FoodRUs

Rype

Software Systems Requirement
For Online Restaurant System

Version 1.0

Revision History

Date	Version	Description	Author
11/01/19	1.0	First phase of Online Restaurant System through the System Requirement Specification	Bujar Sefa Danish Faruqi Denny Liang Munif Nagi

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to document the process of building an application. The project at hand is to create an application that connects managers, salespeople, cooks, and delivery people to make the entire delivery process seamless. The delivery platform allows customers to order food, which is first approved by the manager of the restaurant. After the delivery is successful, the customer is then able to rate the cook and the delivery person. Similar to Yelp, but it has a delivery service built on top of it. Cooks and managers are also included in the platform's ecosystem.

1.2 Scope

This software system will be an end to end food delivery app. It allows customers and staff to communicate and lets managers organize and supervise the entire system. This system has 5 users; customer user, delivery person user, salesperson user, cook users, and manager users. Customer users can order, pay, and rate the food/cook. Delivery person users can bid, choose the shortest route, and rate the users. Salesperson users can negotiate prices about supplies for restaurants. Cook users determine food and menu prices and determine food quality. Managers manage customer complaints, salesperson salary, cook salary, and pays delivery people.

1.3 Definitions, Acronyms, and Abbreviations

Terms	**Definition**
SU (Manager/Super User)	The administrator of the system
DU (Delivery person User)	The person who delivers, bids, and rates CU
SPU (Salesperson user)	The person who negotiates prices
COU (Cook user)	The people who decides food prices/menu for the system

CU (Customer user)	The person who use the app to get food/perform transactions.
RCU (Registered customer user)	A CU who has an account/is registered with the system.
VI (visitor)	A CU who does not have an account, or is not registered with the system. This CU receives the highest price on food.
VIP	A RC that receives the lowest price for food.
Database	Collection of all the information monitored by this system
Software Requirements Specification	A document that completely describes all of the functions of a proposed system and the constraints under which it must operate. For example, this document.
Python	A dynamic, strongly typed, object oriented, multipurpose programming language
HTML (Hypertext Markup Language)	A standardized system for tagging text files to achieve font, color, graphic, and hyperlink effects on World Wide Web pages.
UML (Unified Modeling Language)	As a language, UML can be used to describe software developed using the traditional paradigm or any of the many versions of the object-oriented paradigm, including the Unified Process. In other words, UML is a notation, not a methodology. (Schach, 2011, p.571).
Flask	An extensible web microframework for

	building web applications with Python.
PostgreSQL/Postgres	A free and open-source relational database management system emphasizing extensibility and technical standards compliance.
Javascript	A high-level, interpreted scripting language that conforms to the ECMAScript specification.
Bootstrap	A free and open-source CSS framework directed at responsive, mobile-first front-end web development.
npm	A package manager for the JavaScript programming language.

1.4 References

Professor Jei Wei SRS template -
http://www-cs.ccny.cuny.edu/~csjie/322/spec_sample.pdf

Project requirements from the Professor - proj_spring19.docx

Schach, B. N. (2011). *Object-Oriented and Classical Software Engineering*(8th ed.). New York, NY: McGraw-Hill Companies

1.5 Overview

Sections ahead give an overview of the software's functionality.

Section 2
This section will cover the Use Case Model Survey, which will contain the actors present in this document. Also, the latter section of section 2 will highlight any other related technicality assumptions that may be seen in this report that help enhance our system as a whole.

Section 3

This section will cultivate the specific requirements of our software product to a level of highly descriptive detail and information. This will help designers and testers be able to understand our new online restaurant system. Finally, this section will also detail any supplement information as well as use case reports.

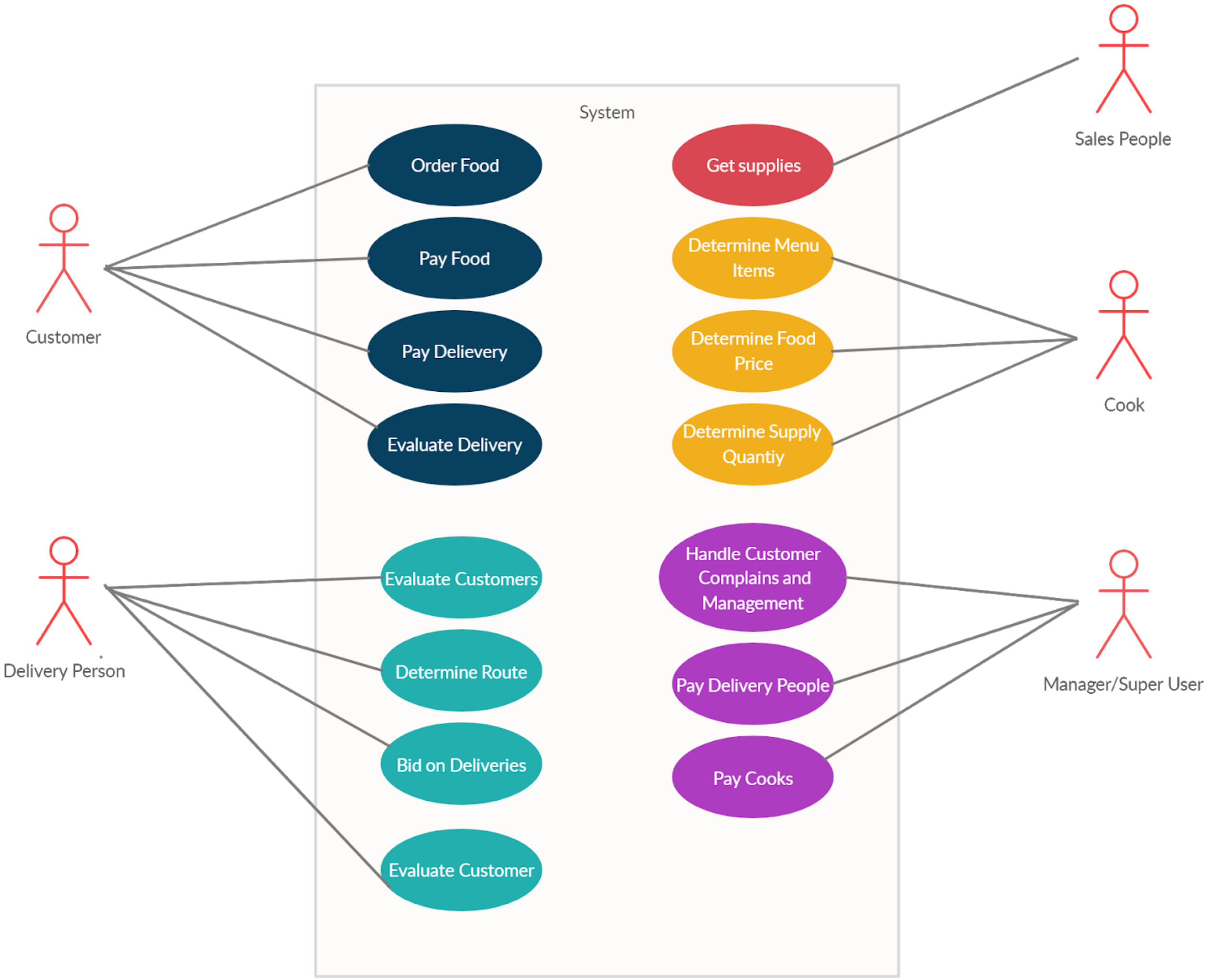
Section 4

This section will include any supporting information necessary to complete the SRS. also this section is depicted above in the table of contents.

2. Overall Description

This section will depict the overall view of the system. The system will be described through actors and use-cases by use-case modeling.

2.1 Use-Case Model Survey



There are five types of users: Super User (SU). Delivery person User (DU), Salesperson User (SPU), Cook User (COU), and Customer User (CU).

CU can order food, rate food, and pay for delivery. They can communicate with the cook to comment on the food. They are also able to communicate with the delivery person about the delivery as well. A registered CU who has more than 3 orders with an average rating of greater than 4 stars is automatically promoted to a VIP. A registered customer making more than 3 orders with an average rating of less than 2 but more than 1 is demoted to a visitor, if the average rating is 1 then the customer is put in the customer blacklist who can never be a registered user, customers can't change their names. COU can determine the prices of the food and the items on the menu. They receive products from the SPU and decide the quality of the food. They also decide the quality of the order; there are at least 2 cooks per restaurant. SPU can only negotiate food prices for the restaurant, along with the quality of the food. There are 2 salespeople per restaurant; their commission is determined on the quality of the supplies.

The COU decides their ratings, and with consecutive 3 5star ratings, they receive a promotion and a 10% increase in commissions. With 3 consecutive complaints, they get demoted with a 10% deduction in commissions. With 3 warnings, they get laid off. The DU delivers orders and bids on deliveries; they use the shortest route to get there. A delivery person receiving an average rating of less than 2 for the last 3 deliveries will receive a warning, which can be erased by the manager. A delivery person with more than 3 warnings will be laid off. The delivery person rates the customer. SU for each store decides commissions of salespeople, pays cooks, delivery people, handles complaints, and manages customers. The manager starts the bidding procedure and decides if the customer is approved to be part of the restaurant.

2.2 Assumptions and Dependencies

We are assuming that our GUI app will be sufficient enough to process multiple user transactions to order food and that once the user claims their received food, they can then go ahead and rate it. We are also assuming that our users will behave adequately and not perform any attacks or exploits on our system.
Additionally, we are assuming that we can have delivery users bid for the minimal cost of delivering food. There is time set aside (1 or 2 minutes) between when the food/transaction is processed/accepted; They should respond appropriately and not behave erratically.

3. Specific Requirements

3.1 Use-Case Reports

3.1.1 Authentication

Description

Users must be able to be authenticated and log in

Requirements

1. A user is presented a login/signup page
2. The user has the option to register as a customer/manager/chef/delivery/salesman or continue as a visitor
3. The visitor needs the approval of the manager to view the restaurant

3.1.2 Food Recommendation

Description

When a customer visits a restaurant page, he/she will be presented the top 3 recommended food choices

Requirements

1. Check the customer to see if they are approved by the manager
2. Check the order history of the customer and compare it to the restaurant's menu

3.1.3 Discounts

Description

Registered/VIP customers receive perks while visitors do not

Requirements

1. Check the customer's privilege
2. Based on their privilege (registered, visitor, VIP) and display different prices based on each

3.1.3 Bidding System

Description

A bidding system is in place for delivery people to price their service.

Requirements

1. Every delivery person puts a price on their service.
2. All the prices get compared with each other, and the lowest ask receives the order to be fulfilled.
3. The most efficient route is then curated in a google map interface for the delivery person based on traffic.

3.1.4 Rating System

Description

Customers have the option to rate food/cooks and delivery person. The delivery person also has the option to rate the customer.

Requirements

1. Once the food is delivered, the customer can give a 1-5 star rating to both the food/chef and delivery person.
2. If the rating is lower than 2 stars for the last 3 deliveries, they will be given a warning, which can be erased by the manager. If more than 3 warnings, their accounts will be banned.
3. The delivery person also has the option to rate the customer after the delivery.
4. If the rating is lower than a 3 star, then it will be filed as complaint towards the other party.
5. Check customer's # of complaints, and demote/promote their privileges accordingly. 3 orders or more with an average of 4+ stars get promoted to VIP. In between 1-2 stars will be demoted to visitor. Finally less than 1 star will be blacklisted.

3.1.5 Salesman commision

Description

A salesperson's rating is based on their quality of food.

Requirements

1. When the food is delivered to the restaurant, the cooks have the option to rate the person who sold the ingredients to them from 1-5 stars.
2. Check the salesperson's history and rating
3. If 5 stars for the last 3 deliveries give 10% raise. If 3 complaints were filed, then reduce wage by 10%

4. To be friendly to vision challenged customers, a voice-based order feature should be available.

Use Case: Apply to be an RCU
User: CU
To apply for an account, the CU needs to log in and sign up.
Description: A CU can submit a login/register request with the application to become an RCU. The user should provide their name, address, type of food liked, as well as a password. If the SU approves the application, then the CU becomes an RCU.

Use Case: Approve VI to be RCU
User: SU
For a VI to become an RCU, it must be approved.
Description: A VI can submit a login/registration to become an RCU. Once the application is submitted, an SU can then approve the VI to be an RCU. If the SU approves, then the VI's data gets stored on the application, and the VI has an account. If the SU rejects the VI, the VI continues to remain a VI. The manager can check the customer record of the restaurant to decide if this customer should be approved, any customer who is on the blacklist of the restaurant should be automatically denied.

Use Case: Pay COU
User: SU
For a COU to be paid, the SU must log onto the system.
Description: A SU can submit the payments/checks to the CU by logging onto the application and paying the CU based on the commission they received, which is determined by the rating of food.

Use Case: Pay DU
User: SU
For a DU to be paid, the SU must log onto the system.
Description: A SU can submit the payments/checks to the DU by logging onto the application and paying the DU based on their rating and regular fee.

Use Case: Determine the SPU Commission
User: SU
To determine how much an SPU gets paid, the SU must log onto the system and provide a commission.

Description: A SU can submit the SPU's commission onto the system, which then determines how much money they get from the total profit.

Use Case: Handle/Manage CU complaints
User: SU
For a CU's complaint to be resolved, SU must log into the system and address it.
Description: A SU can log onto the system, and see a list of CU complaints, that were addressed to the restaurant. The SUN can then go and send messages/refunds as deemed fit.

Use Case: Login
User: SU, CU (VIP, RCU), COU, SPU
For a SU, VIP, RCU, COU, SPU to connect to the system, they must log in.
Description: An SU, VIP, RCU, COU, SPU have an account in the system, which they can log into and interact or conduct their activities.

Use Case: Order food
User: CU
To order food, the CU must be on the app.
Description: The CU goes on the application, and either log in or continue as a VI and then selects the food that they want and press order.

Use Case: Pay for food
User: CU
To pay for food, the CU must first order the food.
Description: When the CU goes on the application and orders the food that they want, they must then pay for the transaction.

Use Case: Rate food
User: CU (RCU, VIP)
To rate the food, the user must have received their food.
Description: A CU, which includes RCU, and VIP, can rate the food that they received on a scale of 1-5, 1 being lousy quality or taste, and 5 being high quality or taste.

Use Case: Rate Delivery
User: CU (RCU, VIP)
To rate the DU, the CU must have received their food.
Description: A CU, which includes RCI, and VIP, can rate the delivery of the food on a scale of 1-5, 1 being poor delivery service, and 5 being excellent delivery service.

Use Case: Order food supplies
User: SPU
To order supplies for the restaurant, SPUs must be logged on the app.
Description: The SPU is in charge or getting the best deals from suppliers for the best food supplies by purchasing supplies on the app.

Use Case: Determine food on the app
User: COU
To determine what food is available by the restaurant, the COU must log in and list the foods available.
Description: The COU is in charge of logging onto the system and entering which foods from the menu, are available for customers to purchase. This is them selecting which foods from the current menu are available.

Use Case: Determining the menu for the restaurant
User: COU
To determine what foods the restaurant makes, the COU must log in and create an extensive menu for the restaurant.
Description: The COU is in charge of logging onto the system, and inputting all of the restaurant menu items. Note, not all food may be available to the customers at a given time, however that is a list of foods that will be made at one point.

Use Case: Rate supply Quality
User: COU
To determine if an SPU continues to get supplies for COU.
Description: A COU must log onto the system and provide a rating to the SPU based on the quality of the supplies that the COU uses for their dishes.

Use Case: Rate Customer
User: DU
To rate the CU (RCU, VIP), the DU must have delivered the food to the CU and be logged in.
Description: A DU delivers the food to the customer and can rate the CU based on the interaction between handing the food.

Use Case: Bid on Delivery
User: DU
To ship food from restaurants to CU, the DU must bid for the lowest price to win.

Description: Once a CU order the food, and the restaurant's process it, DU's will have to bid within 2 minutes to be the one to deliver it. The DU with the lowest price within the 2 minutes, then gets to deliver the food to the CU.

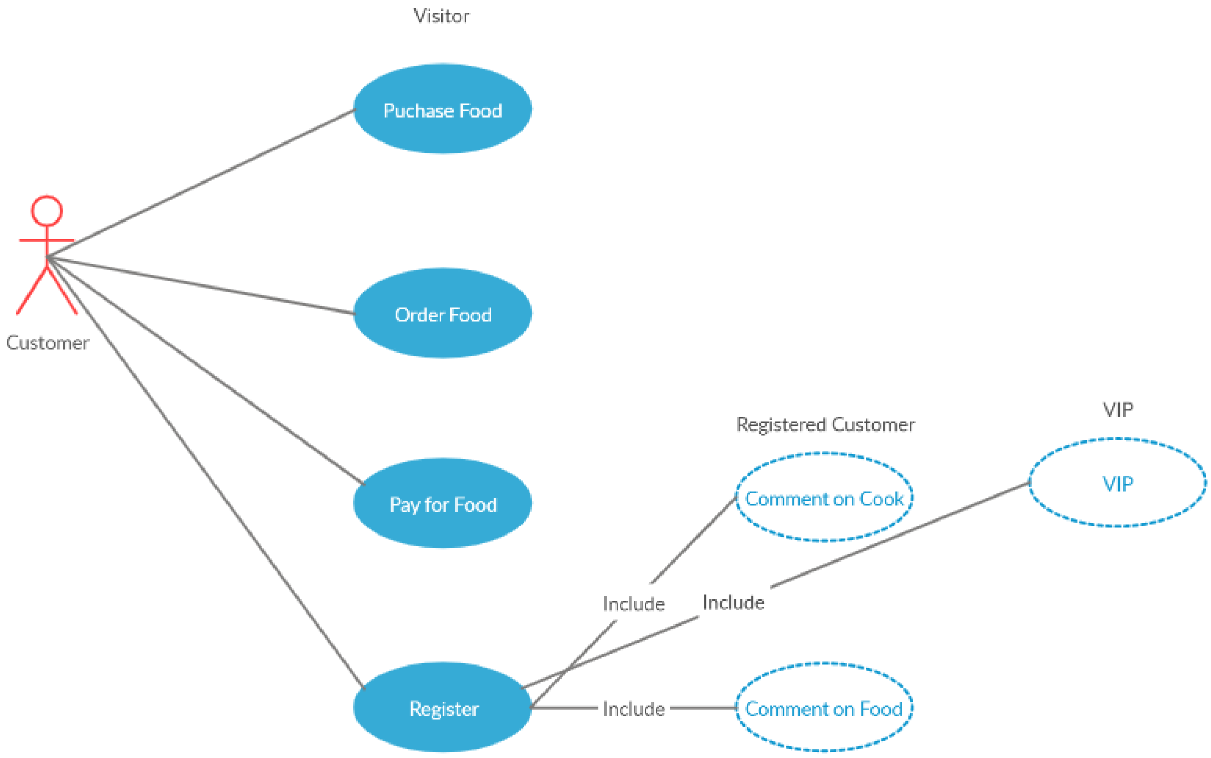
Use Case: Determine Route

User: DU

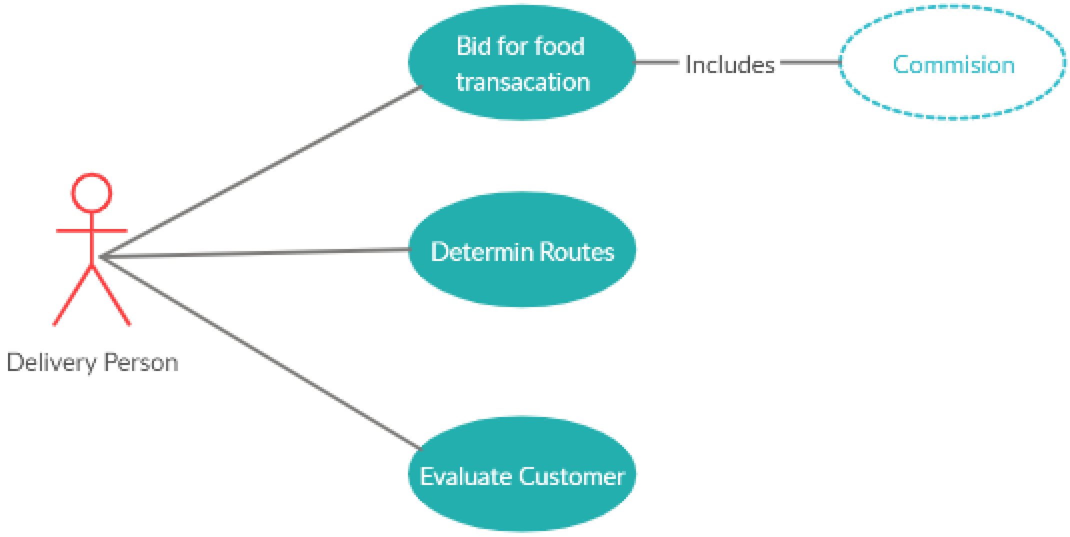
To ship the food from the restaurant to CU, the Du must choose the route based on the options provided.

Description: Once the DU has the food ad has to deliver it to the CU; the DU must decide, based on the map, which routes to take in, which they believe will bring the food to the CU in an orderly fashion.

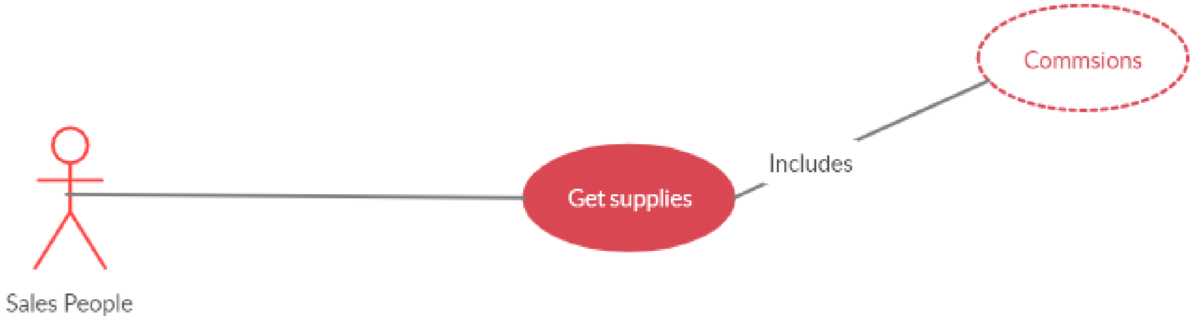
Customer (CU)/ Visitor (VI)/ Registered Customer (RCU)/ VIP:



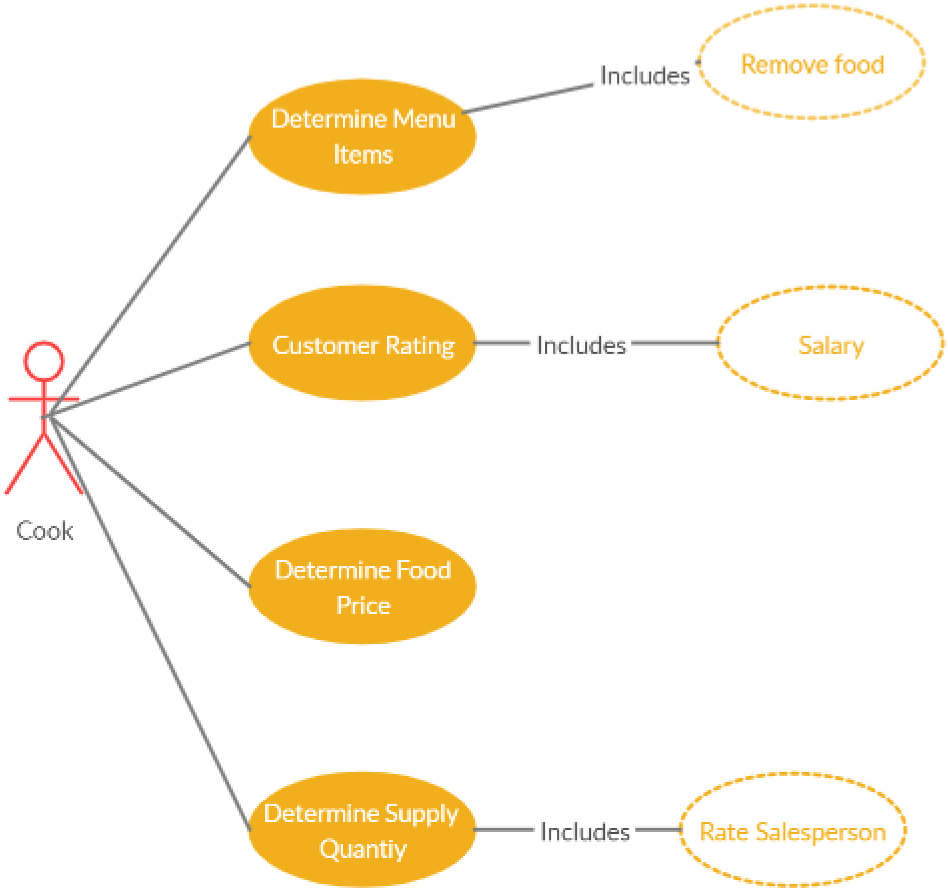
Delivery Person User (DU):



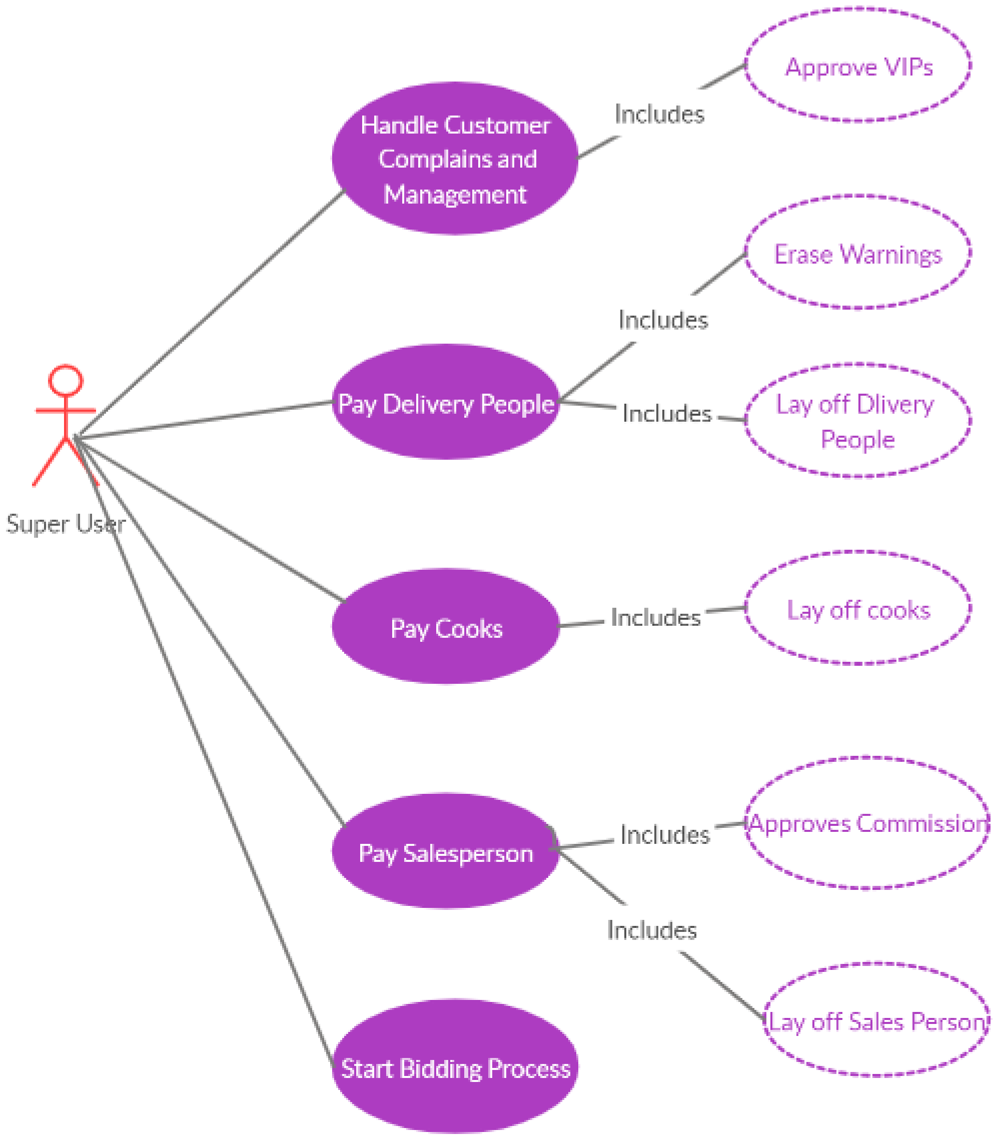
Salesperson User (SPU):



Cook User (COU):



Manager/ Super User (SU):


Super User
Handle Customer Complains and Management
Includes
Approve VIPs
Pay Delivery People
Includes
Erase Warnings
Includes
Lay off Dlivery People
Pay Cooks
Includes
Lay off cooks
Pay Salesperson
Includes
Approves Commission
Includes
Lay off Sales Person
Start Bidding Process

3.2 Supplementary Requirements

If we want our product to work, we need to ensure a few more things:

3.2.1 The system can handle scalability:

Since we aren't sure how many users would be accessing the app at the same time, we need to be open to scaling up and optimizing the system to work faster with a more massive load. Since this is a development process, further challenges in scalability should be considered.

As long as we meet our specifications, we should be able to expand out of a tiny subgroup of users who can access the system at once; We should be ready to have continuous integration in our system as well as be open to significant changes with the volume of users. We can take care of these things through software updates, replacement of functionality, and system upgrades.

3.2.2 The code needs to be easy to maintain:

Keeping Kolmogorov complexity in mind, we need to maintain a clean and readable code structure. This maintenance is essential as it will make the code hard to read if read by someone not actively working in the code. And for people who are actively working on the code, things can get confusing fast. The code needs to be readable and easy to maintain because as we add more features, the code will quickly become convoluted if it's not organized in any coherent way - making everyone's life hard. As such, we require the code to be maintained actively.

3.2.3 The accuracy and timeliness of the system processing:

Accurate and timely treatment is necessary for the performance of the system. We need to be able to meet any inbound traffic with speed and accuracy to ensure a backlog does not get generated, which would only create more dependencies and problems in the system. The system design and development process should dictate efficiency as it will reduce and help manage future workload in request processing.

4. Supporting Information

This Software Requirements Specification includes:

1. Table of Contents
2. Index