



<https://github.com/ajay/ROSE>

First report (Specification only) - Full Report 1

Software Engineering (14:332:452)

Professor Ivan Marsic

Due: February 21, 2016

Group 9

Ajay Srivastava, Srihari Chekuri, Cedric Blake, Brice Howard,
Vineet Sepaha, Neil Patel, Jonathan Cheng

Table of Contents

Individual Contributions Breakdown	3
Section 1: Customer Statement of Requirements	4
a) Problem Statement	4
b) Glossary of Terms	7
i) Technical Terms	7
ii) Non-technical Terms	7
Section 2: User Stories	8
a) Customer's perspective	9
b) Manager's perspective	10
c) The User Experience	11
i) Customer User Interface	12
ii) Manager User Interface	14
Section 3: Functional Requirements Specification	15
Use Case Diagram	16
Use Cases	17
Section 4: User Interface Specification	22
User Effort Estimation	25
Section 5: Domain Analysis	26
a) Domain Model	26
i) Concept Definitions	26
ii) Association Definitions	28
iii) Attribute Definitions	31
b) Systems Operation Contracts	36
c) Mathematical Models	38
Section 6: Plan of Work	39
References	41

Individual Contributions Breakdown

Responsibility Levels	Points	Team Member Name							Totals
		Ajay Srivastava	Srihari Chekuri	Cedric Blake	Brice Howard	Vineet Sepaha	Neil Patel	Jonath an Cheng	
Project management	10	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	100 %
Section 1: Customer Statement of Requirements	9	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	100 %
Section 2: System Requirements	6	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	100 %
Section 3: Functional Requirements Specification	30	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	100 %
Section 4: User Interface Specs	15	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	100 %
Section 5: Domain Analysis	25	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	100 %
Section 6: Plan of Work	5	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	100 %
Total	100	14.3	14.3	14.3	14.3	14.3	14.3	14.3	100
Total Percentage	-	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	14.3 %	100 %

Section 1: Customer Statement of Requirements (CSR)

a) Problem Statement

Establishing a new restaurant is like having a child: no one will take as much interest or pride in it as you. Don't you love hearing how much people like coming to your restaurant? There are very few jobs that make you feel as proud and happy as owning a successful business. You are your own boss, but at the same time, you also have a lot of responsibilities. The good and the bad falls on your shoulders. Being a restaurant owner is not all fun and games with free food. There are many drawbacks such as:

- Long days and nights - As an owner of a restaurant, I have to put in very long hours to make sure everything goes smoothly. And even when I'm home I have to make several calls a day to make sure everything is fine. I'm constantly thinking of how I can improve my restaurant, and what I can do in order to reach more customers and improve their overall experience.
- No weekends - Weekends are the busiest time for most restaurants and are the time when the majority of customers have free time to stop by my restaurant. This means I have to sacrifice the time I originally had with my family and my free time in order to try to make my restaurant the best it can be.
- Unstable Income - I remember when business slowed down, and my checkbook was in the red, and guess who the first to go without a check was? Me. The person who does the most work, is the person who is often paid the least. Of course, sometimes small business owners have to make these sacrifices in order to keep the business afloat.
- Benefits - Health insurance is a major concern for all small business owners. I have to buy private insurance for myself and my employees and provide 401K plan and/or some sort of other additional benefits as well.

As a restaurant owner, I want to give my customers the best possible dining experience I can, so that I can turn them into repeat customers. Even if my food is great, if the customer doesn't like my service, it is highly unlikely that they will return to eat at my restaurant again. Often times customer service falls short due to the lack of policies set by myself, or the lack of quality in those I hire (waiters, chefs, and such). Several problems that I have run into with my employees include:

- Tardiness - An employee who is late once or twice may need a reminder about company policies. An employee who is consistently tardy can throw off the shift schedule and strain his co-workers which will impact the quality of your restaurant. Controlling employees and changing their habits are hard; a habitually late employee will most likely continue to be late even after being given warnings. Repercussions such as firing employees will then lead to other issues that require time and money, such as searching for new employees and then training them when they are found.

- Hygiene or appearance problems - One employee's foul hygiene or poor appearance will give rise to poor perceptions of the food quality, cleanliness of the restaurant and overall atmosphere. Issues similar to tardiness will arise.
- Poor attitude - Customers come to a restaurant for friendly, polite service and a pleasant experience. Service from someone with a poor attitude taints the entire dining experience and reduces patronage. It is hard to control attitudes of employees, as well as control specific moods they are in.
- Poor job execution - When an employee cannot perform his or her job at an acceptable level, additional training may be required, unless there is a deeper problem like poor attitude or laziness. Again, training requires additional time and money to be spent.
- Poor customer service - It is essential that all employees be well versed in customer service and table etiquette. Hosts and hostesses, servers and bussers should all exhibit quality service techniques.
- Insubordination - Problems with supervisors or managers are common plights in many businesses. A disrespectful staff member can leave a blemish in an otherwise positive workplace.
- Theft - Employees have been known to find methods of stealing money, supplies or food from a restaurant. This not only hurts business profits but also fosters an environment of dishonesty.

As a restaurant owner, I have more than enough on my plate already and I want to minimize the number of factors that are not in my control. This includes the efficiency at which my restaurant is running and reducing the number of employees I have to depend on.

When customers order a meal and don't receive what they ordered, they can either suffer through it or send it back. Either way, the situation isn't ideal because eating something they don't really want is, well, bad, and sending it back can lead to an awkward confrontation which will ruin their dining experience. If they were to have a confrontation, they're usually afraid of upsetting the server and paranoid about the historical claims of getting something extra when their meal returns (saliva, or worse?), or they come across a jerky and a little entitled waiter, in which case you, as the restaurant owner have to deal with more than you bargained for. To avoid such problems, you as the restaurant owner should first figure out how to avoid such disturbances in the dining process. Like many existing restaurants such as Chili's and Applebee's, I would like a customized web application that can be accessed via tablets at each table in my restaurant. The application should allow my customers to order food at their own pace and they no longer have to ask for the menus every time they wish to order something else. By enabling the customer to make changes to their order personally, I can reduce the number of mistakes in the orders, due to a waiter's negligence.

By introducing the web application, the restaurant will run more efficiently and hopefully prompt the customers to come back again. I also want to reduce the number of employees I hire because it adds onto my responsibility of running the restaurant. Most waiters and waitresses do not cause problems, but paying them and providing them with benefits cuts into my paycheck

every month. A waiter's major role includes taking orders, delivering the food, giving the customer their check and then cleaning up. I would like a robot that is able to navigate through my restaurant without running into any obstacles and avoiding collisions with customers and/or staff. The robot should be able to differentiate between different foods and drinks. Then based on the selections made on the web application, the robot should bring the ordered food and drinks to the customers' table. This robot should reduce the required number of employees required to run the restaurant efficiently while increasing profits.

I would also like administrator access to the robot. In the web application, I would like a staff member to have login access to the robot status. They should be able to use a simple login and see the robot's current status such as its location, battery level and the option to turn on or off the robot. Having access to such information is necessary for me to maintain the robots so I don't run into problems with them (It would not be very nice if a robot ran out of battery in the middle of my restaurant).

We conclude what features our users might want to have in our software/hardware as follows:

Web application requirements:

1. Order food - choose between entrees, appetizers, drinks and make modifications to the order
2. Request an employee (for some problem)
3. Pay their bill at the end of their meal.
4. An administrator login
5. Access to robot status (Location/battery and such)
6. Ability to turn on and off robot

Robot Requirements:

1. Navigate through my restaurant without running into any obstacles
2. Differentiate between different types of foods and drinks
3. Be able to grab different types of food and drinks after recognizing them
4. Localization (It should know where the tables are)
5. Deliver food from kitchen to the customers' table
6. Differentiate between different customers (or where they are seated) in order to deliver the correct order to the correct customer

b) Glossary of Terms

Technical Terms

→ Robot terms:

- Robot - a machine that is capable of carrying out a series of actions automatically

→ Web application terms:

- User interface (UI) - refers to the “screen” of the application and the components within it that users (i.e. customers) see and interact with.
 - i) Customer-side UI - The user interface with which the customers interact.
 - ii) Administrator-side UI - The user interface with which the manager interacts.
- Web Application (webapp) - a program accessible through the internet that performs some task. In the case of this project, “webapp” refers to the application used by customers to request food or drinks, and used by managers to synchronize with the restaurant’s menu and attend to customers who seek them.
- Menu - a graphical display on the webapp containing different selectable options. Not to be confused with the non-technical term, Menu.

→ Server/database terms:

- Database - a collection of information that is organized for ease of access, management, and updating
- Server - a computer program or machine that waits for and responds to a request sent by another computer
- Request - in the perspective of computers, this is defined as a message sent from one device to another

Non-Technical Terms

- Manager - the individual who controls the operations of the restaurant
- Customer - an individual who is served by the restaurant
- Order - a request of food and/or drink specified by a customer
- Menu - a list of food and drink items available to select. Not to be confused with the technical term, Menu.
- Administrative Options - the options provided to the manager to update the restaurants menus. This includes options to add/ remove items, change item descriptions, and change item prices.
- Robot Conditions - This includes the robot’s battery life and the mechanical conditions of the robot.
- Stock - a generalization of any location in which food or drinks can be served. This document assumes that the food or drink is premade and ready for the robot to retrieve.

Section 2: User Stories

Size Key

For each user story we include a size of 1 - 10 points. The sizes are rough estimates of how much effort is needed to implement each user story based on our group's technical skills and each user story's perceived complexity.

Priority Key

ST-X-#: **High** priority (essential feature)

ST-X-#: **Middle** priority (Important feature)

ST-X-#: **Low** priority (less important feature)

ST-X-#: **Lowest** priority (optional feature)

Structure

- The user stories have a very specific structure:
 - As a <insert actor here>, I am able to <insert action here> by means of <insert device here>, which will allow me to <insert benefit here>
- ST-Cu-# for customer stories, ST-M-# for manager stories

a) As the customer...

Index Number	User Story (As the customer)	Size (Points)
ST-Cu-1	I am able to request food and drinks by way of the webapp on my phone, which ROSE will then bring to me when it is ready.	7 points
ST-Cu-2	I am able to request a refill of my drink by using my phone, and ROSE will get it for me.	5 points
ST-Cu-3	I am able to traverse the restaurant without ROSE getting in my way.	4 points
ST-Cu-4	I am able to request additional condiments and utensils in case I need any.	2 points
ST-Cu-5	I am able to check the exact item charges for what I ordered, so that I know what I am paying for.	2 points
ST-Cu-6	I am able to request a change in my order before ROSE notifies the chef of my order.	2 points
ST-Cu-7	I am able to ask for the manager through the webapp on my phone, so that I'll be able to speak to him or her regarding my food, service, or feedback.	1 point
ST-Cu-8	I am able to tailor my order to my preferences (well-done steak, dressing on the side, etc) when ordering food using the webapp, so that I may have the food the way I like it.	2 points

b) As the manager...

Index Number	User Story (As the manager)	Size (Points)
ST-M-1	I am able to access the customer menu from the restaurant's PC in order to determine what my customer sees when viewing the menu.	2 points
ST-M-2	I am able to access administrative options of the restaurant's menu on the restaurant's PC with a username and password so that I will be the only person that can change the restaurant's menus.	3 points
ST-M-3	I can add, remove, and alter menu items through the administrative options on the restaurant's PC for the sake of updating customers about the available items and their prices.	5 points
ST-M-4	I am able to view the quantity of purchase for each item and the restaurant's profits with administrative access so that I can monitor the state of the restaurant and make changes to the menu based on item popularity.	5 points
ST-M-5	I can monitor the current status of the robot, which includes positioning, table assignment and its current conditions, through the restaurant's PC so that I can determine if the robot has malfunctioned and, in the event that it has, find out where the robot is so that I can find and handle it physically.	8 points

c) The User Experience

Our project will in essence have 2 types of users, the manager/restaurant owner, and the restaurant customer.

The manager should have administrative access to the user interface. They should be able to do things such as add/ remove items, alter price of items, display when items are temporary available/ unavailable (i.e. item of the day, ran out of materials for that item, etc.). In addition, the manager should be able to monitor and control various properties of the robot such as table assignment of the robot, current action of the robot, condition of robot (i.e. battery life, any mechanical malfunctions).

The customer should be able to view the restaurant menu items. This includes the Item description, the price of the item, and possibly a picture of the item. In addition, the customer should be able to place an order and purchase for their meal through the device. This includes purchasing for multiple people from one device (allow customers to select their meals from separate devices, but accept the purchase of all items from one device). Furthermore, the customer should be allowed to monitor the state of their meal and be given an estimated time of service.

The process of interacting with the robot and its features will depend on the actor that wants to communicate with the robot. There will be two types of actors. Actor 1 will be the restaurant customer, which will order items from a device provided to them by the restaurant. Actor 2 will be the restaurant manager, which will have access to the administrative features of the web application.

The manager of the restaurant will be able to access the web application through a personal computer within the restaurant. Once accessed, the manager will have a variety of features to access for the sake of monitoring the state of the restaurant and the state of the robots in service. As far as monitoring and updating the restaurant is concerned, the manager will have administrative access to the menu of the restaurant. Administrative access will be checked by a userID and password, so that the system not only knows that the user is an administrator, but also knows which administrator is using the web application. In addition, the userID will be used so that any administrator knows whoever made changes to the web application and when they made the changes. Once logged on, the administrator will be able to monitor data pertaining to their restaurant and will be able to make changes to the menu items that the restaurant customers will see. From the restaurant menu, in addition to the item data that is seen by the restaurant customer, the administrator will be able to view data about menu items such as the quantity that has been sold of that food item and the profit generated by selling that item throughout the day.

Customer User Interface:





Please select a category

Drinks	Appetizers	Entrees	Desserts
--------	------------	---------	----------

Help
Extras

Total: \$18.67 [Checkout =>](#)

[< Back](#) Entrees

Beef 	Chicken 	Pork 	Vegetarian 
from \$13.95	from \$7.99	from \$11.75	from \$6.99

[I want to customize my order](#)

Help

Checkout

Item	Price
Mozzarella sticks (1).....	\$6.99
Chicken Alfredo Pasta (1).....	\$12.99
Dr. Pepper.....	\$1.99

Subtotal: \$21.97

Tax (7%): \$1.54

Total: \$23.51

Help

Please choose your method of payment:

CASH CHECK CREDIT/DEBIT CARD

Figure 1: The customer user interface on the web application. The first picture depicts the home page, where customers can add items to their order; the next one illustrates the process of selecting a particular item from a specific category. The last image shows the checkout interface, enumerating the chosen items and allowing multiple methods of payment.

Manager User Interface:

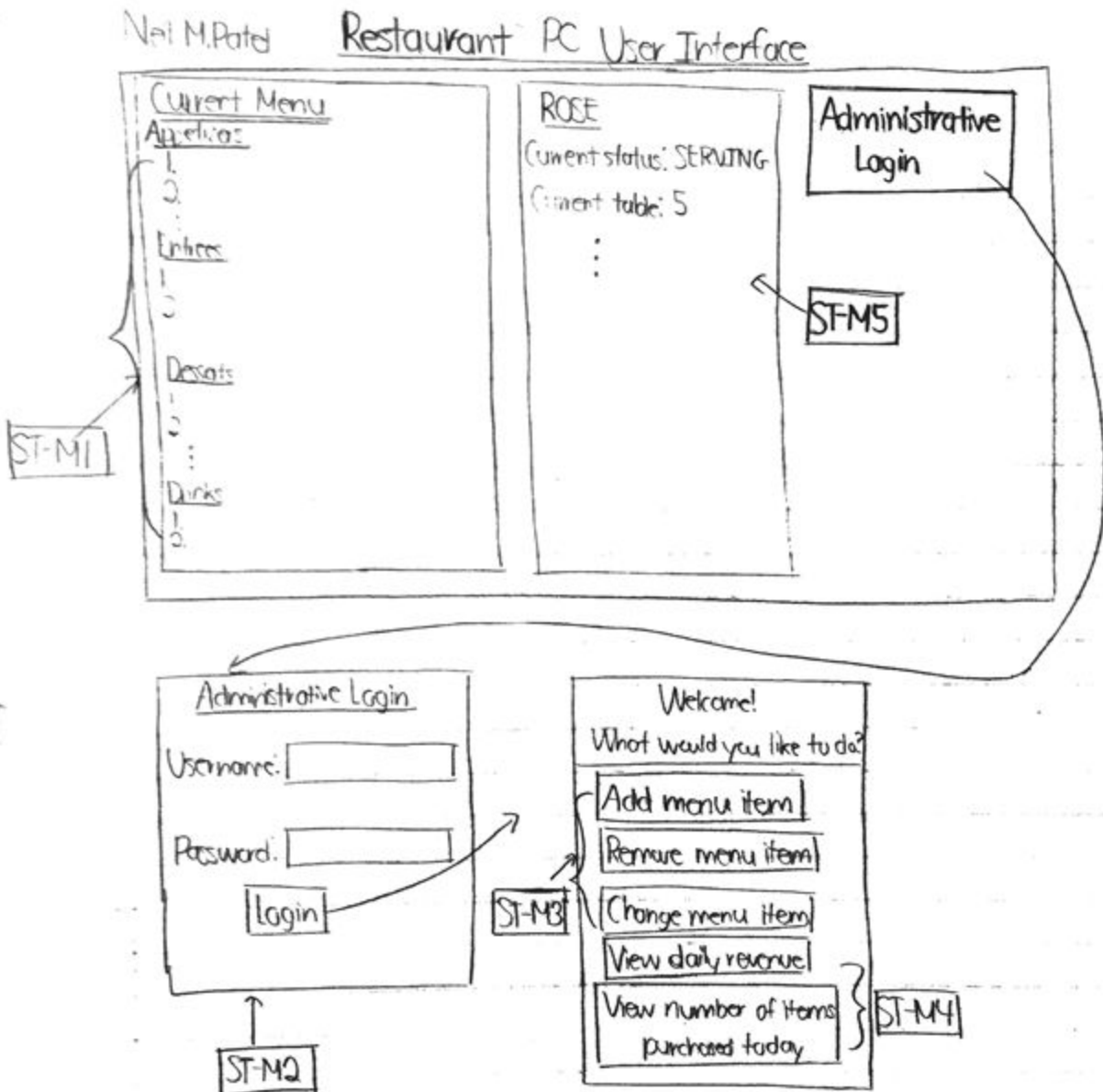


Figure 2: The manager's user interface on the web application. Note that this aspect of the web application interface is accessed through an administrative login on the homepage of the application.

Section 3: Functional Requirements Specification

- a) Stakeholders: Manager, restaurant organization, chef, customers, manufacturing companies, developers
- b) Actors and Goals:
 - Main goal – build robot to deliver food to customer
 - Initiating Actors – Robot, Customer, Manager
 - Supporting Actors – Tablet, Main terminal
 - Off Stage Actors – Stock

Casual Descriptions of Use Case:

Our group's user stories will serve as the casual descriptions of our use cases

Use Case Diagram

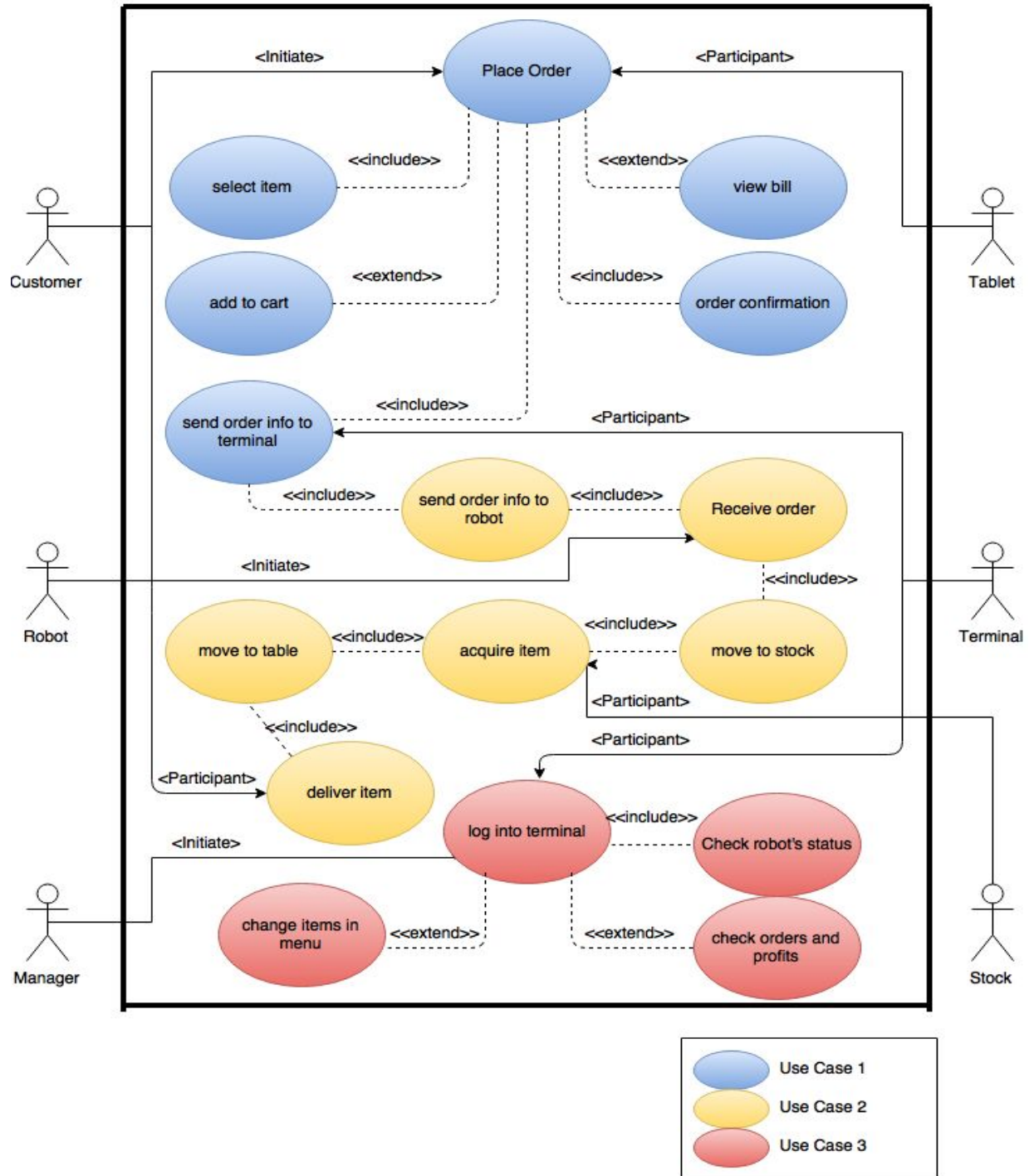


Figure 3: Use case diagram, illustrating the interactions between the actors and the components of the system.

Use Cases

1. Placing the Order

Initiating Actor: Customer

Goal: Receive food using app on tablet

Participating Actors: Tablet, Main Terminal

Precondition: The customer has a tablet present on the table to place their order

Postcondition: The customer's order is placed into the database queue

Main Success Scenario:

- Customer opens restaurant menu on web app
- Customer views the items present on the menu through categories
- Customer will add items to cart
- Customers continue the process until they have all the desired items in the cart
- Customers press the checkout button to send the order to the main terminal
- Customers will receive a confirmation on screen that the order has been placed

Alternate Scenario (Extended main):

- Customer clicks a button to request a change in the order
- Database checks if the robot has received the ordered item. If not, customers will be allowed to change their order
- Customer can add/remove items as desired
- Customer submits request to change their order in the database
- Customer will receive a confirmation on screen that the order has been placed

2. Delivering the Order

Initiating Actor: Robot

Actor's Goal: deliver food to the customer

Participating Actors: Stock, Customer, Main Terminal

Precondition: The requested item is present in the database of the main terminal

Postcondition: The customer has received his/her order

Main Success Scenario:

- Robot request order from the main terminal database
- Order information is delivered to the robot
- Robot moves to stock
- Robot acquires the order from stock
- Robot updates its status every certain time intervals
- Robot transports the order to the customer's table
- The customer receives the order from the robot

Alternate Scenario (Out of Stock):

- Robot does not find the requested item
- Robot tells the main terminal that the food item is out of stock
- The main terminal sends information to the web app that the food item is out of stock

Alternate Scenario (Robot Malfunction or Connection Issues):

- Robot fails to send update status to main terminal or stops performing a certain action
- Terminal sends a distress signal to the manager
- Manager receives signal from the terminal
- Manager handles the malfunction as needed
- Terminal alerts the user of this malfunction and asks them to wait while the manager resolves the issue

3. Review the System

Initiating Actor: Manager

Actor's Goal: View status and condition of the robot

Participating Actors: Main Terminal, Robot

Precondition: Robot updates the main terminal of its current condition and status

Postcondition: Manager is notified on all aspects of the robot in real time

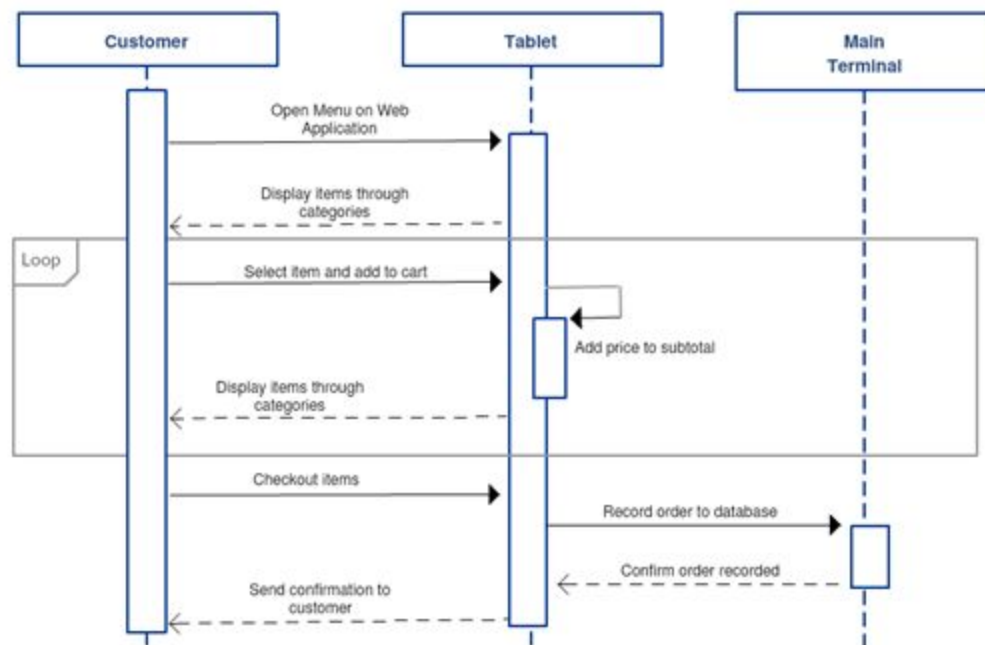
Main Success Scenario:

- Robot sends information to the terminal about its current location, action and condition in discrete time intervals
- Manager logs into the main terminal
- Manager checks the status and condition of the robot
- Manager logs out of the system

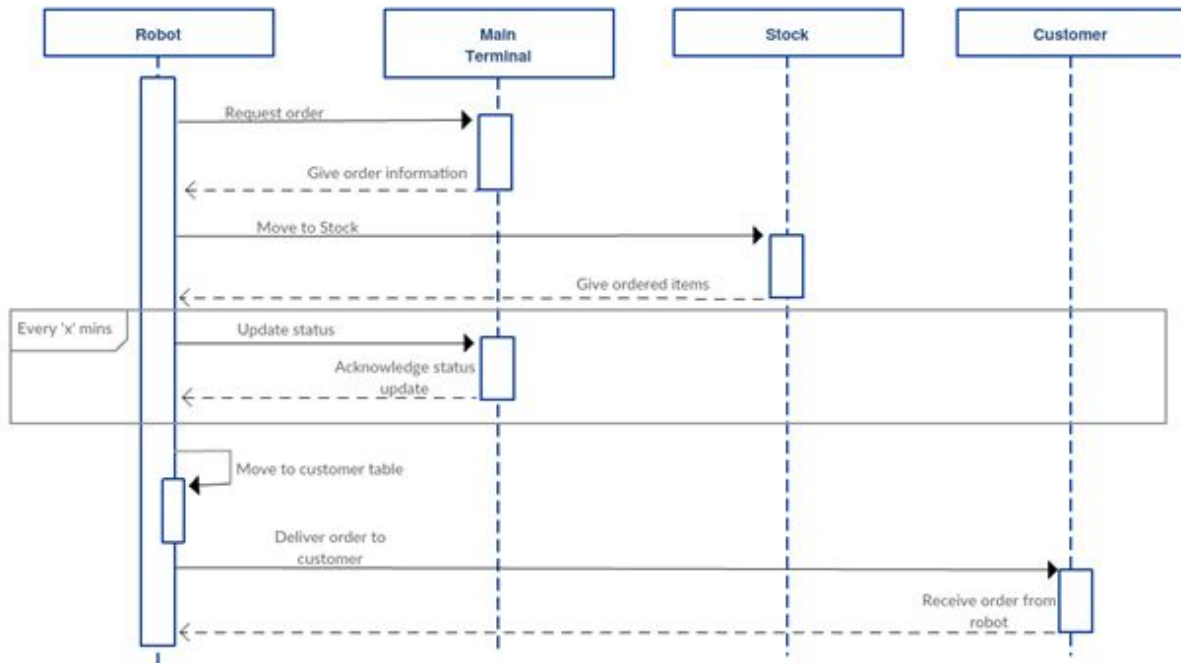
Alternate Scenario:

- Robot sends signal to the main terminal
- The main terminal stops sending orders and signals to the robot
- The main terminal decides which type of malfunction occurred
- The main terminal pings the manager to let him know that a malfunction has occurred
- The manager can check and see what type of malfunction has occurred

Place Order Sequence Diagram



Delivering Sequence Diagram



Review the System Sequence Diagram

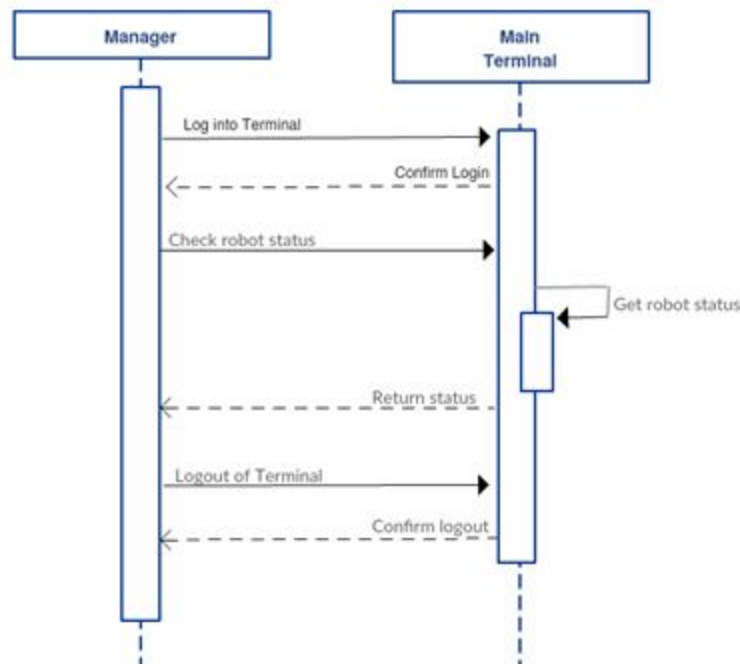


Figure 4: System sequence diagrams for each of the use cases previously listed. These diagrams illustrate the interactions between the entities involved in each particular use case.

Traceability Matrix

<u>Requirement</u>	<u>PW</u>	<u>UC-1</u>	<u>UC-2</u>	<u>UC-3</u>
1	3			
2	2			
3	3			
4	3			
5	4			
6	5			
7	5			
8	3			
9	5			

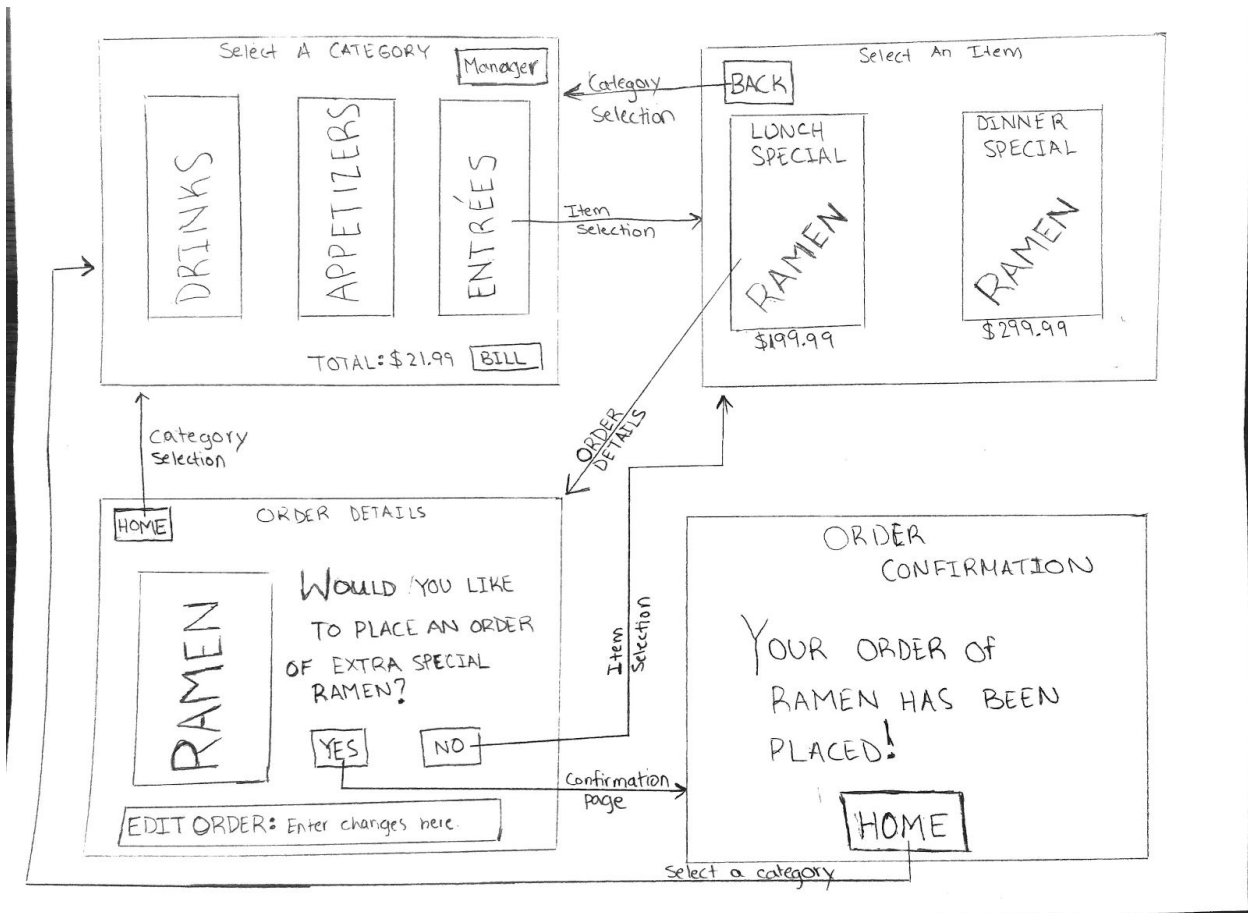
Figure 5: The traceability matrix, linking the requirements with the use cases for this project.

Requirements:

1. Provide user interface for the customer
2. Allow the user to select items from the user interface
3. Notify the terminal of the items selected when the order is placed
4. Notify the robot of the items selected when the order is placed
5. Robot selects items to deliver from the stock based on the items ordered by the customer
6. Monitor the robot's location, actions, and conditions
7. Deliver the requested items by robot
8. notify the terminal that the robot has completed delivery and is ready to make a new delivery
9. Notify the terminal when the robot malfunctions

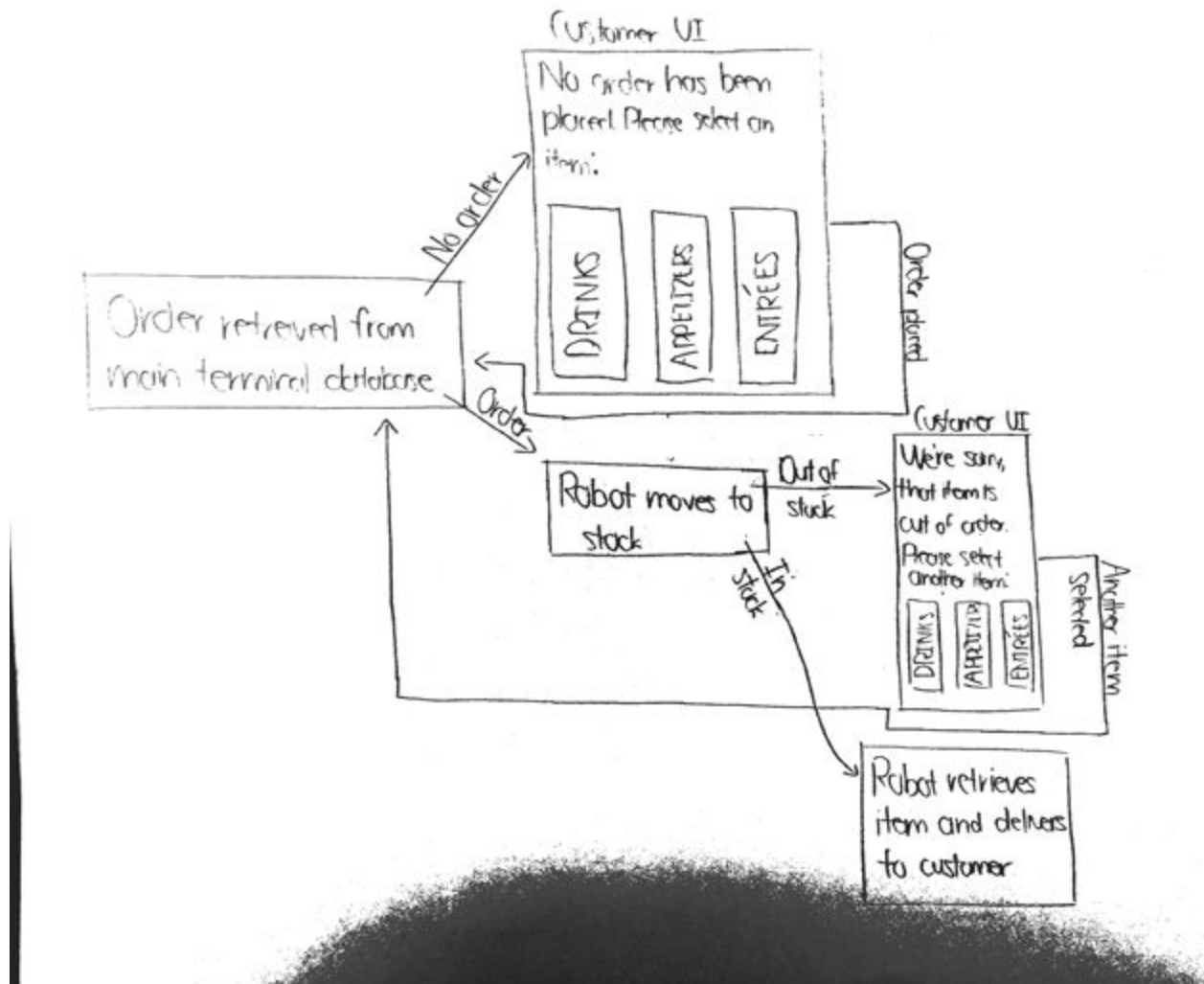
Section 4: User Interface Specification

Front End of Web App for Customers:



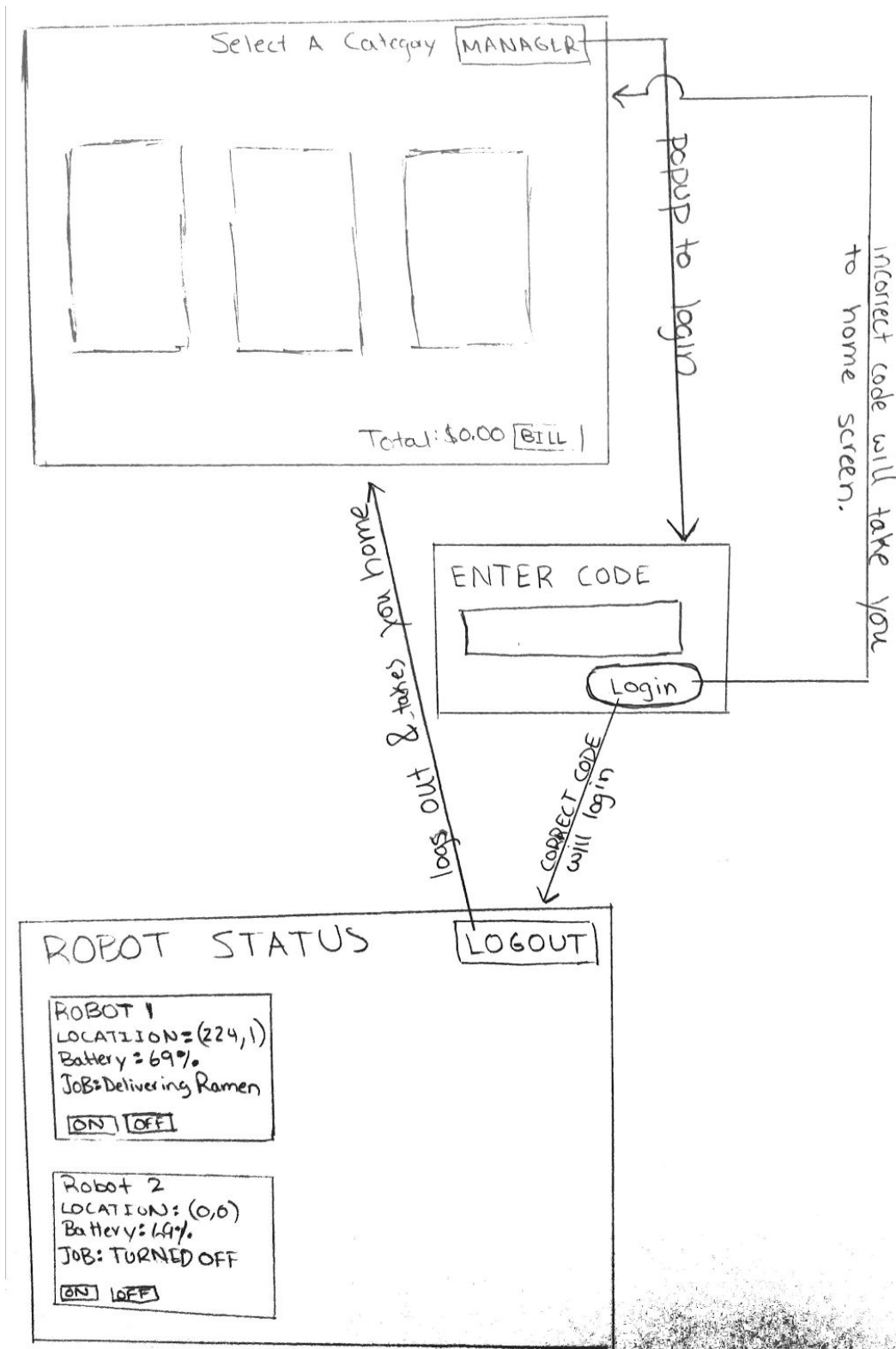
On the first page of the web application the user will be allowed to choose a category. By clicking on a category, they will be taken to the item selection. When an item is selected they will be taken to the order details page. This page will give a brief description of the food and allow the user to add changes to the order. When they place an order, they will get a notification that the order has been placed. Each view also has a back button, that goes to the previous view.

Use Case #2: Delivering the Order



When delivering an order, the database on the main terminal is first inspected to determine whether or not an order has in fact been made. If it has not, the customer is alerted of this and is prompted to place an order, after which the main terminal fetches the placed order. After this, the robot moves to the stock to retrieve the order item; if it is out of stock, the customer is notified and prompted to again select another item to order. If it is in stock, then the robot brings it to the customer.

Manager Login and Robot Status on Web Application:



You can click on the manager button which will give you a popup view asking you to enter a code to login. If the correct code is entered you can access the Robot status page.

User Effort Estimation

- 1) Customer placing an order for 1 drink, 1 appetizer, and 1 entree, without making any changes to his/her order

Adding an item from each category requires 4 button presses:

- (1) Selecting the item category
- (2) Selecting the item from the category
- (3) Reviewing the details of the item
- (4) Returning to the home screen upon order confirmation

Lastly, the customer must press the “Bill” button on the home screen after he/she is finished with his/her meal to receive the bill. Therefore, this use case scenario requires $4*3+1 = 13$ button presses and 0 keystrokes (assuming no order changes made).

UI navigation: $1*3$ (for selecting each item category) + $1*3$ (for returning to home screen) + 1 (for bill request) = 7/13 button presses

Data entry: $1*3$ (for selecting each item) + $1*3$ (for confirming the order for each item) = 6/13 button presses

Note: Order confirmation has been considered “data entry” because customers can specify order changes within this screen.

- 2) Manager authenticating into web application to view robot status, then logging out of system

On the home screen, the manager first presses “manager”, and then enters the required security code. He/she then clicks “login”, and assuming that he/she correctly enters the security code, accesses data regarding robots in the facility. Lastly, the manager clicks “logout” to return to the home screen.

Therefore, this task requires 3 button presses and 1 instance of data entry.

UI navigation: 1 (“manager” click) + 1 (code entry) + 1 (“login” click) + 1 (“logout” click) = 4/4

Data entry: 0/4

The above results are due to the fact that the manager simply wants to observe robot data and is not aiming to modify it in any way for the purposes of this use case.

Section 5: Domain Analysis

a) Domain Model

In the following model derivations, the following definitions are used:

Responsibility - a necessary action that the system must fulfill

Type - a classification of aforementioned responsibilities

Type D, "Doing": a responsibility that requires action, such as manipulation of data or physical interaction of components

Type K, "Knowing": a responsibility that designates the storage of data

Concept Name - label that represents aforementioned responsibilities

(i) Concept Definitions

Use Case 1: Placing the Order

Responsibility	Type	Concept Name
Coordinate and delegate responsibilities of all concepts.	D	Controller
Accept user's touch input.	D	TouchInterface
Container for user's order data, consisting of list of foods and prices.	K	Order
Container for all of a given user's orders.	K	Bill
Container for all food items in the restaurant.	K	FoodList
Push order to the user's bill.	D	OrderAdder
Container for all customers' orders.	K	OrderQueue
Add item to user's order.	D	ItemAdder
Confirm with user that he/she would like to take an action.	D	WarningPopup
Push order to the robot's order queue.	D	OrderPusher
Receive payment for bill.	D	PaymentAcceptor
Display menu items for the user.	D	MenuDisplay
Show confirmation that the order has been pushed.	D	ConfirmPopup

Figure 6: Extracted concepts and their responsibilities for Use Case 1.

Use Case 2: Delivering the Order

Responsibility	Type	Concept Name
Coordinate and delegate responsibilities of all concepts.	D	Controller
Implement a procedure to pull and remove database records as robot processes orders.	D	DatabaseConnection
Block additional requests for delivering orders if the robot is already processing an order.	D	StatusChecker
Split a given order into the queue of actions that constitute it.	D	OrderParser
Correctly identify the item to be delivered from the available stock.	D	ItemChecker
Continuously check a timer to determine when to display a robot's status.	D	StatusChecker
Deliver the order to the correct table.	D	OrderDeliverer
Allow web app to receive messages regarding out-of-stock items.	D	StockChecker

Figure 7: Extracted concepts and their responsibilities for Use Case 2.

Use Case 3: Review the System

Responsibility	Type	Concept Name
Authenticate user credentials.	D	Authenticator
Container for user's information.	K	User
User information storage module.	K	UserList
Display the map of the restaurant, and robot's location, the current action, and current operating status of the robot.	D	DataDisplayer
List of queued actions.	K	ActionQueue
Receive information (location, operating status, current action) from the robot.	D	DataReceiver

Figure 8: Extracted concepts and their responsibilities for Use Case 3.

(ii) Association Definitions

Identifying associations for Use Case 1: Placing the Order

Concept Pair	Association Description	Association Name
Controller ⇔ TouchInterface	The Controller receives input from the TouchInterface, which is generated by user interaction, to ensure the user can interact with the device.	Touch Request
Order ⇔ OrderAdder	OrderAdder will store data to the Order container so that the user's bill is updated	Provide Data
OrderPusher ⇔ OrderQueue	The OrderPusher data is stored in OrderQueue to ensure that the order information is saved for the user	Order Request
MenuDisplay ⇔ Controller	The Controller should initiate MenuDisplay so that the user will be able to see what is available to order	Show Menu
PaymentAcceptor ⇔ Bill	Payments should be paid according to the amount calculated in the Bill, so that the PaymentAcceptor debits the correct amount from the user.	Pay bill
WarningPopup ⇔ TouchInterface	The user must be able to confirm his or her action by touching the choices that WarningPopup has, through the TouchInterface.	Confirmation
ConfirmPopup ⇔ TouchInterface	TouchInterface will display a message to the user indicating that the confirmation has been received.	Confirmation Received
FoodList ⇔ MenuDisplay	MenuDisplay should extract container of information regarding items from the FoodList to show users the restaurant's menu.	Prepare data

Figure 9: Relationships between concepts seen in Use Case 1 and their importance.

Identifying associations for Use Case 2: Delivering the Order

Concept Pair	Association Description	Association Name
Controller ⇔ DatabaseConnection	Controller ensures that there is a connection between the application and the server by checking DataBaseConnection. The server connection is needed to store data from users and associated systems.	Check Connection
Controller ⇔ StatusChecker	Controller would see if robot is delivering an order so that the robot can finish its current task. The Controller will also establish an interval by which to execute this concept.	Check status
Controller ⇔ OrderParser	The Controller will break down the order so that the robot logically follows specific steps to obtain the requested items in that order.	Generate Actions
Controller ⇔ ItemChecker	The Controller allows the robot to distinguish features a requested item to ensure that the item is correctly identified.	Identify item
Controller ⇔ OrderDeliverer	OrderDeliverer receives coordination data from the Controller so that the robot delivers the order to its correct destination.	Delivery
Controller ⇔ StockChecker	Controller returns processed data to give StockChecker an updated count of what items are available in the restaurant.	Check stock

Figure 10: This table describes the relationships between each concept in Use Case 2 and explains why they need to work together.

Identifying associations for Use Case 3: Review the System

Concept Pair	Association Description	Association Name
Authenticator ⇔ User	Authenticator takes username and password entered through User and temporarily holds the login request for verification.	Begin Authentication
Authenticator ⇔ UserList	Authenticator attempts to match credentials stored in UserList to match the credentials of a user that is attempting to log in.	Verify User
DataDisplay ⇔ ActionQueue	The ActionQueue has a list of queued actions that DataDisplay checks in order to give an update on the display	Show actions
DataReceiver ⇔ DataDisplay	The DataReceiver obtains data about the robot that DataDisplay continually checks in order to provide live updates.	Send Info

Figure 11: This table lists relationships based on Use Case 3 and explains why they need to work together.

(iii) Attribute Definitions

Attributes for Use Case 1: Placing the Order

Concept	Attributes	Attribute Description
Controller	Coordination	Used to interact with various components simultaneously to ensure smooth transitions and multi-component effort to execute tasks.
	Task Distribution	Ensures that information and commands are sent to the correct component that can execute such requests.
TouchInterface	Deciphering user input	Ability to convert touch of screen by user into readable data by the system. Necessary to simulate interaction between user and display.
PaymentAcceptor	Transaction Record	Tracks transactions for restaurant's records to document method of payment, approval of credit or debit card, and time of transaction.
MenuDisplay	Data-to-image conversion	Any data, information, or confirmation that the device is programmed to output must be readable to the user via the display.

Figure 12: Attribute definitions based on Use Case 1.

Attributes for Use Case 2: Delivering the Order

Concept	Attributes	Attribute Description
Controller	Coordination	Exchanges data with robot for positioning and movement tracking in order to deliver the correct order in a reasonable time.
	Task Distribution	Breaks down data from orders so that robot can follow a sequential set of commands to retrieve items and orders to be delivered.
DatabaseConnection	Pull request	Obtains and, if needed, removes data from the database that the webapp stores from orders and other information about the user.
ItemChecker	Object identifier	Program to visually identify items by key features.
StatusChecker	Order interrupter	Temporarily holds a series of commands that delivers an order if another order is being delivered.
StockChecker	User notifier	Allows message to be displayed after stock for a specific item is unavailable.
OrderDeliverer	Delivery confirmation	Once an order is delivered to its correct table, the next order in queue is looked at.
OrderParser	Data division	Information lumped into an order is split into logical steps that the robot can follow.

Figure 13: Attribute definitions based on Use Case 2.

Attributes for Use Case 3: Review the System

Concept	Attributes	Attribute Description
Authenticator	Verify identity	Checks to see if user input matches pre-existing credentials that are stored in a database.
DataReceiver	Find location	Locates robot and records position within restaurant.
	Check status	Pings the robot to see if it is operating correctly
	Track actions	Retrieves information about what the robot is doing
UserList	Check role	While credentials are being verified, if a match is made, the role of the user logging in is checked for access privileges.
User	Username container	Field provided to user to enter user name
	Password container	Field provided to user to enter password
DataDisplay	Periodic checking	Makes sure that data being displayed is up-to-date by performing status checks in intervals.

Figure 14: Attribute definitions based on Use Case 3.

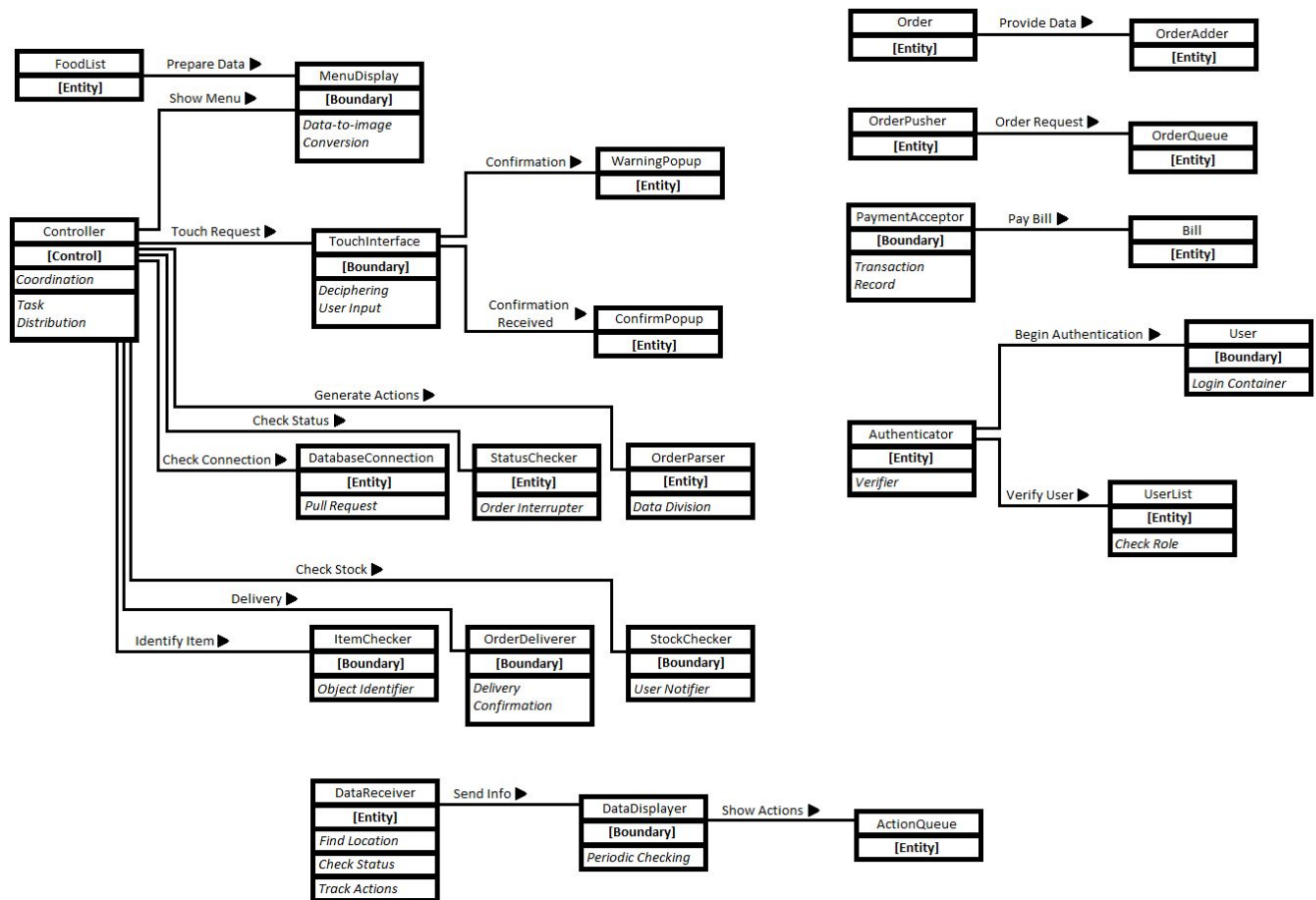


Figure 15: The Domain Model

The domain model was constructed by first extracting the responsibilities that needed to be completed from each use case, and by delegating these tasks to appropriate concepts. After this, the attributes of each concept were extracted so that these concepts can contain the information needed to perform their operations. Moreover, associations/relationships between different concepts were determined; these attributes and associations were then incorporated into the concept illustration, and the domain model was completed. For convenience, each concept was labelled as one of the following: control (which is responsible for delegating tasks to specific concepts), boundary (a concept that directly interacts with the external actors), or entity (a concept that is internal to the system and that therefore does not interact with the actors).

(iv) Traceability matrix

Use Case	PW	Domain Concepts																								
		Controller	<u>TouchInterface</u>	Order	<u>FoodList</u>	Bill	<u>OrderAdder</u>	<u>OrderQueue</u>	<u>ItemAdder</u>	<u>WarningPopup</u>	<u>OrderPusher</u>	<u>PaymentAccept</u> <u>or</u>	<u>MenuDisplay</u>	<u>ConfirmPopup</u>	<u>DatabaseConnec</u> <u>tion</u>	<u>StatusChecker</u>	<u>OrderParser</u>	<u>ItemChecker</u>	<u>OrderDeliverer</u>	<u>StockChecker</u>	Authenticator	User	<u>UserList</u>	<u>DataDisplay</u>	<u>ActionQueue</u>	<u>DataReceiver</u>
UC-1	8	X	X	X	X	X	X	X	X	X	X	X	X													
UC-2	28	X												X	X	X	X	X	X							
UC-3	19																			X	X	X	X	X	X	X

Figure 16: Shows how our use cases map to our domain concepts.

b) Systems Operation Contracts

Operations Contract Table 1

Name	Place Order
Responsibilities	Allow customer to give order to robot through Web App on a mobile device
Use Cases	UC-1
Exception	No exceptions
Preconditions	Display a menu through a Web App on a tablet or mobile device.
Postconditions	Order is placed in the database to be carried out by the robot.

Operations Contract Table 2

Name	Deliver Order
Responsibilities	Based on items requested by customer, the robot will obtain the item from the stock and deliver it to the customer.
Use Cases	UC-2
Exception	Item is out of stock
Preconditions	Order is placed in the database by the customer.
Postconditions	Order has been received by the customer.

Operations Contract Table 3

Name	Request information from terminal
Responsibilities	Allow manager to access information about the robot through the restaurant's server.
Use Cases	UC-3
Exception	No exception
Preconditions	Robot sends information to the terminal about its' condition.
Postconditions	Manager is informed about the state of the robot and the robot's current action.

c) Mathematical Models

Particle Filter / Localization:

We use the particle filter algorithm for localization of our robot. Given a map of the environment, the algorithm estimates the position and orientation of a robot as it moves and senses the environment. The algorithm uses a particle filter to represent the distribution of likely states, with each particle representing a possible state, a hypothesis of where the robot is. The algorithm typically starts with a uniform random distribution of particles over the configuration space, meaning the robot has no information about where it is and assumes it is equally likely to be at any point in space. Whenever the robot moves, it shifts the particles to predict its new state after the movement. Whenever the robot senses something, the particles are resampled based on recursive Bayesian estimation (how well the actual sensed data correlate with the predicted state). Ultimately, the particles should converge towards the actual position of the robot. The use of fiducial markers to act as landmarks (chilitags in our case) helps with the implementation of this algorithm on our physical robot.

Edge Detection / Chilitags:

We will use edge detection to aid in the localization, object detection and identification. Edge detection is an image processing technique for finding the boundaries of objects within images. It works by detecting discontinuities in brightness. Edge detection is used for image segmentation and data extraction. The algorithm used to detect chilitags utilizes edge detection to determine if a set of edges is a chilitag and also determine its id, in the case it is one.

PID / Motion:

We will be using PID(proportional–integral–derivative) controller to make the robot's movements more accurate. A PID controller continuously calculates an error value as the difference between a measured process variable and a desired setpoint. The controller attempts to minimize the error over time by adjustment of a control variable, such as the position of a control valve, a damper, or the power supplied to a heating element, to a new value determined by a weighted sum. With the PID controller the robot will take into consideration the external forces and still function accurately (for example moving in a straight line or picking up an object).

Inverse Kinematics / Motion of arm:

We will use kinematics equations of the robot to determine the joint parameters that provide a desired position of the end-effector. Inverse kinematics transforms the motion plan into joint actuator trajectories for the robot. The movement of a kinematic chain of a robot is modeled by the kinematics equations of the chain. These equations define the configuration of the chain in terms of its joint parameters. Forward kinematics uses the joint parameters to compute the configuration of the chain, and inverse kinematics reverses this calculation to determine the joint parameters that achieves a desired configuration. Inverse kinematics formulas allow calculation of the joint parameters that position a robot arm to pick up a part.

Section 6: Plan of Work

The product ownership section from the proposal is as follows:

3 people → Work on robot (build / movement / motion)

2 people → Computer vision (predictive analysis)

2 people → Web application

This breakdown was modified in that the first 2-member subgroup was delegated the task of establishing the MongoDB database. The breakdown of responsibilities are as follows:

Neil:

- I have begun using Flask and Python to create the web application. In particular, I have established a simple offline website that routes the user to different webpages based on an option selection.
- I am currently learning how to use PyMongo to allow the web application and database to communicate with each other. After this, I will work on developing an application that allows a user to select from a number of buttons, each of which corresponds to a particular function implemented by the robot (e.g. move to a specified location, grab a specified item, etc.) in order to ensure proper robot operation.
- In the future, I will construct the actual web application, developing the front-end and ensuring that each customer selection calls the appropriate robot function.
- I have also assisted in coordinating between subgroups, especially in ensuring that any dependencies between them are handled efficiently.

Vineet:

- I have began creating basic web applications using Flask; these applications can create an offline website which can route the user to different pages depending on which button he or she presses.
- Currently I am learning about PyMongo to allow the web application to communicate to the Mongo database. After this is accomplished, I will work on the user being able to instruct the robot to implement different functions through the web application.
- Further down the road, I will work on making this web application display a small menu for the customer to order from and an administrator login for the manager to view the robot status.

Brice:

- I have created a simple web application in HTML, with two user input buttons that affect on screen elements.
- I am currently adding the feature of sending packets of data to a Mongo database using these buttons.
- In the future, the simple input methods will be expanded to include more information, and will be formatted to a standard.

Ajay:

- Worked on building the robot / arm
- Created framework for managing entire robotic system
- Developed serial communications protocol between nvidia system on chip and arduino microcontroller, allow motor control and sensor feedback from robot
- Currently working on localization with particle filters. In testing phase.
- Exploring the use of an a* path finding algorithm in order to traverse restaurant

Srihari:

- Worked on building the robot / arm
- Helped in programing the robot. Helped parse the encoder values.
- Currently working on learning opencv, to detect objects and determine depth. Further exploring the use of fiducial markers (e.g. chilitags) in order to ease depth determination.

Cedric:

- Currently working on the Server and communication between the User Interface and the database
- Will work on communication between the robot and the database
- Will work on the operation of the robot's arm

Jonathan:

- Working on Server and involved in setting up communication between User Interface and database that will utilize the Server
- Will be aiding in construction of robot
- Will be assisting in establishing communication between robot and the database

Current overall progress:

We have reached a point where we have a basic robot frame that is able to move around. This frame has a camera attached to it, and we are currently working on using fiducial markers (the chilitags) as landmarks in a particle filter in order to determine our location in a predetermined map of our restaurant. This map will always be static, and this turns this into something manageable. Current work on the robot involves perfecting this algorithm and testing it on the robot to confirm its operation.

The barebones skeleton of the web app has been created, and is in the process of being connected to the database so it can send information back and forth. The schema of the database is being outlined, and the plan is to connect this webapp to the current robot framework through the database, in order to test for latency and proper operation.

References

<http://www.foodservicewarehouse.com/blog/managing-problem-employees/>

https://en.wikipedia.org/wiki/Inverse_kinematics

<http://www.mathworks.com/discovery/edge-detection.html?requestedDomain=www.mathworks.com>

https://en.wikipedia.org/wiki/PID_controller

https://en.wikipedia.org/wiki/Particle_filter