

---

**Granny's Sweet**

**Granny's  
Sweets  
Software Requirements Specification  
For Web Application  
Version <1.0.1>**

Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

## Revision History

Date	Version	Description	Author
16/OCT/20	1.0	Phase I – Software Requirements Specification	Abir Deb, Samuel Fils, Andrey Goryuk, Michal Moryosef, Syed Sadman
13/NOV/20	1.0.1	Phase II – Design report & updated UseCase Diagram from Phase I	Abir Deb, Samuel Fils, Andrey Goryuk, Michal Moryosef, Syed Sadman

Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

## Table of Contents

<b>1. Introduction</b>	<b>5</b>
1.1 Purpose	5
1.2 Scope	5
1.3 Definitions, Acronyms, and Abbreviations	6
1.4 References	6
1.5 Overview	6
<b>2. Use Cases</b>	<b>7</b>
2.1 Use-Case Model Survey	9
2.2 Use-Case Reports	11
2.2.1 [Unregistered (Surfer) Customer Use Case]	11
2.2.2 [Registered Customer Use Case]	11
2.2.3 [VIP Customer Use Case]	13
2.2.4 [Banned Customer Use Case]	16
2.2.5 [Chef Use Case]	17
2.2.6 [Manager Use Case]	18
<b>3. E-R Diagram</b>	<b>19</b>
<b>4. Detailed design</b>	<b>21</b>
<b>5. System screens</b>	<b>26</b>
<b>6. Group Meetings</b>	<b>29</b>
<b>7. Github Repositories</b>	<b>30</b>

Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

# Design report & updated UseCase Diagram from Phase I

## 1. Introduction

### 1.1 Purpose

This software requirement specification will provide a detailed description of the Online Restaurant System (ORS). In this document one can find the Use Case Models, System Requirements and specifications, detailed description of use cases, The Petri Net Diagram for each use cases, ER Diagram for all entities and their relationship inside the database, the detailed design of our API, diagram for API endpoints and pictures of the System screens for the frontend of our website.

### 1.2 Scope

The Food Ordering System (FOS) will be a web based application where users can order a variety of food items.

This ORS, includes two use-case diagrams: the first use-case diagram describes the customer functionalities (order, deposit money etc.) The second use-case describes the employees class: chef and manager. Each employee has its own separate specifications according to the task description provided (see *References*). In this document includes the software requirements such as frameworks which will be used, as well as relevant dependencies used.

Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

## 1.3 Definitions, Acronyms, and Abbreviations

Terms	Definitions
ORS	Online Restaurant System
FOS	Food Ordering System
ReactJS	React is a front end JavaScript library for building user interfaces maintained by Facebook.
Java	Java is a class-based, object-oriented programming language.
Spring Boot	Spring Boot is an open source Java-based framework used to create a micro service.
SQL	Database querying language/management system
NodeJS	Javascript runtime environment that runs outside the web browser

## 1.4 References

*Software Requirements Specification*, [www-cs.ccny.cuny.edu/~csjie/322/spec\\_sample.pdf](http://www-cs.ccny.cuny.edu/~csjie/322/spec_sample.pdf).

## 1.5 Overview

The “Use Case” section of this document illustrates the ORS features of the food ordering system application and its accessibility for different types of users. The “Use Case Model Service” goes into detail about each type of user permission and privileges. It contains a complete overview of the users scope throughout the application. The ORS diagram is also included as a top down view of the system as a whole through use-case diagrams which link different types of users to features in the ORS.

Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

The “Use Case Reports” section of the document illustrates an overview of the system and its relation to users. It includes the Class Diagrams and Petrinets for each of the use cases. There are normal and exceptional scenarios containing a brief overview of the cases.

The “ER Diagram” section contains the attributes and keys for each class encompassing the entire system.

The “Detailed Design” section contains critical methods from services and controllers in the application. For simplicity and readability, pseudo code is displayed instead of the code used in the application.

The “API Endpoint Chart” displays the mappings for the controller and methods used to make REST API requests. These endpoints are used to persist data to the SQL database using Spring Data and Hibernate. The JSON form will be displayed in the frontend.

Finally, the “Screenshots” section displays some progress with the frontend. All events are temporarily hardcoded. At the conclusion of the use cases, the data from the database will be used to populate the fields.

## 2. Use Cases

The ORS features an online food ordering service that is accessible by different types of users. Designing the Restaurant app where each chef of the restaurant has their own menu of dishes that all surfers (non-registered User) and registered users can browse, Registered User can also order food that will use USD from the balance in their account. The Chef then receives orders for his Dish and once the order is made a Delivery agent picks the order and delivers it based on the customer information provided during registration. Each user can rate both delivery man and dishes. It features a GUI web based application that provides participants with an interface to create and manage order while incorporating a delivery system that drives competition between food delivery agents. The delivery agents can pick the orders they want to deliver from the queue, that is set up by the recency of the order. It also features a dashboard for chefs to be able to present their menu

Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

items, providing various food options for customers, setting the prices. These menu items can also be rated from 1 to 5 stars. Managers have various jobs such as approving every account manually including delivery, chef and customer accounts, manually reviewing banned user's unban requests. This app also features a warning system for chef and delivery man with too many warnings or bad ratings getting fired and a customer with too many warnings getting deregistered or demoted from their VIP status. This can be done manually by managers who can also choose to fire each of his employees or give them a warning based on reviews by the customers. Managers can also create a list of taboo words that if a customer uses, will receive warnings by the system. Customers with 50 orders or 500 dollars spent also can acquire the VIP status that leads to their rating counting as twice the normal rating and having access to 10% discount on the menu items.

Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

## 2.1 Use-Case Model Survey

In our use case diagram, we have 3 types of ordinary users and 3 types of privileged users. The ordinary users are the Surfers (users with no account), Customers (users with account), and VIP Customers (customers that made over 50 orders or > 500\$).

**Surfers:** Surfers are Users with no accounts and can only view menu items and dishes and register for an account to become an user

**Customer:** A Customer is a registered user who has an account balance to order menu items. Customers can browse and view Menu items just like surfers but also order the items with their account balance. Once the order is delivered they have to confirm that the order is received. They can rate both the delivery guy, chef and the Dish from a rating of 1 to 5 stars. Customers can also interact and write comments with their review and if they write words that are detected as taboo words they receive warnings. 3 warnings deregister a customer.

**VIP:** VIP status is given to registered users (customers) who have ordered 50 times or ordered items worth 500 dollars or more. VIPs can browse menu, order menu items, rate dishes, delivery agent, chefs and leave reviews just as a customer can but their rating counts as two times the rating of a regular customer. They also receive a 10% discount on every order made. If a VIP uses profanity (Taboo words) they also receive warnings and VIPs with 3 warnings get demoted to regular customers and lose all their perks.

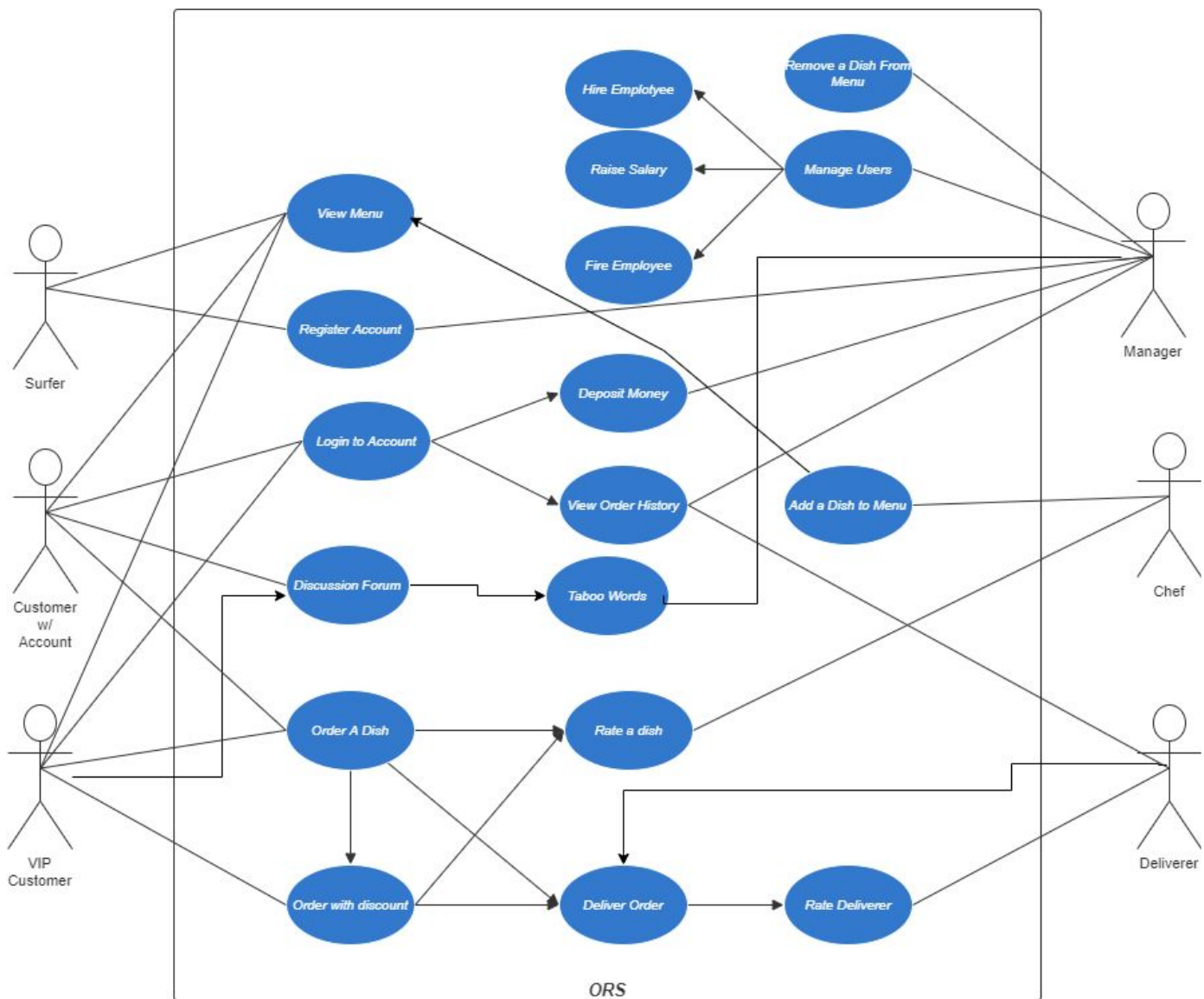
**Manager:** The restaurant has one manager who is an employee and can control, modify and has access to many of the use cases of the restaurant system. Managers can remove any dish from the menu and make changes to a dish without the chef's permission. They have access to the entire order database and see different user's orders. They can manage users and other employees giving them warnings and promotions as necessary, including hiring and firing employees, giving them a salary raise or ban users based on warnings.

**Chef:** chefs are also employees of the restaurant with the ability to create their own menu that only the specific chef and managers have access to. Each menu item belongs to a specific chef that only that chef and the manager can modify and delete. Chef's can also see the order that contains the dish belonging to them so they know dishes to cook if an order for their dish is made. Chefs can be rated by customers and are prone to being fired if too many bad ratings come in.



Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

**Deliverer:** Deliverers are also employees of the restaurant that can see orders that have been recently made and are not marked by the completed tag. If an uncompleted order appears a deliverer can choose to do the from the queue of uncompleted orders. Deliverers can also be rated by customers and can be fired by the manager if they receive too many negative ratings.



Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

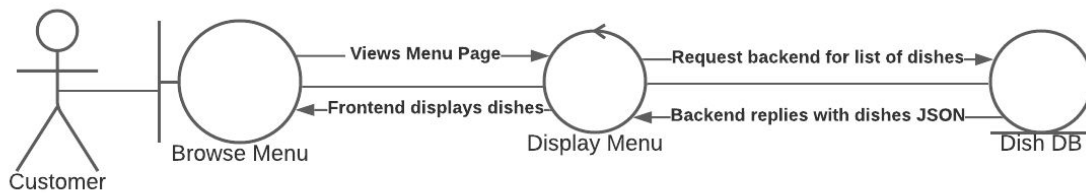
## 2.2 Use-Case Reports

### 2.2.1 [Unregistered (Surfer) Customer Use Case]

Use-Case: Browse menu items

Normal Use Case: Surfer will be able to access the menu page on the website and look through available dishes.

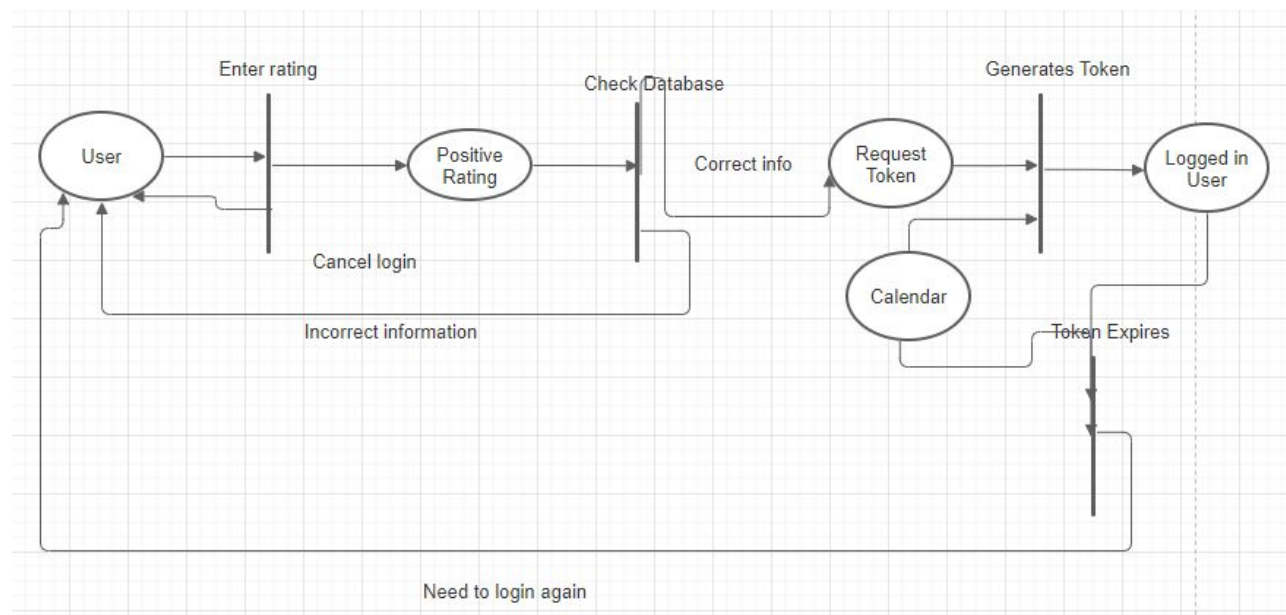
Exceptional Scenario: None



### 2.2.2 [Registered Customer Use Case]

Use-Case: Login

Description: Customer must provide a username and password for authentication.

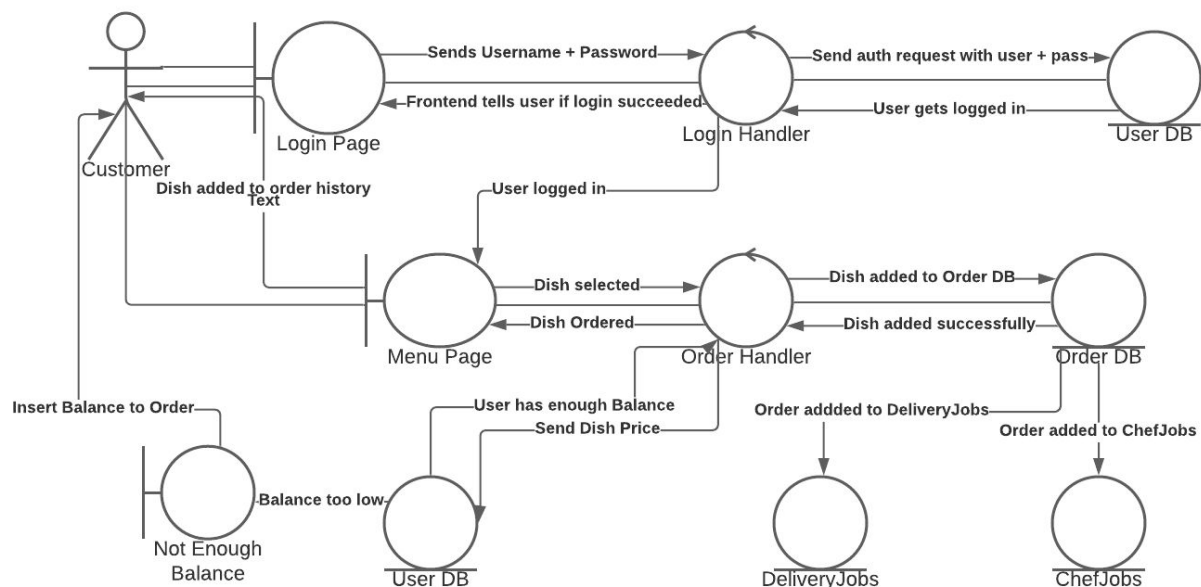


Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

## Use-Case: Order Dishes

Description: Customers can order dishes from the menu.

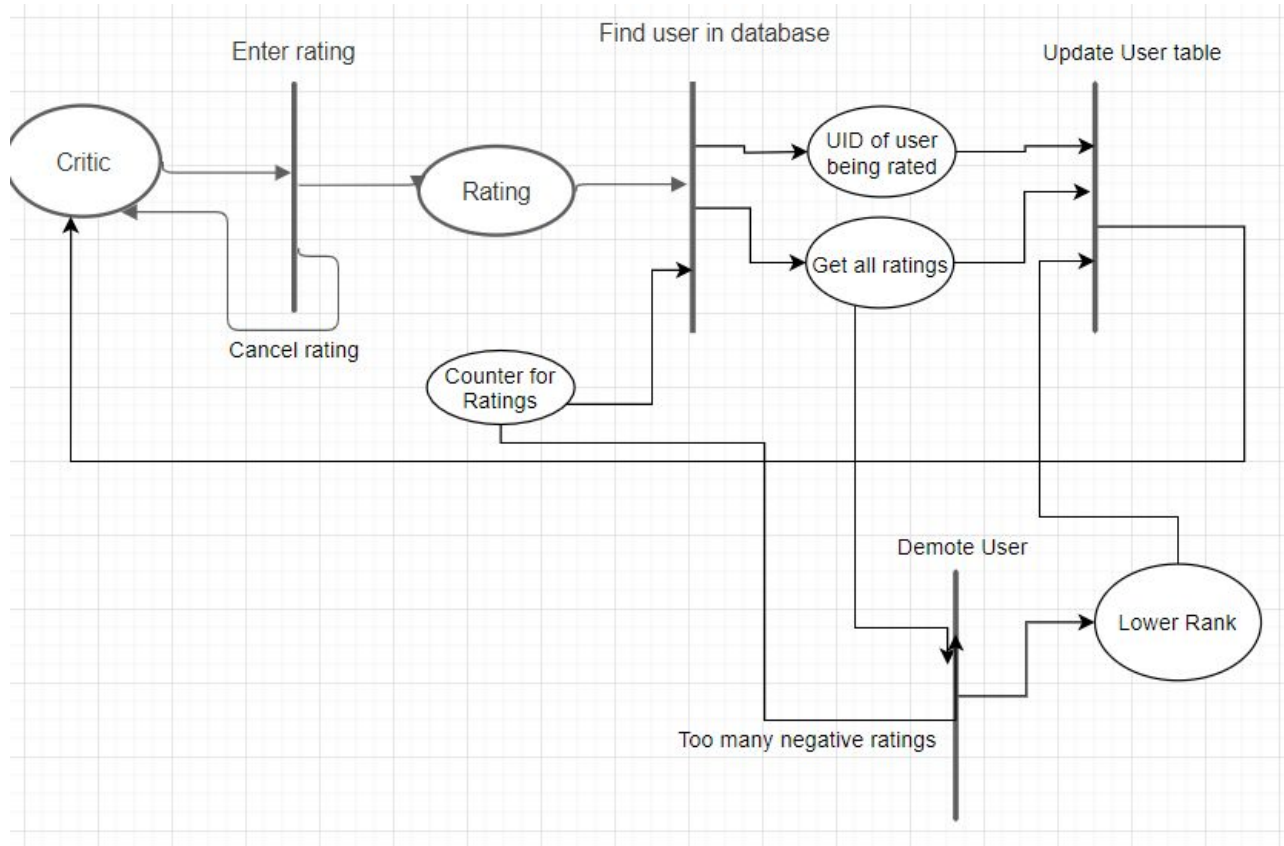
Normal Use Case: For a user to order a dish, they must first be signed in and be a registered customer. By signing in, customers are able to access the menu page and add a dish from the menu to their order. Before an order is submitted, it is compared with the user's balance. If the balance is enough the order is submitted otherwise, the user is sent a message to add balance. Once an order is submitted, it is added to the Chef Jobs database and Delivery Jobs database.



Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

### Use-Case: Rate Dishes

Description: Customers can rate dishes that they ordered before through their order history page.



### Use-Case: View Order History

Description: Customers will have a profile page where they can see a list of their previous orders.

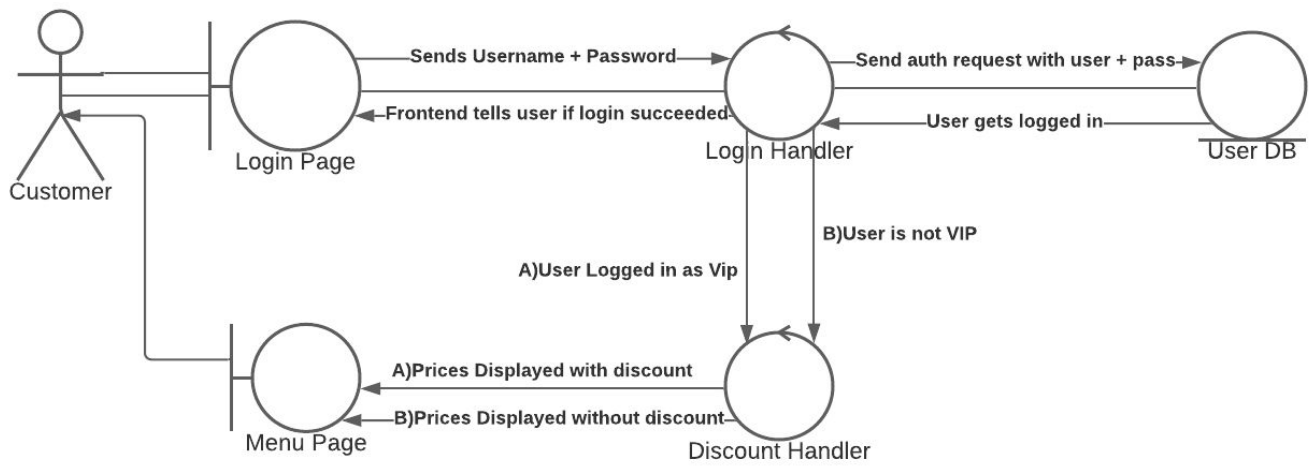
## 2.2.3 [VIP Customer Use Case]

### Use-Case: Use 10% Internal Discount

Description: VIP customers will be access to a 10% Discount on all orders.

Normal Use Case: In order to make an order, the user must be logged in. When logged in, the frontend is given access to the currently logged in user's data, one of which is boolean value for whether the user is a vip. If the user is a vip, the frontend will adjust the prices it gets from the DishDB on the menu page to reflect a discount.

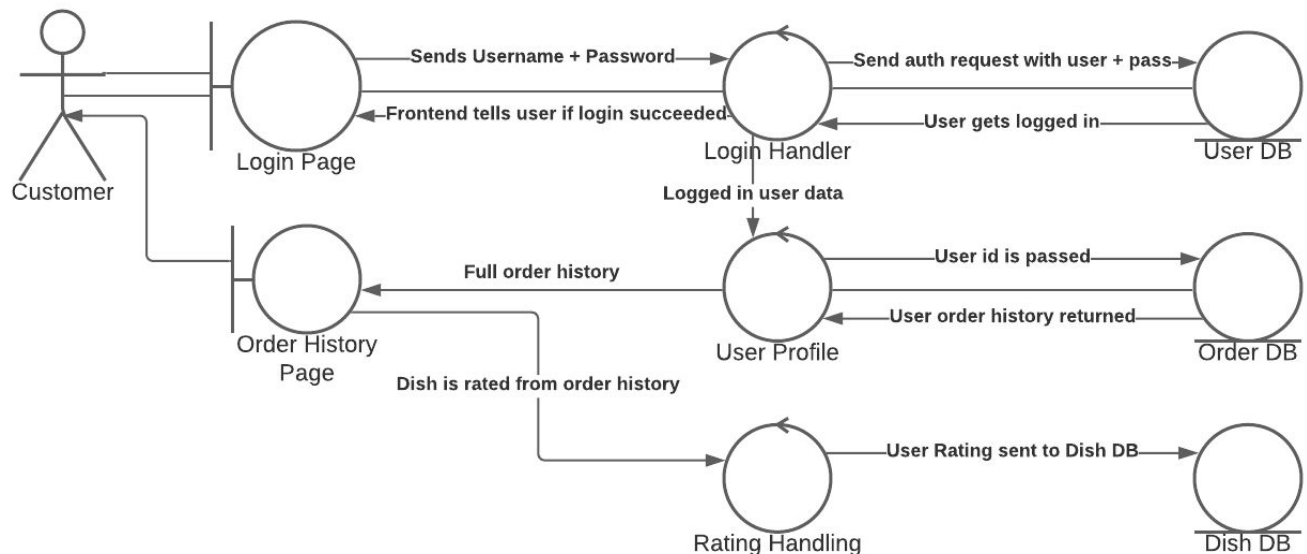
Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	



### Use-Case: Rate Dishes

Description: Customers can rate dishes that are associated with their orders.

Normal Use-Case: In order to rate a dish, a user must be a customer and has to be signed in. After signing in, a user has access to their order history. From there, users can mark their dish as delivered and must enter a rating for the dish.

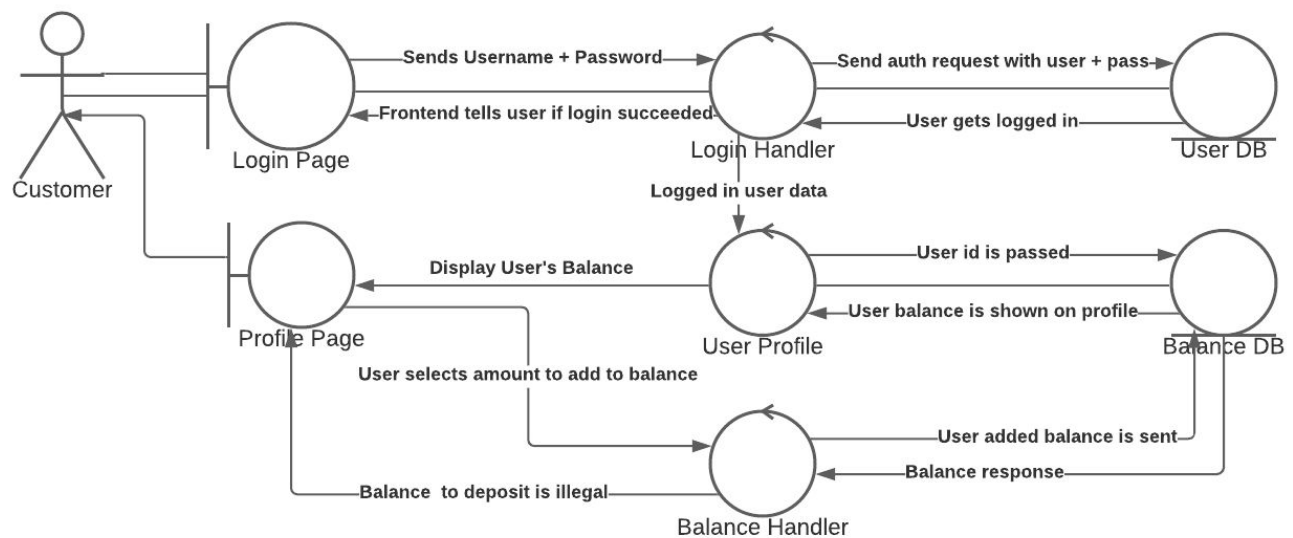


Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

### Use-Case: Deposit Money

Description: Customers can deposit money to account balance.

Normal Use Case: A user can add a balance to their account if and only if they are a customer and are able to sign in. After signing in, they have access to their profile page where they can view and add value to their balance. The balance to add is sent to the balance handler which checks if the balance added value is a legal amount (not negative and not over 500). If the amount is valid, the balance is added to the balance database.

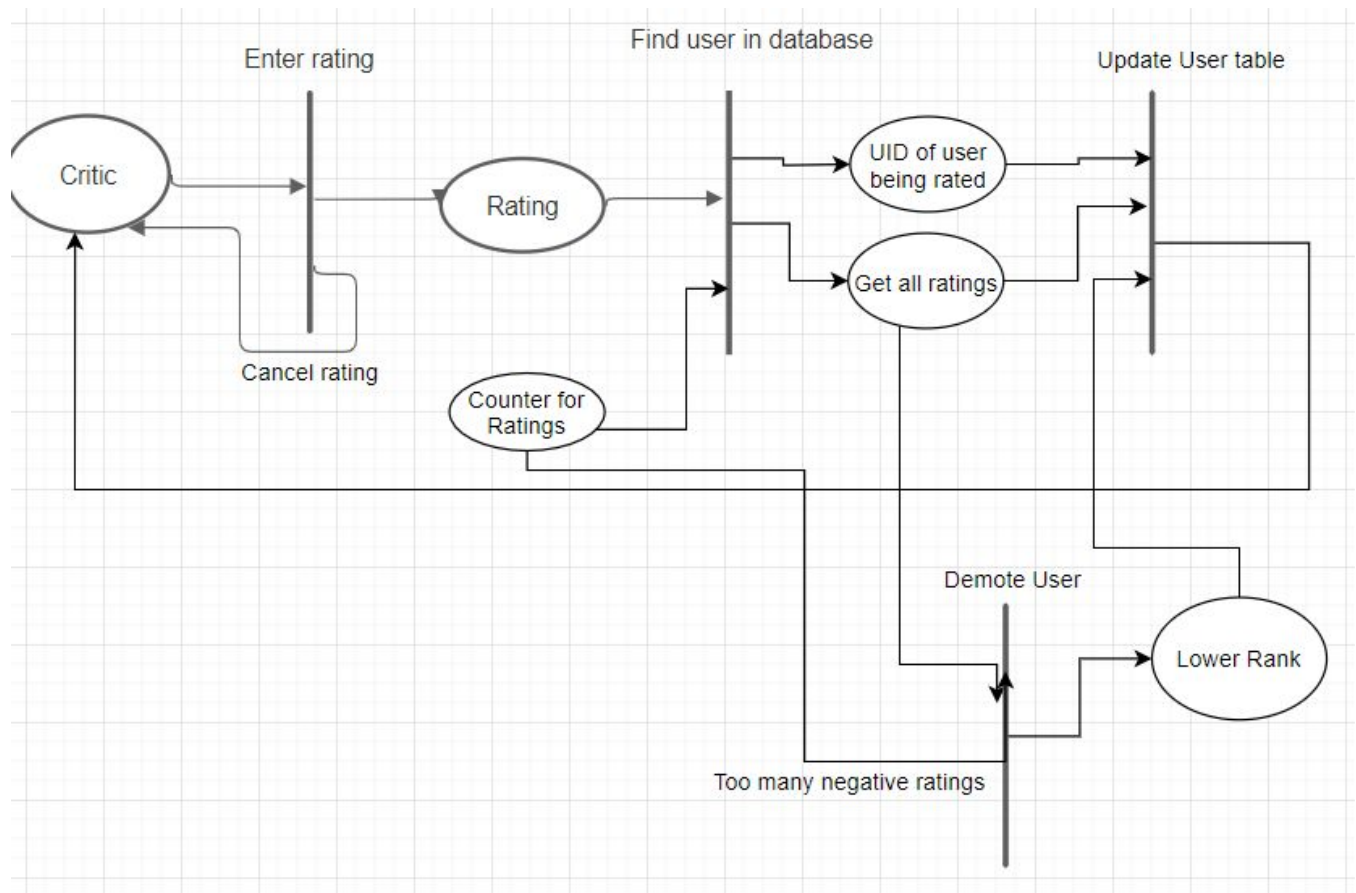


Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

## 2.2.4 [Banned Customer Use Case]

### Use-Case: Request Account Review

Description: Banned customers can contact the manager for account review. They are restricted from accessing the above use cases.

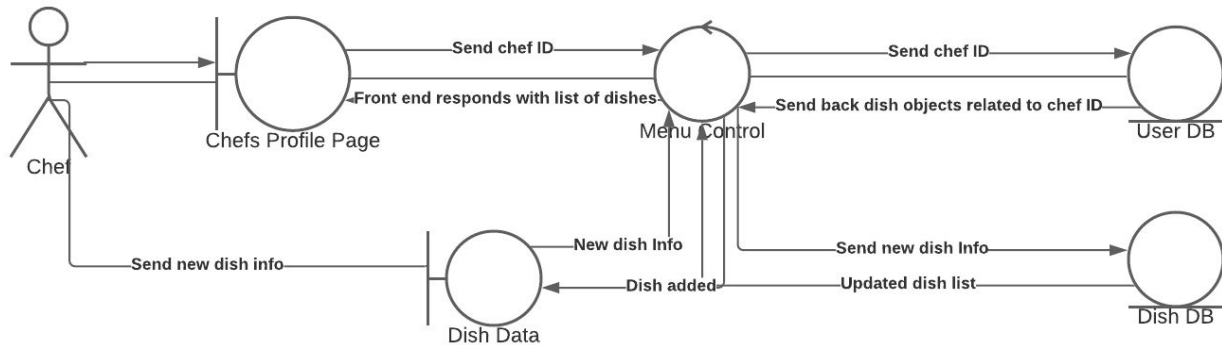


Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

## 2.2.5 [Chef Use Case]

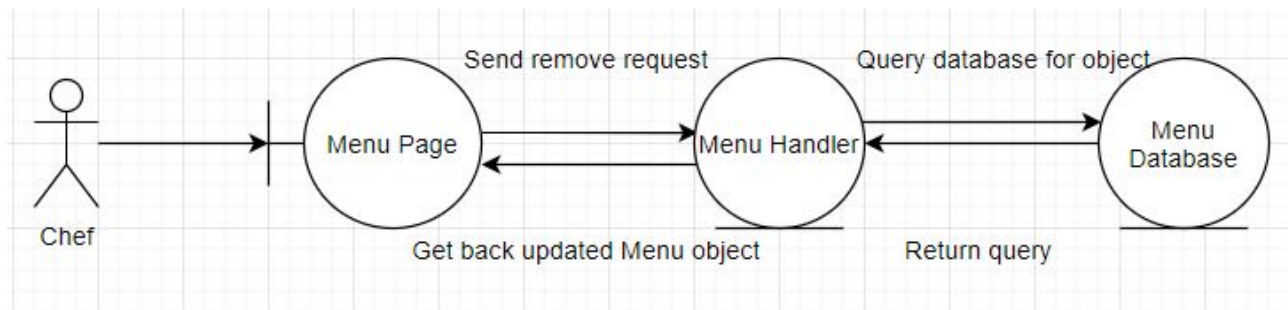
### Use-case: Add Menu Item

Description: Chef is able to add an item to the menu. Chef must include a name, brief description, and price. *\*Optional: Chef can include a picture of the item*



### Use-case: Remove Menu Item

Description: Chef can remove items from the menu.



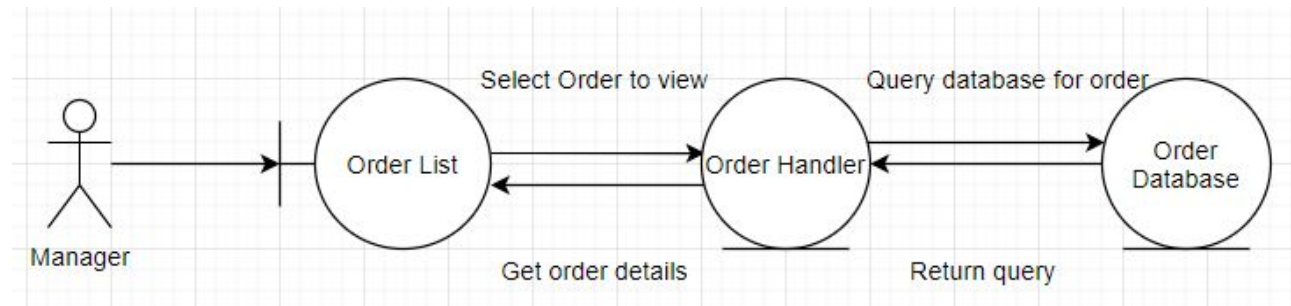


Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

## 2.2.6 [Manager Use Case]

### Use-case: Access Customer Orders

Description: Manager has a tab to see a list of all orders placed



### Use-case: Managing User accounts

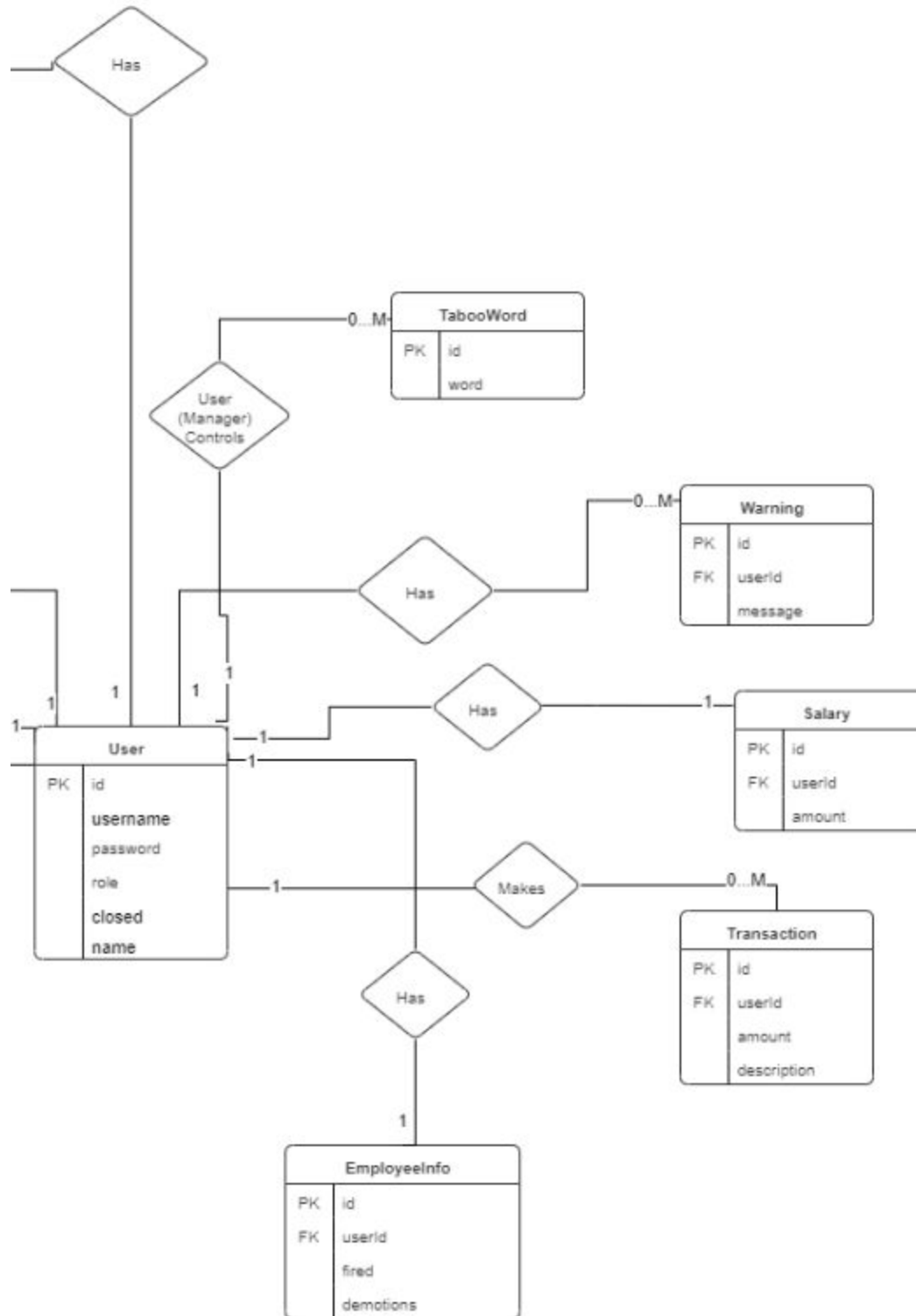
Description: Managers can remove banned users from the system or change them back to registered customers

### Use-case: Managing Employee Accounts

Description: Managers can remove/fire employee accounts (Chef or Deliverer)



Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	



Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

## 4. Detailed design

### Login Service

```

Login(jsonwebtoken){
    isValid = valid(jsonwebtoken );
    isAUser = hasUserInDB(jsonwebtoken.username)
    if(isValid && isAUser) {
        Return User
    }
}

```

### Censor Service

```

Censor(text, user) {
    Count = 0
    censorString = text.split(" ").map(
        word -> if word in censorList {count ++}
        Return "***"
    )
    user.setWarning("you said a bad word")
    if(count >= 3) return ResponseError(too many bad words)
    Else {return censorString}
}

```

```

Discount(user, amount){
    if(user.role == VIP) Amount = amount - amount * .1
    Return amount
}

```

### Dish Service

```

HTTP-GET
GetAllDishesForAChef(userId) {

```

Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

```

    return List<Dishes> in userId
}

```

```

GetDishByKeyWord(String Keyword) {
    Return List<Dish> associated with KeyWord
}

```

```

GetDishByRating(Long Rating){
    Return List<Dish> By average customer rating
}

```

### **Order Service**

#### HTTP PUT

```

UpdateOrder(Order updatedObject, Long orderId) {
    Order = findInDatabase
    Order = updatedObject
    Save(order)
}

```

```

DeleteOrder(OrderId)
    orderRepository.deletebyID(OrderID)
    Update (Transaction)
    Refund USD
    Update (balance)

```

### **User Service**

#### HTTP POST

```

FireChef(chef){
    negativeRatings = findInDatabase
    If negativeRatings > 2
    demoteTheChef()
}

```

Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

}

HTTP POST

```
Punish(user){
    User_warning = countWarningFromDatabase
    If User_warning > 3 And user role is VIP
        UserRole = Regular Customer
    If If User_warning > 3 And user role is Customer
        Deregister User
}
```

HTTP POST

```
PromoteCustomer(customer){
    If customer spends > 500
        Customer.role = VIP
}
```

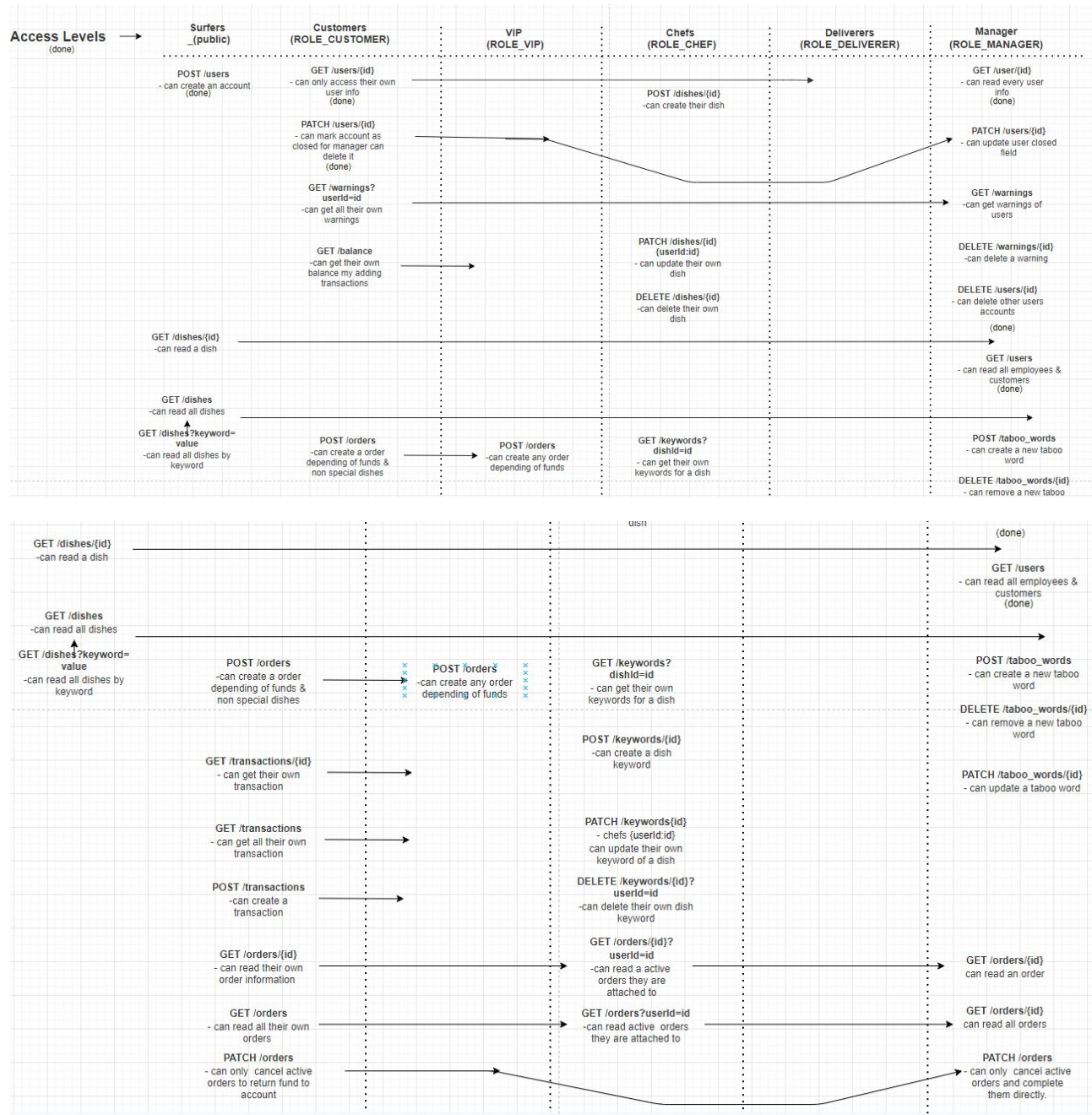
### **Keyword Service (Tags for dishes ex:Dessert)**

HTTP.GET

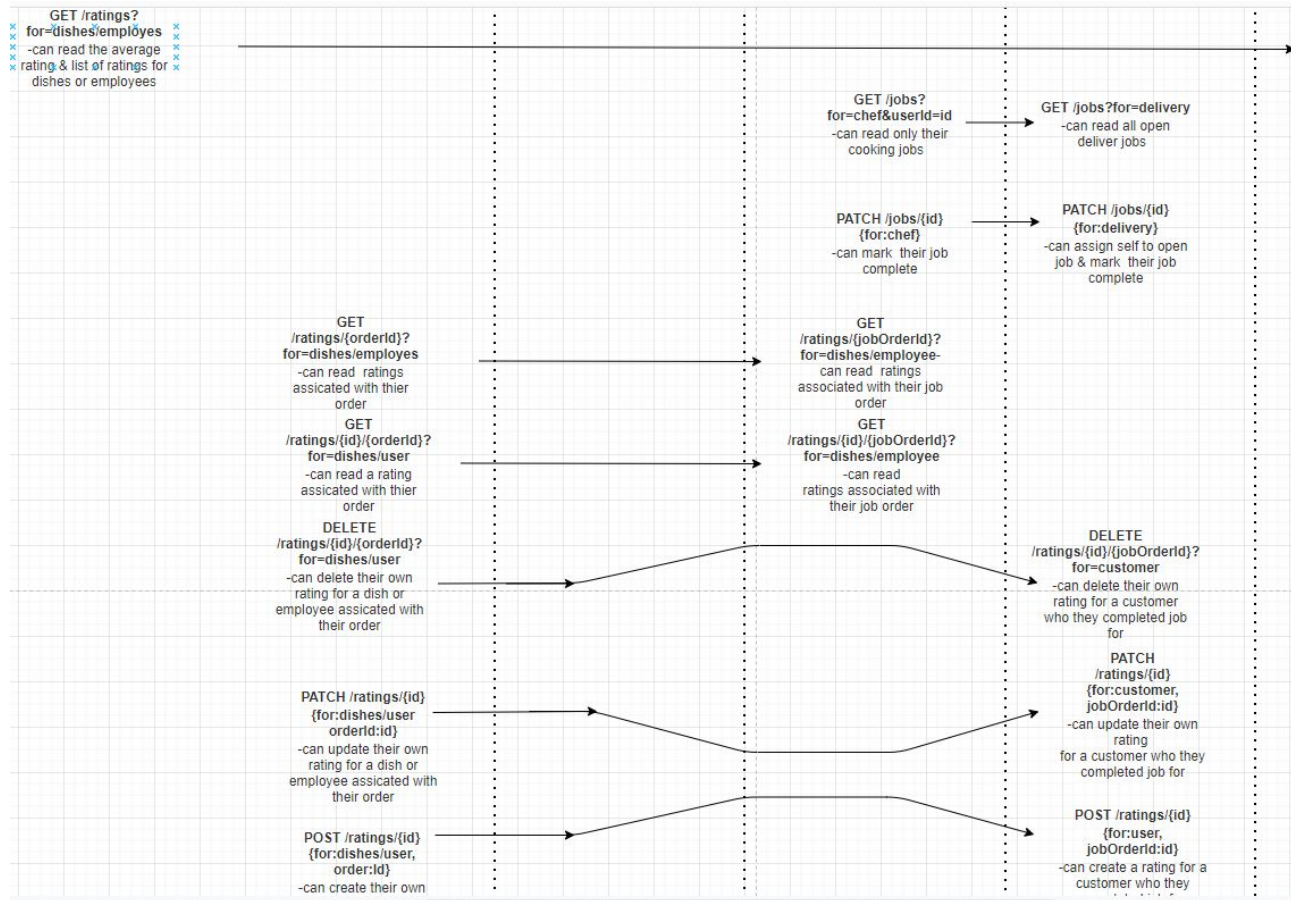
```
GetKeyWordsByDish(DishId) {
    Return List<keyWord> associated with DishId
}
```

## API Endpoint Chart with Security

[https://drive.google.com/file/d/1Dxz59CZGtVx\\_a6ltahc1Ny42QITRo5sW/view?usp=s](https://drive.google.com/file/d/1Dxz59CZGtVx_a6ltahc1Ny42QITRo5sW/view?usp=s)  
haring



Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	






Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

## 5. System screens

---

# Sign-In



Username

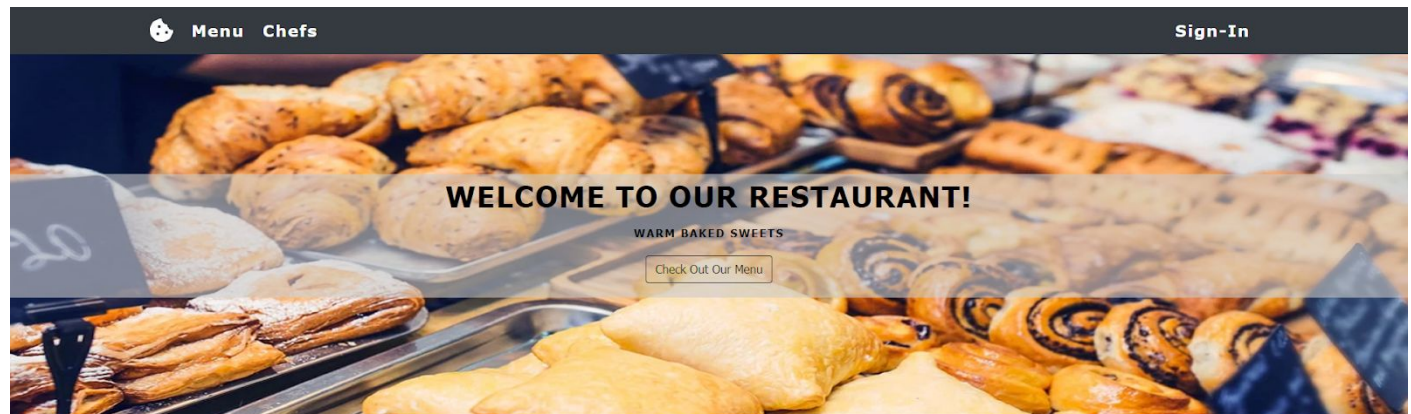
Password

Sign In

---

Create Account

Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	



## About Us



Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.



## Our Chefs



Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

## Menu






### Cup Cake

Some sort of description for a dish

\$5.99



### Cheese Cake

Some sort of description for a dish except this one is a bit longer and ...

\$8.99



### Birthday Cake

Some sort of description for a dish

\$2.99



### Cookie Cake

Some sort of description for a dish

\$15.99



### Some Other Type Of Cake





### Vegan Cake




### Boston Cream Donut







### Vanilla Cream Donut

Logo





Katarina Smith

 Customers
  Products
  Account

TOTAL CUSTOMERS  
**1,600**  
 16% Since last month

TOTAL PROFIT  
**\$23,200**

Latest Orders

Order Ref	Customer	Date ↓	Status
CDD1049	Ekaterina Tankova	11/04/2019	<span>pending</span>
CDD1048	Cao Yu	11/04/2019	<span>delivered</span>
CDD1047	Alexa Richardson	10/04/2019	<span>refunded</span>
CDD1046	Anje Keizer	08/04/2019	<span>pending</span>
CDD1045	Clarke Gillebert	07/04/2019	<span>delivered</span>
CDD1044	Adam Denisov	07/04/2019	<span>delivered</span>

VIEW ALL

Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

## 6. Group Meetings

<b>Meeting #</b>	<b>Date</b>	<b>Time</b>	<b>Topic</b>
1	09/25/2020	2 HRS	Introduction
2	10/02/2020	2 HRS	Project initialization
3	10/09/2020	2 HRS	Discussion
4	10/16/2020	3 HRS	Phase Report 1
5	10/23/2020	3 HRS	Planning
6	10/30/2020	3 HRS	Discussion
7	11/06/2020	4 HRS	Debugging and programming
8	11/13/2020	5 HRS	Phase report 2

Online Restaurant System	Version: 1.0.1
Software Requirements Specification	Date: 12/NOV/20
Phase II Report	

## 7. Github Repositories

- Frontend:  
<https://github.com/syedsadman16/Online-Restaurant-System-Frontend>
- Backend:  
<https://github.com/syedsadman16/Online-Restaurant-Backend>