TEAM X

On-line Restaurant Order and Delivery System
Design Report
For Online Restaurant Application

Version 2.0

grade: 95
comments: overall a nice report. The main flaws are the petri-nets: either are wrong or didn't quite reflect the details you described.
I don't see much activity in your repo yet, most python codes are skeletons done by one member, hope the entire team can work together having a nice working system.

Revision History

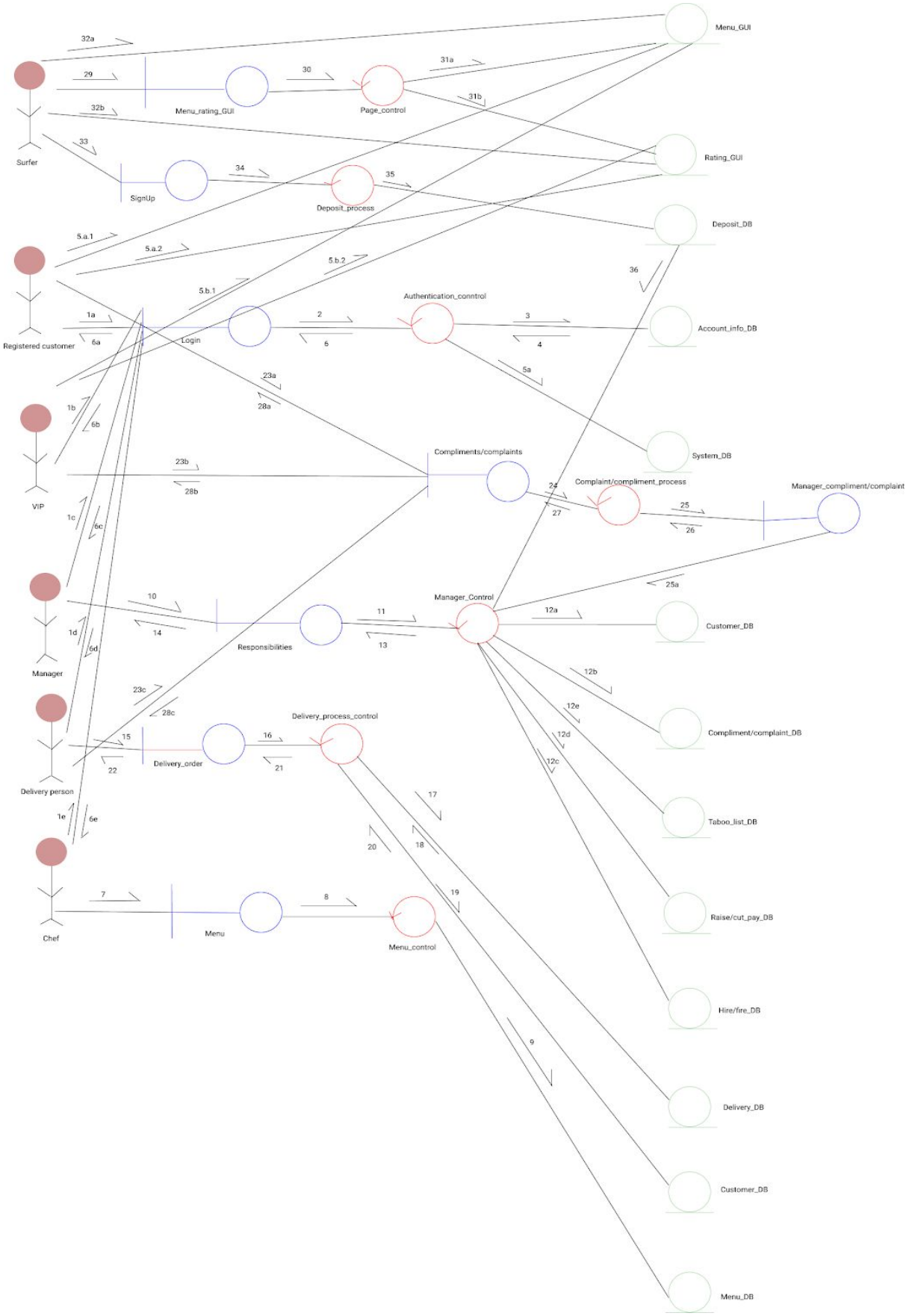
Date	Version	Description	Author
13/10/2020	1.0	Creating a login screen	Saiful Islam
18/10/2020	1.0	Specification report for Phase 1	Hafsa Nadim Tanzil Baraskar Saiful Islam Daniel Lichter Dante Betancourt
9/11/2020	2.0	Database setup for system	Daniel Litcher
14/11/2020	2.0	Function prototypes added to fulfill specifications	Saiful Islam
17/11/2020	2.0	More function prototypes added to fulfill specifications	Saiful Islam
17/11/2020	2.0	Design report for Phase 2: includes the data structure, logic, diagrams, pseudo-code, and images to carry out the functionalities dictated by the specification.	Hafsa Nadim Tanzil Baraskar Saiful Islam Daniel Lichter Dante Betancourt

Table of Contents

1. Introduction

The following diagram is a collaboration diagram, which provides an overall picture of the online restaurant system. This report presents the details of the designed system in depth by showing various diagrams, such as Petri-net, collaboration class, and E-R diagrams. The report also contains pseudocodes for every method, demonstrations of major GUI screens of the system, information on group meetings, and more.

1.1 Collaboration Diagram of System



Description of Numbered Actions

1a. Registered customer login into account

1b. VIP login into account

1c. Manager login into account

1d. Delivery person login into account

1e. Chef login into account

2. Account is getting processed

3. Account is getting checked

4. Feedback response

5a. If the logging into account is successful, then registered customer, VIP, manager, delivery person, or chef are in the system

5.a.1.Registered customer views the menu

5.a.2. Registered customer views the ratings

5.b.1 VIP views the menu

5.b.2. VIP views the ratings

6. If logging into account is unsuccessful, then registered customer, VIP, manager, delivery person, or chef are not in the system

6a. Registered customer has to sign in again

6b. VIP has to sign in again

6c. Manager has to sign in again

6d. Delivery person has to sign in again

6e. Chef has to sign in again

7. Chef puts in the description

8. Chef decides the menu

9. Menu is getting put together

10. Manager manages the restaurant

11. Takes on the responsibilities that a manager should have when dealing with customers (registered customer and VIP) and employees (chef and delivery person)

12a. Processes customer registration

12b. Handles the complaints and compliments

12c. Handles firing and hiring of the employees (chef and delivery person)

12d. Handles giving a raise or cutting pay for the employees (chef and delivery person)

12e. Keeps a list of taboo words

13. Mangers receives feedback

14. Manager receives feedback

15. Delivery person receives the delivery order

16. Have the delivery be processed

17. Check the delivery details

18. Respond

19. Delivery details such: there is enough deposit money in the account, and the address of the customer is correct

20. Customer receives the delivery

21. Delivery is successful

22. Delivery order is a success

23a. Registered customer compliments/complains

23b. VIP compliments/complains

23c. Delivery person compliments/complains

24. Compliments/complaints get processed

25. Manager is informed about these compliments/complaints

25a. Manager takes charge/control

26. Compliments/complaints feedback

27. Send response to the correct user (registered customer, VIP, or delivery person)

28a. Manager's response to registered customer

28b. Manager's response to VIP

28c. Manager's response to delivery person

29. Surfer view/browse the menu/ratings

30. Menu and ratings being processed

31a. Menu with the top three most popular dishes listed

31b. Top three highest ratings appear

32a. Surfers view the top three most popular dishes on the menu

32b. Surfers view the top three highest ratings

33. Surfers apply to be registered customers

34. With fixed amount of deposit money

35. Account is processed with the fixed amount of deposit money

36. Deposit database needs manager control

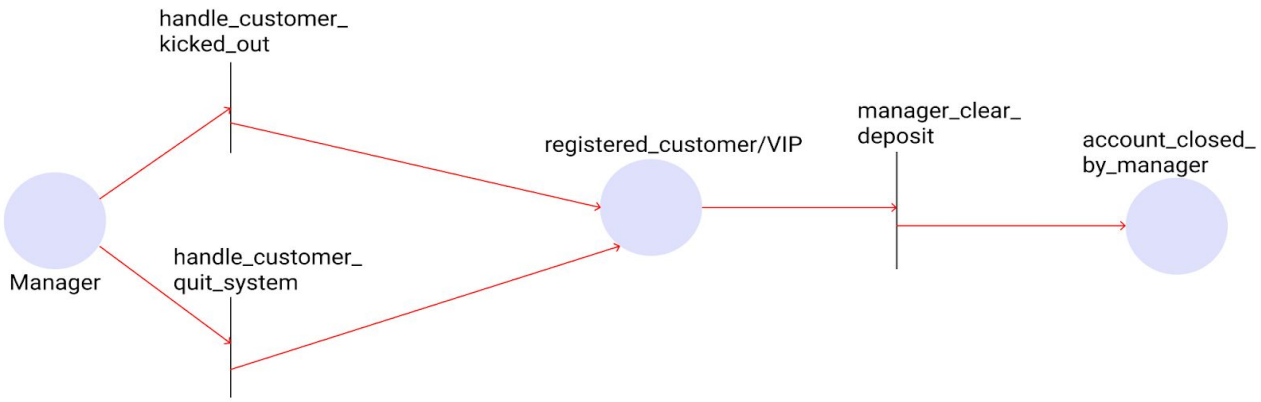
2. Scenarios and Diagrams for All Use Cases

A. Manager Diagrams

1. Customer kicked out/quit system (Petri-net)

Normal Scenario: If a customer (registered or VIP customer) is kicked out of the system or decides to quit the system, the manager will clear the deposit and close the account.

Exceptional Scenario: Customer had no money left in the account before they got kicked out of the system or quit the system.

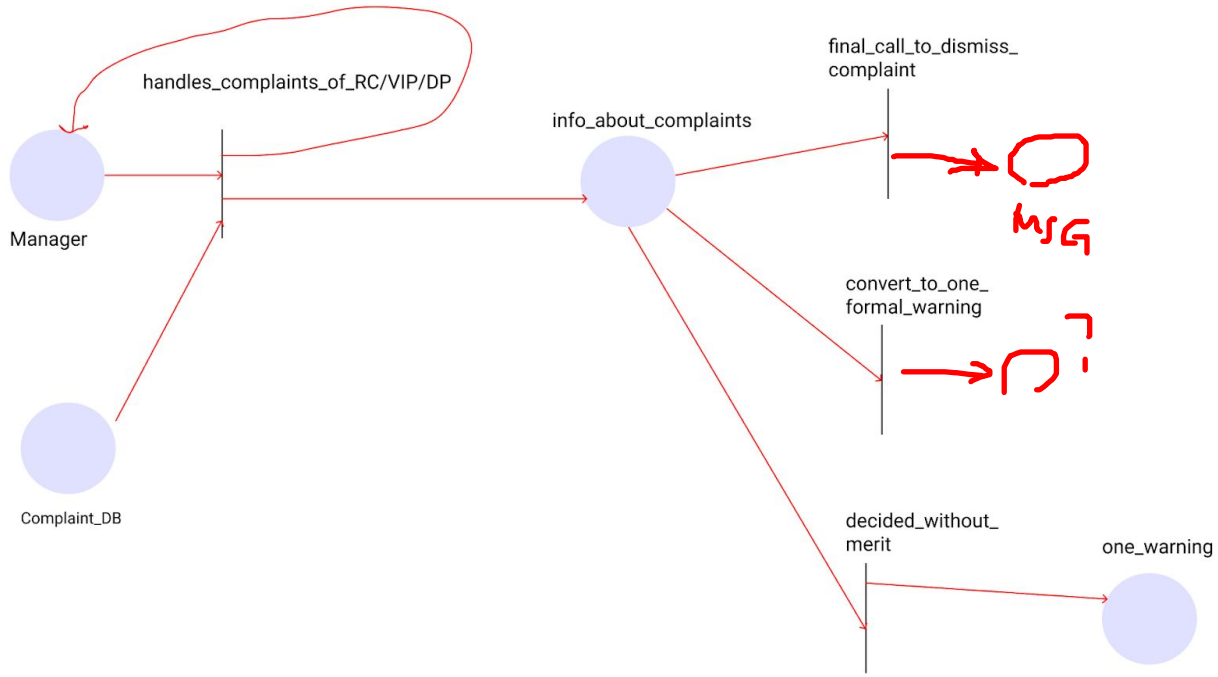


2. Manager handles complaints (Petri-net)

Normal Scenario: The manager handles the customers' complaints (registered customer and VIP) and delivery people. The complaint database contains all complaints. When handling the complaints, the manager will receive information about them and decide whether to dismiss the complaint or convert it into one formal warning. If the manager chooses a complaint made by a customer or delivery person to be without merit, then the respective customer or delivery person receives a warning.

Exceptional Scenario:

1. No warnings are given
2. Customer or delivery person already has prior warnings

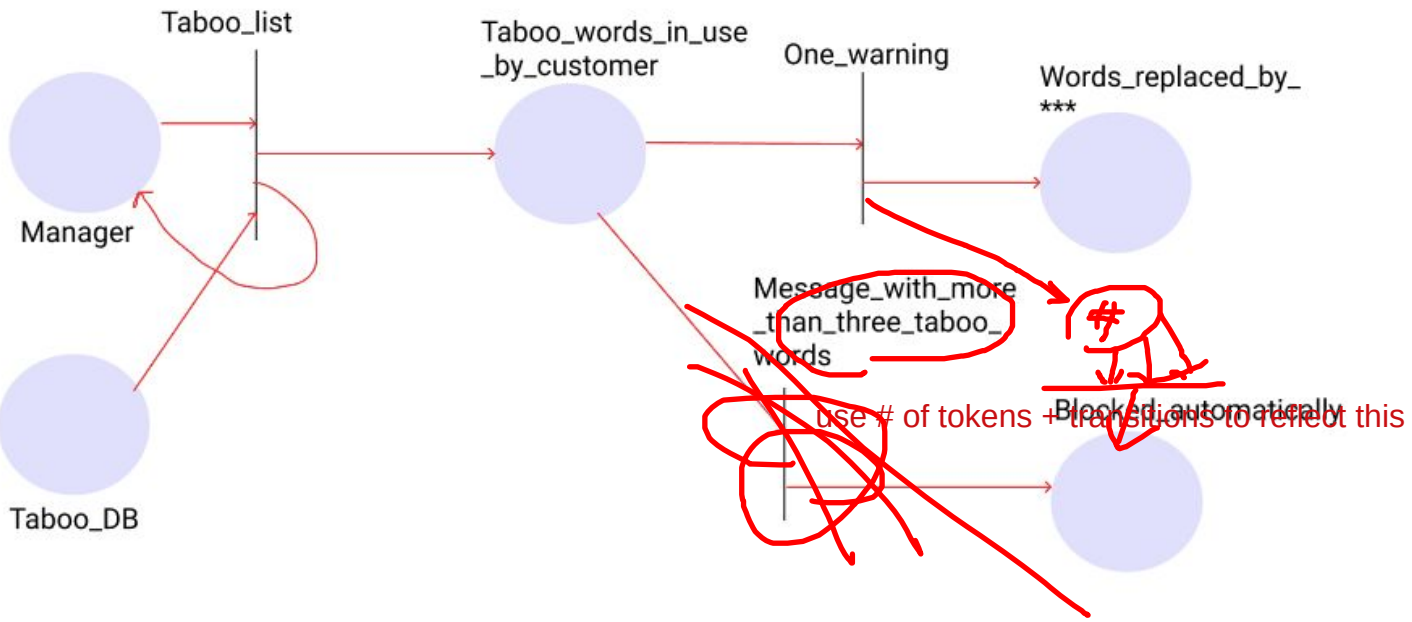


3. Taboo List (Petri-net)

Normal Scenario: The taboo database contains a list of taboo words. Customers will receive one warning automatically, and the taboo words will be replaced with ****. A message with more than three taboo words is blocked automatically.

Exceptional Scenario:

1. If a message includes taboo words that are separated by characters or combined with other words
2. If a message includes no taboo words
3. If a message contains exactly three taboo words

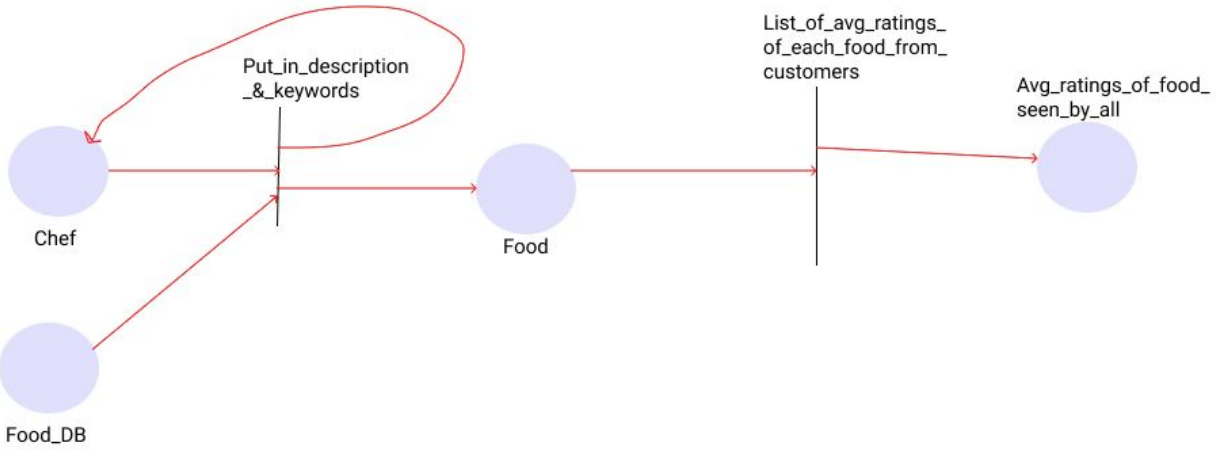


B. Chef Diagrams

1. Chef puts in menu items for the users to view (Petri-net)

Normal Scenario: Chef puts in the description and keywords for the food. The food database stores information about the menu items. The average ratings for each menu item, written by the customers, are available for any user to view.

Exceptional Scenario: The chef forgets to put in the description or keywords of the menu items



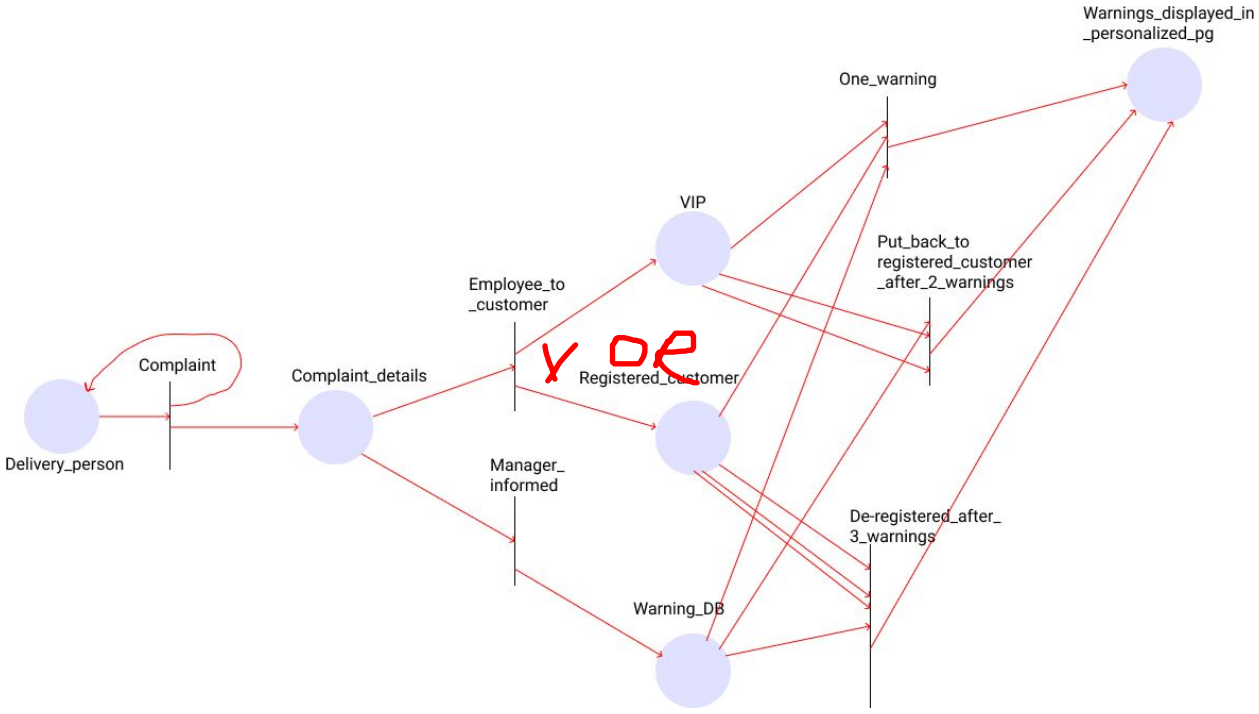
C. Delivery Person Diagram

1. Delivery person complains against customers

Normal Scenario: The delivery people make complaints about both registered customers and VIPs. The delivery people's complaints are informed to the manager. VIPs and registered customers receive a warning if the delivery people make a complaint about them. If a VIP customer receives two warnings, then they are back to being registered customers on the system. If a registered customer receives three warnings, then they are de-registered from the system. All these warnings are displayed on the personalized page for each type of customer, and the warning database stores all the warnings.

Exceptional Scenario:

1. Manager disapproves of the complaint, so no warnings are assigned.

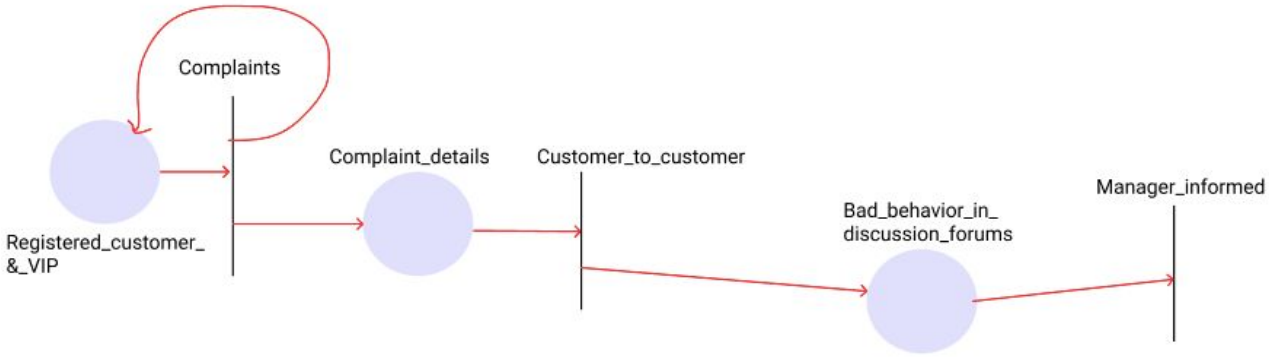


D. Registered and VIP Customer Diagrams

1. Complaints from customers about other customers (Petri-net)

Normal Scenario: Customers (registered and VIP) make complaints about other customers that do not behave appropriately in the discussion forums, and the manager is informed.

Exceptional Scenario: If there was no bad behavior in the discussion forums and a customer still makes a complaint about it



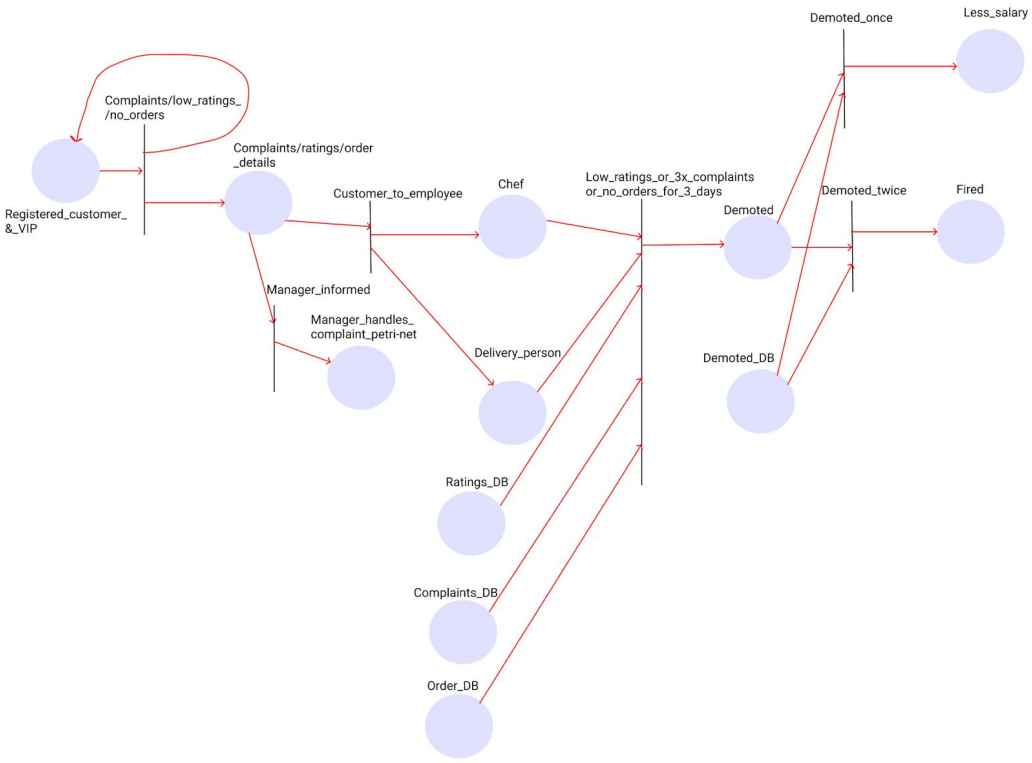
2. Complaints, low ratings, or no orders from customers on a menu item (Petri-net)

Normal Scenario: Customers can do the following things which will affect the chefs and delivery people: complain, give low ratings, or not order a menu item for three or more days. The manager is informed of all of this, and then he or she deals with the complaints (associated with manager_handles_complaint_Petri-net). If either the chef's dishes or delivery people receive low ratings or three complaints or no order for three days, they are demoted. Three databases called rating, complaints, and order are created for these low ratings/complaints/no orders, and a

demoted database is created for demotion. If a chef or delivery person is demoted once, they receive less salary. If demoted twice, a chef or delivery person is fired.

Exceptional Scenario:

1. If there are no complaints filed
2. If the chef or delivery person receives only one complaint
3. If the chef or delivery person receives two complaints
4. If the chef or delivery person does not receive orders for one or two days
5. If there are four complaints or more

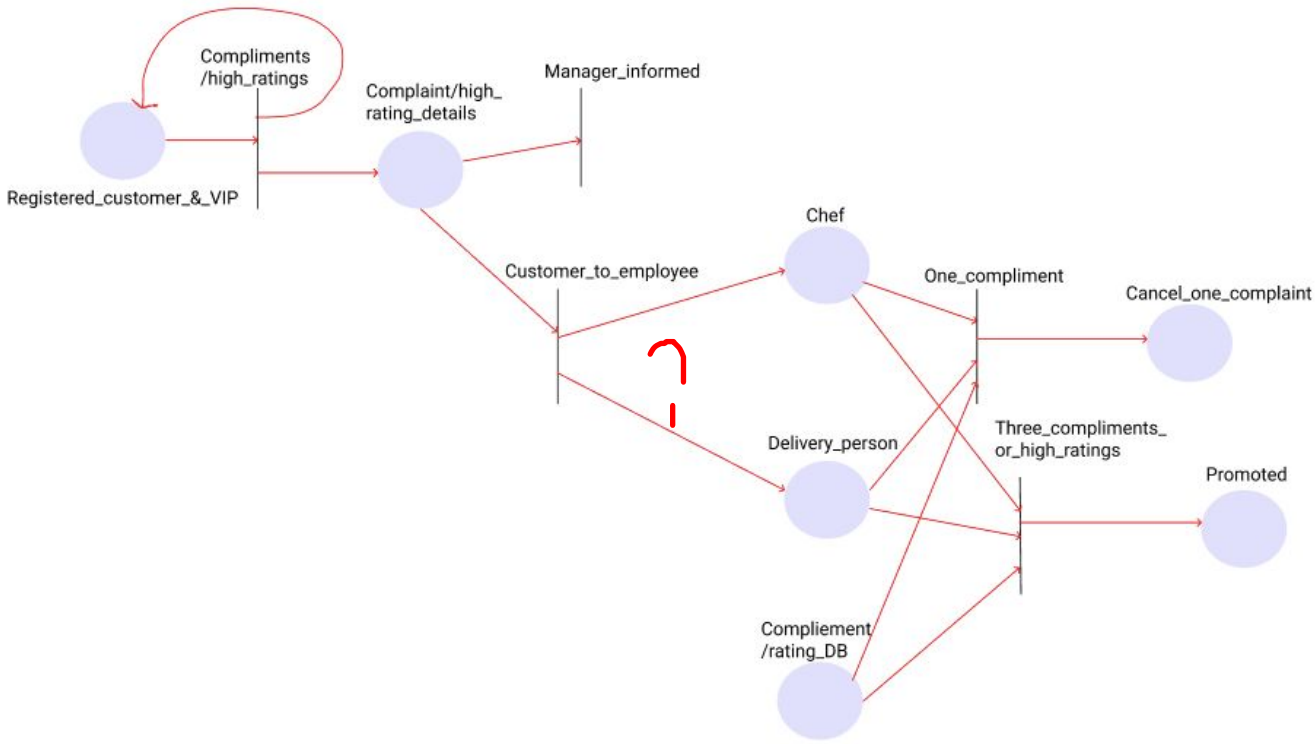


3. Compliments/ratings from customers (Petri-net)

Normal Scenario: The customers (registered and VIP) compliment/give high ratings to the employees (chef and delivery person). These high ratings/compliment details are informed to the manager. The compliment/rating database stores the compliments/high ratings. If the chef or delivery person receives one compliment, one complaint is canceled. If the chef or delivery person receives three compliments or high ratings, they are promoted.

Exceptional Scenario:

1. If registered customer or VIP give two compliments to either the chef or delivery person
2. If registered customer or VIP give more than three compliments to either the chef or delivery person

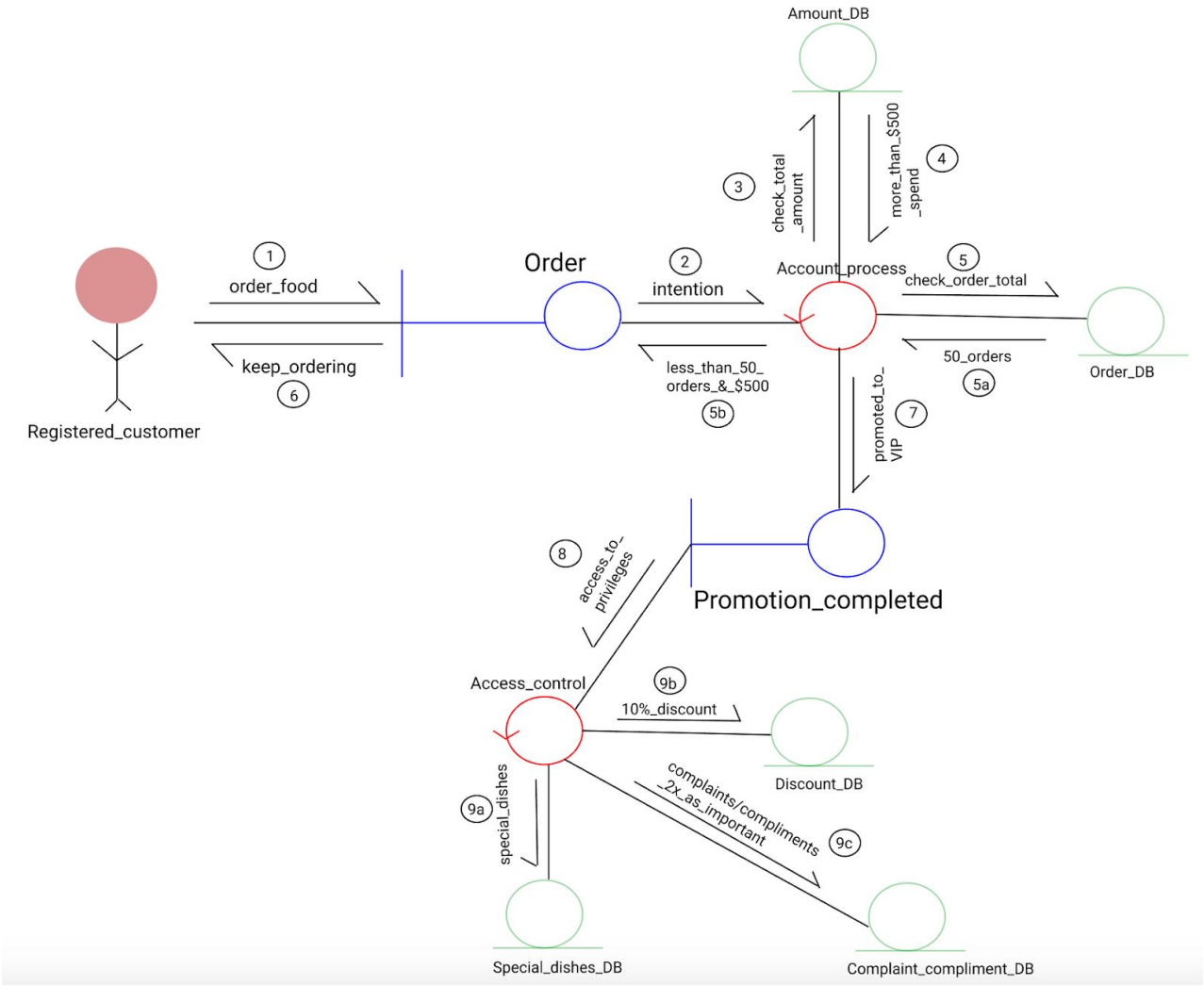


4. Registered customer to VIP customer

Normal Scenario: Registered customers can become VIP customers as long as they spend more than $500 or place 50 orders. If the registered customers do either of these two actions, they are promoted to VIP users. With this promotion, they have three special privileges: receiving a 10% discount on their ordinary orders, having access to specially developed dishes, and their compliments/complaints being counted twice as important.

Exceptional Scenario:

1. If the registered customer places more than 50 orders
2. If the registered customer spends exactly $500

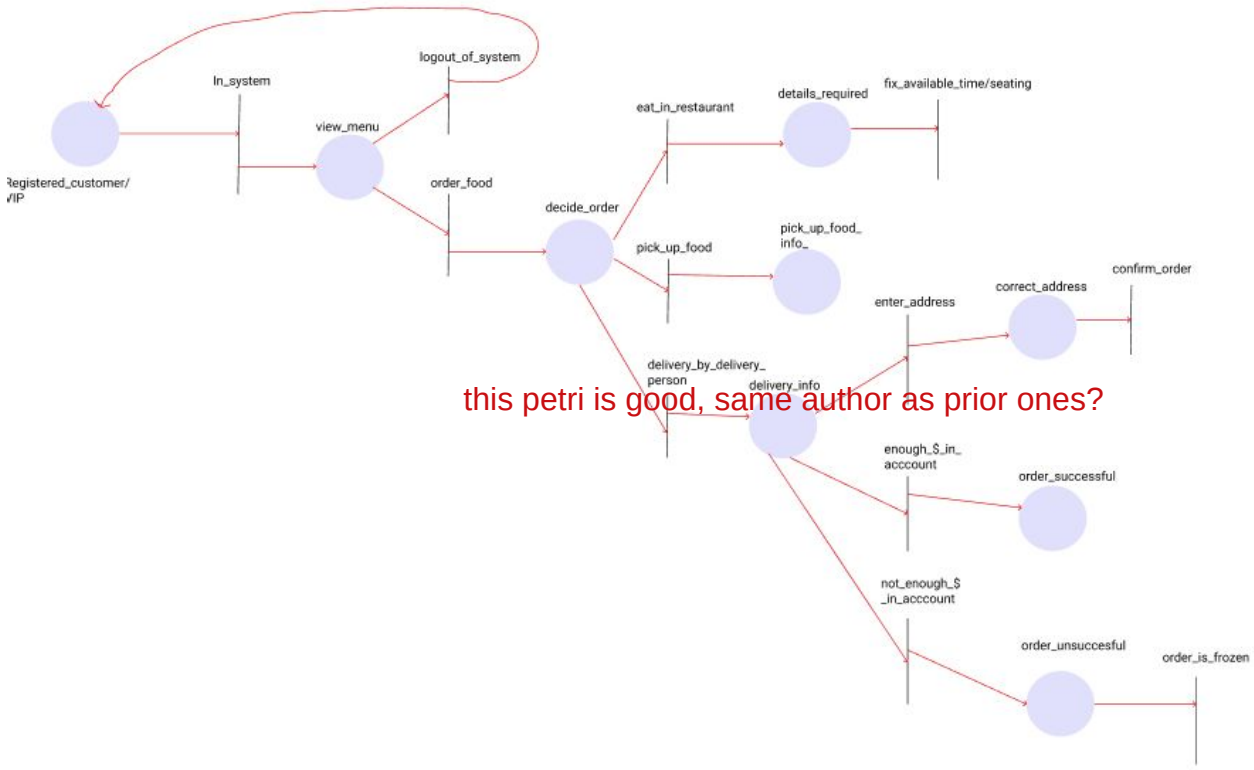


5. Three menu items upon customer login

Normal Scenario: Customers (registered and VIP) will log in to the system, and once they are in, they view the menu and order their food. Customers have three options: eat food at the restaurant, pick up the food from the restaurant, or have the food delivered to their house by a delivery person. Eating at the restaurant will require customers to fix available time/seating. Picking up the food will require details such as knowing when to pick it up. The delivery option involves information such as the customer's address and makes sure the account has enough money. If the address is correct, the order will be confirmed, and if there is enough money in the account, the order will be successful. If the account does not have enough money, then the order is unsuccessful, and it is frozen.

Exceptional Scenario:

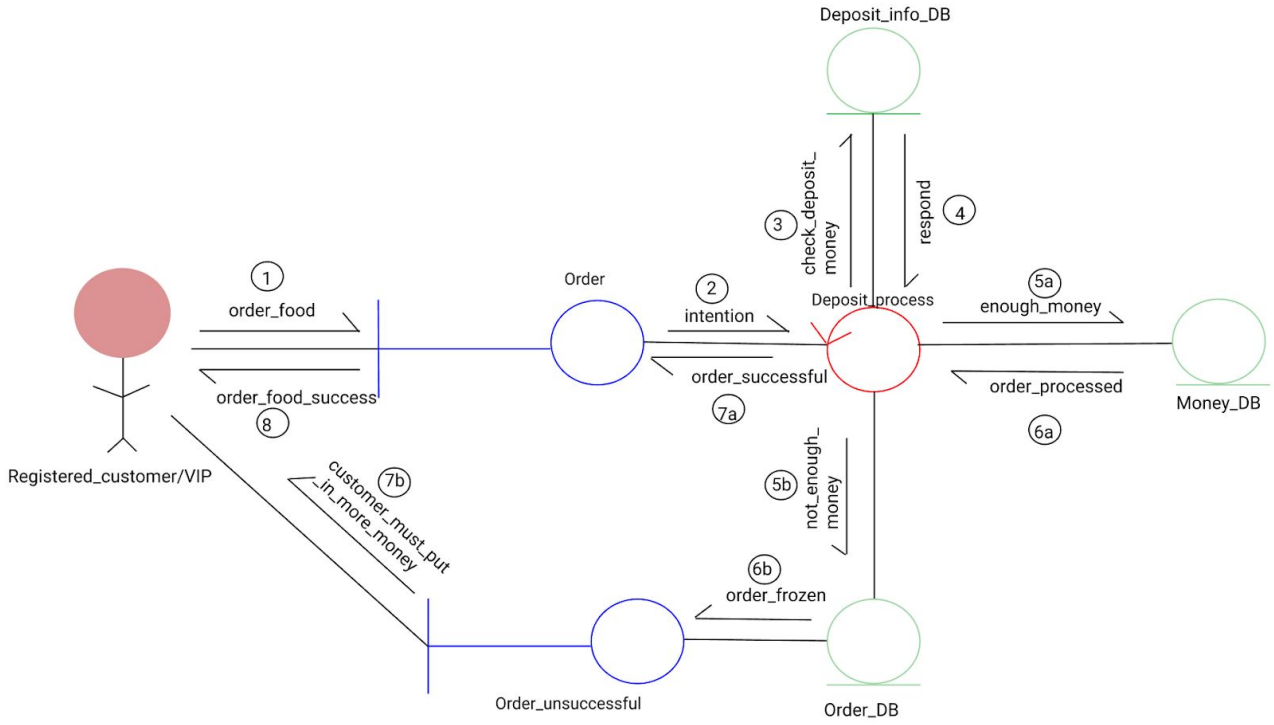
1. The customer enters an incorrect address.
2. Attempt to login by a user who has been kicked out of the system or decided to quit the system.


logout_of_system
In_system
view_menu
Registered_customer/
VIP
order_food
decide_order
eat_in_restaurant
details_required
fix_available_time/seating
pick_up_food
pick_up_food_
info_
confirm_order
correct_address
enter_address
delivery_by_delivery_
person
delivery_info
this petri is good, same author as prior ones?
enough_$_in_
acccount
order_successful
not_enough_$
_in_acccount
order_unsuccesful
order_is_frozen

6. Customer sufficient deposit check (Class diagram)

Normal Scenario: After the customer (registered and VIP) orders food, the system will check if the customer has enough deposit money in their account. If there is enough money deposited, then the order will be successful. If the customer does not have enough money deposited, then the order will be frozen, and it will tell the customer to put in more money.

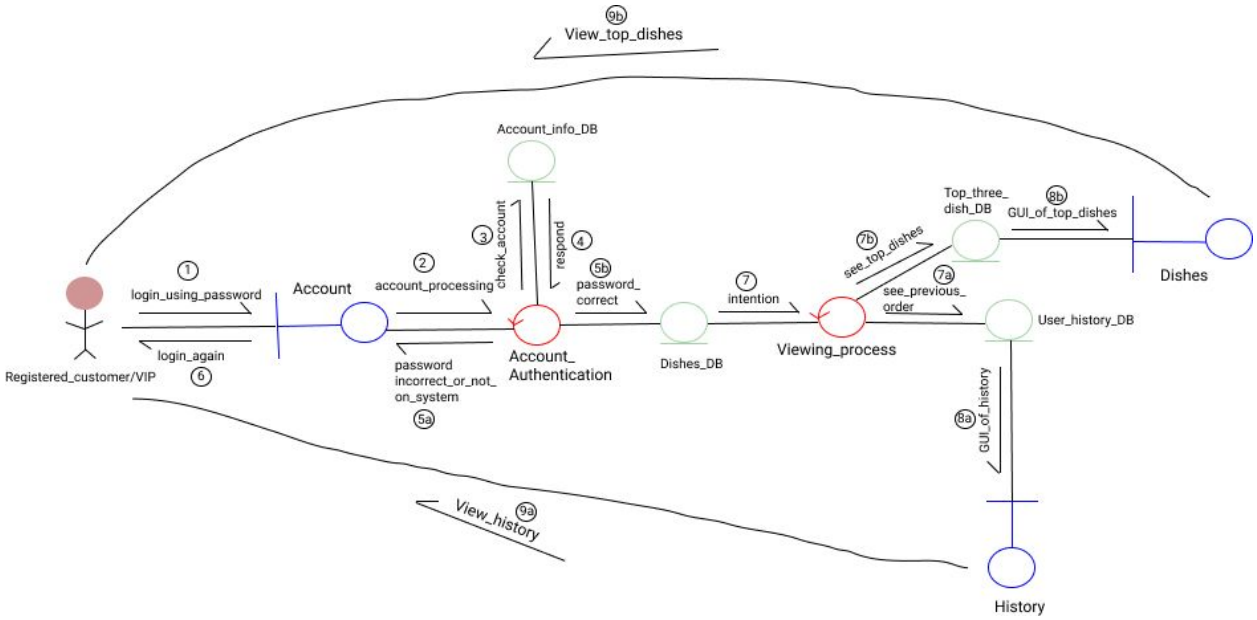
Exceptional Scenario: No exceptional scenario for this



7. Customers view dishes after correct login (Class diagram)

Normal Scenario: Customers (registered customer and VIP) log in using their login information. If the password is incorrect, the customers have to log in again. If the password is correct, customers are in the system and have access to view the dishes. Based on their history of prior choices, customers can view the top 3 listing dishes and view their history.

Exceptional Scenario: If a customer who was kicked out of the system tries to log in, the system will not allow the user to enter.



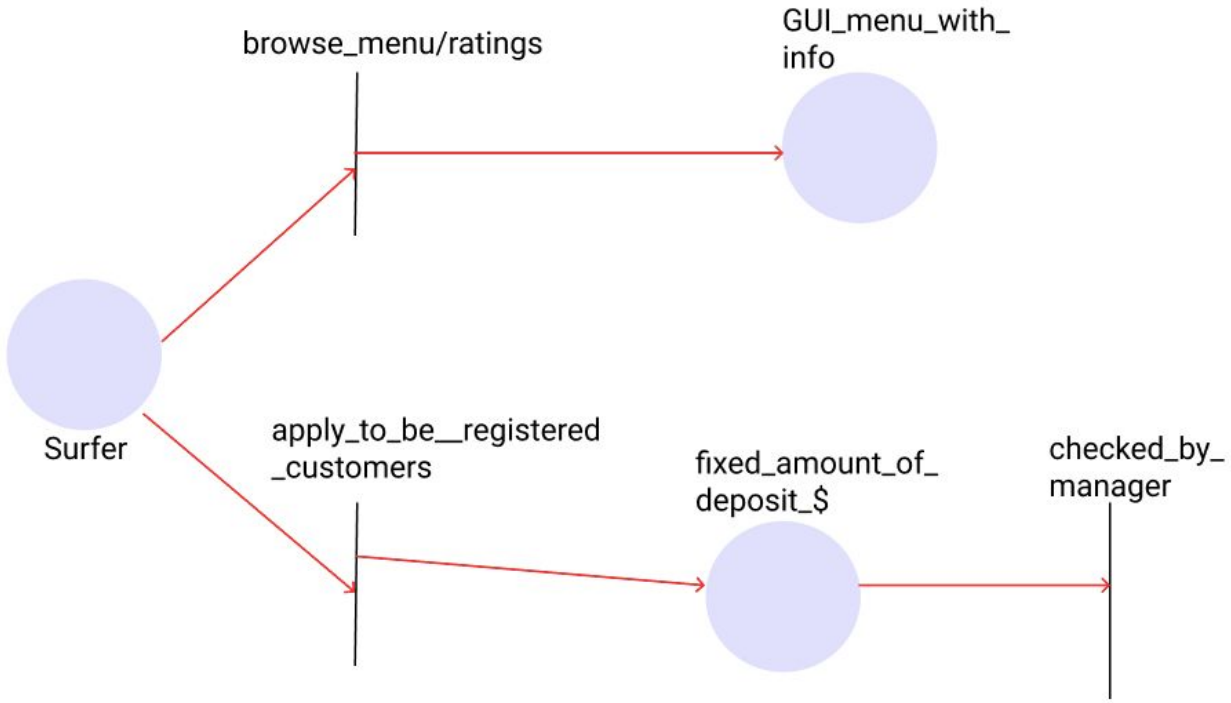
E. Surfer Diagrams

1. Surfer view rating/menu (Petri-net)

Normal Scenario: Surfers can browse the menu and ratings. They can also become registered customers by first depositing money and then getting checked by the manager.

Exceptional Scenario:

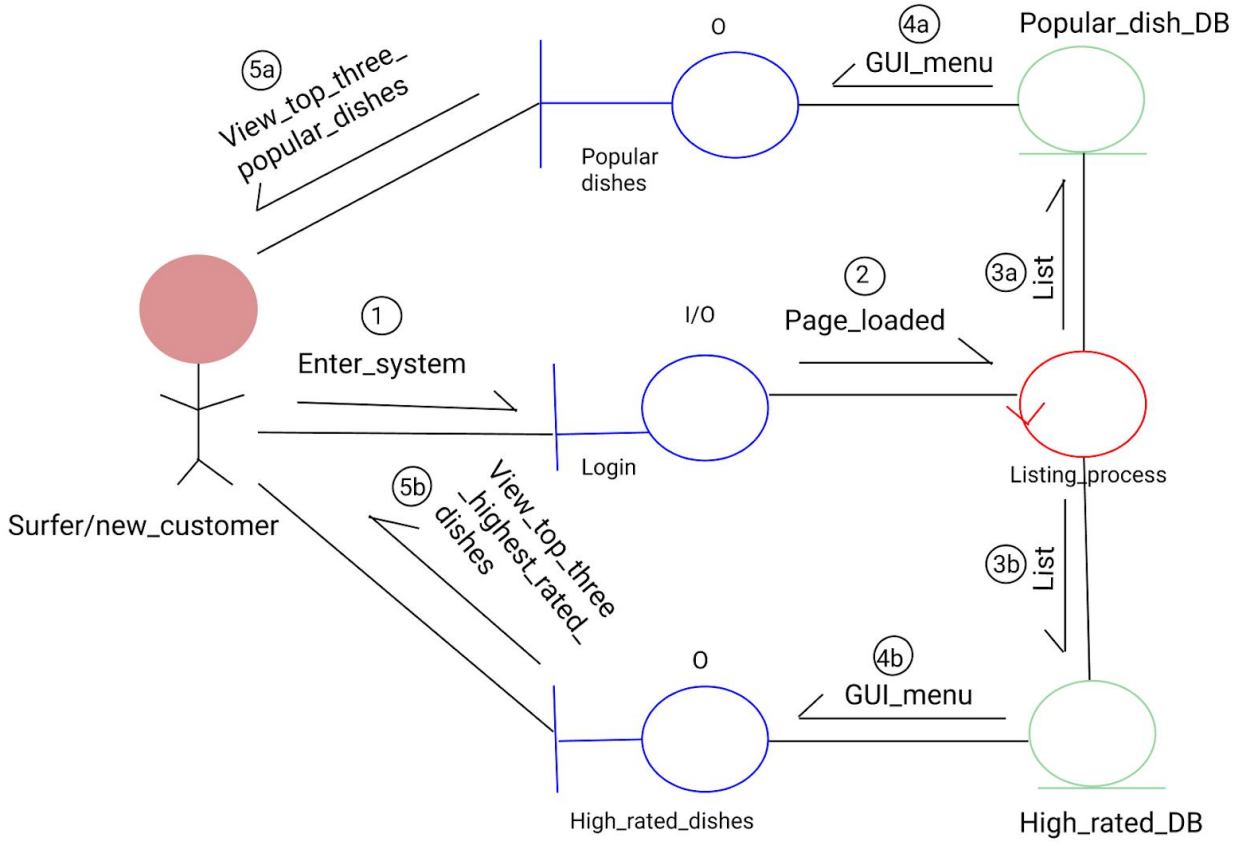
1. If customers did not rate the food, then there will be no ratings to show the surfers.
2. If surfers apply to be customers with wrong credentials, such as credentials that already exist on the system, the process will be rejected.



2. Surfer or new customer view top 3 popular/high rated dishes

Normal Scenario: When surfers or new customers enter the system, they have the option to view the top three most popular dishes and the top three highest-rated dishes.

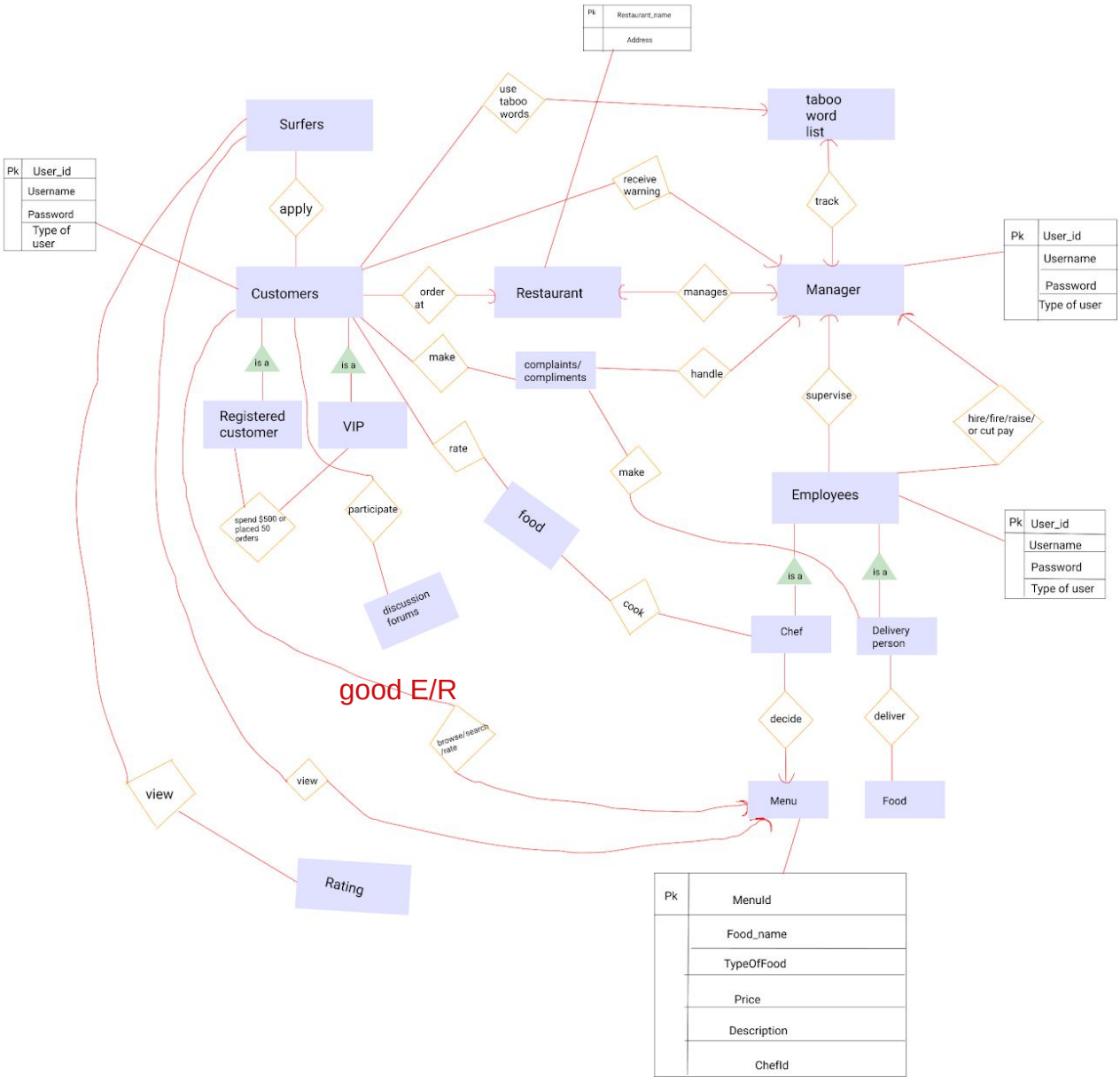
Exceptional Scenario: 1. If there are no orders made by customers, there will be no ratings and no popular dishes to show surfers/new customers



3. E-R Diagram

This E-R diagram shows the main relationship between the entities of the entire system.

It also includes attributes and keys for each class.



4. Detailed Design

This section includes the pseudo-code for every method, which delineates the input/output and main functionalities.

Manager

1. Taboo list

```
def tabooCheck (user_id, customer_message):
        Arr_for_taboo = [list of taboo words]
        if i in Arr_for_taboo:
                if  (customer_message = = i)
                        customer_message = "****"
                        warning_issued = (user_id,  display_warning)
                        if (i > 3):
                                Message_blocked
```

2. Manager checks customer quit/ kick out system

```
def customer_outOf_System (user_id):
        if( vip or registered customer quit system):
                Manager_handles = clear.deposit
                clear.deposit = close.account(user_id)
        elif (vip or registered customer kicked out of system):
                Manager_handles = clear.deposit
                clear.deposit = close.account(user_id)
```

3. Manager handle complaints

```
def manager_recieveComplaints(user_id):
        if(manager receive complaints):
                final_call = dismiss.complaint
        elif:
                warning = give.warning
                inform both parties
        elif:
                withoutMerit = give.warning
```

Chef

4. Chef puts description of food

```
def chef_updates_menu ():
        chef = add.description_food  // chef adds description of the food
        update.food_db                  // database of food is updated
```

5. Demote chef

```
def chef_demote(user_id):
        if chef_rating low:
                demote_chef.user_id = less_salary
                else:
                        chef_complaint = 3:
                        demote_chef.user_id = less_salary
                else:
                        chef_order = 0:
                        demote_chef.user_id = less_salary
```

6. Demote chef twice

```
def chef_demoted_twice(user_id):
        chef_demote() // call the chef_demote function
        if (chef_demote = =2 ):
                employee.fire = user_id.chef_fire
```

Delivery Person

7. Delivery person complaint

```
def deliveryPerson_complaint(user_id):
        deliveryPerson.complaint = manager.informed
        if (deliveryPerson complaint on VIP one time):
                VIP.warning =  update.warning_db
                elif (deliveryPerson complaint on VIP two times):
                        VIP.warning = update.warning_db
                        VIP_to_customer(user_id) // Call the function to demote to registered
customer
        if (deliveryPerson complaint on registered customer one time):
                R_customer.warning = update.warning_db
                elif(deliveryPerson complaint on registered customer three times)
                        R_customer.warning = update.warning_db
        deregister_customer(user_id, account_db) //call the function to deregister
```

8. Demote Delivery Person Salary

```
def deliveryPerson_demote(user_id):
        if delivery_rating low:
                demote_deliveryPerson.user_id = less_salary
                update.complaint_db
                else:
                deliveryPerson_complaint = 3:
                        demote_deliveryPerson.user_id = less_salary
                        update.complaint_db
                else:
                        deliveryPerson_NoDeliveryOrder = 0:
                                demote_deliveryPerson.user_id = less_salary
                                update.complaint_db
```

9. Demote Delivery Person

```
def deliveryPerson_demoted_twice(user_id):
        deliveryPerson_demote() // call the chef_demote function
        if (deliveryPerson = = 2 ):
                employee.fire= user_id.deliveryPerson_fire
```

Customer (VIP and Registered Customers)

10. Promote registered customer to VIP

```
def customer_to_VIP(user_id):
        user_id.setR_customer(VIP)
        update_customerPosition (user_id)
```

11. Demote VIP to registered customer

```
def  VIP_to_customer(user_id):
        user_id.setVIP(R_customer)
        update_customerPosition (user_id)
```

12. De-register registered customer

```
def deregister_customer(user_id):
        if(manager deregisters customer account ):
                user_id.deregistered = user_id.delete
                update.account_db
```

13. Registered customer condition for VIP

```
def r_customer_promote (user_id, order_data, money_spent):
```

```
        if (R_customer.order_data = 50):
                customer_to_VIP(user_id)

        else if (R_customer.money_spent > 500_dollars):
                customer_to_VIP(user_id)
```

14. Customer Ordering Food

```
def customer_order (user_id):
‘‘‘ displays menu’’’
        if(customer account  = True):
        update.account_db

        display options for ordering food:
                if(customer wants to eat in):
                        user_id.eatIn = user_id.fix_time_seat
                elif (customer wants to pick up food):
                        user.id_pickUp = user_id.getUpdates
                elif(customer wants to get food delivered):
                        food_delivery = user_id.getAddress
```

15. Customer Complaint

```
def customer_complaint (user_id):
        customer.complaint = manager.informed // manager is informed when complained
        if (customer complained on employee first time)
                update.complaint_db
                if(complaint on chef):
                        chef_demote()  // call this function to demote chef
                if(complaint on deliveryPerson):
                        deliveryPerson_demote  // call this function to demote deliver
person

        if  (customer complained on employee two times):
                update.complaint_db
                if (complaint on chef):
                        chef_demoted_twice()  // call this function to fire chef
                if (complaint on deliveryPerson two times):
                        deliveryPerson_demoted_twice()  //call this function to fire delivery
        person
```

Surfers

16. Surfers View Menu / Rating

```
def surfer_view_menu/rating():
 ‘‘‘ displays menu’’’
        if (chef uploads menu) :
                viewMenu = show_menu.surfers
                if(customers rate food):
                        viewRating = show_rating.surfers
```

17. Surfer Apply to Customer

```
def surfer_login():
        new_customer = create.user_id
        money_deposit = deposit_money.user_id
        manger_check = check.user_id
        update.customer_db
        for i in customer_kickOut:
                if(new_customer == customer_db): // if new customer matches previous account
                        print("Can not login")
```

Here is a table with the method and their input, output, and main functionalities. Below the table, there are corresponding pseudocodes for each method.

Method	Input/Output and Main Functionalities
1. tabooCheck	Takes in user_id as a parameter and will run on taboo words. It will work on every user's message. It compares words stored in the taboo array, and if a match is found, the taboo word will be converted to *** and a warning is issued. If there are more than three taboo words will cause the message to be blocked.
2. customer_outOf_System	Accepts user_id as a parameter. When the userID is passed, it will be deleted from the system, and the deposit associated with the userID will be cleared. The manager will handle all this.
3. manager_recieveComplaints	Accepts user_id as a parameter. With this function, the manager will receive a complaint

	with the user's name and take further action. Manager can then dismiss the complaint, or they can give a warning
4. chef_updates_menu	This method will take the chef's input. The chef will be able to update the menu by adding descriptions of foods. The food_db will be updated so that it can be used later to show customers and surfers.
5. chef_demote	Takes in user_id as a parameter. It has three cases to demote a chef. If it passes one of them it will demote the chef and will decrease the salary. This will be updated in the complaint_db
6. chef_demoted_twice	Takes in user_id as a parameter. If the chef is demoted twice, then user_id will get fired. This would call the chef_demote method and see check if the chef had been demoted previously.
7. deliveryPerson_complaint	This takes in user_id as a parameter. In this method, deliveryPerson can make complaints on customers. VIP customers and registered customers are handled in different manners.
8. deliveryPerson_demote	Takes in user_id as a parameter. It has three cases to demote a deliveryPerson. If it passes one of them it will demote deliveryPerson and will decrease the salary. This will be updated in the complaint_db
9. deliveryPerson_demoted_twice	Takes in the user_id as a parameter. If deliveryPerson is demoted twice, then user_id

	will get fired. This would call the deliveryPerson_demote method and see check if the deliveryPerson had been demoted previously.
10. customer_to_VIP	This takes in user_id as a parameter. This method's purpose is to update the position of registered customers to VIP. It takes the registered customers' names and updates it to VIP customers.
11. VIP_to_customer	This takes in user_id as a parameter. This method's purpose is to demote the position of VIP customers to registered customers. It takes the VIP customer's name and demotes it to registered customers.
12. deregister_customer	This method takes user_id as an argument. If the manager deregisters a customer, it will take the user_id, and it will delete it. This will also be updated in the customer_db.
13. r_customer_promote	This method takes in the user_id, order_data, money_spent as a parameter. If a registered customer meets either of the conditions, spending five-hundred dollars or having fifty orders, it will be promoted to VIP. This method keeps track of how many orders customers make and how much money they spent. In this method, the function customer_to_VIP(user_id) is called. This method is called so that registered

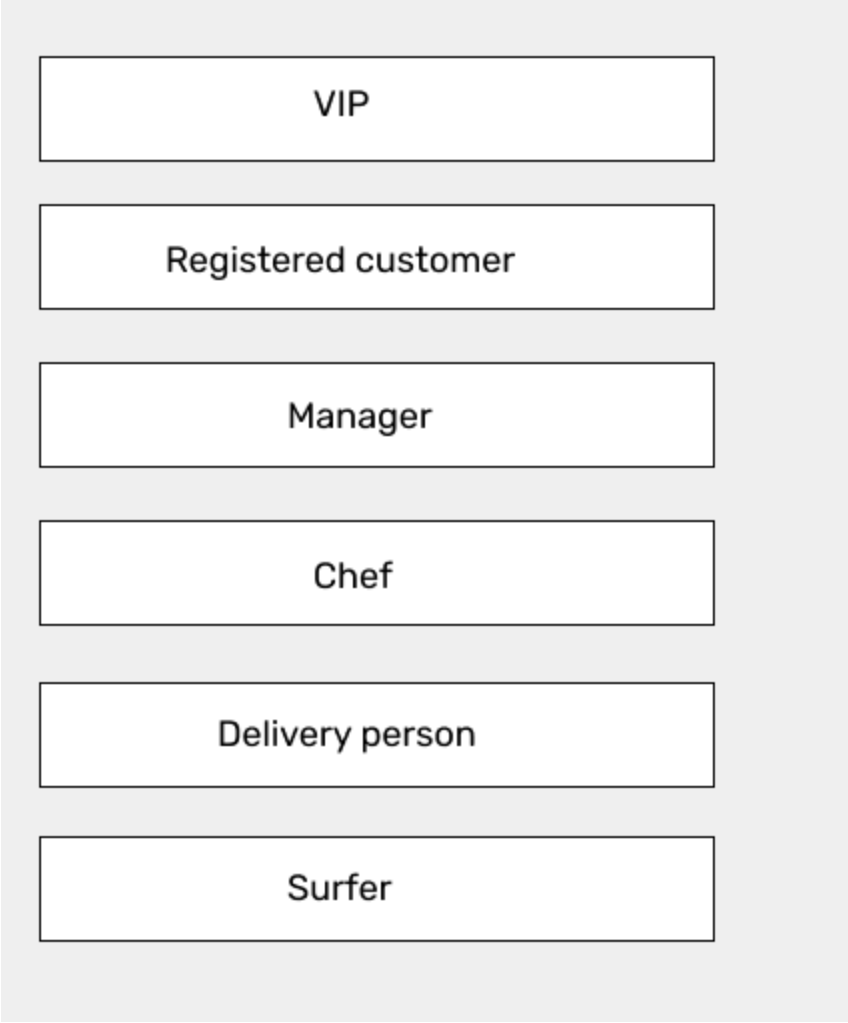
	customers can be promoted to VIP.
14. customer_order	This method takes in user_id as an argument. When customers make an order, it will check if their account is valid such that they have enough money. Then they will be given three options of how they want their order, either by delivery, pick up, or eat in the restaurant. If they chose to eat in the restaurant, they would need to fix their seat and time. If they decided to pick up the order, they would need to go to the restaurant and pick up from there. However, if they decided on delivery, they would need to enter their address.
15. customer_complaint	This method takes in user_id as an argument. The manager is informed. Customers can complain to the chef and delivery man. This method has two conditions. If the complaint was made for the first time on either the chef or delivery man, it will call chef_demote() or deliveryPerson_demote. The second condition is that if the complaint was made for the second time, it will call chef_demoted_twice() or deliveryPerson_demoted_twice()
16. surfer_view_menu/rating	Displays a GUI of the menu uploaded by the chef with ratings for the surfers.
17. surfer_login	This method will take the user's input from the login. This will then compare the user_id with other user_ids in the customer_db to see if this

	user was previously kicked out of the system, and if it was kicked out, it will print a message.

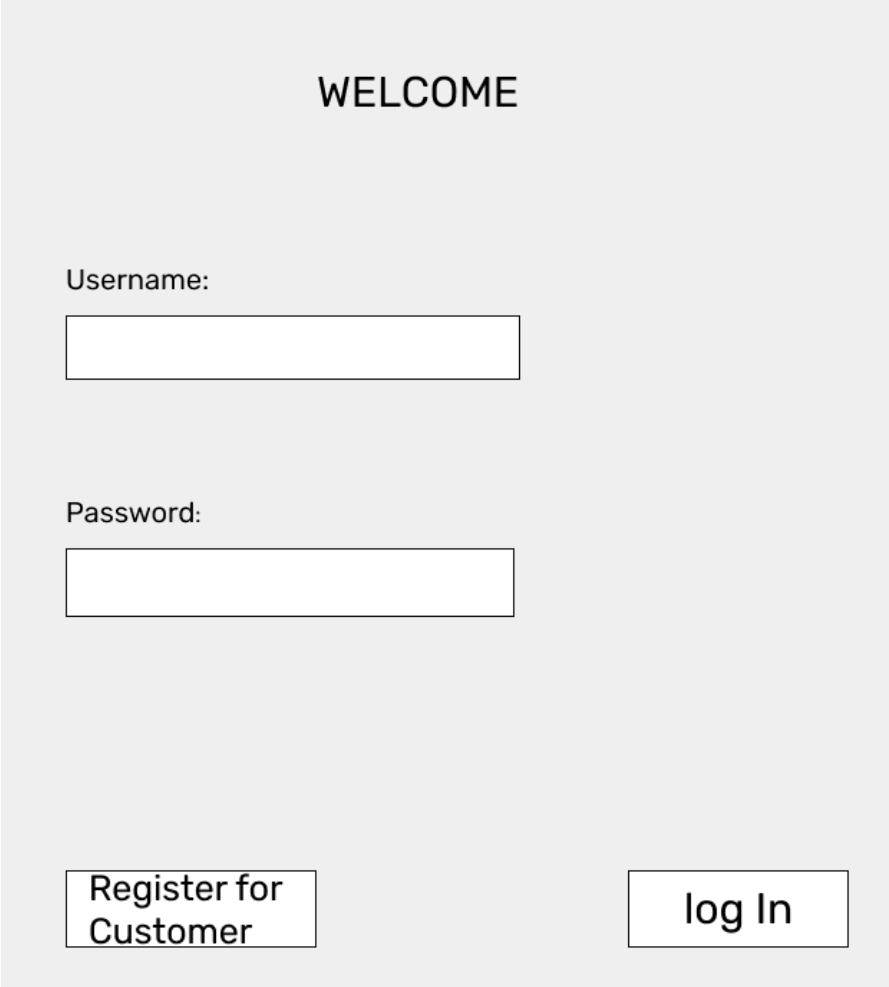
5. System Screen

The following are some major GUI screens in our system.

1. The first screen lists the type of users: VIP, registered customer, manager, chef, delivery person, and surfer. The user selects the one they are associated with.



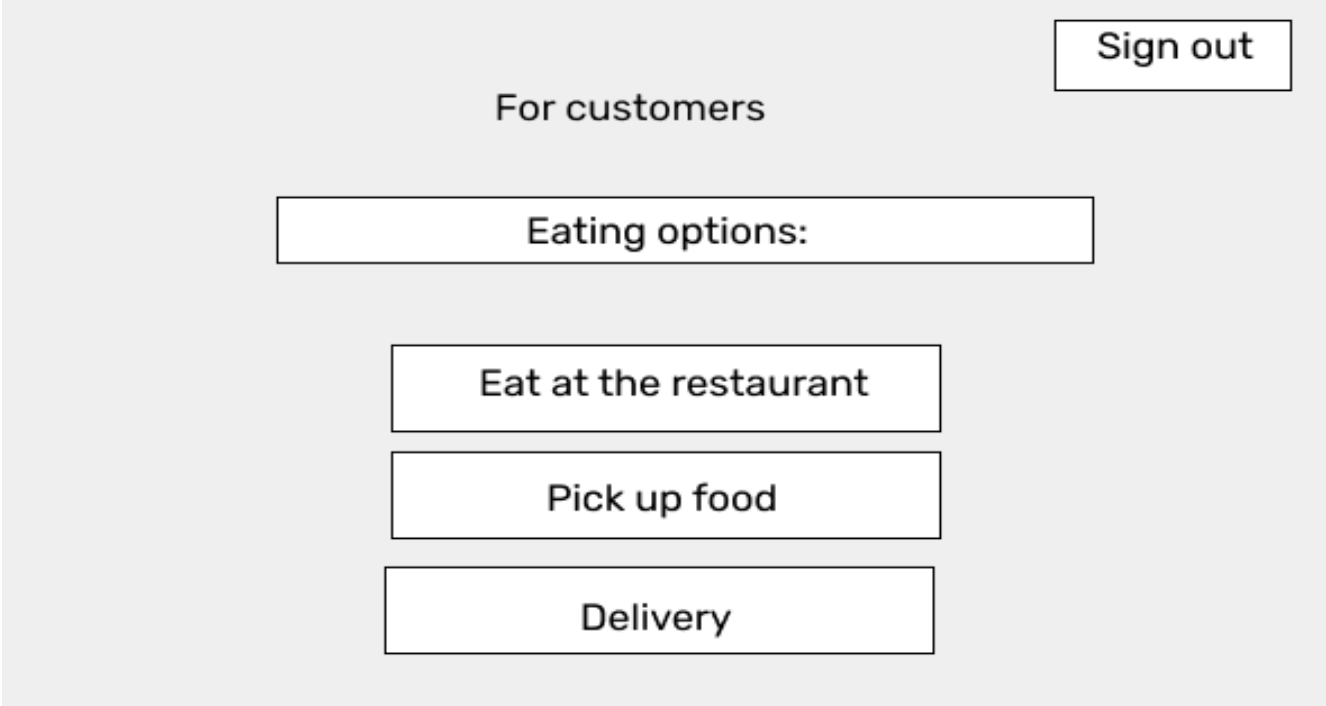
2. The welcome screen displays the following: username, password, register as customer, and log in. The users are allowed to enter their username/password and then click on login. Register for customer option is for surfers/new customers.



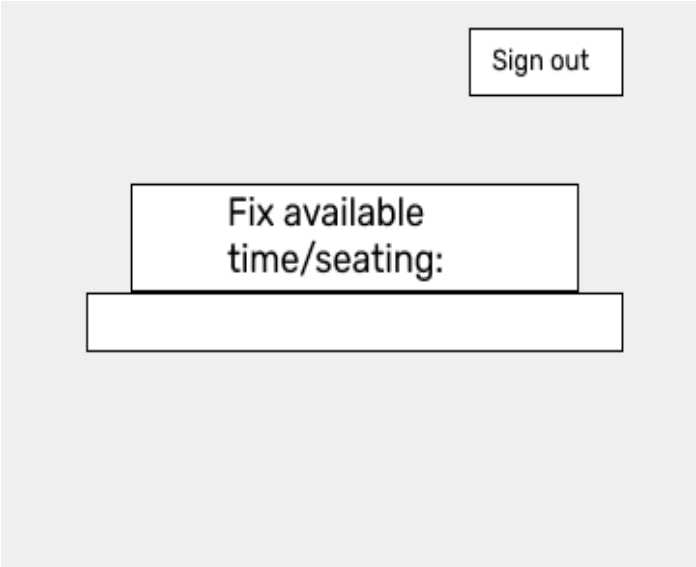
3. This screen is for registered and VIP customers. Based on their history of prior choices, 3 of the listing dishes will appear. For now, we wrote ‘insert image’ since we have to decide what to do here.

4. The following screen is for customers selecting three of the eating options. They can either select “eat at the restaurant,” “pick up food,” or “delivery.” If the customer selects “eat at the restaurant,” they will be taken to the “fix available time/seat” screen. If the customer selects the “pick up food” option, then they are taken to the screen that displays a message saying, “See you soon.” If the customer selects the “delivery” option, they are

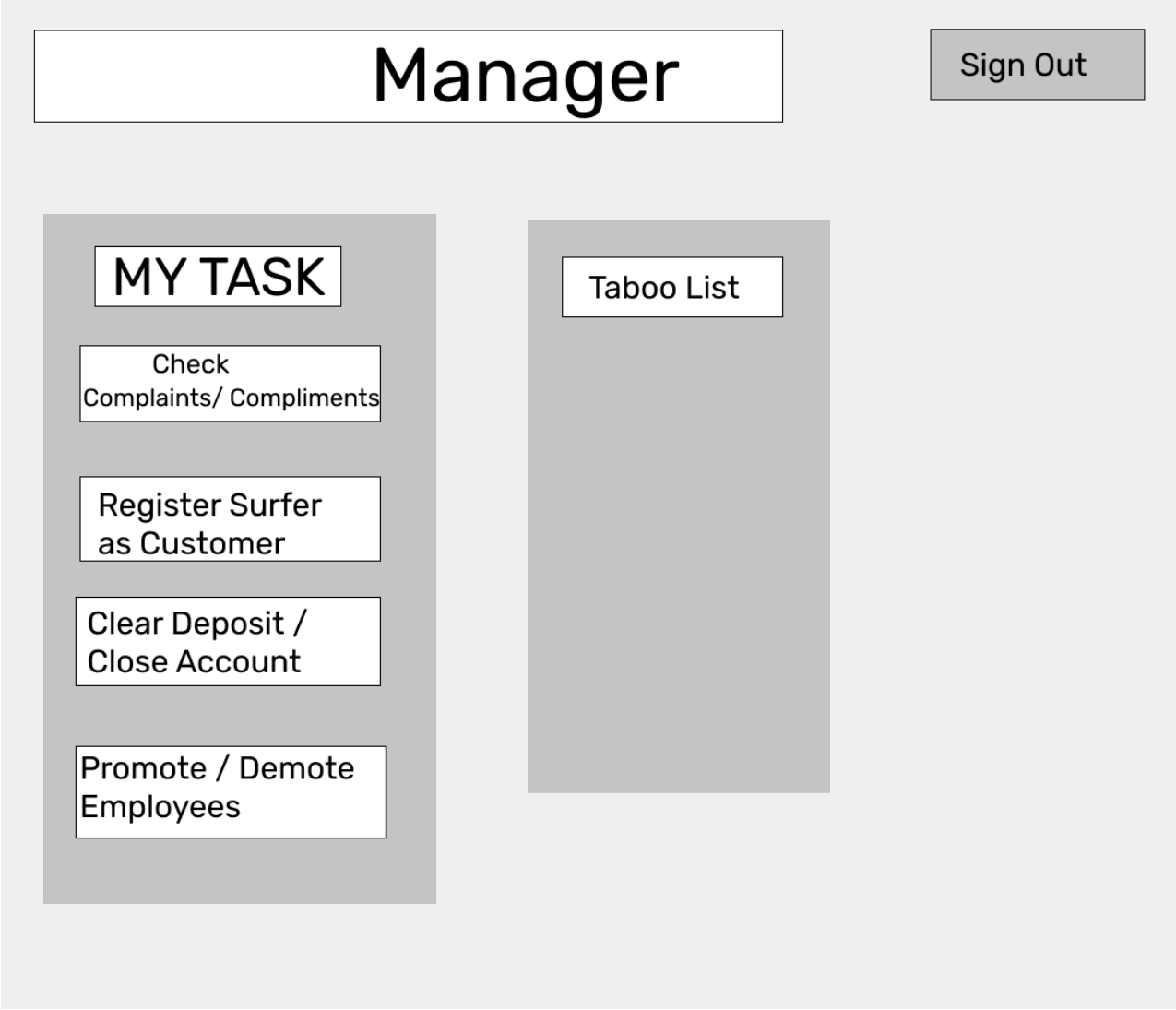
taken to the “enter address” screen.



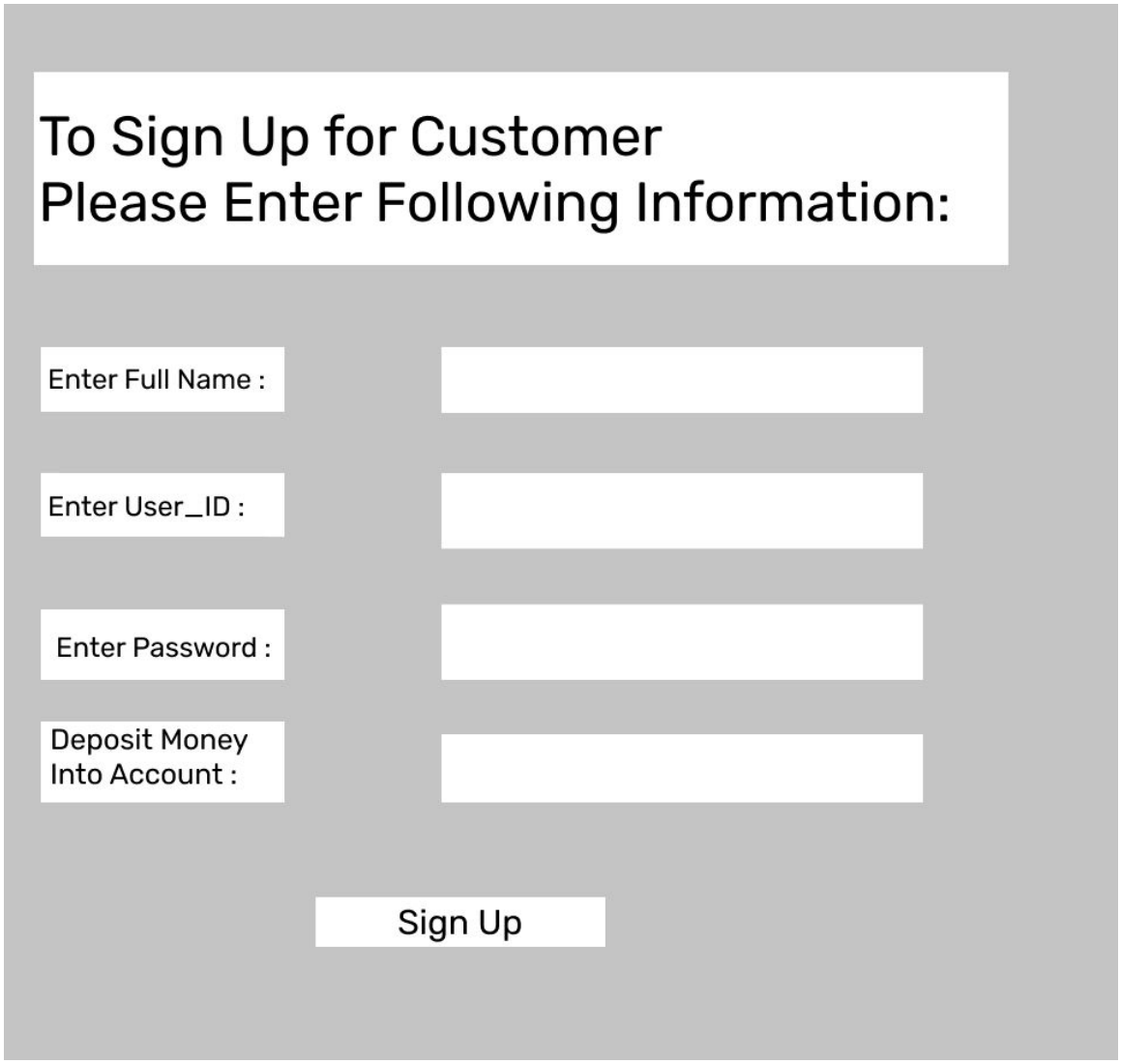

Sign out
See you soon


Sign out
Fix available time/seating:

5. This is the screen for the manager. The Manager has responsibilities that are listed under My Task. It also has another tab called Taboo List, which the manager keeps.



6. After surfers / new customers click the Register for Customer option on the welcome screen, it will take them to this image. New customers will be required to fill in the following information to register.



7. Surfers have the option to see the top three most popular dishes and the top three highest-rated dishes. For this GUI, we simply just said insert image, as we are yet to decide what to do for this part.


Top 3 most popular dishes
1. Popular dish image insert
2. Popular dish image insert
3. Popular dish image insert
Top 3 highest rated dishes
1. Highest rated dish image
2. Highest rated dish image
3. Highest rated dish image

6. Group Meetings

Date:

10/08/2020 -Brief discussion on what each group member can do (15 mins)

-Agreed on using Python programming language for the project (5 mins)

10/8/2020 -Messaged back and forth every day in the group chat about what needs to be

11/16/2020 done for the project

-These chats included details of specific parts that needed to be done for the project

11/14/2020 -Two-hour video chat on what diagrams need to be fixed/changed

-Worked on the normal and exceptional scenarios (25 mins)

11/15/2020 -Two-hour video chat about the diagrams and pseudo-codes. Also, we viewed Github, which contains all project-related files in our repository.

-Three hour Google docs meetup on fixing the diagrams

-Group chat texting about the coding (1 hour)

11/16/2020 -Four-hour talk in the group chat/Google docs about finalizing the diagram, pseudocode, and GUI images

11/17/2020 -Three-hour final review of the second phase report

Our group has been working together and helping each other out for this project. Everyone has been doing their part for the project.

7. GitHub Repository Link

https://github.com/saifulislamny/RestaurantSystem