Design Report
for
Kitchen Zealot

Written by:
Abtahi Chowdhury
Abusaleh Masud
Safwan Shahid
Arman Uddin
Farhan Zaman

Version 2.0

November 23, 2019

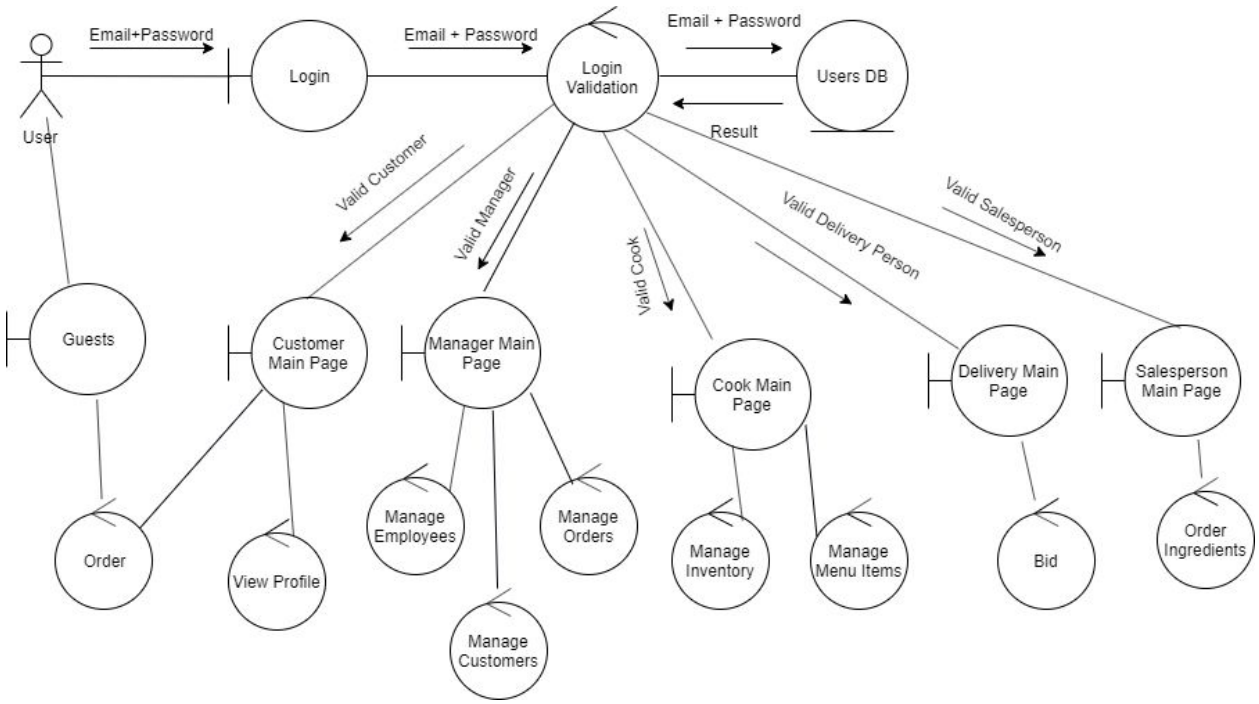
Date	Version	Description	Author
11/2/2019	1.0.0	Initial version of the online restaurant system	Abtahi Chowdhury Abusaleh Masud Safwan Shahid Arman Uddin Farhan Zaman
11/23/2019	2.0.0	This report is meant to provide the data structure and logic to carry out the functionalities dictated by the specification.	Abtahi Chowdhury Abusaleh Masud Safwan Shahid Arman Uddin Farhan Zaman

Table of Contents

1. Introduction

Below you will encounter many diagrams, charts, and pseudocode, as they will assist in breaking down the design of our product. As you continue reading the paper, any questions or confusions that arise will be answered and clarified.
Not only does it allow customers (registered users) and guests (unregistered users) to order and receive food, but also gives managers, salespeople, cooks, and delivery people access to their own page to handle services in the company. Delivery people have access to see all the different orders from customers and guests, and bid on them. Salespeople are given comments from cooks, to know and order ingredients that are needed. Cooks are allowed to request more supplies from salespeople, rate salespeople, and change menu items. Managers can approve guest to customers, view order history, view all ratings, start delivery bidding process per order, pay employees, hire/fire employees, and remove warnings.

The collaboration class diagram below gives an overview on the entire Kitchen Zealot system.



DB: Database

2. Design

2.1 Use Case Scenarios

View Menu
Normal Scenario 1. Guests, or customers click “Menu” on the homepage to see menu. Customers or Guests who are logged in to the site, can click “Order” to view the menu to order.

Login
Normal Scenario 1. User clicks “Login” button in the navigation bar 2. User is asked to enter their email and password 3. User inputs appropriate information 4. Information is checked against database for authenticity 5. User is logged in
Exceptional Scenario 1. User clicks login and enters credentials 2. Credentials are not valid 3. Error message is shown 4. User is asked to try again

Continue as Guest
Normal Scenario 1. User clicks on “Login” and then “Continue as Guest” 2. User inputs their phone number and address when ordering
Exceptional Scenario 1. User clicks on “Login” and then “Continue as Guest” 2. User inputs invalid phone number and/or address 3. Input boxes will highlight red to indicate an error

Register

Normal Scenario

1. Guest clicks login
2. Guest clicks “Sign Up”
3. Guest is asked to enter “Name”, “Email”, “Password”, “Phone Number”
4. Guest clicks “Register”
5. A request is sent to the manager to approve
6. Manager approves or disapproves new
7. If approved, then guest info is sent to the database to be added

Exceptional Scenario

1. Guest clicks login
2. Guest clicks “Sign Up”
3. Guest is asked to enter “Name”, “Email”, “Password”, “Phone Number”
4. Guest enters an email already in use
5. Guest clicks “Register”
6. An error message pops up stating the email is already used

Add/Remove Products to Cart

Normal Scenario

1. Customer clicks on order
2. Customer adds an item to the cart with the quantity they want

Order Food

Normal Scenario

Scenario 1: Customer

1. Customer clicks “Order”
2. Customer views menu and clicks “Add to Cart”
3. Customer adjusts quantity
4. Customer click “Check Out”

Scenario 2: Guest

1. Guest clicks “Menu”
2. Guest adds an item to the cart with the quantity they want
3. Guest clicks “Check Out” to place order
4. Guest is prompted to enter their details: Name, Email, Phone Number, Payment Details

View All Placed Orders

Normal Scenario

1. Customer clicks on “My Orders” to view all placed orders

Rate Food/Delivery

Normal Scenario

1. Customers can rate the food(cooks) and the delivery people individually
2. Customer Views order
3. Rate Food
4. Rate Delivery

Bid on Deliveries

Normal Scenario

1. Delivery person can access the pending bid page via a button on the post-login page
2. Delivery person chooses from a list of pending bids
3. Delivery person places bids on that selected order
4. After 15 minutes time runs out, bid is over

Exceptional Scenario

1. Two users input the same bid at the same time

Give Customers Rating

Normal Scenario

1. Delivery person can access the past delivered ordered in the “Delivered Orders” tab
2. Delivery person can view orders by clicking “View” next to the respective order
3. Delivery person can give customers ratings from 1 to 5 and an optional comment

Exceptional Scenario

1. Can’t give a rating if they view the rating they received from said customer

View Delivered Orders

Normal Scenario

1. Delivery person can access the past delivered ordered in the “Delivered Orders” tab

Modify Menu Items

Normal Scenario

1. Cooks can access the “Modify Menu Page” by clicking a button in the navbar
2. Cook choose a menu item they wish to change or add a new menu item

Exceptional Scenario

1. Two cooks may try to delete the same item at the same time

Request Supplies

Normal Scenario

1. Cooks can click a “Request Supplies” button on the navigation bar
2. On the Request Supplies Page, the cook can order the desired supplies by clicking “Order”

Rate Supplies

Normal Scenario

1. Cooks can click “Rate Past Orders’
2. Cooks can click “View” on a past order
3. Cooks can rate from 1 to 5, and write an optional comment

Purchase Supplies

Normal Scenario

1. Salesperson click on “Purchase Supplies”
2. Salesperson can open a list of items and select which items and in what quantity to purchase

Start Bids

Normal Scenario

1. Managers can click “Manage Orders” to view all orders
2. Managers can then click an order to view order details and start bids if appropriate

Exceptional Scenario

1. No current orders

Manage Employees

Normal Scenario

1. Managers can click the “Employees” tab
2. Managers can click “View” next to any Employee to view employee details
3. Managers can hire new employees by manually adding the required information

Exceptional Scenario

1. May go below minimum number of employees

Approve Registration

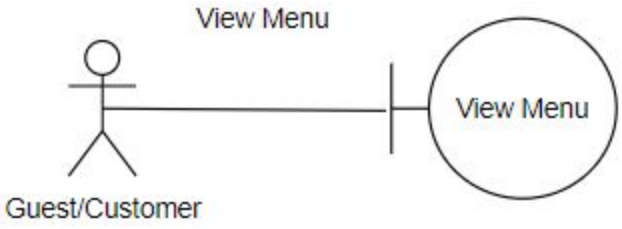
Normal Scenario

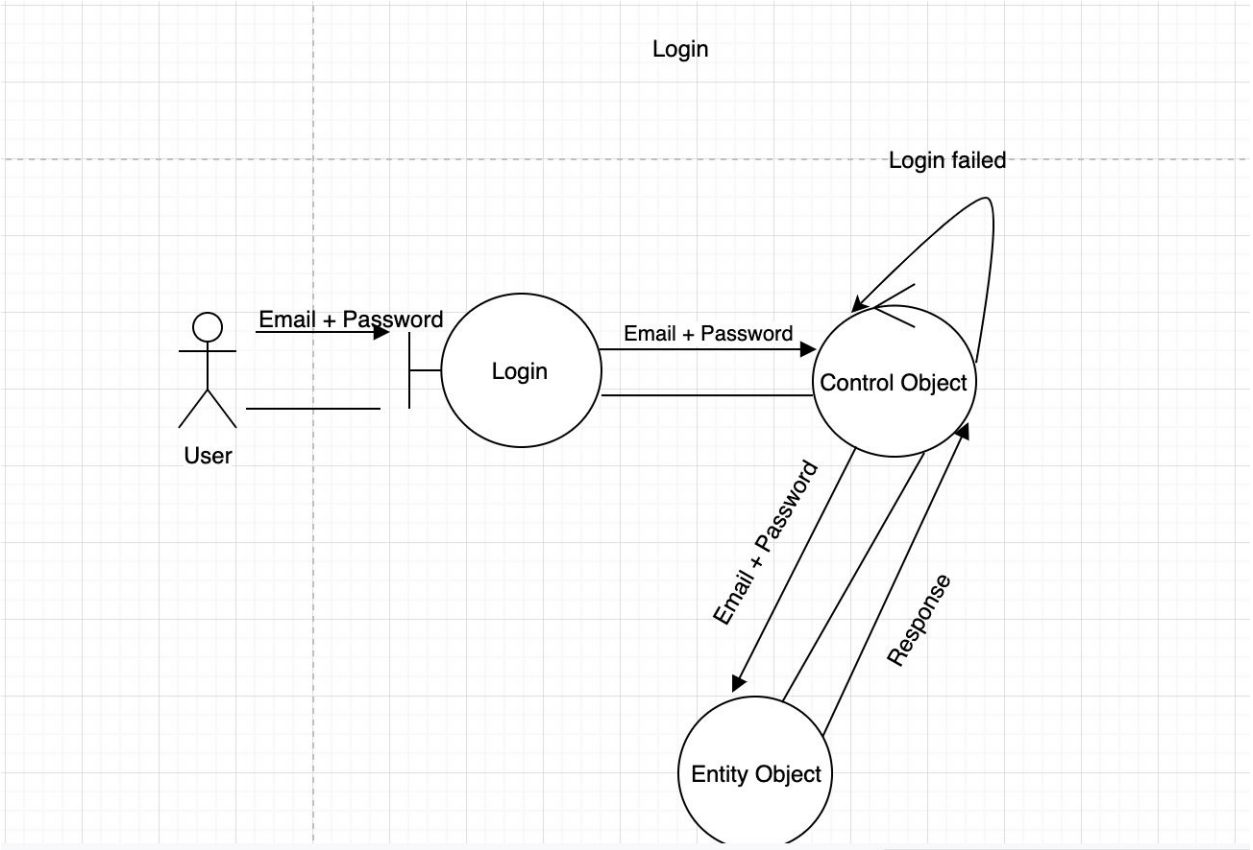
1. Manager clicks “Pending Registrations”
2. Manager can click “View” next to applicants name
3. Manager clicks “Yes” / “No” next to each new guest

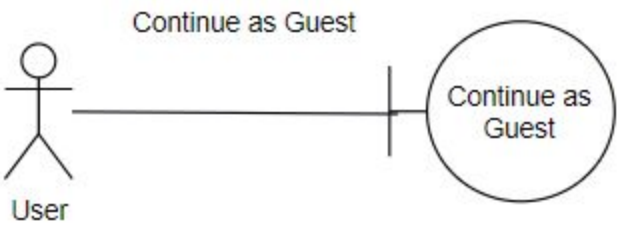
Exceptional Scenario

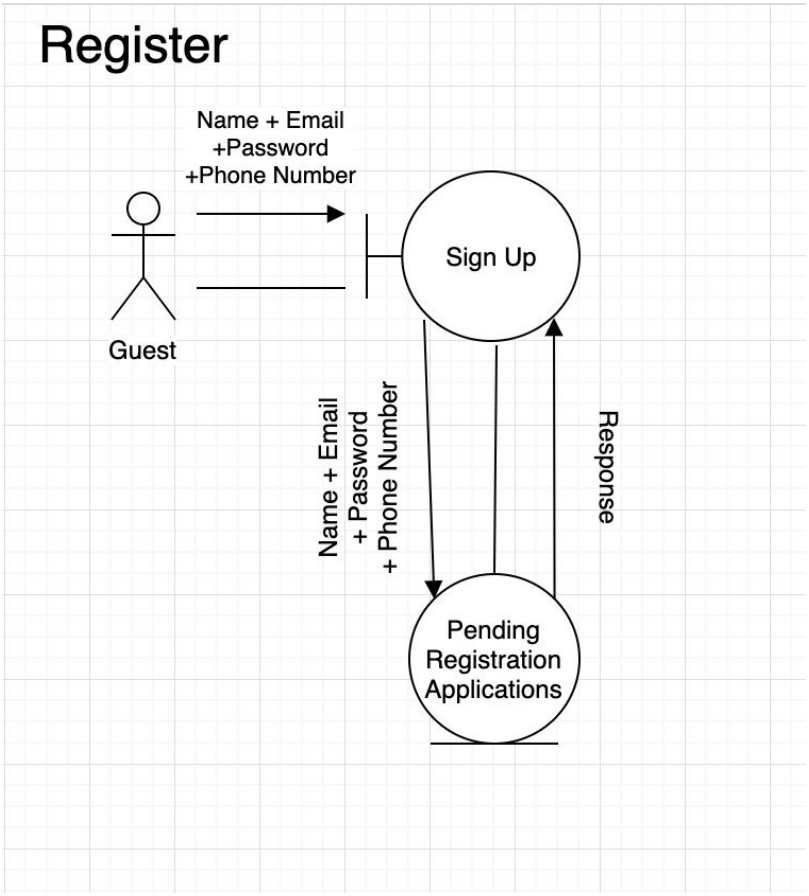
1. Guest may not have order history

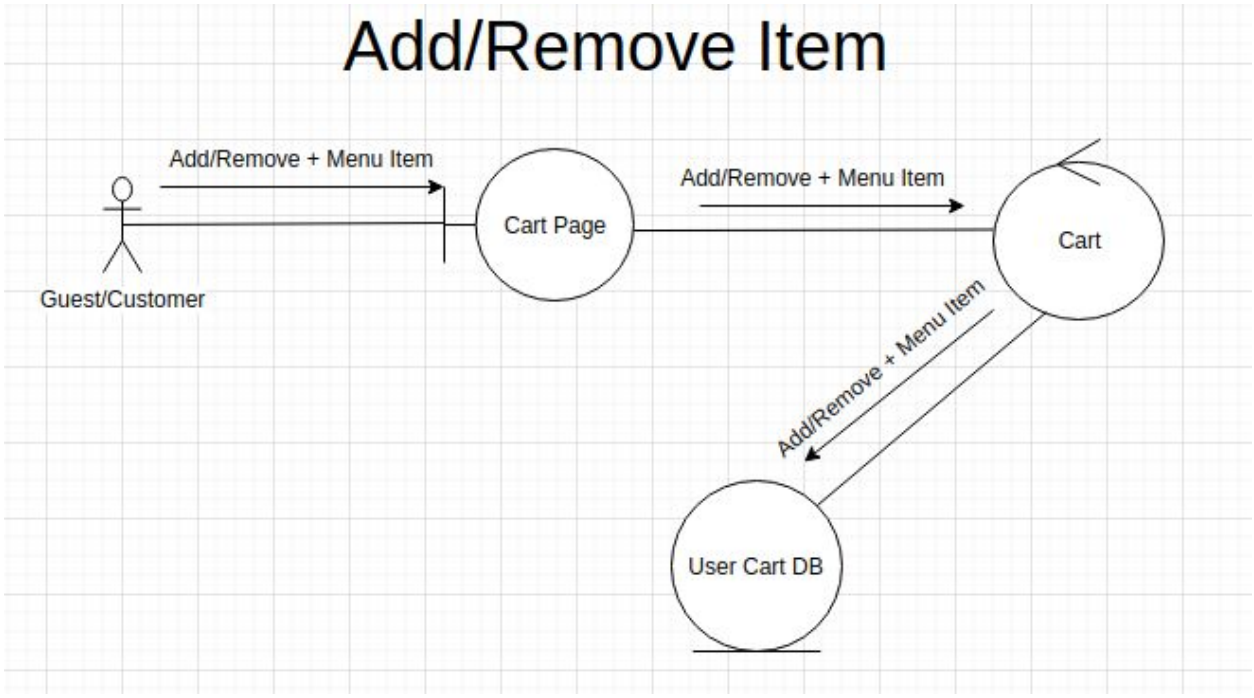
2.2 Collaboration Diagrams

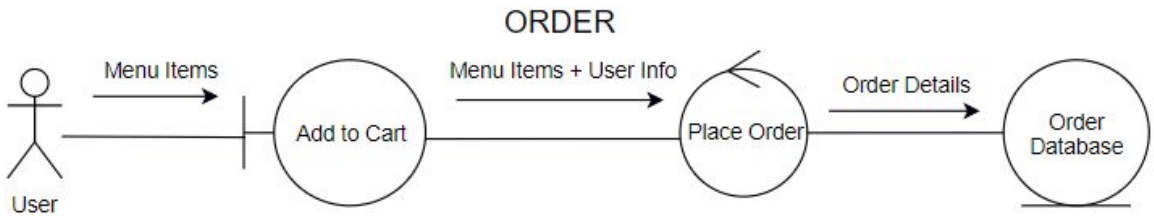


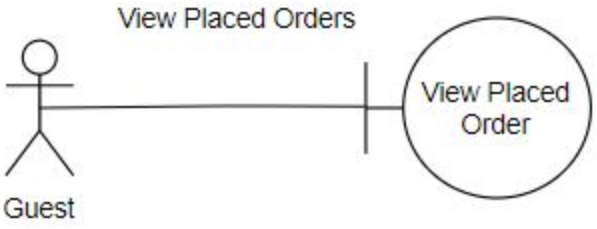


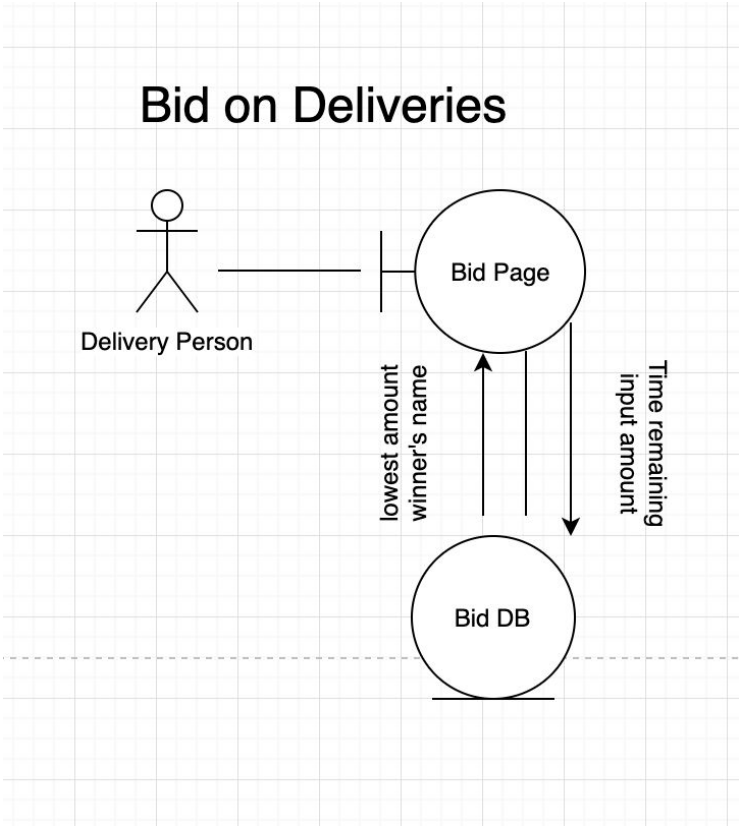

Continue as Guest
Continue as Guest
User

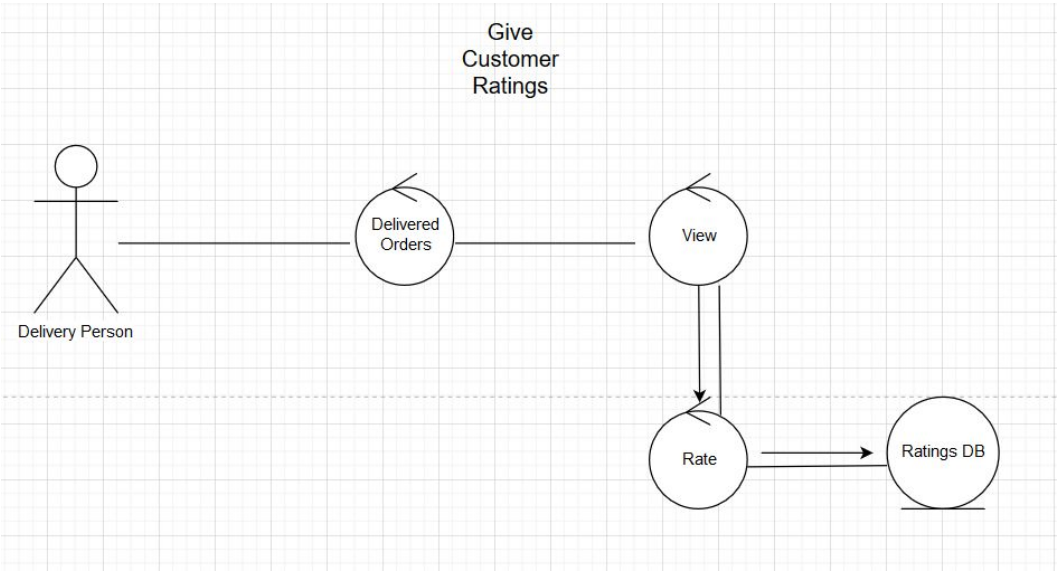

Register
Name + Email
+Password
+Phone Number
Sign Up
Guest
Name + Email
+ Password
+ Phone Number
Response
Pending
Registration
Applications

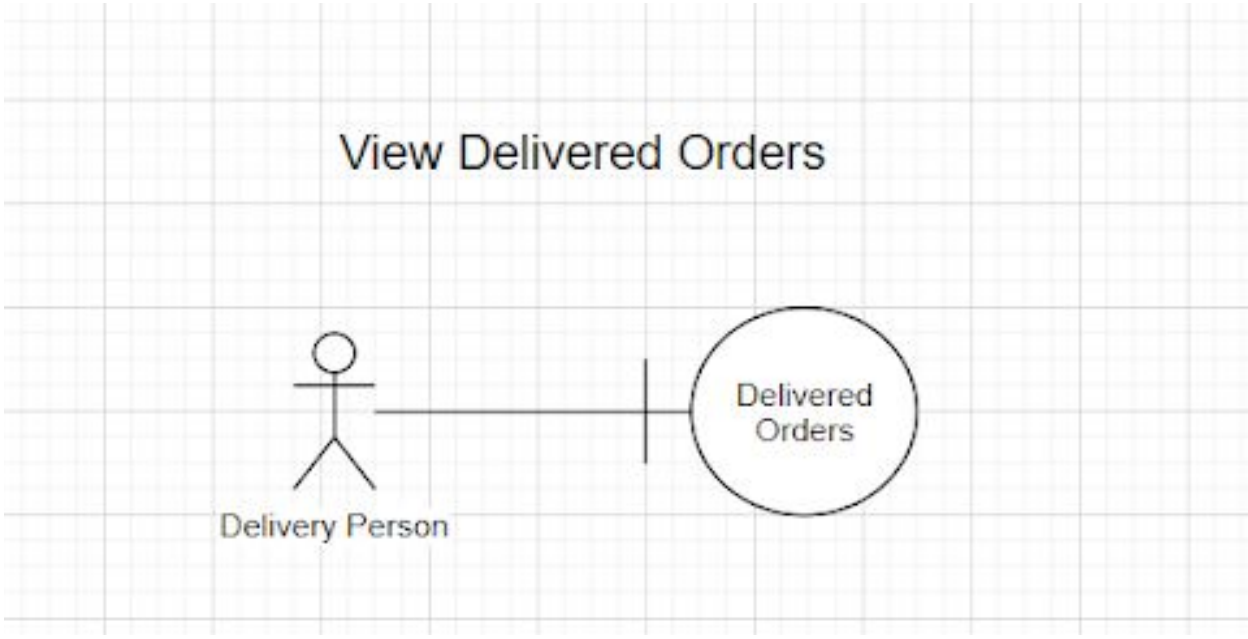

Add/Remove Item
Add/Remove + Menu Item
Guest/Customer
Cart Page
Add/Remove + Menu Item
Cart
Add/Remove + Menu Item
User Cart DB

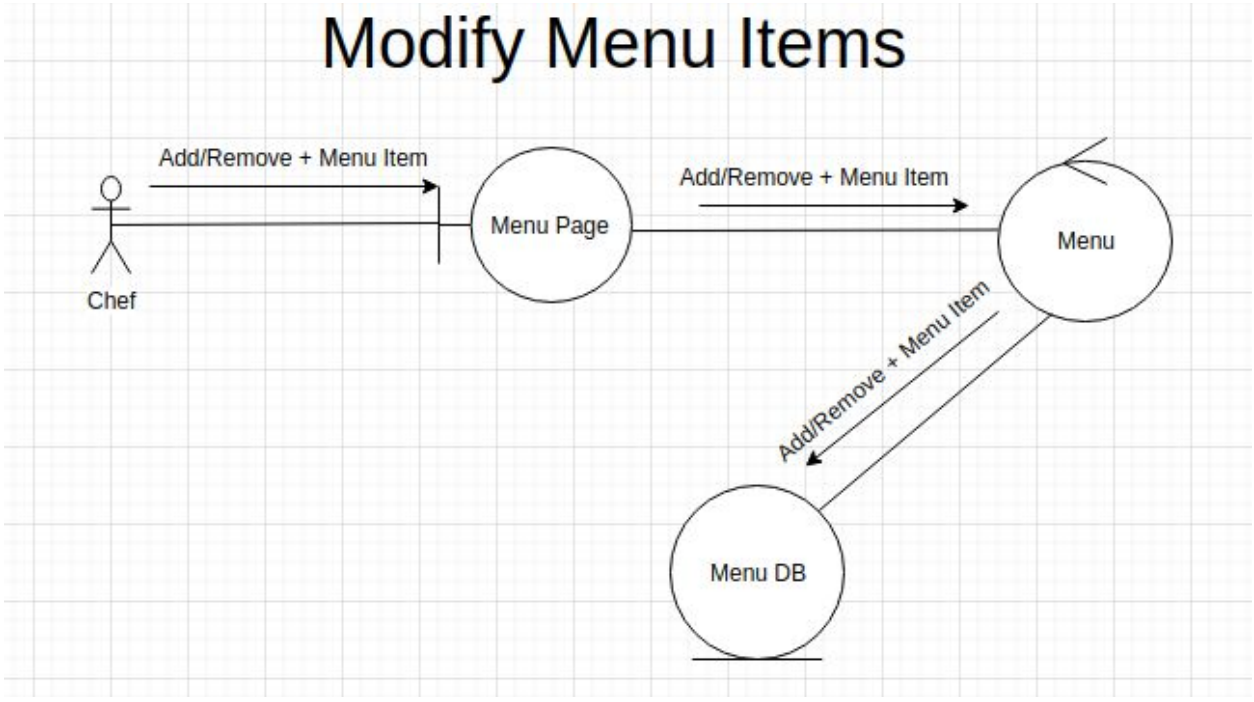

ORDER
Menu Items
User
Add to Cart
Menu Items + User Info
Place Order
Order Details
Order Database

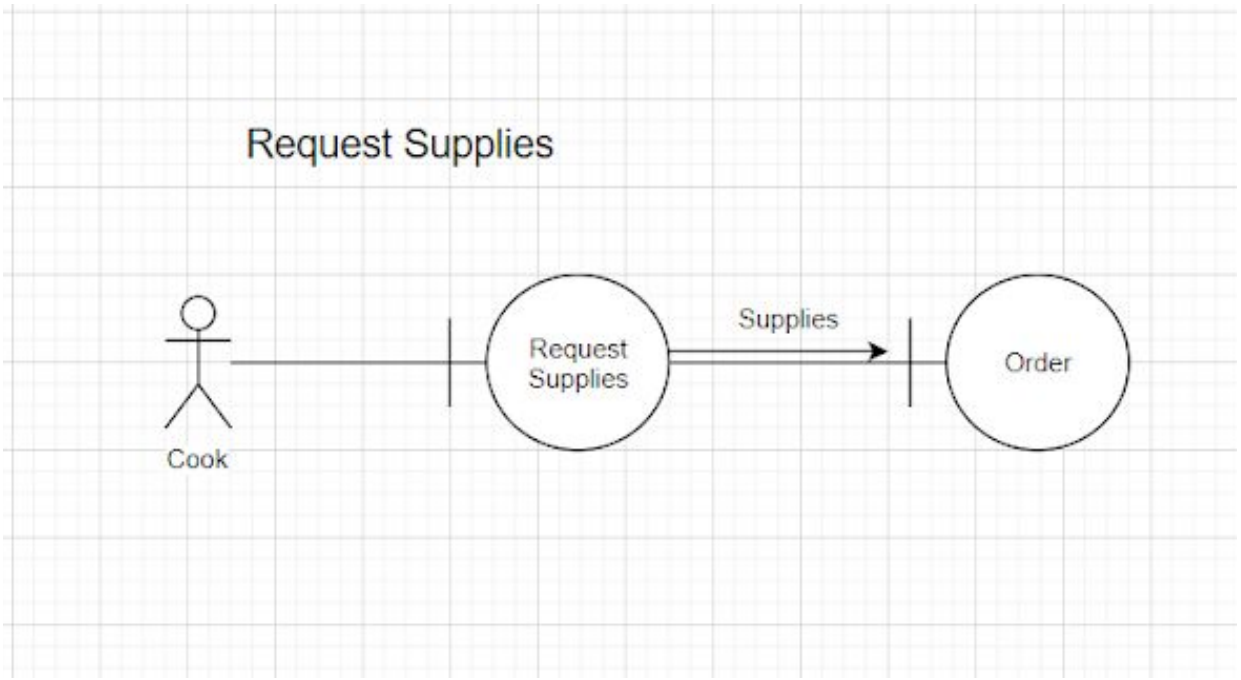

View Placed Orders
Guest
View Placed Order

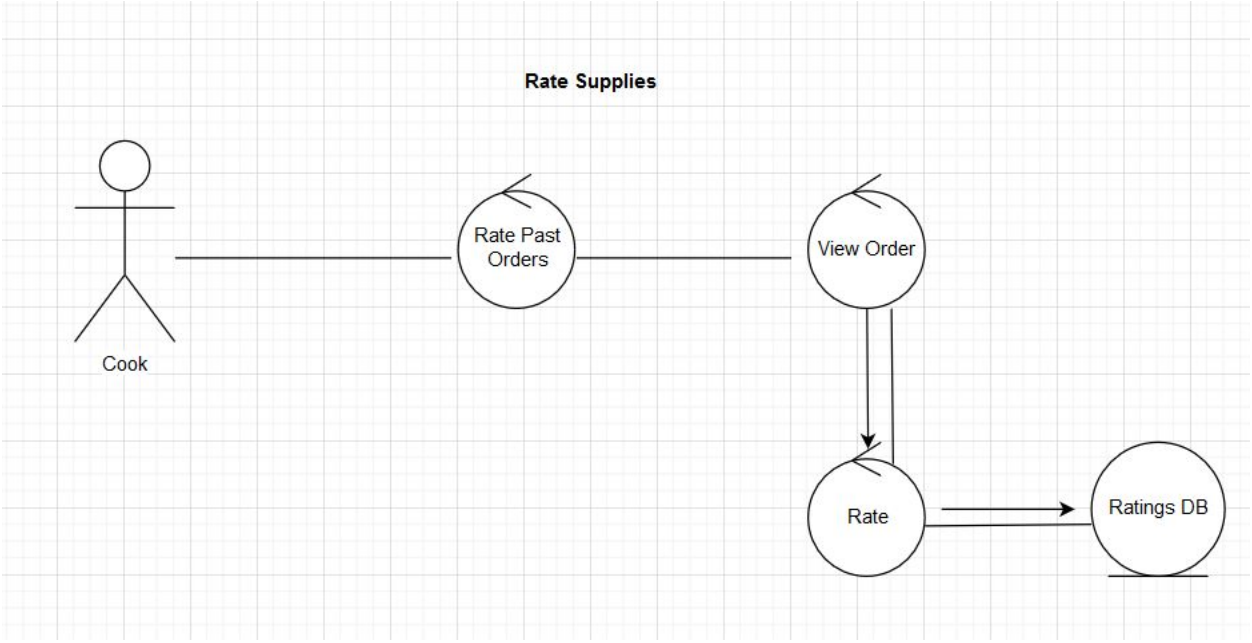

Bid on Deliveries
Delivery Person
Bid Page
lowest amount
winner's name
Time remaining
input amount
Bid DB

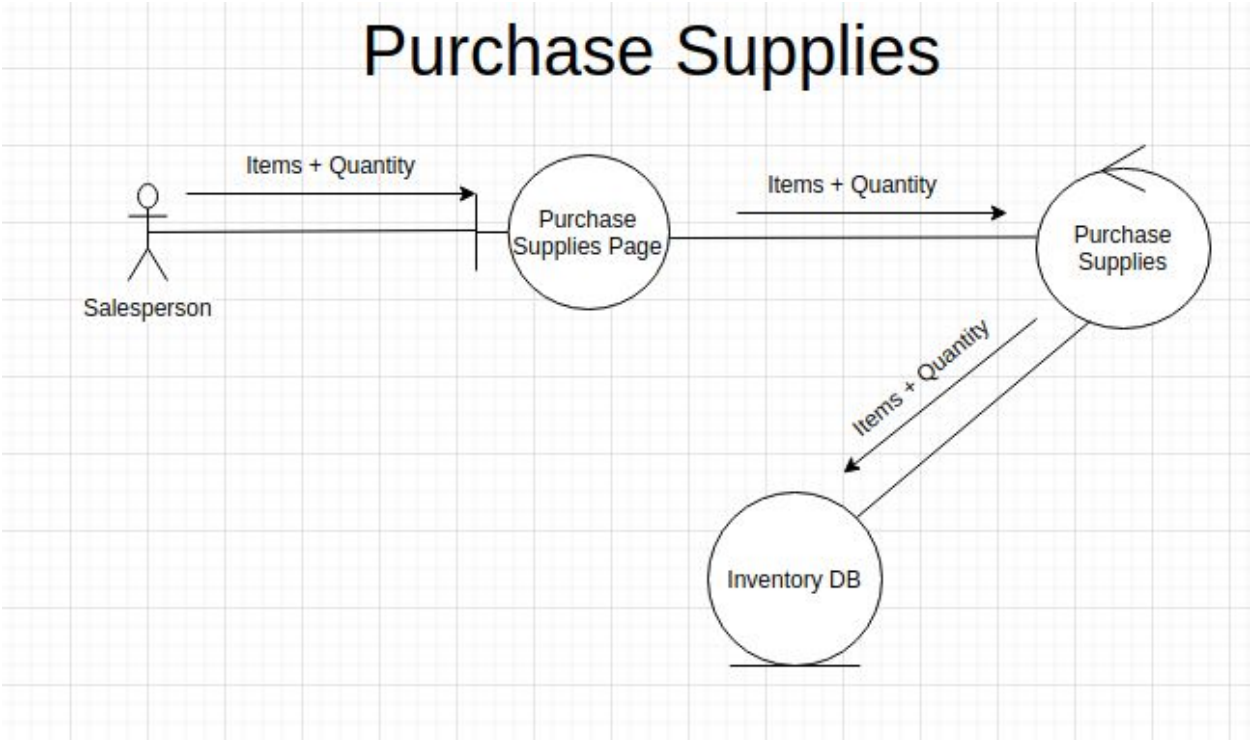

Give
Customer
Ratings
Delivery Person
Delivered
Orders
View
Rate
Ratings DB

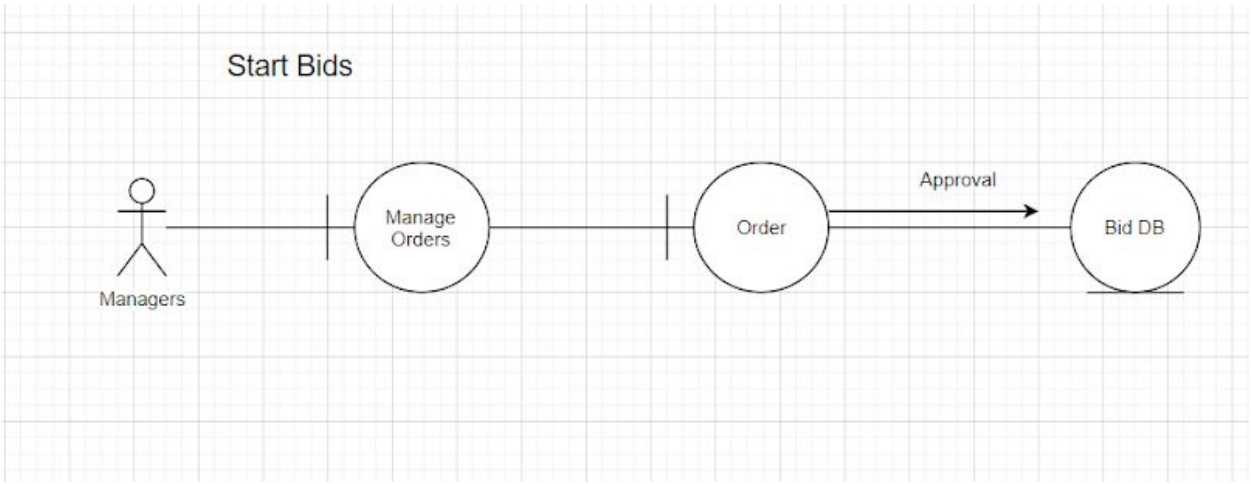

View Delivered Orders
Delivered Orders
Delivery Person

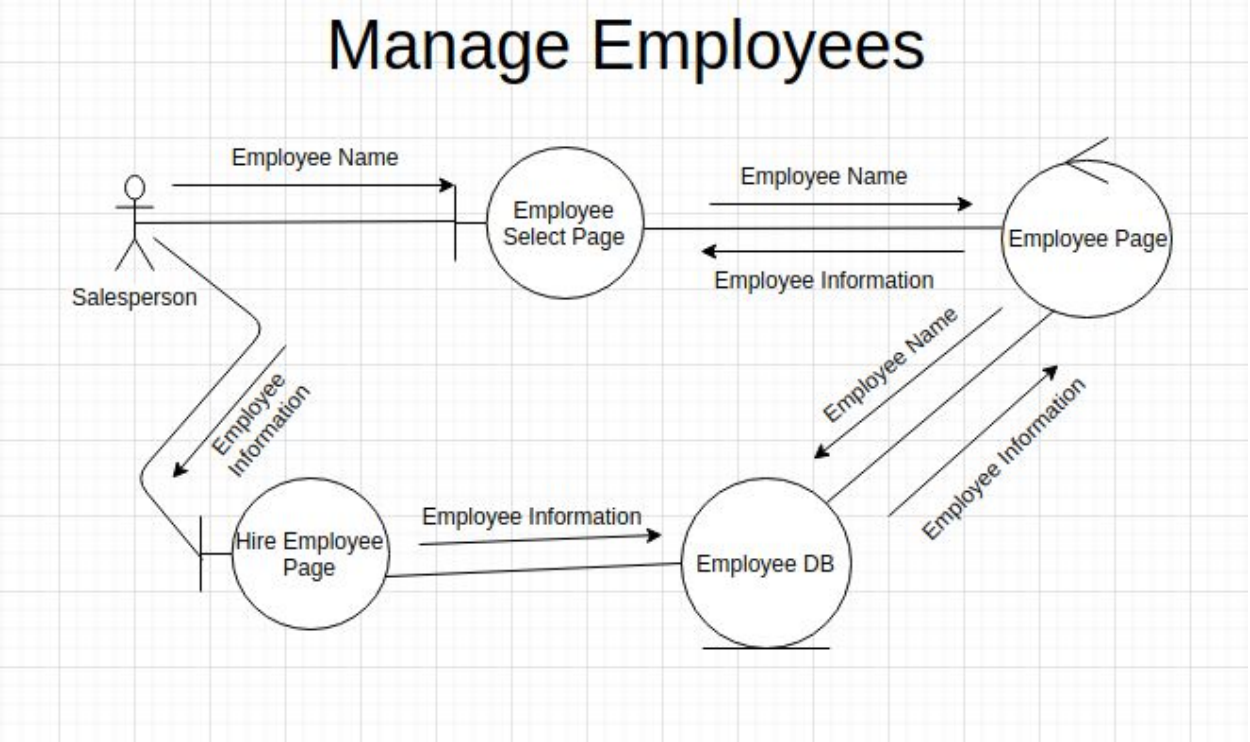

Modify Menu Items
Add/Remove + Menu Item
Menu Page
Add/Remove + Menu Item
Menu
Chef
Add/Remove + Menu Item
Menu DB

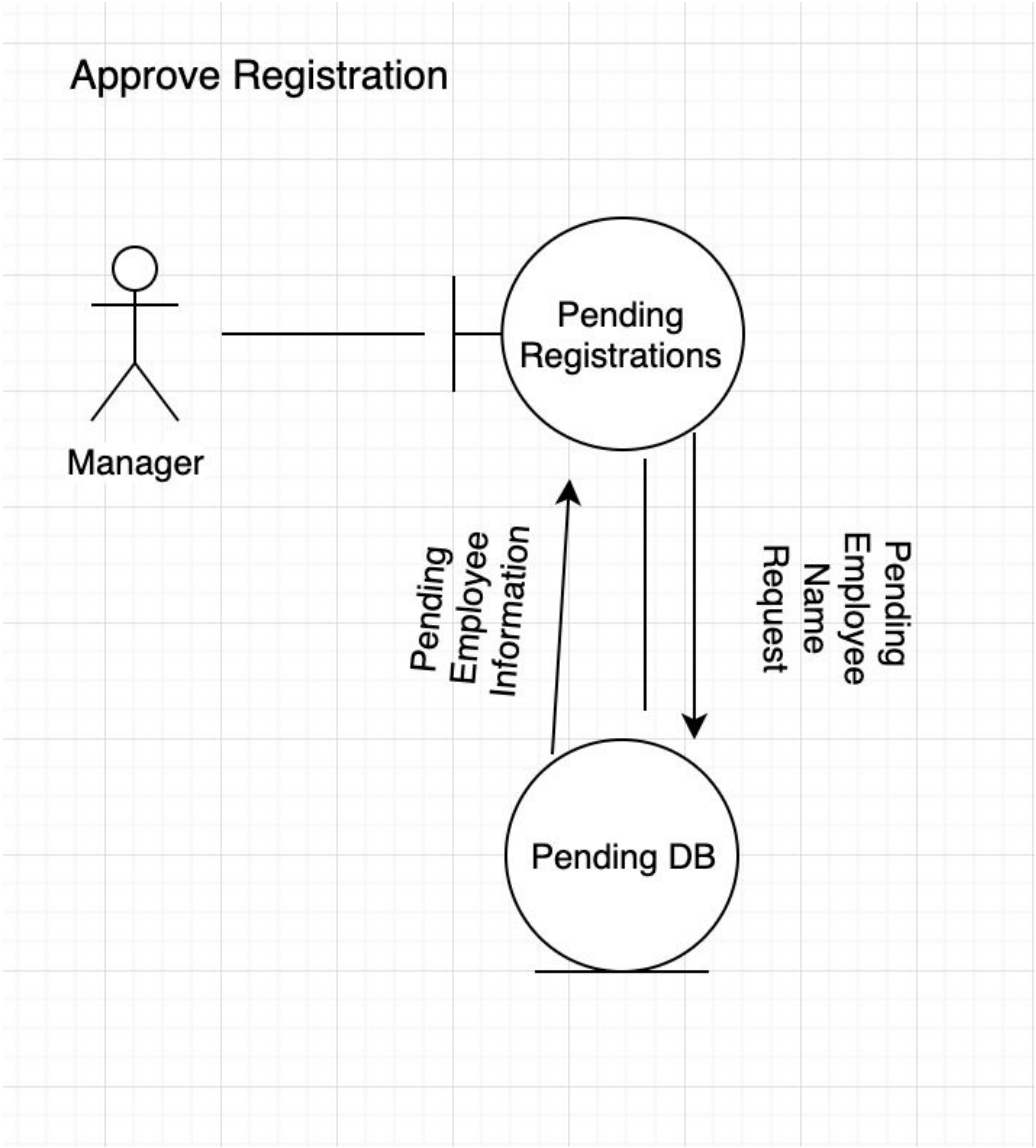

Request Supplies
Cook
Request Supplies
Supplies
Order


Rate Supplies
Cook
Rate Past Orders
View Order
Rate
Ratings DB

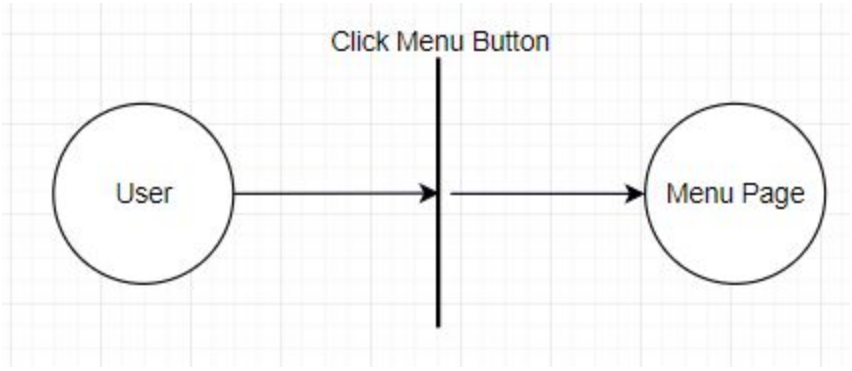

Purchase Supplies
Items + Quantity
Salesperson
Purchase Supplies Page
Items + Quantity
Purchase Supplies
Items + Quantity
Inventory DB

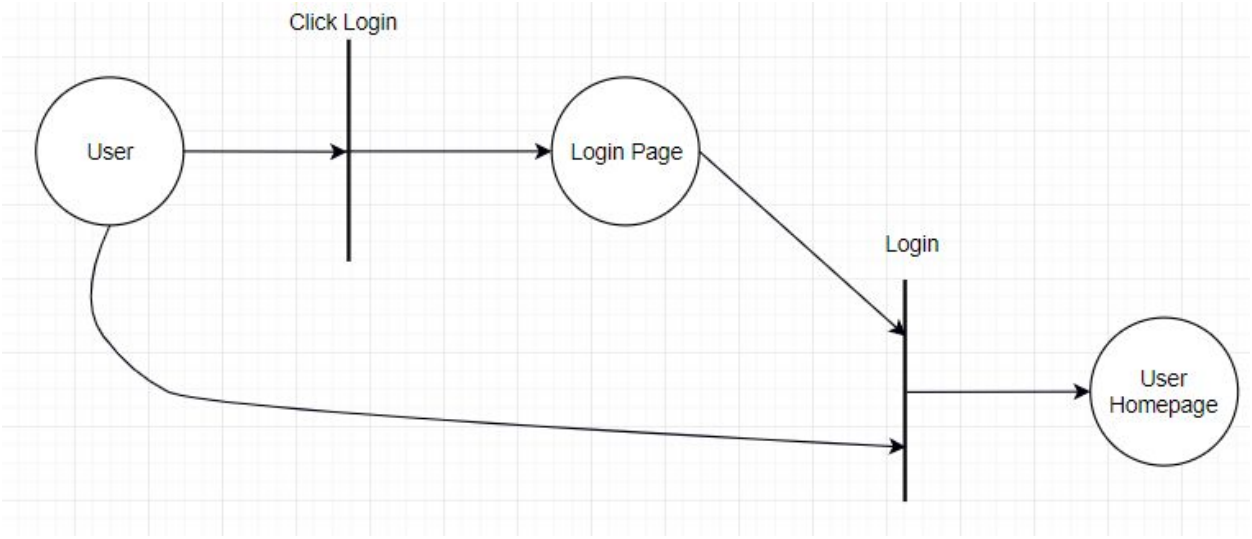

Start Bids
Managers
Manage Orders
Order
Approval
Bid DB

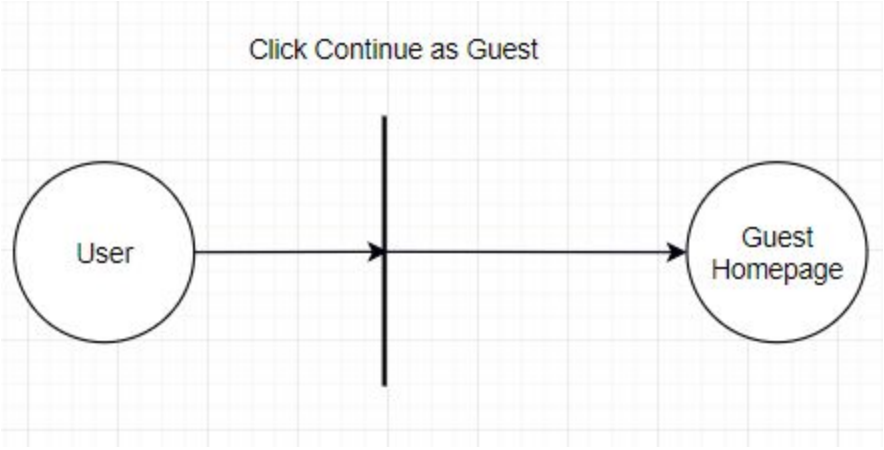

Manage Employees
Employee Name
Employee Select Page
Employee Name
Employee Page
Employee Information
Salesperson
Employee Information
Employee Name
Employee Information
Hire Employee Page
Employee Information
Employee DB

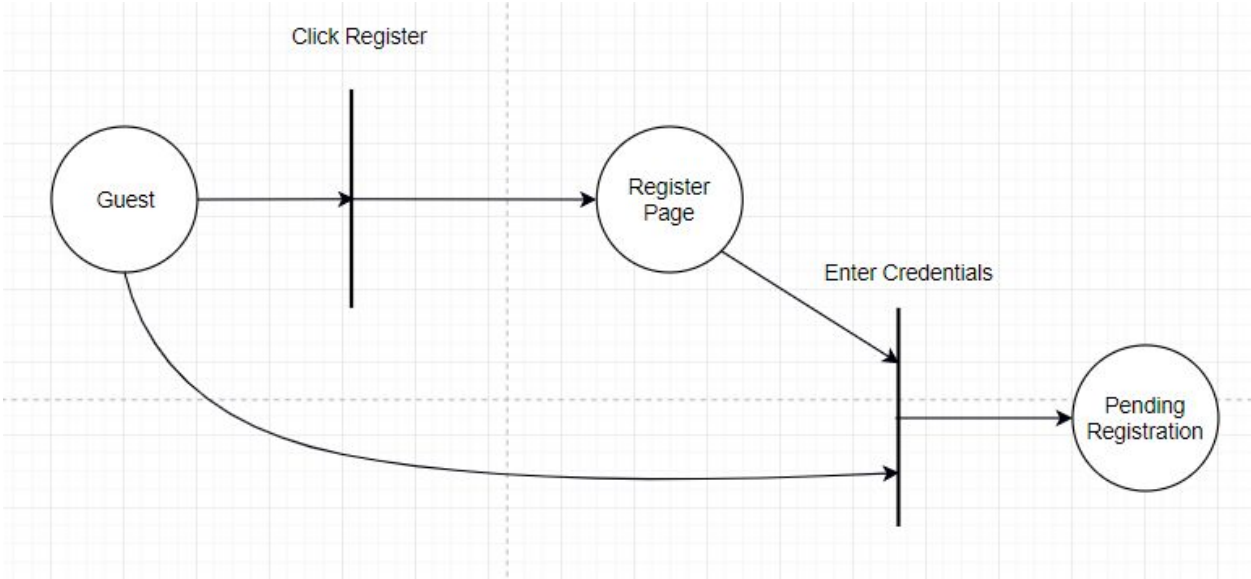

Approve Registration
Manager
Pending Registrations
Pending Employee Information
Pending Employee Name Request
Pending DB

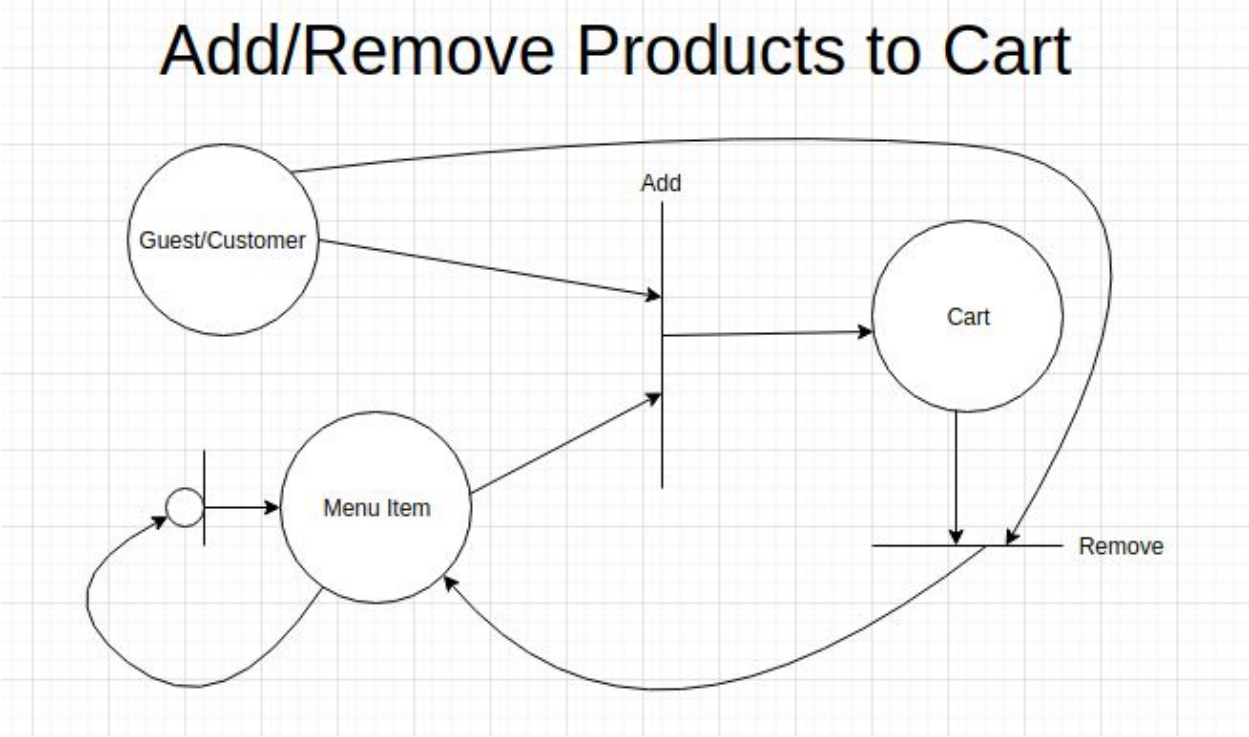
2.3 State Diagrams/ Petri-net

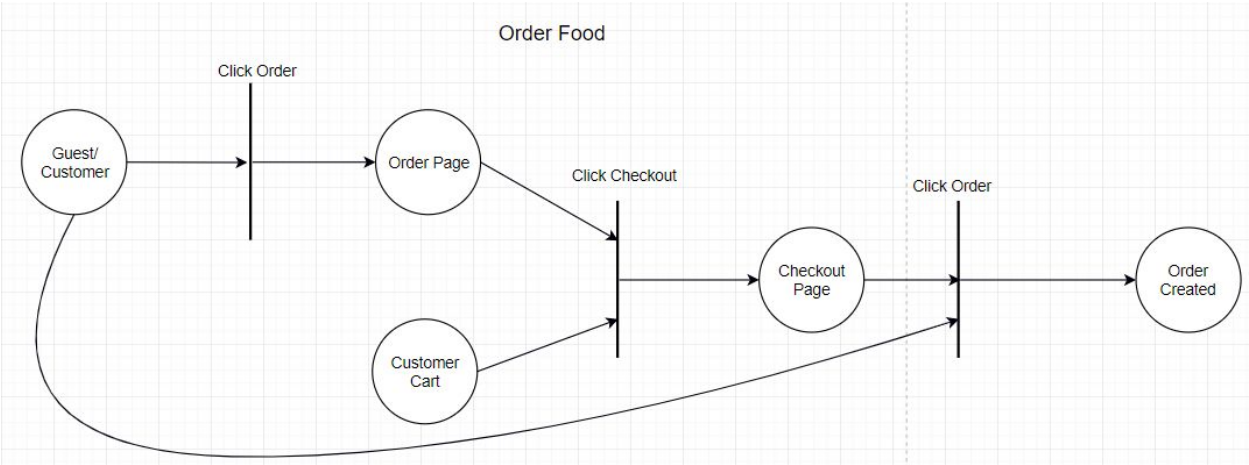


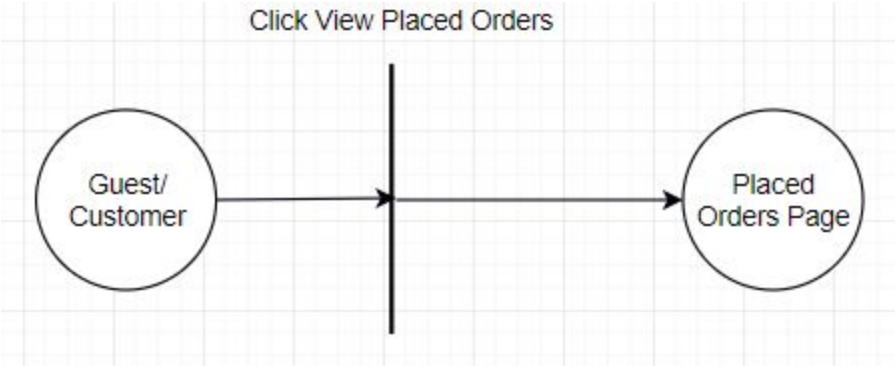


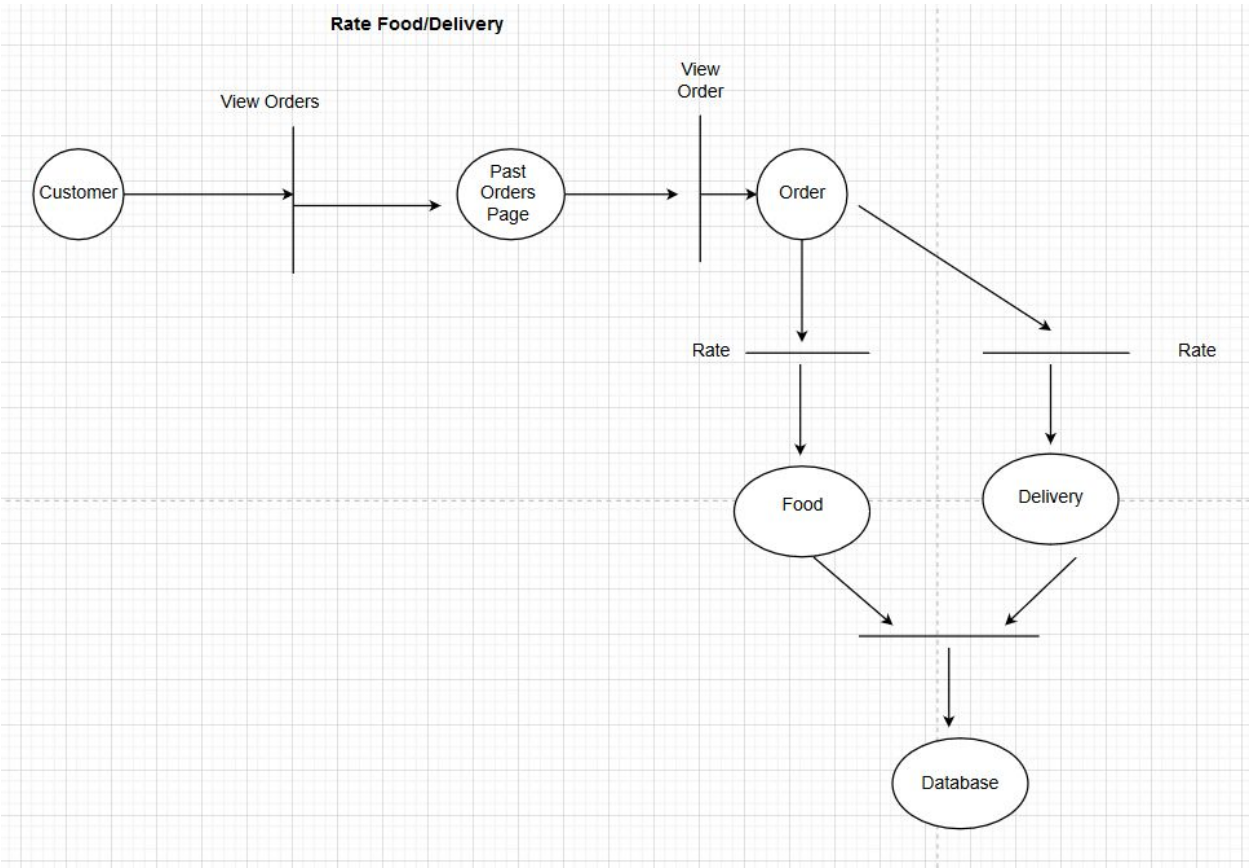


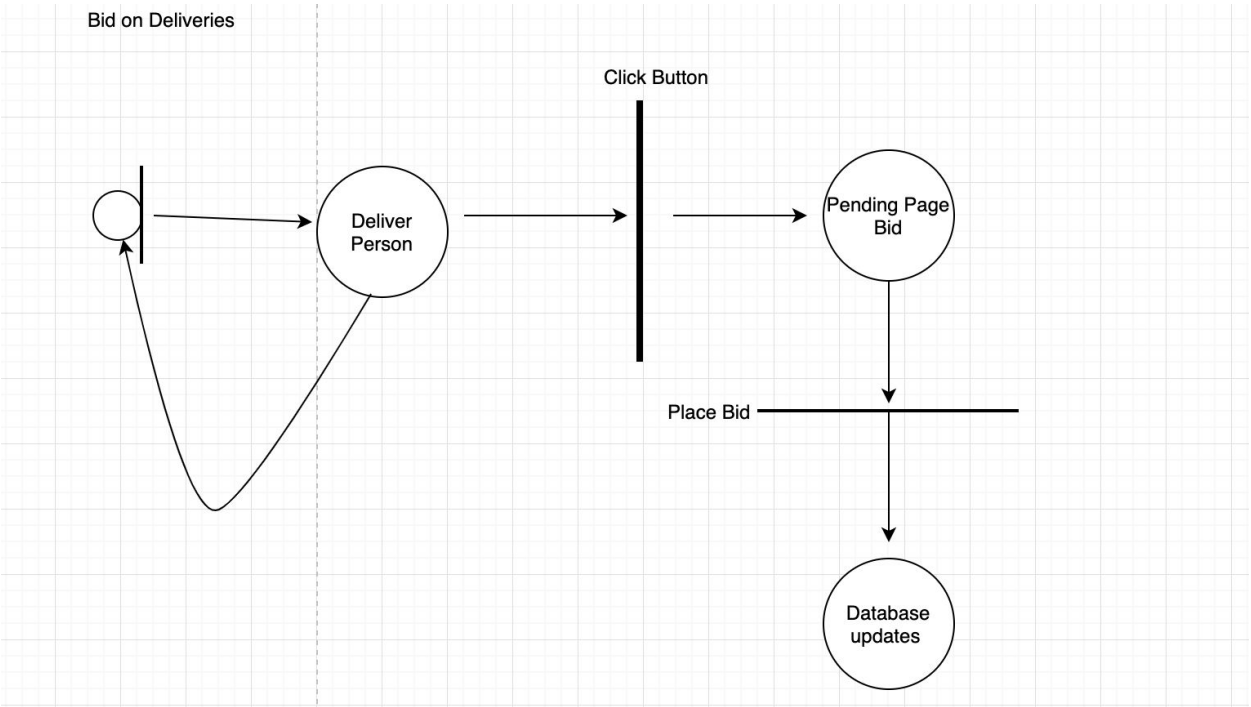

Click Register
Guest
Register Page
Enter Credentials
Pending Registration

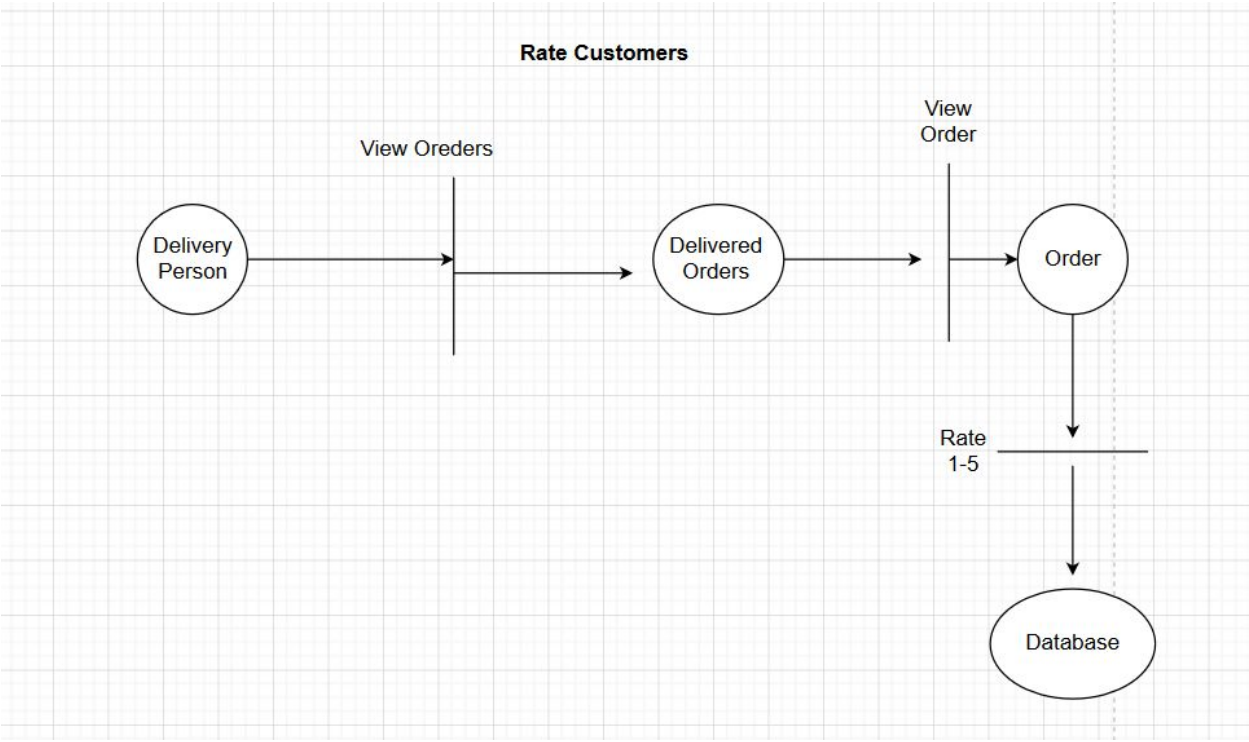

Add/Remove Products to Cart
Guest/Customer
Add
Cart
Menu Item
Remove

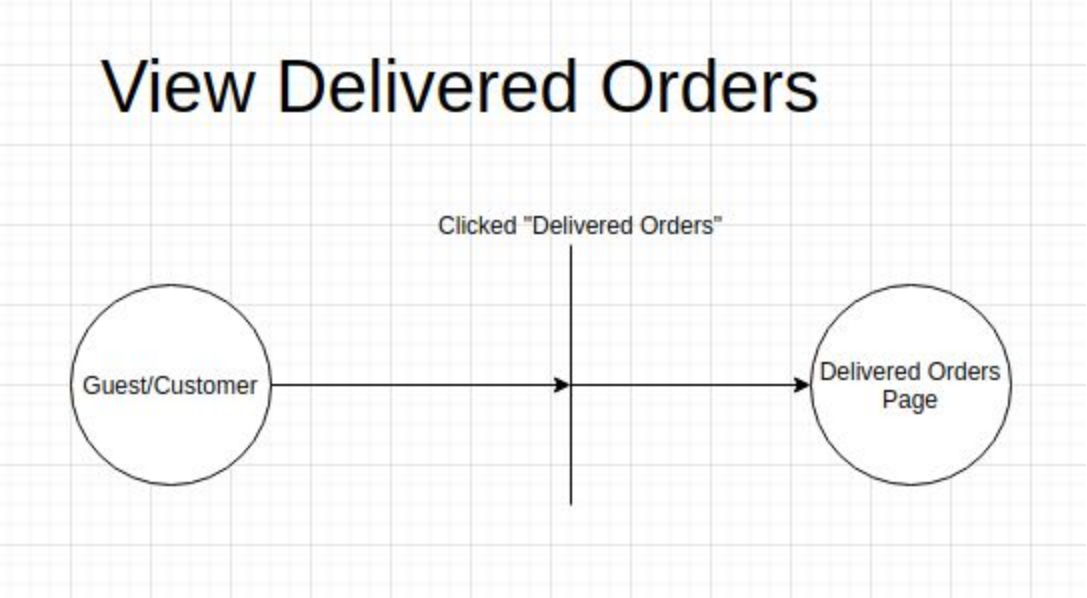

Order Food
Click Order
Guest/
Customer
Order Page
Click Checkout
Click Order
Checkout
Page
Order
Created
Customer
Cart

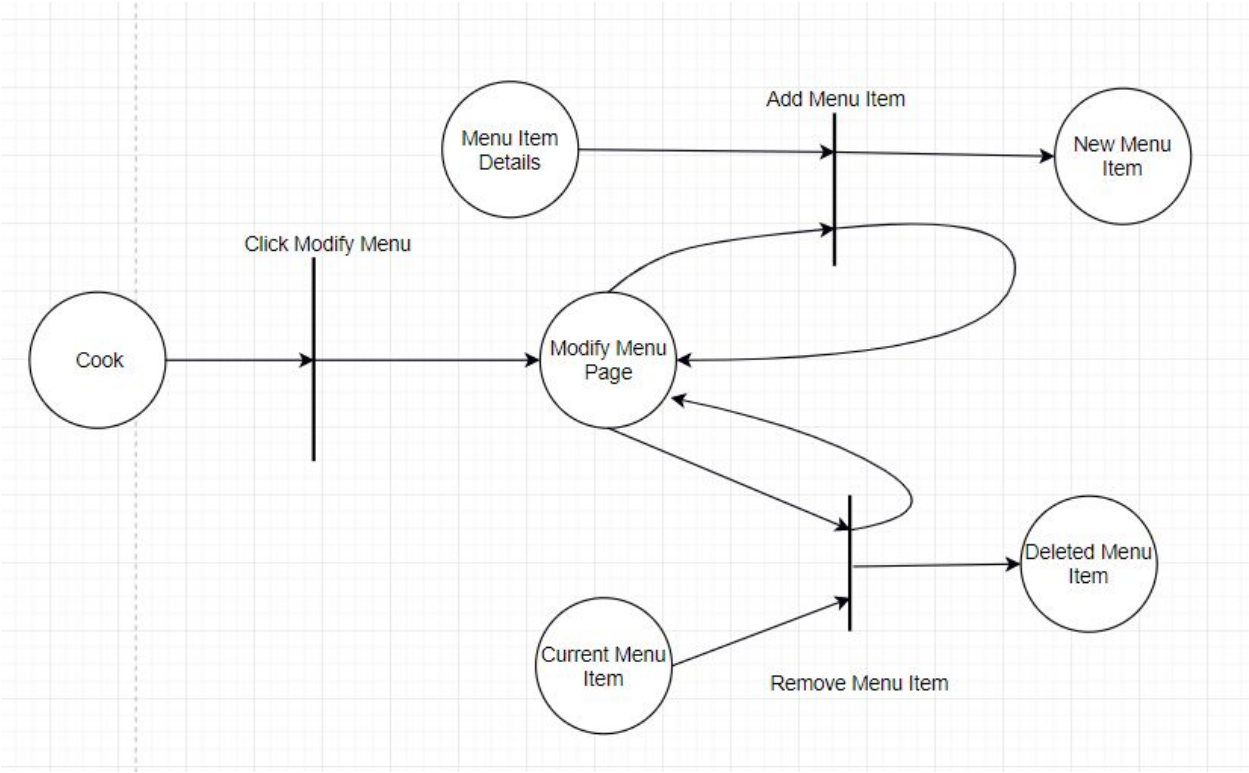

Click View Placed Orders
Guest/
Customer
Placed
Orders Page

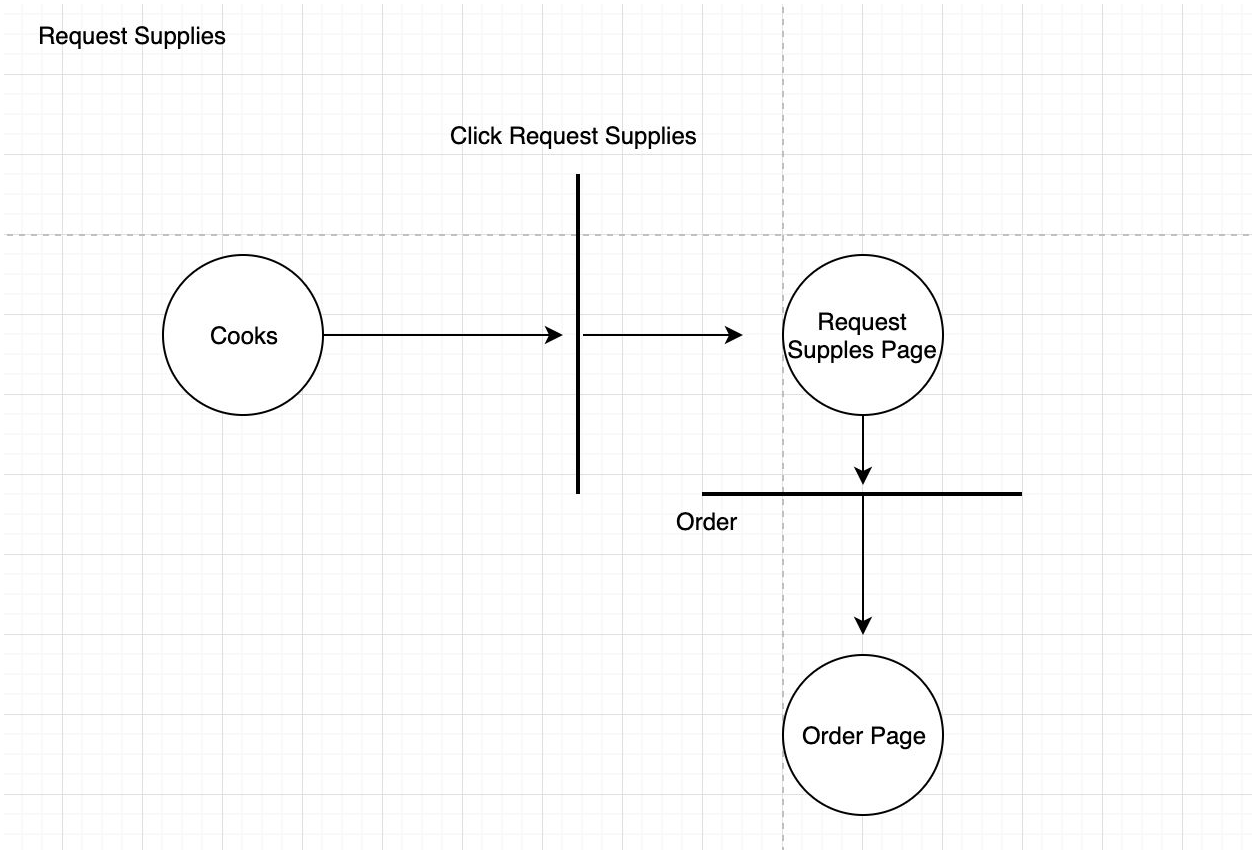

Rate Food/Delivery
View Orders
View Order
Customer
Past Orders Page
Order
Rate
Rate
Food
Delivery
Database

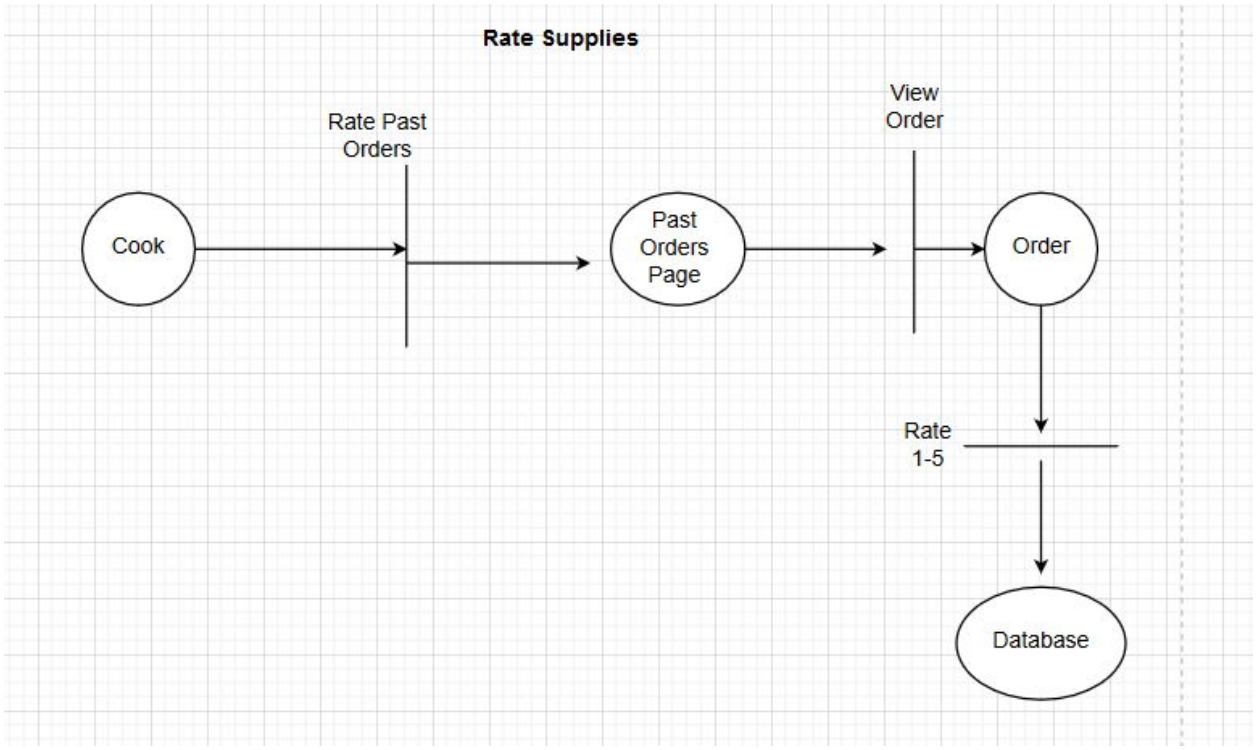

Bid on Deliveries
Click Button
Deliver Person
Pending Page Bid
Place Bid
Database updates

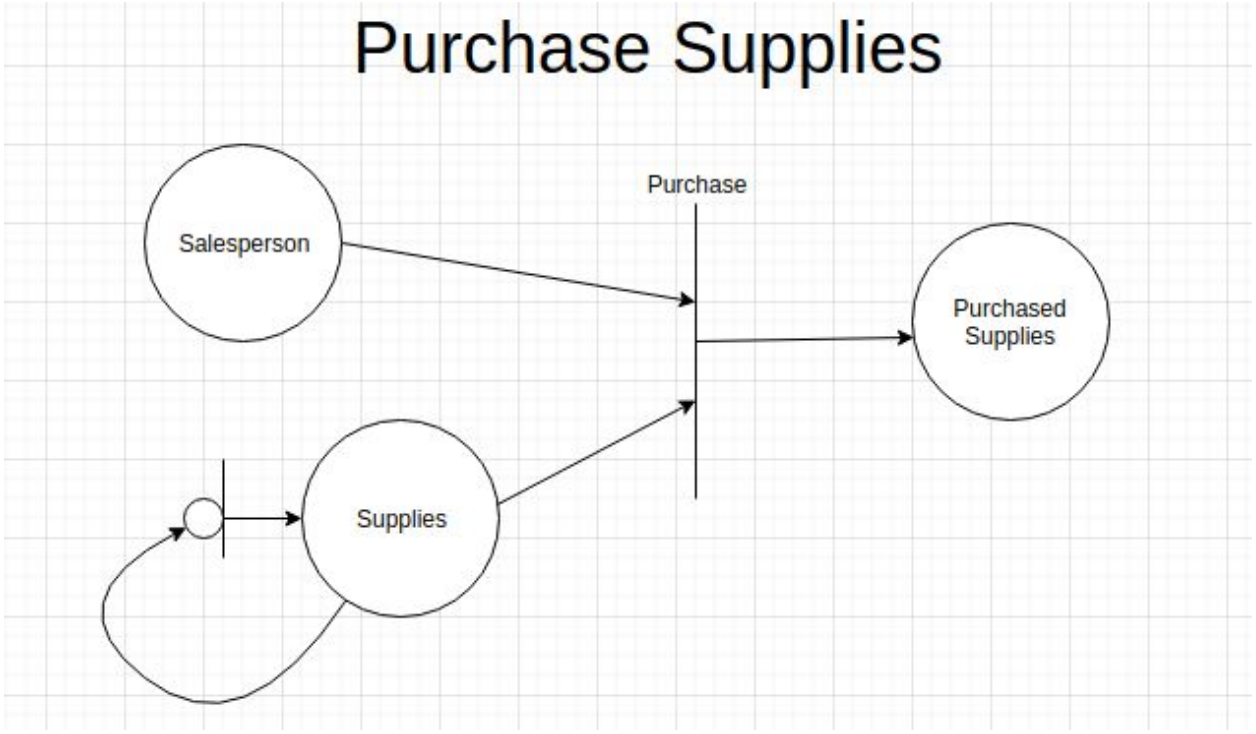

Rate Customers
View Oreders
View Order
Delivery Person
Delivered Orders
Order
Rate 1-5
Database

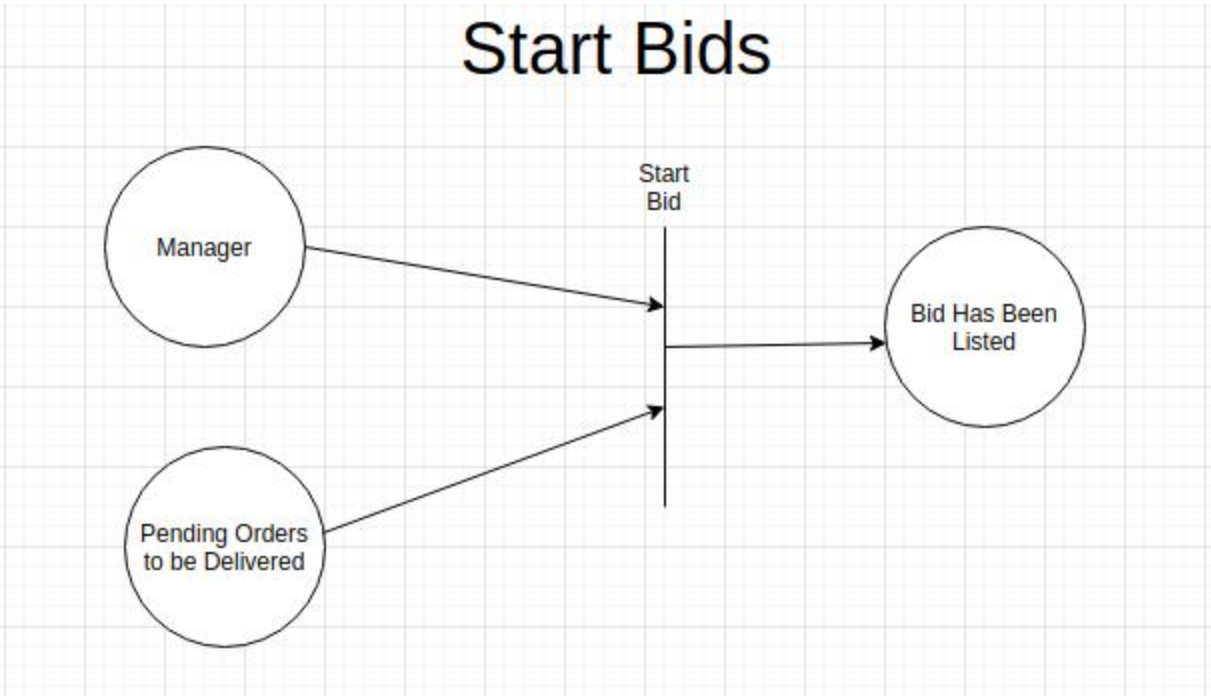

View Delivered Orders
Clicked "Delivered Orders"
Guest/Customer
Delivered Orders Page

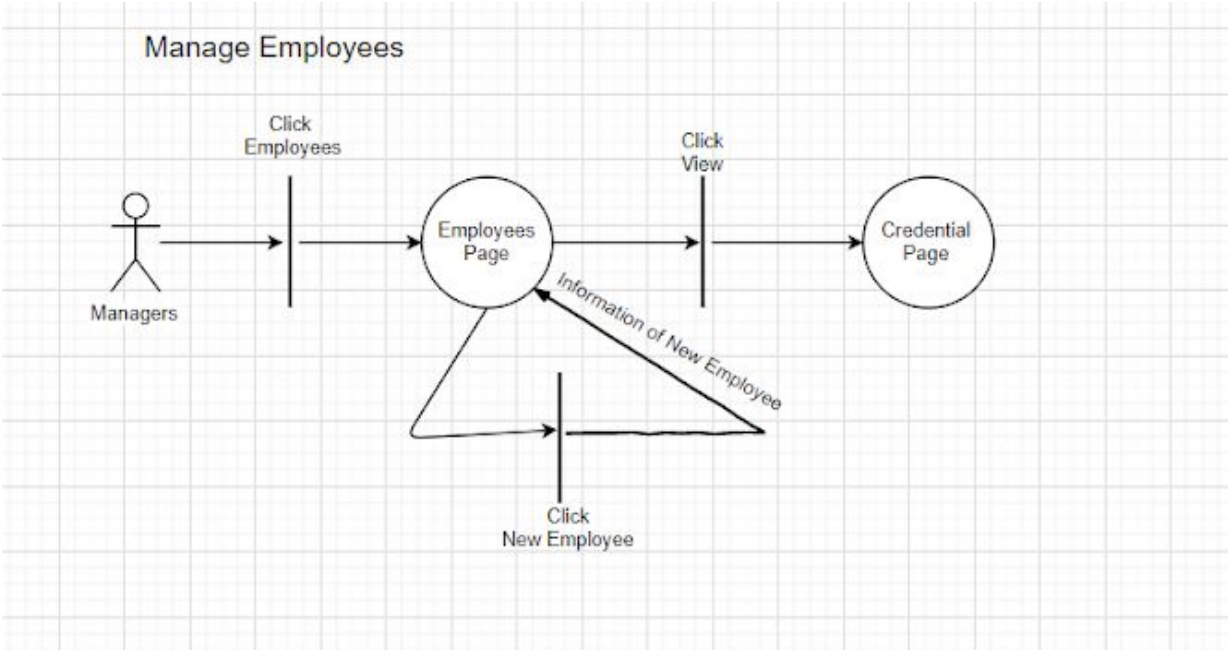

Menu Item Details
Add Menu Item
New Menu Item
Click Modify Menu
Cook
Modify Menu Page
Deleted Menu Item
Current Menu Item
Remove Menu Item

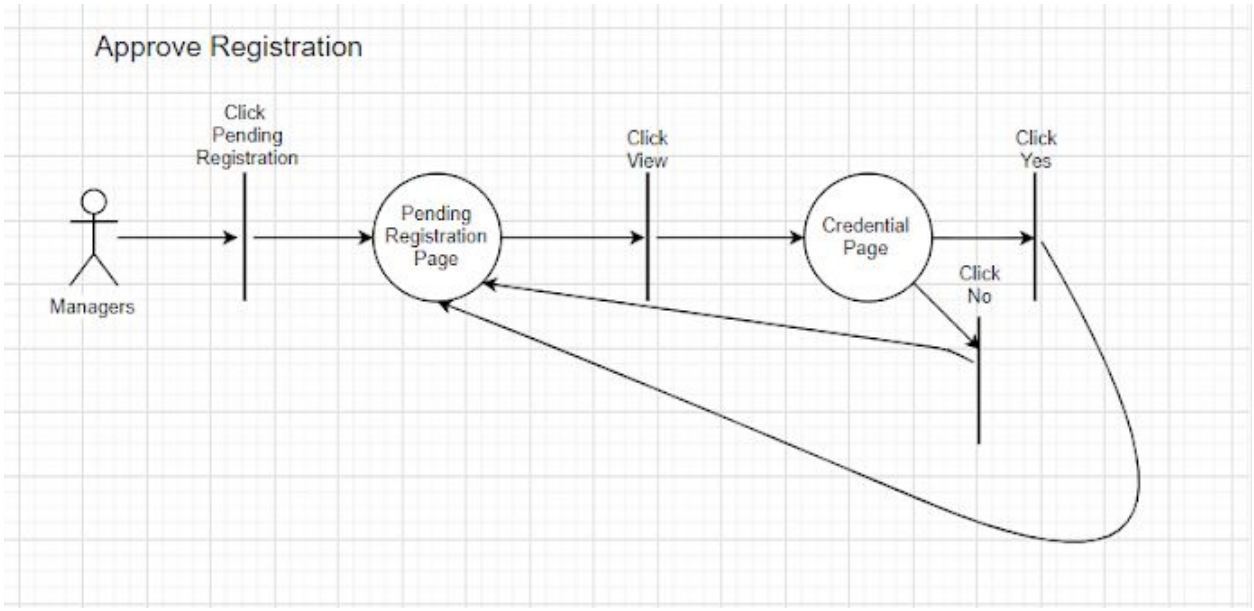

Request Supplies
Click Request Supplies
Cooks
Request
Supples Page
Order
Order Page


Rate Supplies
Rate Past Orders
View Order
Cook
Past Orders Page
Order
Rate 1-5
Database

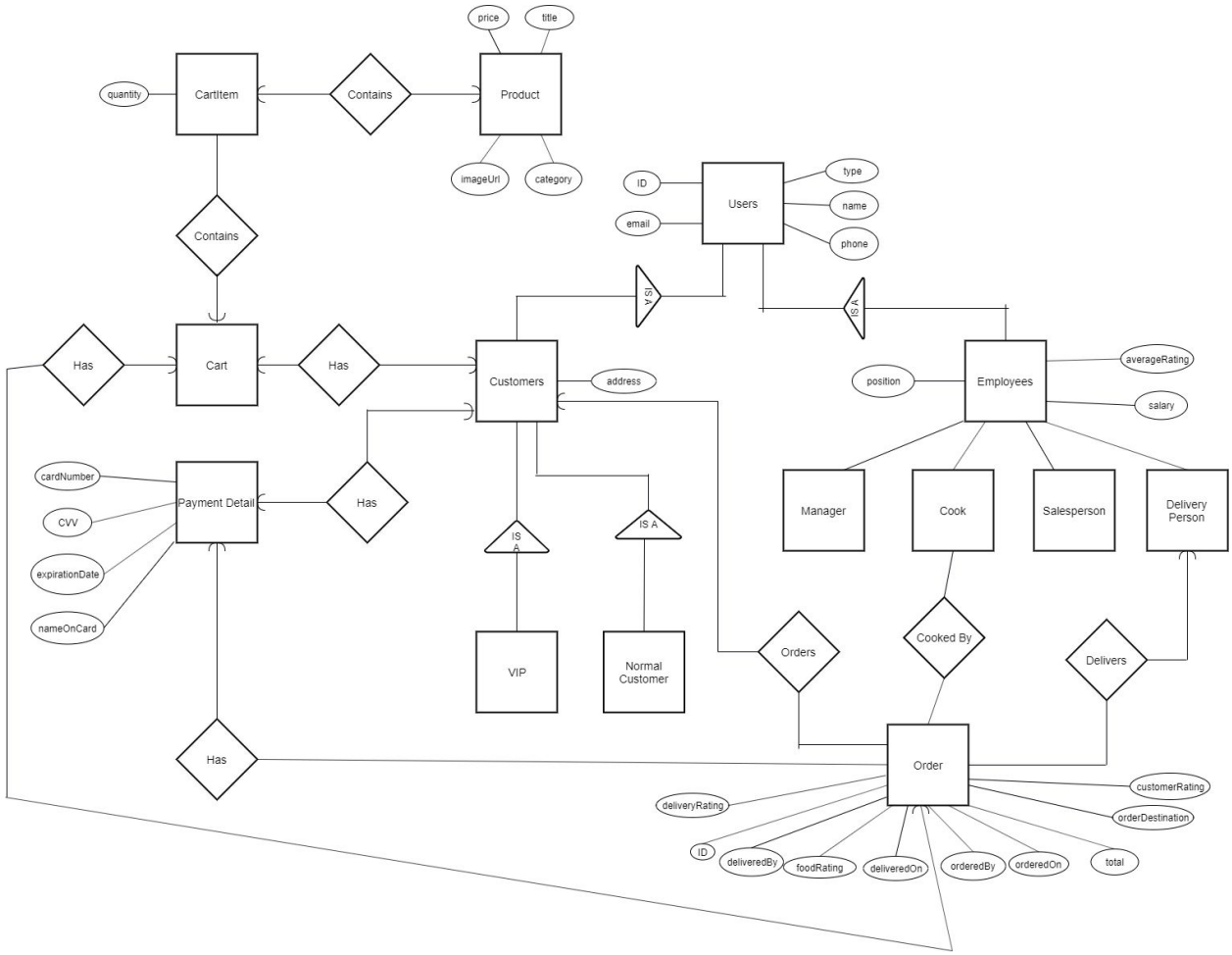

Purchase Supplies
Purchase
Salesperson
Purchased Supplies
Supplies


Start Bids
Manager
Start Bid
Bid Has Been Listed
Pending Orders to be Delivered


Manage Employees
Click Employees
Click View
Employees Page
Credential Page
Managers
Information of New Employee
Click New Employee


Approve Registration
Click
Pending
Registration
Click
View
Click
Yes
Pending
Registration
Page
Credential
Page
Click
No
Managers

3. E/R Diagram



4. Pseudocode

1. addNewEmployee(formValue)

Input: New Employee Information
Output: None
Description: Add new employee's information to the database

```
addNewEmployee(formValue)
 {
   let secondaryApp =
firebase.initializeApp(environment.firebase,"Secondary");
```

```
secondaryApp.auth().createUserWithEmailAndPassword(formValue.email,"123456").then(userCredentials=>{
     let tempUser:User = {uid:userCredentials.user.uid,email:userCredentials.user.email,phone:formValue.phone,name:formValue.name,type:"employee"};
     this.userServe.addUser(tempUser);
     let tempEmployee:Employee = {uid:userCredentials.user.uid,position:formValue.position,salary:formValue.salary};
     this.emplServe.addEmployee(tempEmployee);
     secondaryApp.auth().signOut();
   });
 }
```

2. login(email:string, password:string)

Input: Email, Password
Output: redirected to appropriate URL
Description: Allows a person to login, and DB returns proper type and displays appropriate page

```
 login(email:string, password:string)
 {
   this.afAuth.auth.signInWithEmailAndPassword(email, password).then(userCredentials=>{
     this.user$.pipe(take(1)).subscribe(user=>{
       this.userServe.getUser(user.uid).pipe(take(1)).subscribe(user=>{
         if(user.type=="customer")
         {
           this.router.navigateByUrl("/" + user.type);
         }
         else
         {

this.emplServe.getEmployee(user.uid).pipe(take(1)).subscribe(employee=>{
             this.router.navigateByUrl("/" + employee.position);
           })
         }
```

```
      })
    })
  }).catch(error=>{this.error = error});
}
```

3. register(email:string, password:string)

Input: email, password
Output: None
Description: Creates a customer object with the email and password

```
register(email:string, password:string)
{
this.afAuth.auth.createUserWithEmailAndPassword(email,password
);
 }
```

4. logout()

Input: none
Output: none
Description: Logs current user out

```
logout()
{
  this.afAuth.auth.signOut();
  this.router.navigateByUrl("");
}
```

5. guestLogin()

Input: none
Output: none
Description: Redirects to guest homepage

```
  guestLogin()
  {
    this.afAuth.auth.signInAnonymously().then(guestCredentials=>{
      this.router.navigateByUrl("/guest");
    });
  }
```

6. addToCart(product:Product)

Input: Product
Output: none
Description: Added product to cart

```
addToCart(product:Product)
 {
   this.authServe.user$.pipe(take(1)).subscribe(user=>{

this.custServe.getCustomer(user.uid).pipe(take(1)).subscribe(customer=>{
       for(let cartItem of customer.shoppingCart)
       {
         if(cartItem.product.title == product.title)
         {
           cartItem.quantity += 1;
           this.custServe.updateCustomer(user.uid,customer);
           return;
         }
       }
       customer.shoppingCart.push({product:product,quantity:1});
       this.custServe.updateCustomer(user.uid,customer);
     });
   });
 }
```

7. removeFromCart(product:Product)

Input: product
Output: none
Description: Remove product from cart

```
removeFromCart(product:Product)
 {
   this.authServe.user$.pipe(take(1)).subscribe(user=>{

this.custServe.getCustomer(user.uid).pipe(take(1)).subscribe(customer=>{
       for(let cartIndex =
0;cartIndex<customer.shoppingCart.length;cartIndex++)
       {
```

```
         if(customer.shoppingCart[cartIndex].product.title ==
product.title)
         {
           customer.shoppingCart[cartIndex].quantity -= 1;
           if(customer.shoppingCart[cartIndex].quantity == 0)
           {
             customer.shoppingCart.splice(cartIndex,1);
           }
           this.custServe.updateCustomer(user.uid,customer);
           return;
         }
       }
     });
   });
 }
```