



**Dharmsinh Desai University, Nadiad**

**Faculty of Technology**

**Department of Computer Engineering**

**B.Tech CE Semester- IV**

**Subject: Software Engineering Principle**

**Project title: Online Ice-Cream Shop**

**By:**

- 1. Vrundan Shah, Roll no:CE121, Id:19CEUOS141**
- 2. Jainil Trivedi, Roll no:CE138, Id:19CEUON061**

## **Contents:-**

1. Project Abstract.....	3
2. Introduction.....	4
3. Software Requirement Specification.....	5
4. Design Documents.....	14
5. Implements Details .....	25
6. Work Flow/Layouts .....	27
7. Conclusion .....	30
8. Limitations and Future Extensions .....	31
9. Bibliography.....	32

## **Abstract: -**

“Ice-Cream!!”

I bet your mouth started watering! Ice-cream is a very famous dessert which is favorite of many people around the world. Even when we didn't have that good of a food but having a good ice-cream will always make our meal memorable. During summer season when the temp. is too high, ice-cream will give a relief. Even if we want to eat ice-cream , we sometimes wish to get it sitting inside our home to avoid scorching heat of summer.

Now wouldn't it be great if we could order and get ice-cream delivered home with in few clicks on our cell phone?  
Our web-app helps you fulfill these wishes.

## **Introduction: -**

Online ice-cream shop is a website on internet which will help customer can browse and select ice-cream of their choice. The customer can change the quantity in the menu as well and will also be given recommendations. He can also search for his favorite ice-cream in the search bar. Before proceeding to payment, he will be shown all his ice-creams added to cart in the view cart function. He can change the quantity there also. He will be given few payment options so that he may select the one he prefers.

The main aim of online ice-cream shop is to make a general-purpose e-commerce store where many different flavors of ice-cream are made available to be bought from the comfort of home.

# **Software Requirement Specification**

## 1. Registration of user:

### R.1.1 Add user:

Description: The user input the details and for every successful registration message “Registered successfully” would be printed or not giving proper details message “Please enter valid details” would be displayed.

Input: details of the user

Output: Message registered successfully/ Please enter valid details

### R.1.2 Remove user/ Delete account:

Description: If something went wrong due to the customer’s fault and due to that the shop suffers the loss then the admin may remove the customer’s account or even ban it. Customer can delete his/her account as per their wish.

Input: Account to be deleted or banned

Output: Message account deleted successfully

### R.1.3 Update User Details:

Description: If User want to change some of the information entered previously then he can input the information to be updated and there would be message Information updated successfully

Input: Information to be Updated

Output: Message Information Updated successfully

## 2. Manage ice-cream Menu:

### R.2.1 Add ice-cream:

Description: Whenever there would be a new flavor of ice-cream available then the menu can be updated by adding details of the new flavor and message “ice cream added successfully” would be outputted.

Input: Details of ice-cream like cost, flavor and type.

Output: ice cream added successfully

### R.2.2 Remove ice cream:

Description: If the shop is no longer selling a particular ice cream, we can remove that particular ice-cream by entering its Name and message “Ice cream Removed!” would be outputted.

Input: Name of the ice-cream and its unique id

Output: That ice –cream will be deleted and a message “Ice cream Removed!”

### R.2.3 Modify item description:

Description: If there is some change in the flavor and details of the ice cream then these changes would be reflected by inputting the change to be processed and as the output the change would be visible on the website.

Input: New details of ice-cream

Output: New details of ice-cream seen on website

### R.2.4 Search in menu:

Description: Customers would be able to search the name of their favorite ice cream by entering the name of the ice-cream and in output the details of the ice-cream would be shown

Input: Name of ice-cream

Output: Ice-cream list or ice-cream details would be shown

### 3. Manage ice-cream Cart and Order:

#### R.3.1 Add to cart:

Description: Add ice-cream to cart (from cart all selected ice-creams will be shown before payment). It can be added to the cart by clicking on add to cart button.

Input: Click add to cart button

Output: Ice-cream added to the cart successfully.

#### R.3.2 View Cart:

Description: To see the items added in the cart and modify them if needed. Cart can be viewed by clicking on the View Cart button. And as the output all the items added in the cart previously would be shown.

Input: Click on the View cart button

Output: Cart items are shown



### R.3.3 History and reorder:

Description: Customer will be able to see the history of their orders and also able to reorder them. By clicking on View history button. As a output History (previous orders if any) would be shown.

Input: Click on the View history button

Output: History would be shown

### R.3.4 Place order:

Description: The customer will have a look at his/her order and select payment method. By clicking on place order the customer would be able to place order and select payment method.

Input: Selecting appropriate paying method and paying if needed

Output: Order placed and paid successfully

### R.3.5 Cancel order:

Description: Within 5 minutes form order only, the request to cancel order can be sent. It is also necessary that order was pick up and not home delivery.

Input: Click request cancelation of order

Output: Message showing if order has been canceled or not

#### R.3.6 Notify Customer:

Description: When order is accepted and when their order is ready (pick up mode) or when delivery guys has reached their location, notification will be sent.

Input: Notification message (E.g. Order accepted/Order ready/ Order delivered)

Output: Message sent successfully

### 4. Manage Payment:

#### R.4.1 Select Payment Method:

Description: Buyer is provided with multiple payments method like credit/debit card, net banking, 3rd party payment apps, Cash On Delivery

Input: Select a payment type from multiple options.

Output: Payment method selected successfully

#### R.4.2 Generate Invoice/Bill Receipt:

Description: Buyer can generate invoice for the products he bought and the payment he made.

Input: Click on 'generate bill' button.

Output: Invoice generated successfully

## 5. Feedback:

### R.5.1 Giving Feedback:

Description: User can give feedback for the order he placed and facilities provided by the shop. By typing his feedback in the feedback block or by rating the taste of the ice-cream and the service provided by the shop.

Input: User's suggestions

Output: Message "Thanks for your feedback"

### R.5.2 View Feedback:

Description: Before buying, the user can view the feedback submitted before.

Input: Type of feedback you want to view

Output: Feedbacks of similar kind

## 6. Viewing and analysis of database:

### R.6.1 See number of orders and its detail:

Description: Admin can see all the orders placed and check for most ice-cream ordered or total profit gained by the shop. Admin compare it with the previous year's analysis. He can also check for current month/years targets. Input: Time period in which analysis has to be done.

Output: Orders placed in that time period

### R.6.2 See total users:

Description: Admin can see total number of users or users registered in particular time period registered. To analyze the growth done on the online platform.

Input: Time period in which analysis is to be done.

Output: List of users registered

### R.6.3 Popular ice-cream:

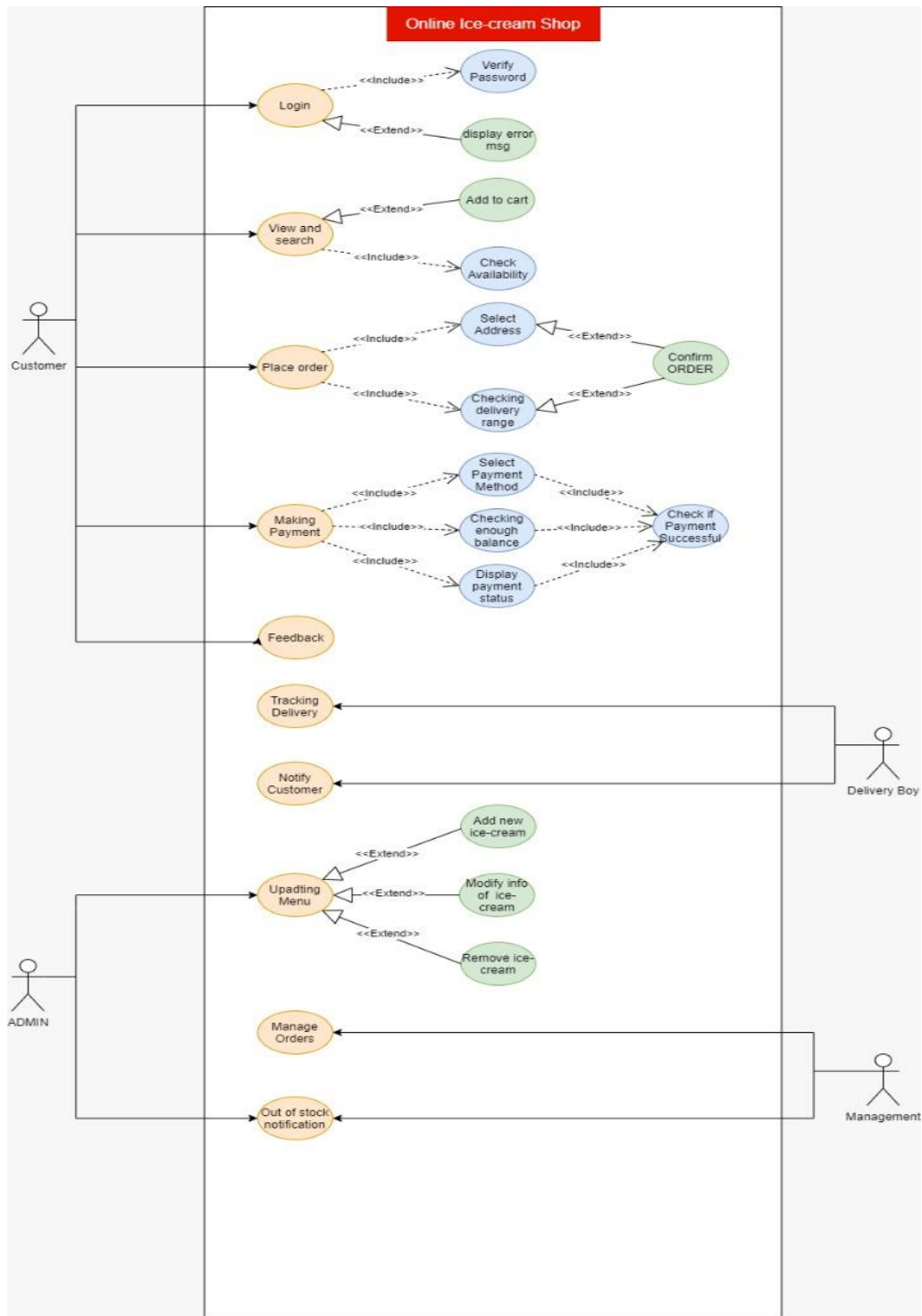
Description: User can see all the popular ice-cream of different flavors and types by clicking on the popular (most ordered) ice-creams list. And as output all the popular (most ordered) ice-cream would be displayed

Input: Time period in which analysis is to be done.

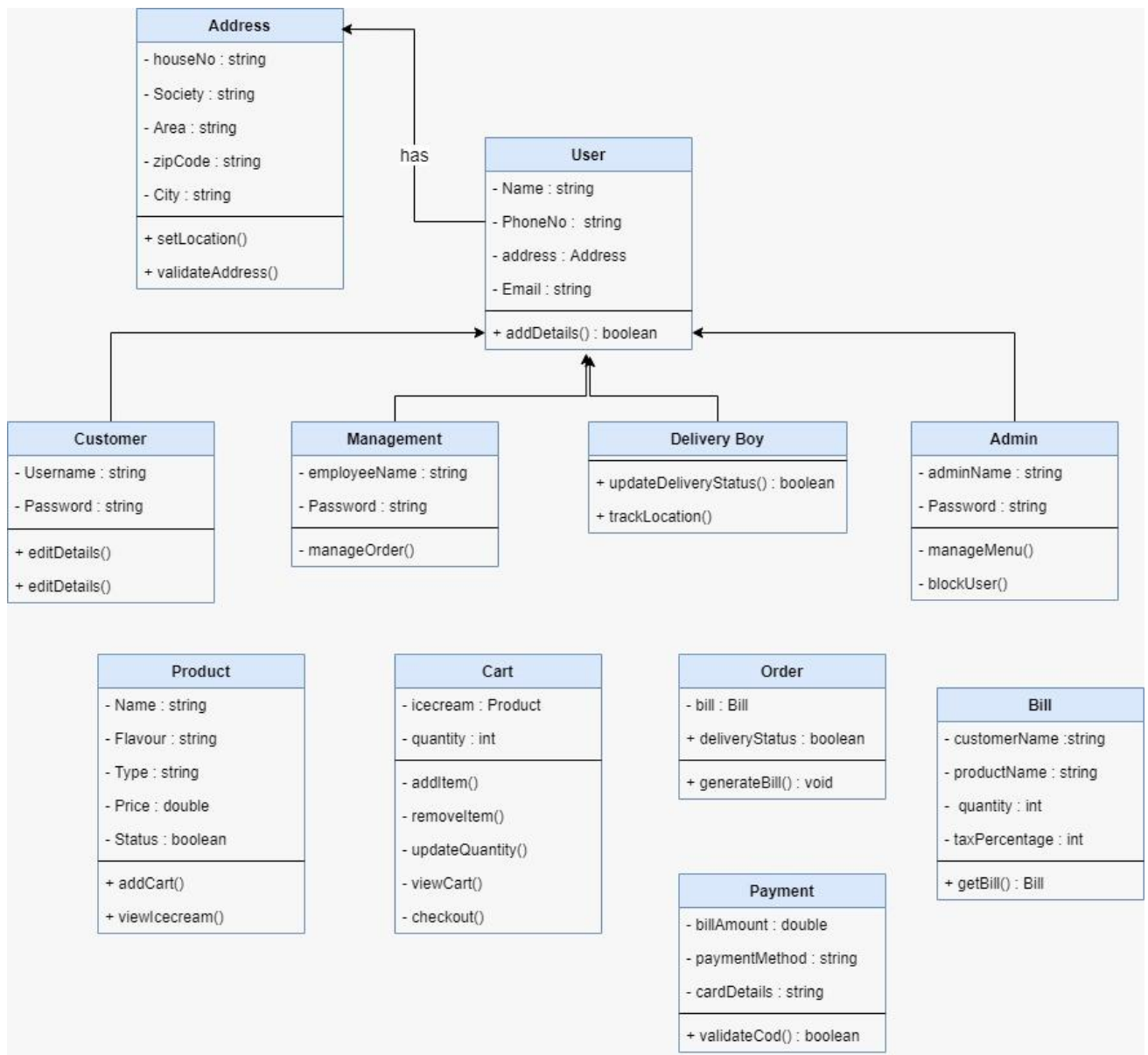
Output: Name and details of the ice-cream

# Design Document: -

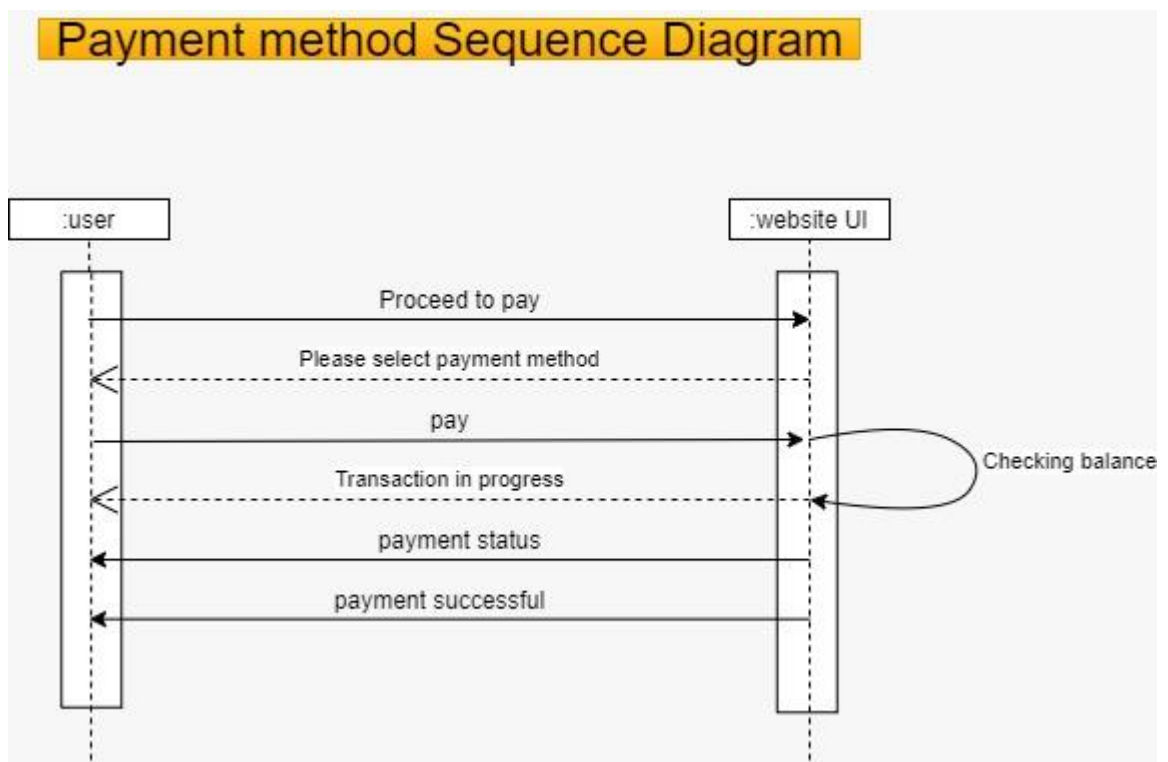
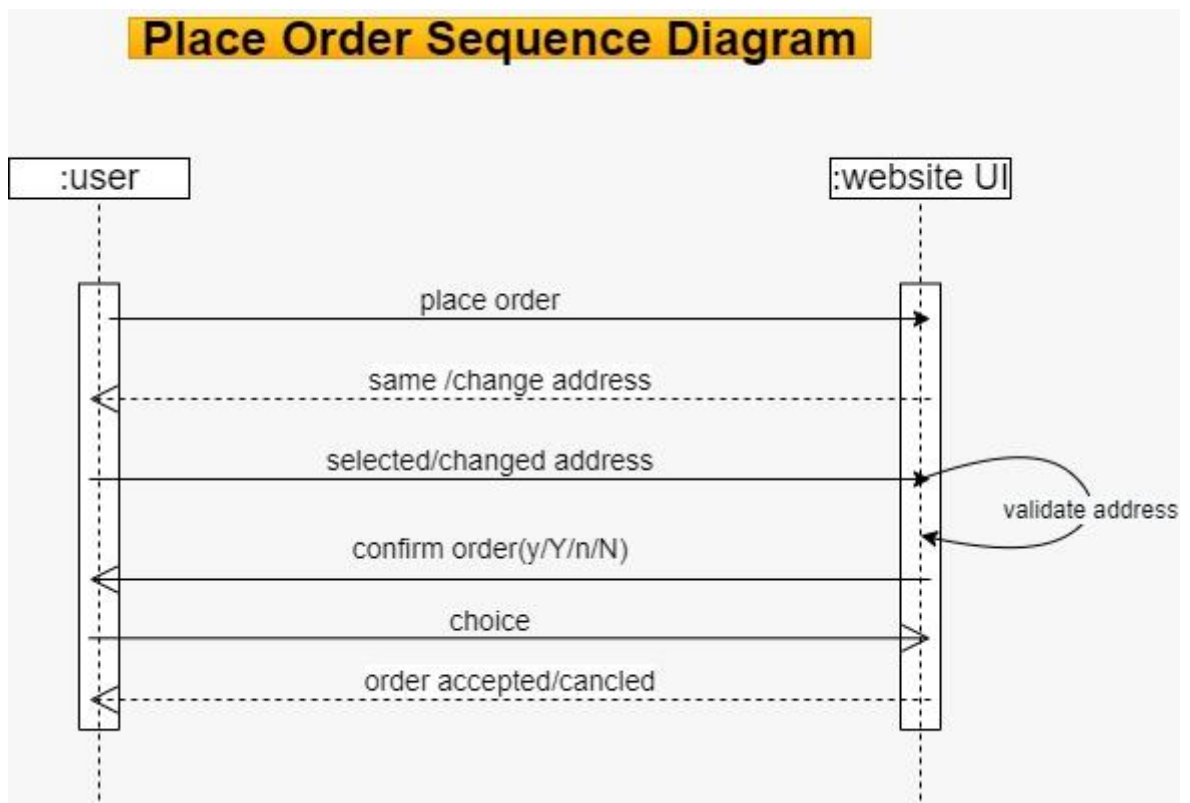
## 1. Use-Case Diagram:-



## 2. Class Diagram:-



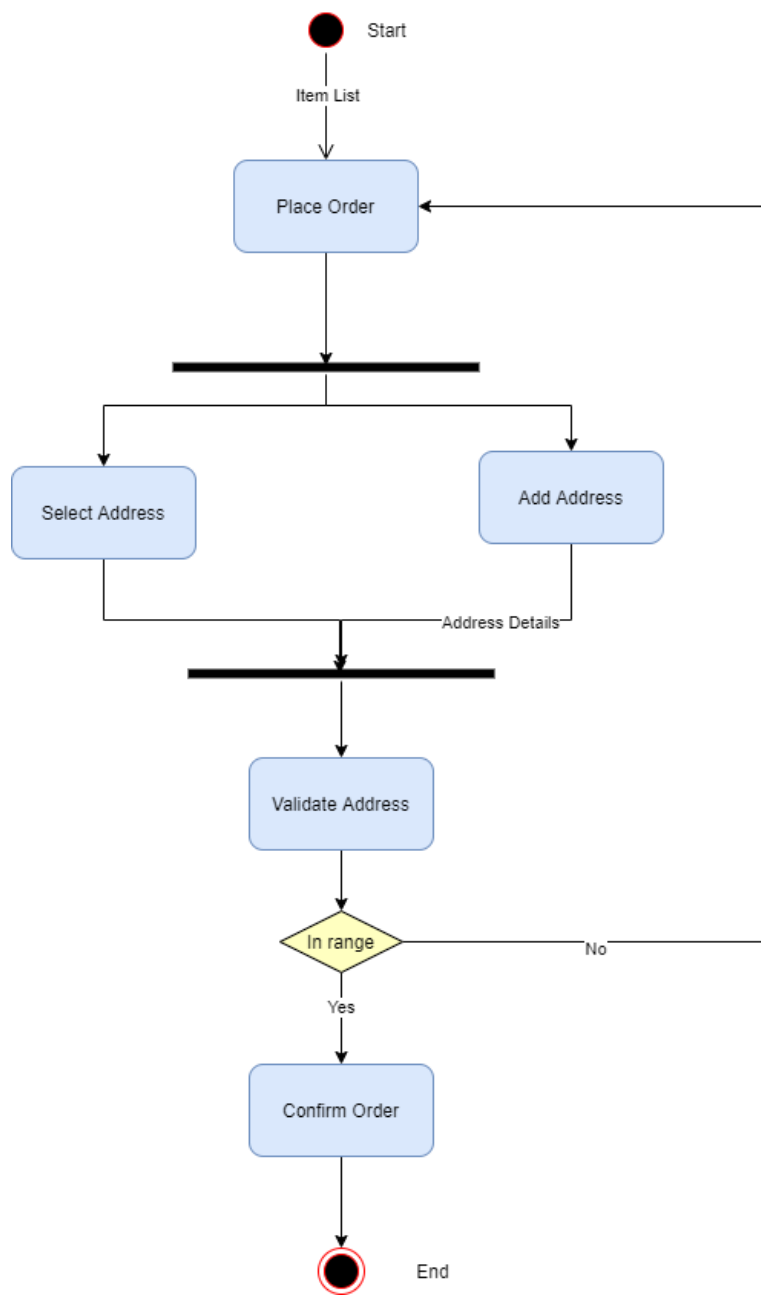
### 3. Sequence Diagram:-



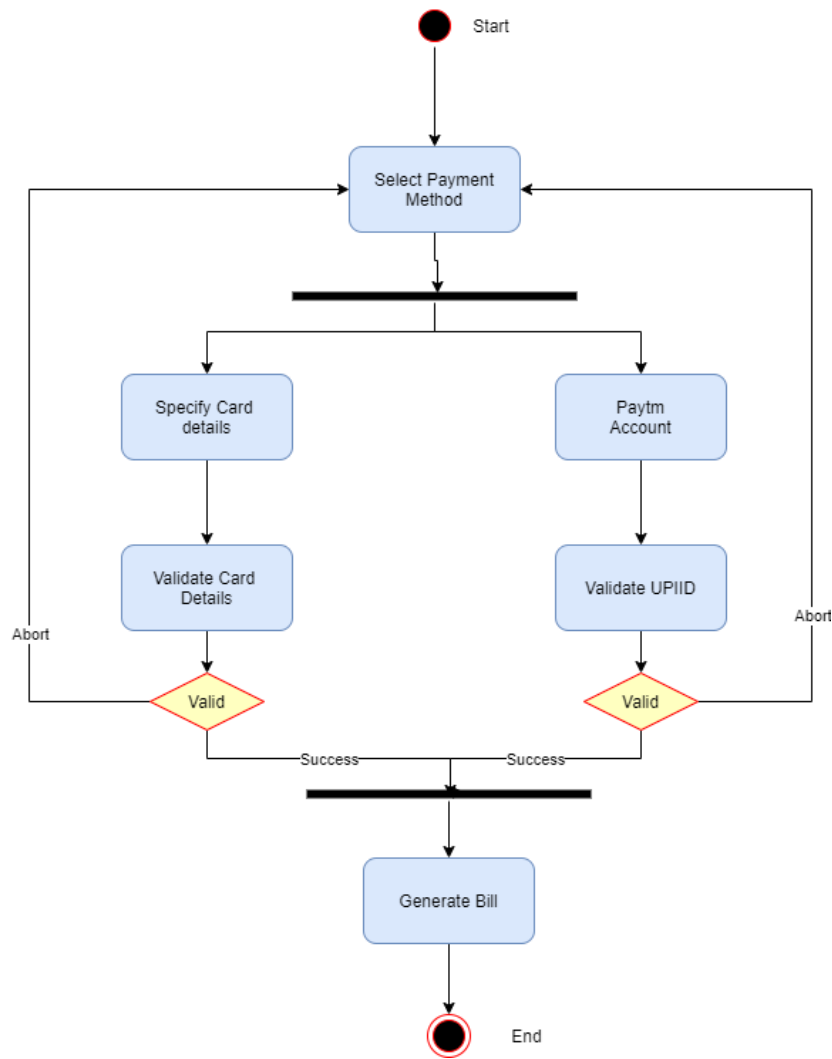


#### 4. Activity Diagram:-

Activity Diagram For Place Order

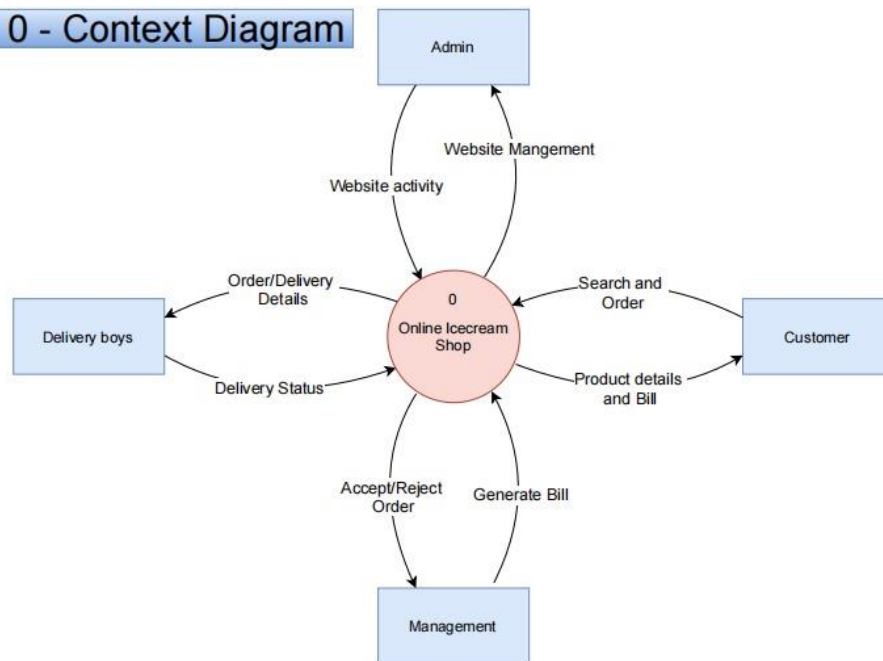


### Activity Diagram For Payment

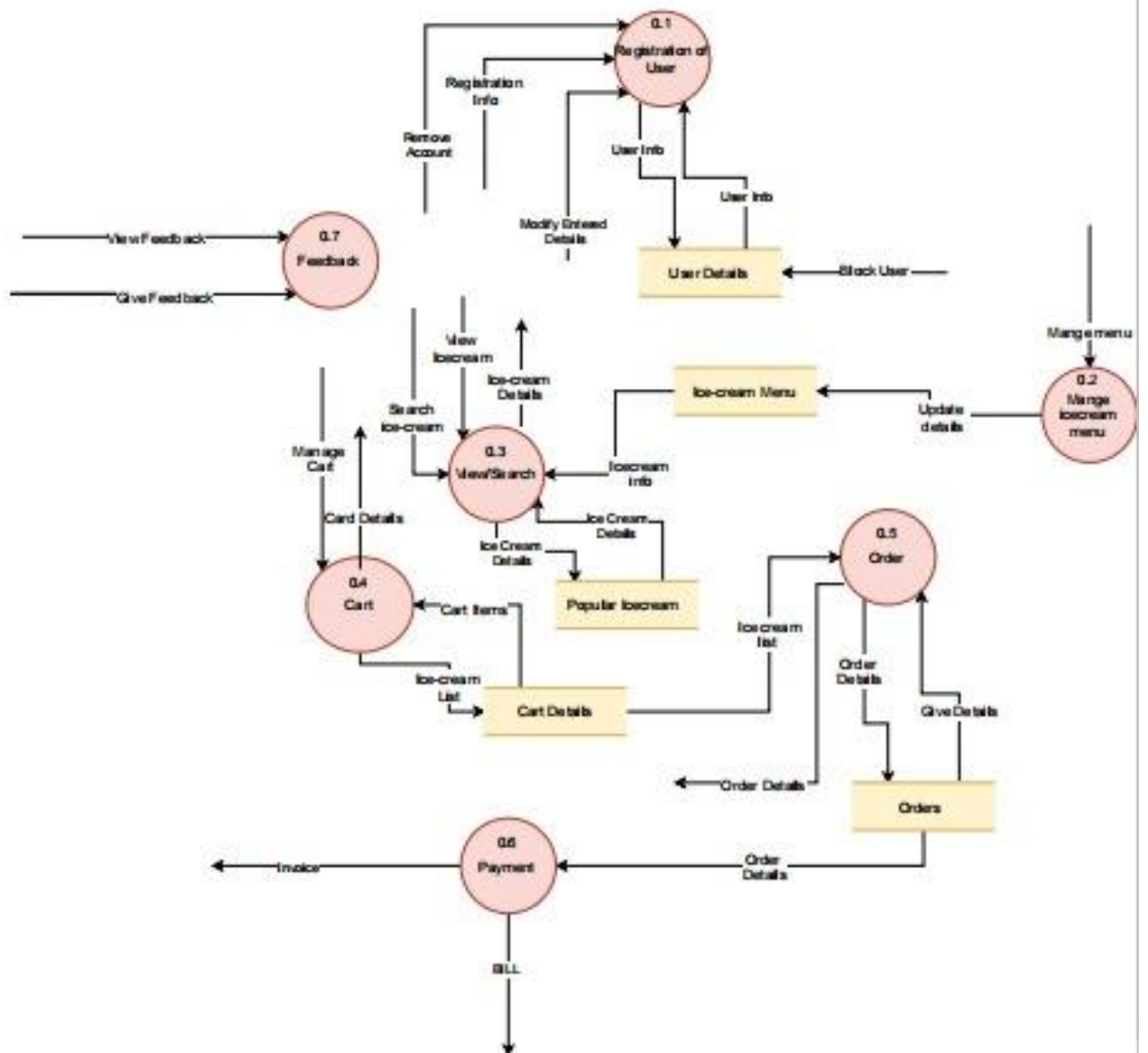


### 5. Data Flow Diagram:-

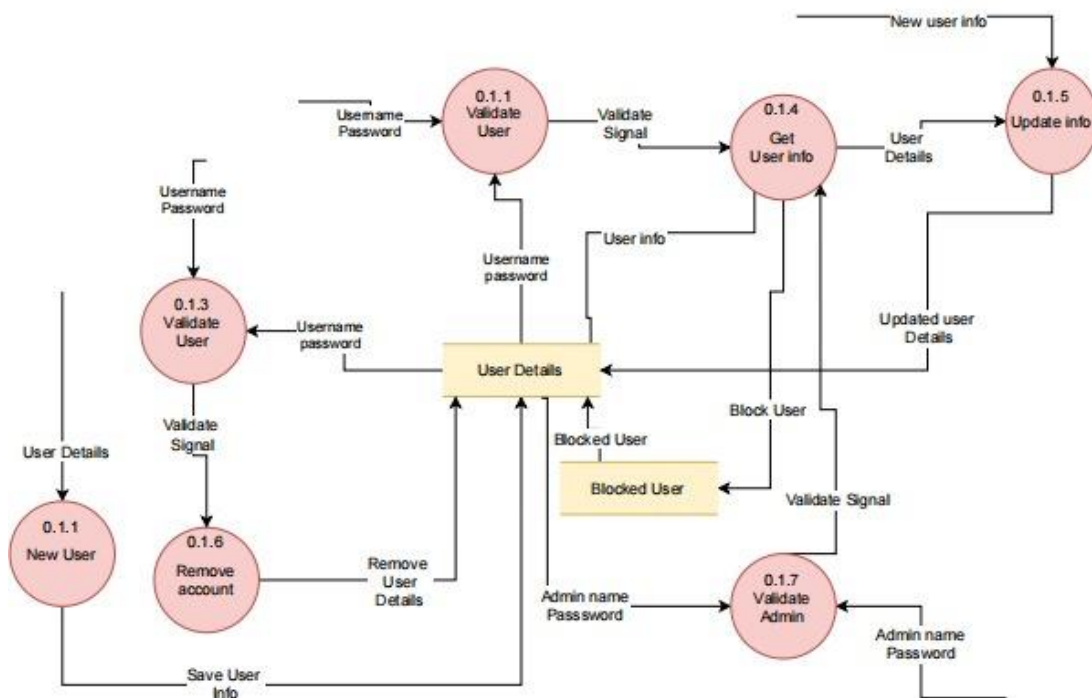
## Level 0 - Context Diagram



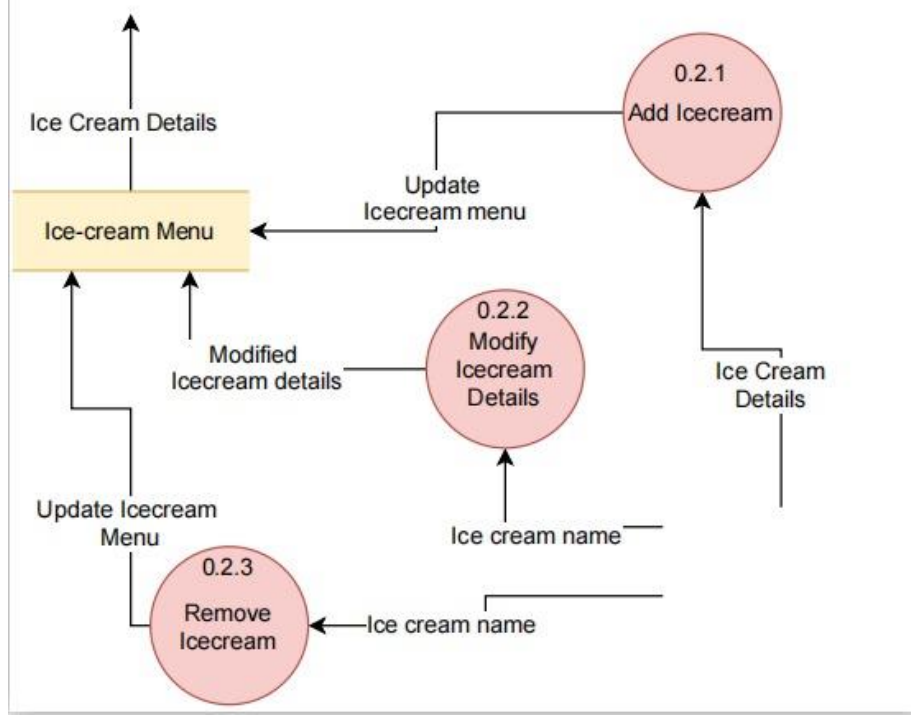
## Level-1 DFD



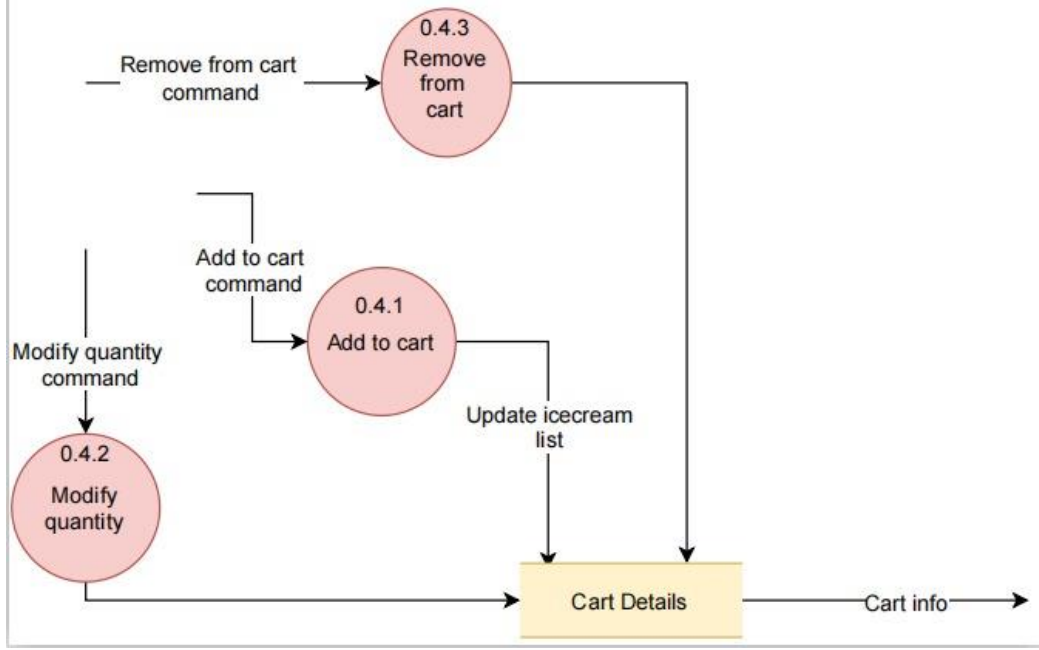
## Level-2 DFD (Decomposed 0.1)



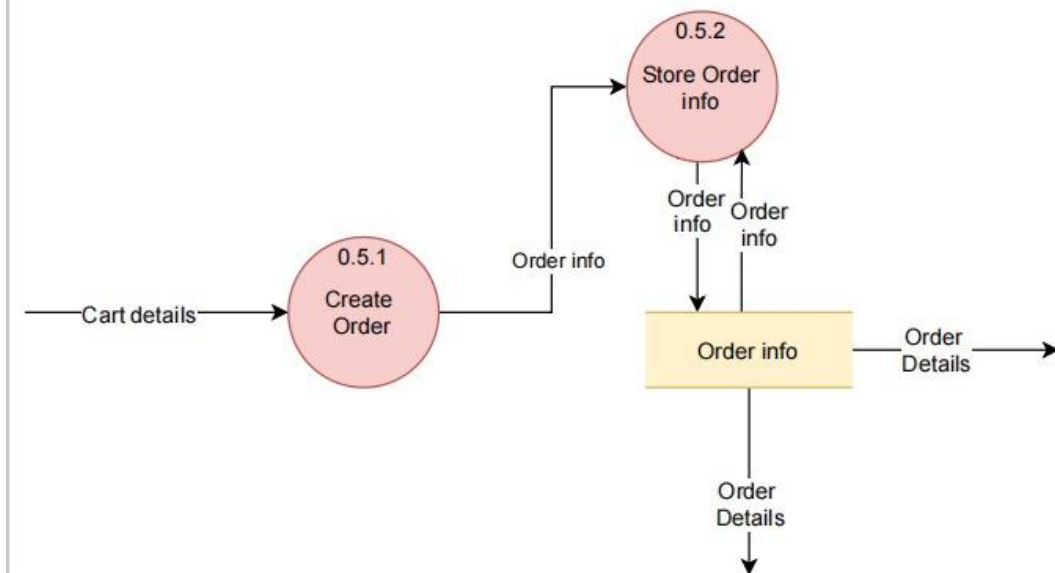
## Level-2 DFD (Decomposed 0.2)



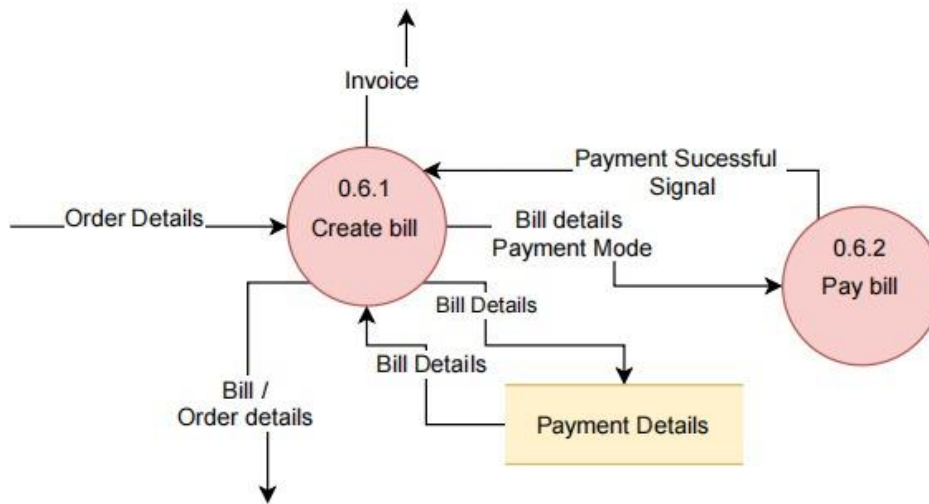
## Level-2 DFD (Decomposed 0.4)



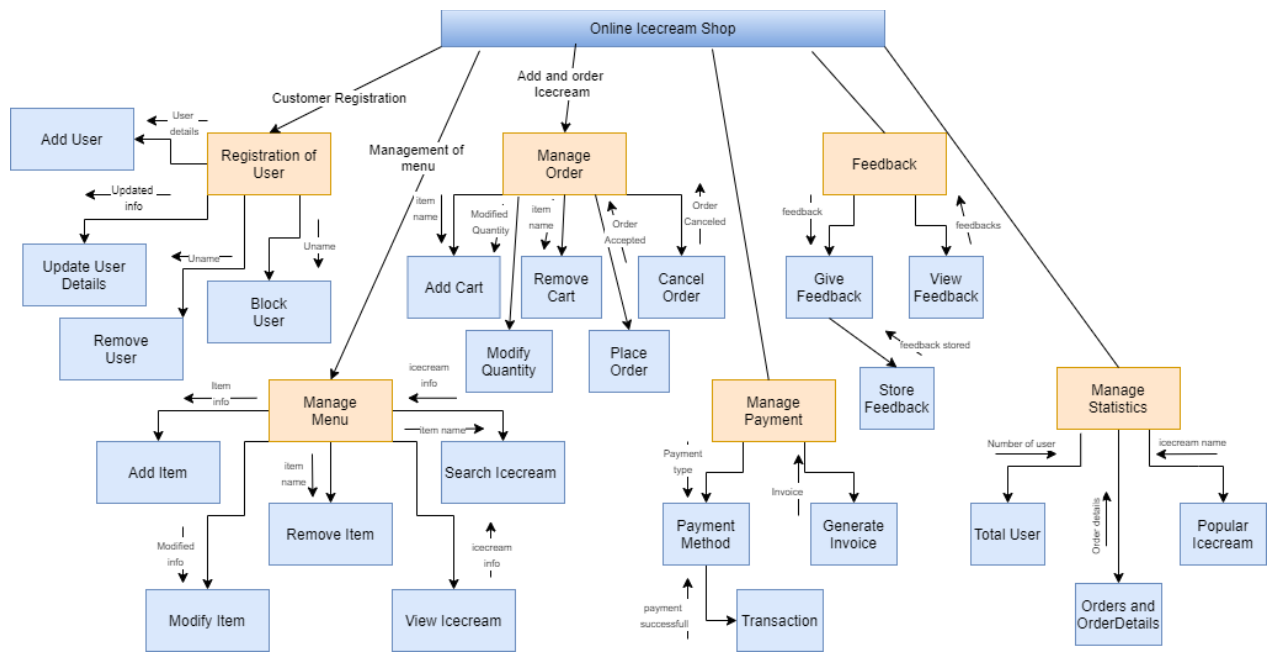
## Level-2 DFD (Decomposed 0.5)



## Level-2 DFD (Decomposed 0.6)



## **6. Structure Chart:-**





# Implementation Details:-

## Major Functionalities:

- **Login or register** : If the user is new customer he may register , old already registered customer may login

```
def log_in(request):
    if request.method == "POST":
        user = auth.authenticate(username=request.POST.get('username'),password=request.POST.get('password'))

        if user is not None:
            auth.login(request, user)
            return HttpResponseRedirect('/Menu/viewMenu/')
        else:
            return render(request,'log_in.html',{'error':"Invalid Username or password"})
    else:
        return render(request,'log_in.html')
```

- **View and search** : To find their favourite ice-cream by using their name(or even by using photos).

```
def search(request):
    if request.method == "POST":
        selectedId=request.POST.get('id')
        itemInCart=Product_details.objects.get(id=selectedId)
        try:
            curr_user = request.user
            alreadythere = cart.objects.get(name=itemInCart.product_name, user_id=curr_user.id)
            qty = int(request.POST.get('quantity'))
            if qty > 0:
                alreadythere.quantity = qty
                alreadythere.save()
            else:
                c = cart.objects.get(id = alreadythere.id)
                c.delete()

            except cart.DoesNotExist :
                carts=cart(name=itemInCart.product_name,price=itemInCart.price,total=0,quantity=1,img_link=itemInCart.im
                carts.save()
        products_cart = cart.objects.all()

        search = request.GET["search"]
        productName = Product_details.objects.filter(product_name__icontains = search)
        productType = Product_details.objects.filter(type__icontains = search)
        productFlavour = Product_details.objects.filter(flavour__icontains = search)
        products = productName.union(productType)
        products = products.union(productFlavour)

        return render(request,'filter.html',{'products':products,'products_cart':products_cart,'search':search})
```

- **Place order**: Can add the item to their cart that they are interested in.

- **Change Order:** Can remove the item from their cart that they are not interested in (in some particular time period and particular orders).


```
qty = int(request.POST.get('quantity'))
if qty > 0:
    alreadythere.quantity = qty
    alreadythere.save()
else:
    c = cart.objects.get(id = alreadythere.id)
    c.delete()
```

- **Select payment method :** Decide whether to pay online or COD


```
<h3>Please Select the Payment Type</h3><br>
<form action="/order/Credit_card/" method="POST">
    {% csrf_token %}
    <button type="submit" class="btn btn-primary">
        Proceed to pay using credit card
    </button><br><br>
</form>
<form action="/order/Paytm/" method="POST">
    {% csrf_token %}
    <button type="submit" class="btn btn-primary">
        Proceed to pay using Paytm
    </button>
</form>
```

# Work Flow/ Layouts: -

## 1. Login

 Ice-Cream

Login




### Log in

LOG IN

[Forgot Password](#)


[New User ? Click Here](#)

## 2. Menu


 Ice-Cream 

Flavours ▾ Type ▾

Search


Update Details | Feedback |  | Logout

### People's Choice




Chocobar  
Price : 20.00  


Add to cart



Golden Ice  
Price : 75.00  
Quantity :




Oreo Crunch  
Price : 75.00  
Quantity :




★ RocketStrike  
Price : 25.00  

Add to cart



★★ Chocobar  
Price : 20.00  

Add to cart




Mango Dolly  
Price : 20.00  


Add to cart

★ Speciality  
★★ Most Bought  
» Recommended


<https://classroom.google.com/h>

### 3. Cart

 Ice-Cream



Price: 75  
Quantity



Name: Golden Ice  
Price: 75  
Quantity

Place Order

### 4. Place Order

Name	Quantity	Price
Oreo Crunch	1	75
Golden Ice	3	75
Total Ice-Creams		4
Total price		300

Proceed to Pay

### 5. Proceed to pay


Please Select the Payment Type

Proceed to pay using credit card

Proceed to pay using Paytm

Payment Successfull

## 6. Feedback

 Ice-Cream

Feedbacks are always appreciated

Feedback

submit

## **Conclusion: -**

After the deployment of our website people will be able to buy/pickup for out from nearest shop in their city. It will be of great help to people in summer as we are providing home delivery of ice-cream so our customer doesn't have to come out in the scorching heat of summer to eat their favourite ice-cream. We have also added many payment methods so that customer doesn't have to undergo any trouble. Customer can also search their favourite ice-cream or flavour of their choice.

### **Limitations: -**

- Order Tracking not available.
- API has not been used at some places to improve the performance.
- Login Required Functionality Not added.

### **Future Extensions:-**

- Login Required functionality can be added to increase the security.
- APIs can be added to include some functionality for user convenience. Such as Order Tracking, etc.
- Update Details and Password resetting can be further improved.

## **Bibliography: -**

[https://www.youtube.com/watch?v=\\_uQrJ0TkZlc&t=20956s](https://www.youtube.com/watch?v=_uQrJ0TkZlc&t=20956s)

<https://www.youtube.com/watch?v=HlWKerBOBK4>

<https://django-filter.readthedocs.io/en/stable/>