

7.12 FR 12: Accept order request

a. **Introduction/Functionality:** This functionality allows the delivery driver to view the meal order placed and accept the order requested. The driver can look at the restaurant address.

b. **Traced:** UC-5

c. **Inputs:** The meal order requested from the patron including the restaurant address.

d. **Processing:** The system will alert the restaurant about the food delivery driver information.

e. **Outputs:** Display a confirmation message that the meal order has been accepted.

f. **Error Handling**: there is no error handling for this functionality.

7.13 FR 13: Pickup meal from the restaurant for the delivery

a. **Introduction/Functionality:** This functionality allows the delivery driver to get an alert from the restaurant owner whenever the meal order is ready to pick up.

b. **Traced:** UC-5

c. **Inputs:** Confirmation that the meal order is ready to pick up from the restaurant.

d. **Processing:** The system will alert the delivery driver after the restaurant owner has confirmed that the meal order is ready.

e. **Outputs:** Display a message with the order number and restaurant address.

f. **Error Handling**: Display an error message if there is any issue with sending the confirmation message that the food is ready to pick up.

7.14 FR 14: Packaging order request from the customer

a. **Introduction/Functionality:** This functionality allows the cafeteria staff to view each meal order from the patrons. They should be able to prepare and pack the food based on each of the orders.

b. **Traced:** Package order request

c. **Inputs:** The meal order from the patrons.

d. **Processing:** The system shall include each meal order information under a separate link with unique identification.

e. **Outputs:** Display the meal order information when the cafeteria staff click on the link.

f. **Error Handling:** Display an error if there is any issue with loading an order.

7.15 FR 15: Tracking meal order status

a. **Introduction/Functionality:** This functionality allows patrons to track the meal order status and track where it is during the delivery.

b. **Traced:** Wait for product

c. **Inputs:** The meal order status from the restaurant and the current location from the GPS.

d. **Processing:** The system automatically updates the meal order status and the location of the delivery.

e. **Outputs:** The patrons can view if the meal order for delivery has been accepted, in preparation, or delivery includes the current location.

f. **Error Handling**: Display an error message if the current location cannot load properly.

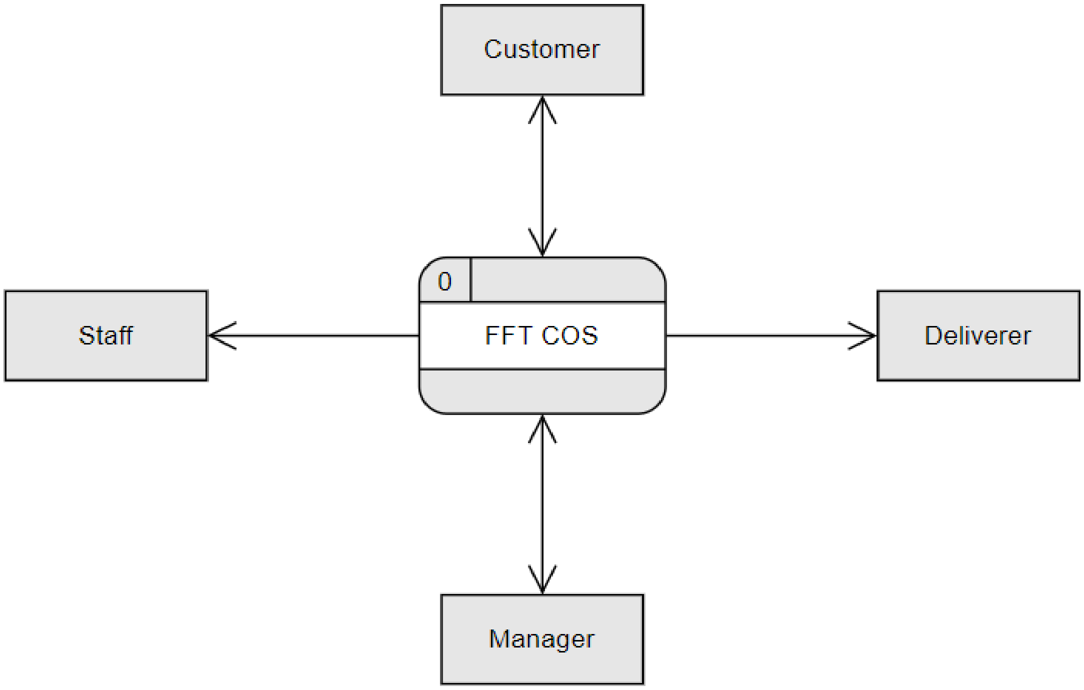
TABLE 2: Traceability Matrix (Use-cases & FRs).

	Related FRs
UC-01	**FR-3**
UC-02	**FR-3 , FR-11**
UC-03	**FR-1, FR-2, FR-7**
UC-04	**FR-1, FR-2, FR-4, FR-5, FR-7, FR-10**
UC-05	**FR-12, FR-13**
UC-06	**FR-8**

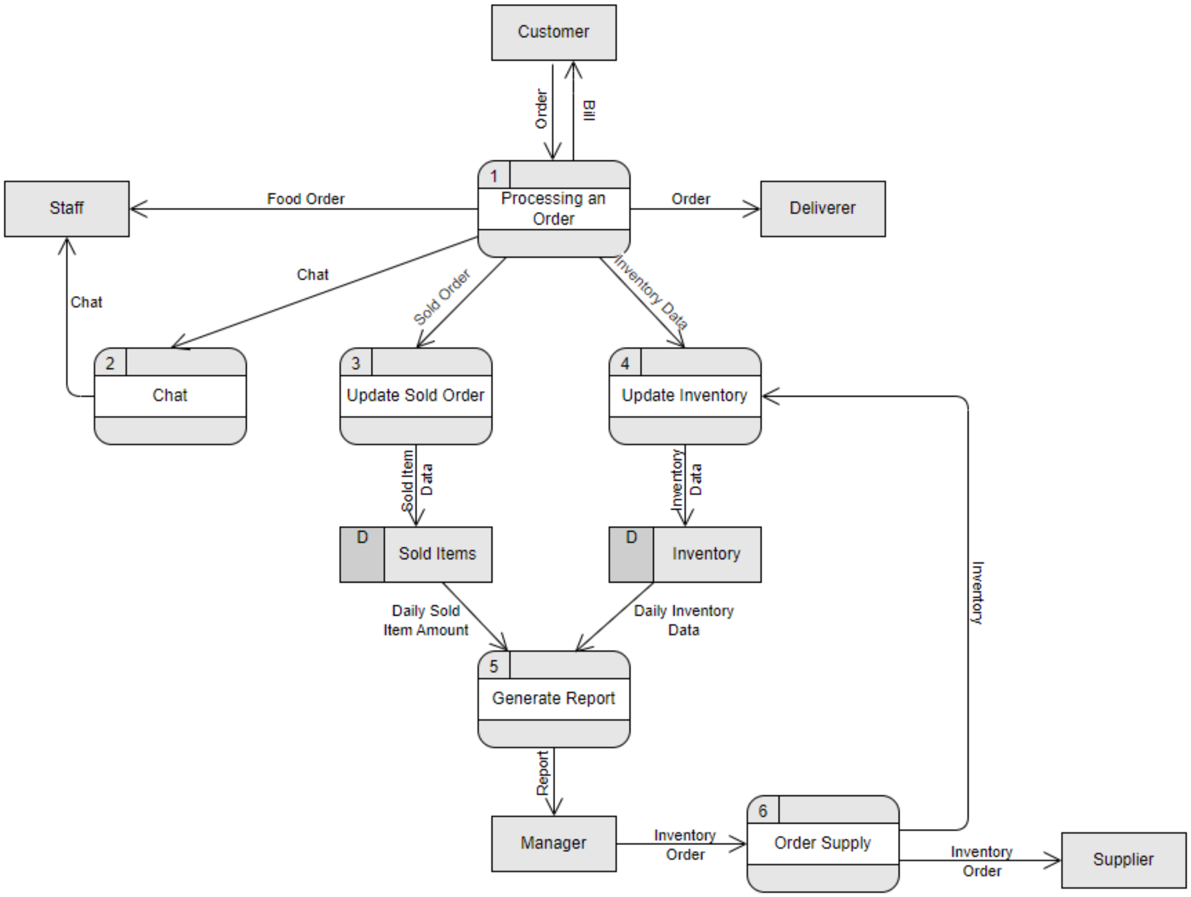
UC-07	**FR-9**

8. Functional Modeling (DFDs)

8.1 Context Diagram (level 0)

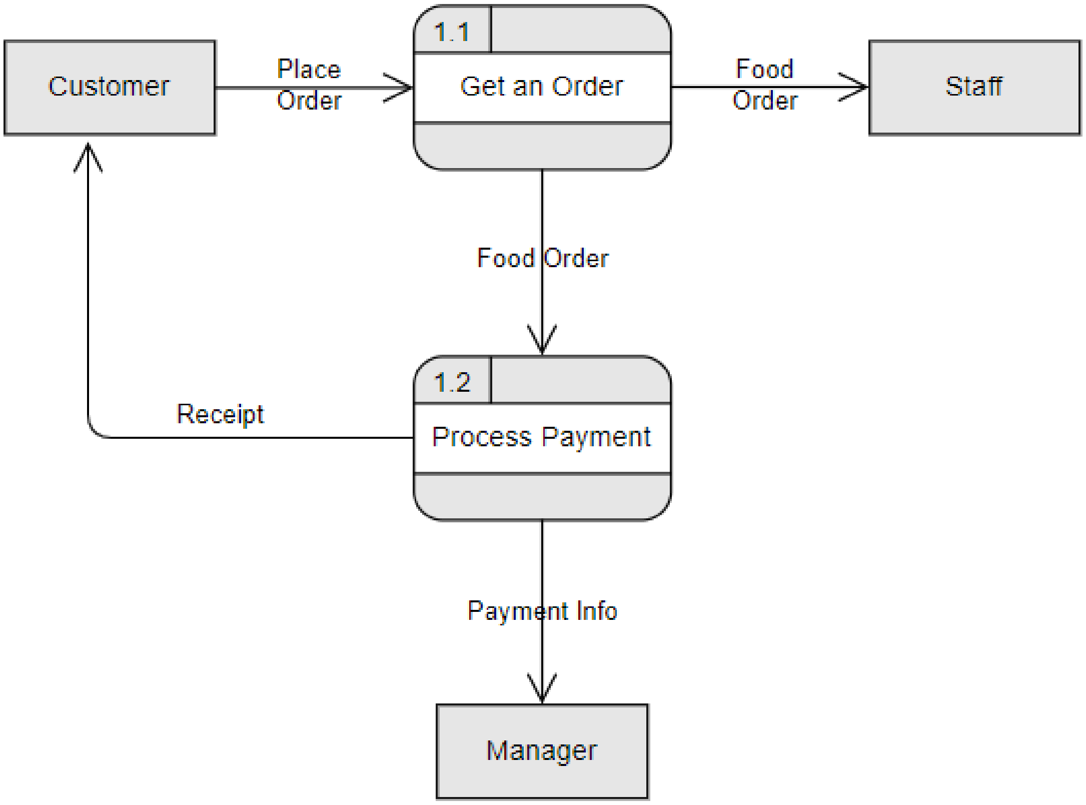


8.2 Level 1

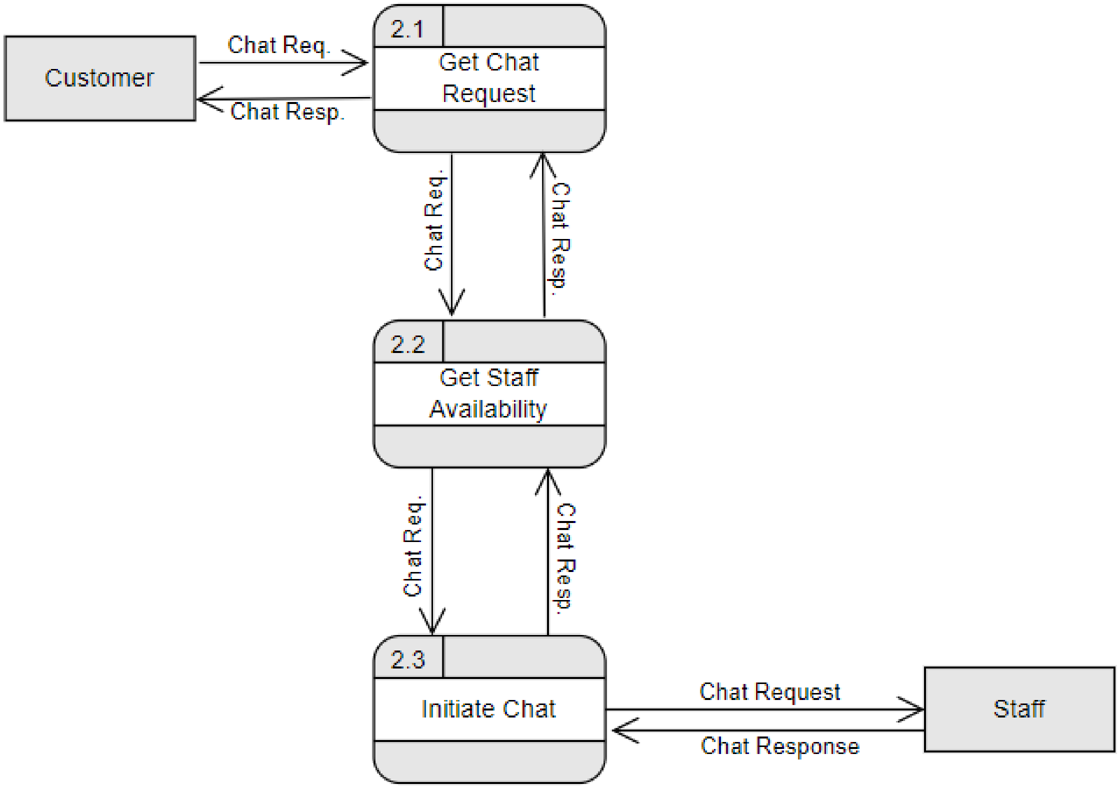

Customer
Order
Bill
1
Processing an Order
Staff
Food Order
Order
Deliverer
Chat
Chat
Sold Order
Inventory Data
2
Chat
3
Update Sold Order
4
Update Inventory
Sold Item Data
Inventory Data
D
Sold Items
D
Inventory
Inventory
Daily Sold Item Amount
Daily Inventory Data
5
Generate Report
Report
Manager
Inventory Order
6
Order Supply
Inventory Order
Supplier

8.3 Level 2

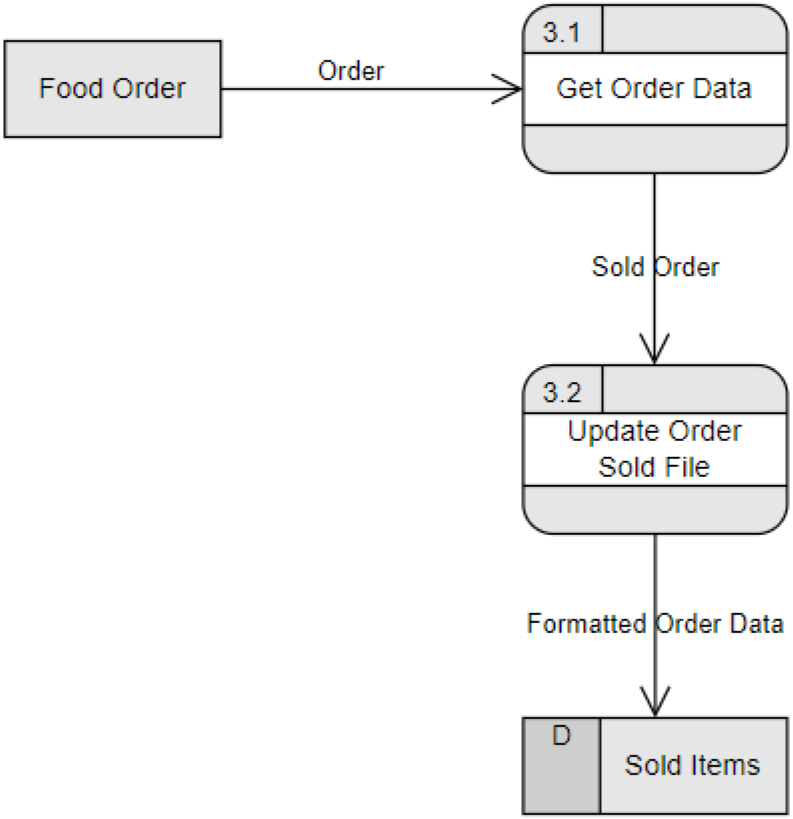
1. Process an order



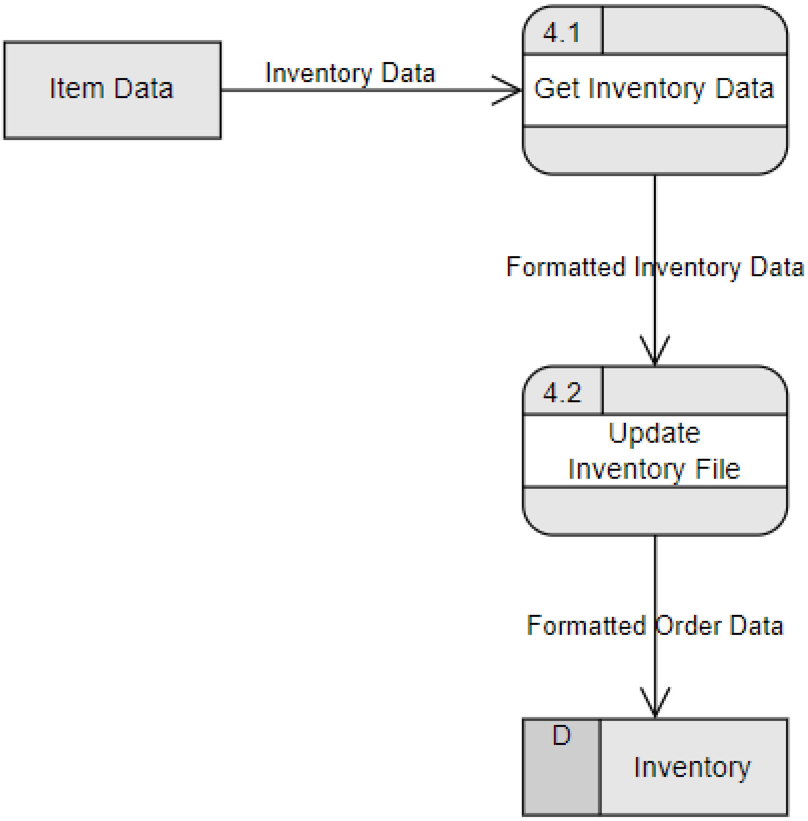
2. Process a chat



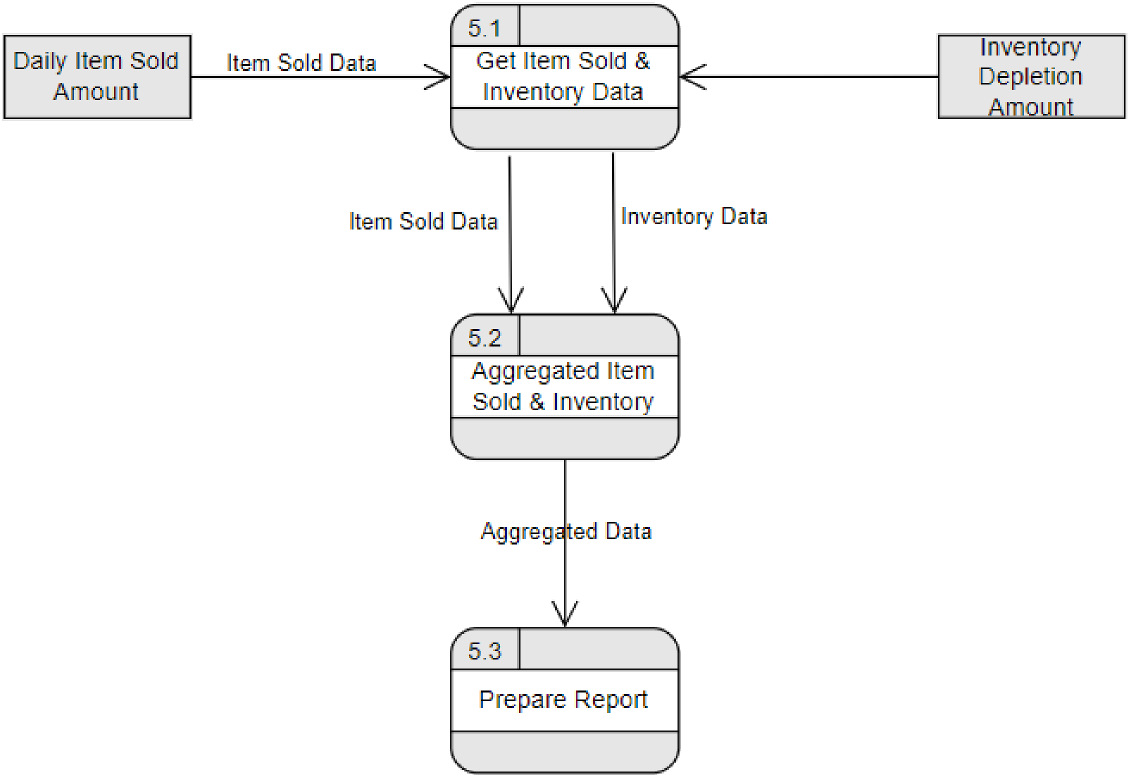
3. Update Sold Orders



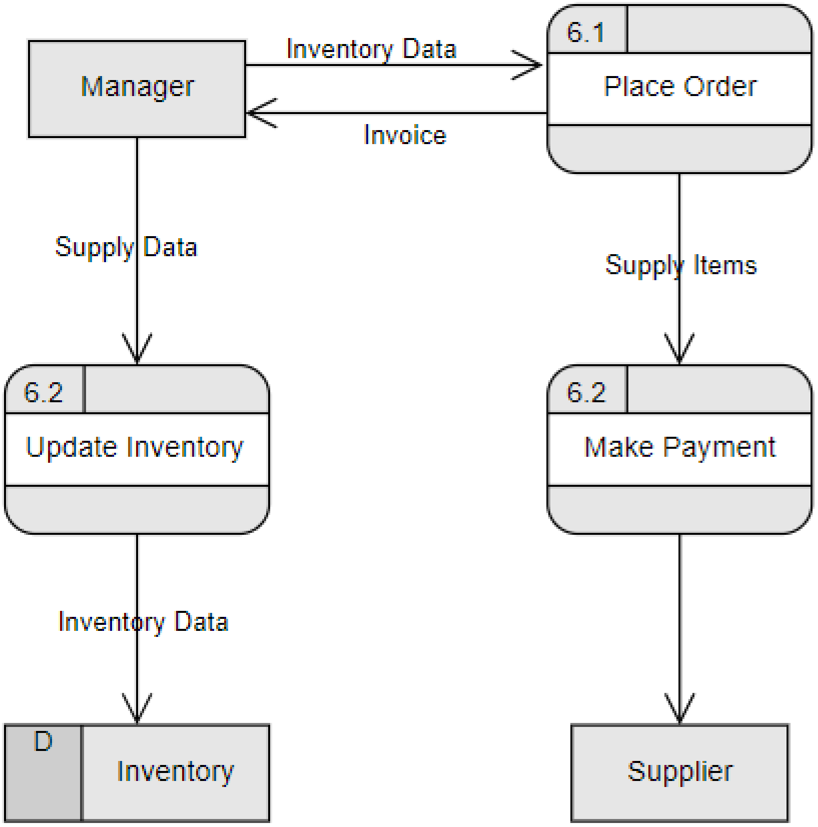
4. Update Inventory



5. Generate Report



6. Order Supply



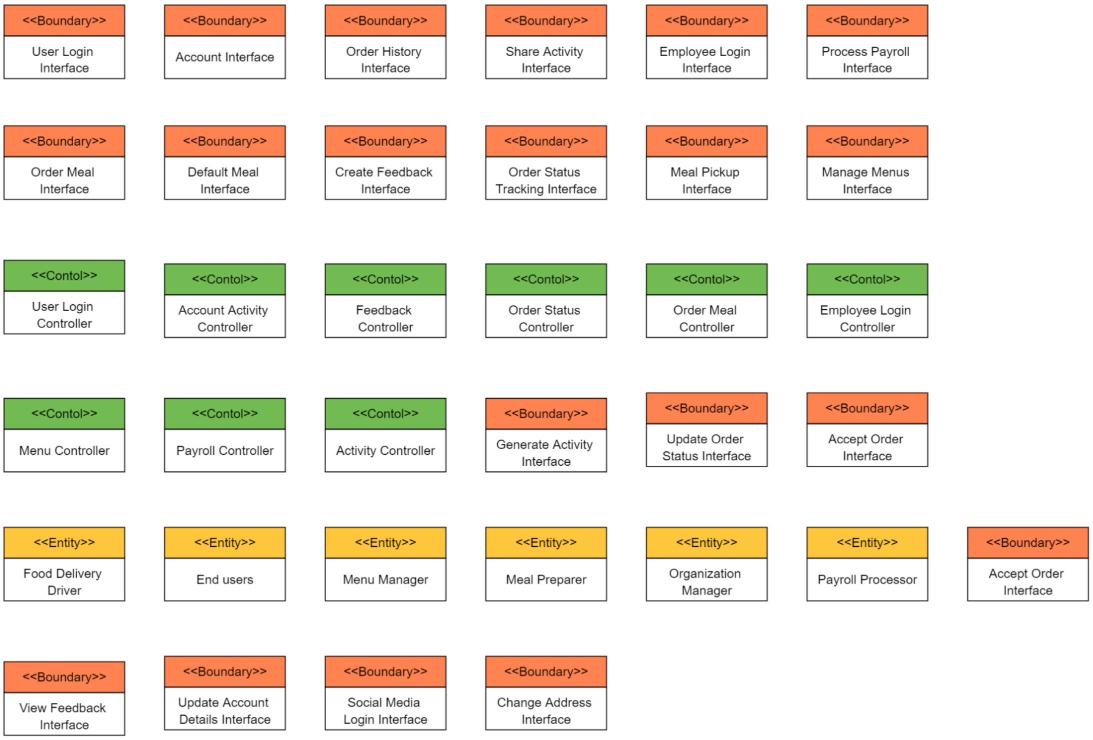
8.n Level n <Put Level 1 Diagrams here>

TABLE 3: Traceability Matrix (FRs and DFD Processes)

	All Related Processes in DFD
FR1	
FR2	
……	
FRn	

9. Class Analysis Modeling

9.1 Initial Class Diagram (ICs)


<<Boundary>> User Login Interface
<<Boundary>> Account Interface
<<Boundary>> Order History Interface
<<Boundary>> Share Activity Interface
<<Boundary>> Employee Login Interface
<<Boundary>> Process Payroll Interface
<<Boundary>> Order Meal Interface
<<Boundary>> Default Meal Interface
<<Boundary>> Create Feedback Interface
<<Boundary>> Order Status Tracking Interface
<<Boundary>> Meal Pickup Interface
<<Boundary>> Manage Menus Interface
<<Contol>> User Login Controller
<<Contol>> Account Activity Controller
<<Contol>> Feedback Controller
<<Contol>> Order Status Controller
<<Contol>> Order Meal Controller
<<Contol>> Employee Login Controller
<<Contol>> Menu Controller
<<Contol>> Payroll Controller
<<Contol>> Activity Controller
<<Boundary>> Generate Activity Interface
<<Boundary>> Update Order Status Interface
<<Boundary>> Accept Order Interface
<<Entity>> Food Delivery Driver
<<Entity>> End users
<<Entity>> Menu Manager
<<Entity>> Meal Preparer
<<Entity>> Organization Manager
<<Entity>> Payroll Processor
<<Boundary>> Accept Order Interface
<<Boundary>> View Feedback Interface
<<Boundary>> Update Account Details Interface
<<Boundary>> Social Media Login Interface
<<Boundary>> Change Address Interface

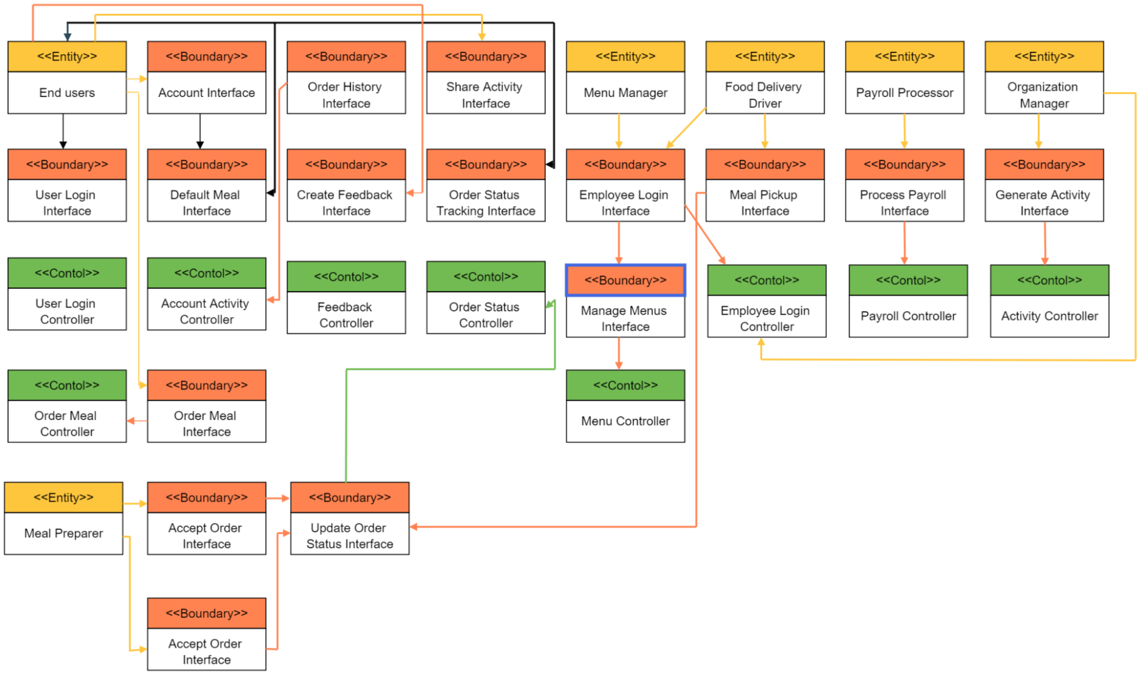
KEY	
Color Code	**Class Name**
🟧	BOUNDARY CLASS
🟩	CONTROL CLASS
🟨	ENTITY CLASS

TABLE 4: Traceability Matrix (FRs and Initial Classes)

	Related ICs
FR1	**<<Entity Class>> End Users:** Food is ordered from the cafeteria for pickup or delivery
FR2	**<<Entity Class>> End users** Food ordered from local restaurants to be delivered
FR3	**<<Boundary Class>> Account Interface:** Meal service subscription: option to delete, create, view, or modify
FR4	**<<Entity Class>> Payroll Processor** Meal payment options: credit/debit card
FR5	**<<Entity Class>> End users** Meal delivery request
FR6	**<<Control Class>> Order Meal Controller** Creating, viewing, modifying, or deleting cafeteria menus
FR7	**<<Entity Class>> End Users** Ordering meals not on cafeteria menu

FR8	**<<Control Class>> Feedback Controller** Customer service satisfaction rating
FR9	**<<Control Class>> Feedback Controller** Live chat with customer service representatives
FR10	**<<Boundary Class>> Process Payroll Interface** Multiple payment methods: debit card, credit card, PayPal etc.
FR11	**<<Control Class>> Account Activity Controller** Access FTT app outside organization intranet
FR12	**<<Boundary Class>> Accept Order Interface** Accept order request
FR13	**<<Entity Class>> Food Delivery Driver** Food deliverer picks up meal from cafeteria/restaurant to deliver
FR14	**<<Entity Class>> Meal Preparer** Packaging order request from customer
FR15	**<<Boundary Class>> Order Status Tracking Interface:** Tracking meal status

9.2 Modified Class Diagram (MCs)



KEY	
Color Code	**Class Name**
🟧	BOUNDARY CLASS
🟩	CONTROL CLASS
🟨	ENTITY CLASS

TABLE 5: Traceability Matrix (Initial Classes (IC) and modified Classes (MC))

	Related MCs
IC1 **Entity & Boundary Class**	End Users => User Login Interface => Account Interface => Order Meal Interface => Create Feedback Interface => Share Activity Interface
IC2 **Boundary & Control Class**	Account Interface => Default Meal Interface =>
IC3 **Boundary & Control Class**	Order History Interface => Account Activity Controller
IC4 **Entity, Boundary & Control Class**	Menu Manager => Employee Login Interface => Manage Menus Interface => Menu Controller

IC5 **Entity & Boundary Class**	Food Delivery Driver => Meal Pickup Interface
IC6 **Boundary Class**	Meal Pickup Interface => Update Order Status Interface
IC7 **Entity, Boundary & Control Class**	Payroll Processor => Process Payroll Interface => Payroll Controller
IC8 **Entity, Boundary & Control Class**	Organization Manager => Generate Activity Interface => Activity Controller => Employee Login Controller
IC9 **Boundary & Control Class**	Manage Menus Interface => Menu Controller
IC10 **Boundary & Control Class**	Order Meal Interface => Order Meal Controller

IC11 **Entity & Boundary Class**	Meal Preparer => Accept Order Interface
IC12 **Boundary Class**	Accept Order Interface => Update Order Status Interface

References

Wiegers, K., & Beatty, J. (2013, August 15). Software Requirements (Developer Best Practices) (3rd ed.). Microsoft Press.