School of Engineering and Applied Science, Ahmedabad University
BTech ICT Sem-4
Course: Database Management System

ONLINE FOOD DELIVERY SYSTEM

Muskan Matwani ---1741027
Naishi Shah ---1741033
Yesha Shastri ---1741035
Devshree Patel ---1741075

Description of the Project

The Online Food Delivery System(OFDS) is a medium for users to order food from a variety of restaurants, cafes and food outlets using one platform and with great ease. This system takes basic input from the users to facilitate its functionalities and to let the users order food with ease.

With this system, the users can order different types of food items in varying quantities and track their order till delivery. The payment for all orders is done via wallet. The user must be logged in to his/her account to avail any functionality. All the operations are done using the username or ID of the customer.

The **OFDS** has many features, which are listed below:

1. **Membership:** There are 2 types of memberships, regular and premium.
    a. Regular Member:
        i. Can get delivery from all the restaurants just like a premium member
        ii. Delivery fee based on the distance of the delivery address from the restaurant
        iii. Can avail discounts after successfully placing n orders.

    b. Premium Member:
        i. Can get delivery from all food joints associated with the system
        ii. No delivery fee on any order regardless of the distance between the delivery address and the food outlet
        iii. A certain amount is deducted from premium wallets as a fee.

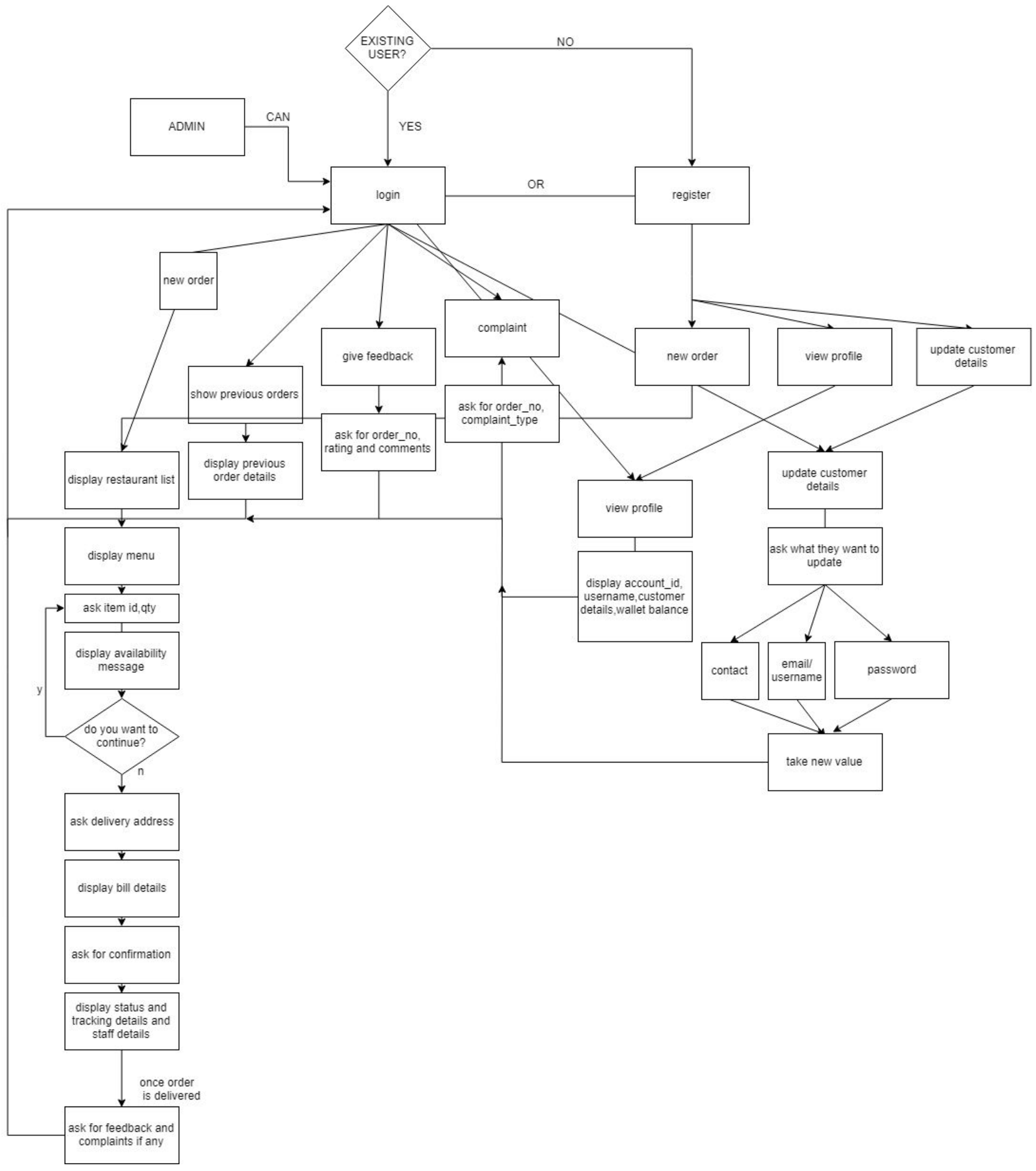
2. **Modes of access:** There are 2 modes of access,namely, administrator and customer. Both hold accounts but have different privileges and rights.
    a. Administrator:
        i. Can access order details and produce reports for a restaurant's performance, the number of orders it has received till date and the top n most active customers that the system hosts.
        ii. Cannot access a customer's personal profile and details

   b. Customer:
      i. Can access his/her own profiles, order details and can update his/her own profiles
      ii. Has no access to meta-level data

3. **Customer privileges:** Customers can do the following tasks:
   a. Place a new order:
      i. All customers can place any number of orders from any restaurants in the system.
      ii. They have to select a restaurant, choose items to be ordered with their respective quantities and provide the delivery address, then he/she can view the bill details and confirm the order.After this, the necessary details are displayed to the customer
   b. Feedbacks and complaints
      i. The customer can give feedback for specific orders and also file complaints if any.
   c. Previous orders
      i. Customers can view their order history

4. **New Registrations:** New customers can register by providing appropriate details for their account and they can then place order,if an already an old member then can view previous orders. Registrations are also open for administrators.

5. **Flow of the system:** The OFDS provides its services to its members only. Thus, you must login, if you have an account, or register to the system by providing imperative details. You can login either as a customer or as an administrator. Once logged in, the customer can perform the above mentioned tasks and similarly for the admin.

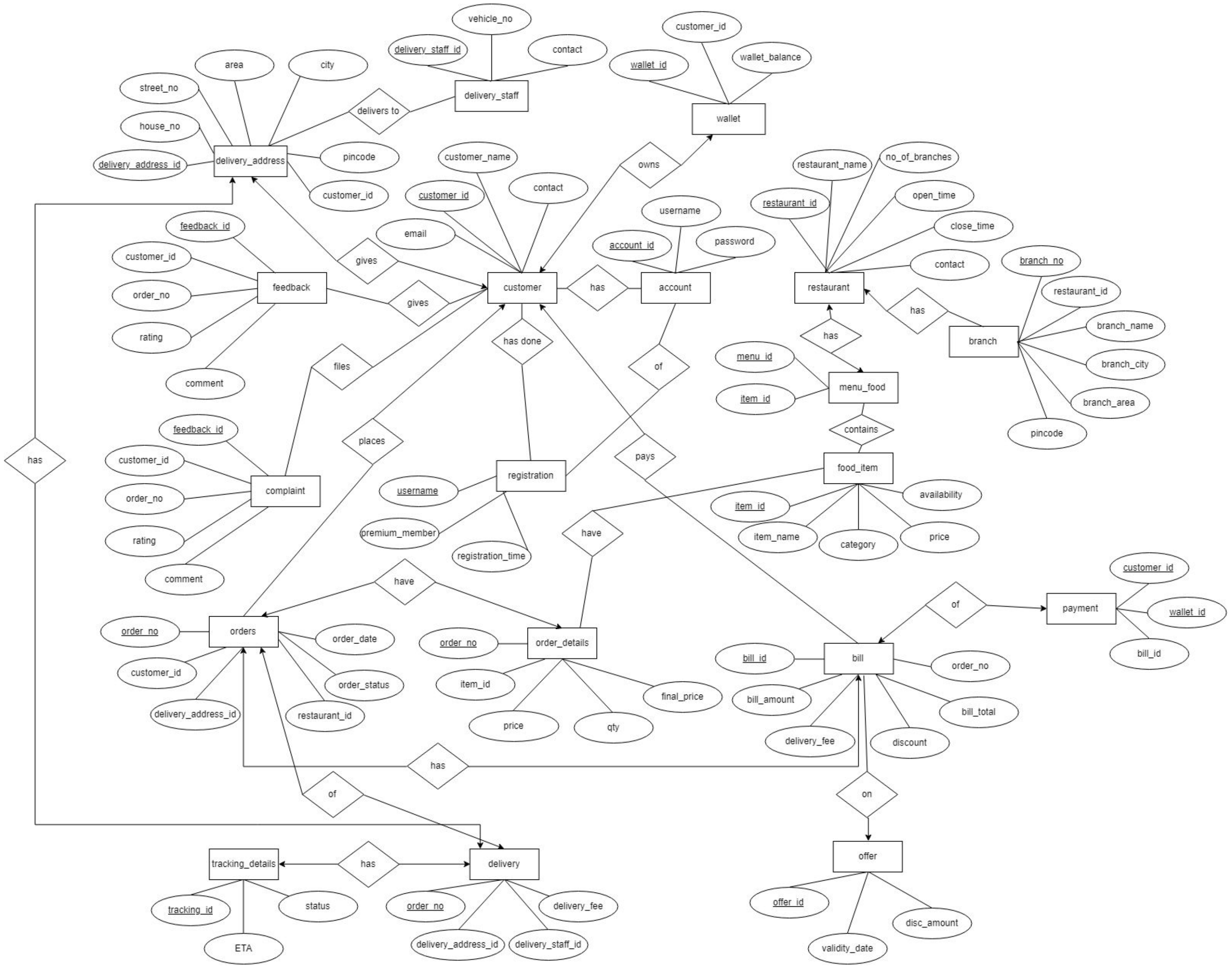


EXISTING USER?
NO
YES
ADMIN
CAN
login
OR
register
new order
show previous orders
give feedback
complaint
new order
view profile
update customer details
ask for order_no, complaint_type
ask for order_no, rating and comments
display restaurant list
display previous order details
view profile
update customer details
display menu
ask what they want to update
ask item id,qty
display account_id, username,customer details,wallet balance
display availability message
contact
email/ username
password
y
do you want to continue?
n
take new value
ask delivery address
display bill details
ask for confirmation
display status and tracking details and staff details
once order is delivered
ask for feedback and complaints if any

6. **Limitations:** This project has its own limitations;
    a. There is only one mode of payment i.e. wallet
    b. One restaurant can have only one menu
    c. ETA cannot be calculated corresponding to real-time data. Also, status cannot be updated based on ETA

7. **Assumptions**:
    a. The username will only be in the form of email.
    b. Some amount is assumed to be already present in the wallet at the time of registration (user need not add the balance).
    c. The wallet is assumed to be refilled automatically when there is not enough balance.
    d. Pincode is assumed as an area id wherein nearby areas will have small difference in the value of area id and vice versa.
    e. ETA is set to be minimum ten minutes and maximum 30 mins according to the distance from restaurant to the delivery address.
    f. Once placed, the order cannot be cancelled.
    g. Delivery fee will not exceed 30 rupees.
    h. Once a customer uses the given offer based on the n number of orders,the customer cannot be valid for the offer again after another n orders.
    i. Once a customer becomes a premium member there is no choice to revert the decision.
    j. Payment can be done only through the wallet. Payment through other modes like COD, credit card, net banking are not allowed.

ER diagram


vehicle_no
delivery_staff_id
contact
delivery_staff
customer_id
wallet_id
wallet_balance
wallet
area
city
street_no
house_no
delivers to
delivery_address_id
delivery_address
pincode
customer_id
owns
customer_name
customer_id
contact
restaurant_name
no_of_branches
open_time
close_time
restaurant_id
contact
restaurant
username
password
account_id
account
email
feedback_id
customer_id
order_no
rating
comment
feedback
gives
gives
customer
has
has done
has
branch_no
restaurant_id
branch_name
branch_city
branch_area
pincode
branch
has
menu_id
item_id
menu_food
contains
food_item
files
of
places
pays
feedback_id
customer_id
order_no
rating
comment
complaint
username
premium_member
registration_time
registration
item_id
item_name
category
price
availability
have
have
order_no
orders
order_date
order_status
customer_id
delivery_address_id
restaurant_id
order_no
order_details
item_id
price
qty
final_price
of
customer_id
payment
wallet_id
bill_id
bill_id
bill
order_no
bill_amount
delivery_fee
discount
bill_total
has
of
on
offer
offer_id
validity_date
disc_amount
tracking_details
tracking_id
ETA
status
has
delivery
order_no
delivery_fee
delivery_address_id
delivery_staff_id

Relational Schema

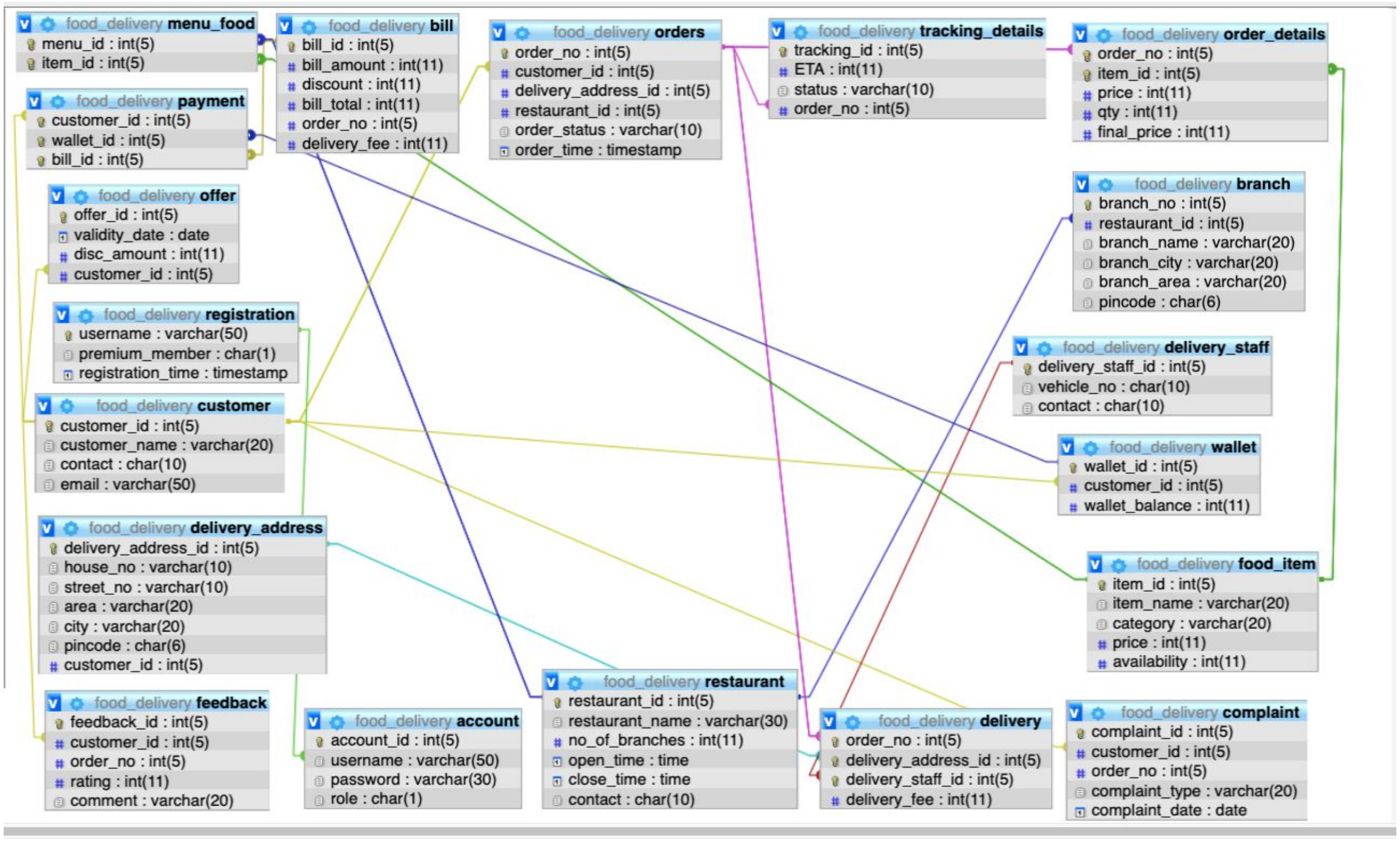


Table Design (Data Dictionary)

1. Table name: *customer*
   Description: This table will store customer details.

Fieldname	Data type	Size	Constraint	Description
customer_id	int	5	PK	Stores the customer ID
customer_name	varchar	20	-	Stores the name of the customer
contact	char	10	-	Stores the contact number of the customer
email	varchar	50	-	Stores the email ID

2. Table name: *orders*
   Description: This table will store order details like the restaurant from which food has been ordered, the customer ID, order status and the order date

Fieldname	Data type	Size	Constraint	Description
order_no	int	5	PK	Stores the order ID
customer_id	int	5	FK referred from customer	Stores the ID of the customer
delivery_address_id	int	5	-	Stores the delivery address ID of the customer
restaurant_id	int	5	-	Stores the restaurant ID
order_status	varchar	10	-	Shows the status of the order
order_time	timestamp	-	-	Stores date and time of the placed order

3. Table name: *account*
   Description: This table stores the account details of customer like his/her username, password and ID

Fieldname	**Data type**	**Size**	**Constraint**	**Description**
customer_id	int	5	PK	Stores the ID of the customer
username	varchar	50	FK referred from registration	Stores the username i.e. the email of the customer
password	varchar	30	-	Stores the password for this account

4. Table name: *delivery*
   Description: This table will store delivery details of orders

Fieldname	**Data type**	**Size**	**Constraint**	**Description**
order_no	int	5	PK, FK referred from orders	Stores the order ID
delivery_address_id	int	5	PK, FK referred from delivery address	Stores the delivery address ID of the customer
delivery_staff_id	int	5	PK, FK referred from delivery staff	Stores the delivery staff ID of the delivery person
delivery_fee	int	11	-	Stores the delivery fee on a delivery

5. Table name: *registration*
   Description: This table stores the registration details of customers like username, premium membership and the registration time

Fieldname	Data type	Size	Constraint	Description
username	varchar	50	PK	Stores the username i.e. the email of the customer
premium_member	char	1	-	Stores y if he/she is a premium member, n otherwise
registration_time	timestamp	-		Stores the time of registration

6. Table name: *restaurant*
   Description: This table stores the restaurant details

Fieldname	Data type	Size	Constraint	Description
restaurant_id	int	5	PK	Stores the restaurant ID
restaurant_name	varchar	30	-	Stores the restaurant name
no_of_branches	int	11	-	Stores the number of branches of a particular restaurant
open_time	time	-	-	Stores the opening time of the restaurant
close_time	time	-	-	Stores the closing time of the restaurant
contact	char	10	-	Stores the contact number of the restaurant

7. Table name: *bill*
   Description: This table stores the bill details of orders

Fieldname	**Data type**	**Size**	**Constraint**	**Description**
bill_id	int	5	PK	Stores the bill ID
bill_amount	int	11	-	Stores the bill amount
delivery_fee	int	11	-	Stores the delivery fee on an order
discount	int	11	-	Stores the discount on an order
bill_total	int	11	-	Stores the total amount of the bill after discount
order_no	int	5	-	Stores the order number of the order corresponding to the bill

8. Table name: *order_details*
   Description: This table stores the order details specifying the items of that order

Fieldname	**Data type**	**Size**	**Constraint**	**Description**
order_no	int	5	PK, FK referred from orders	Stores the order number
item_id	int	5	PK, FK referred from food_item	Stores the item ID
price	float	-	-	Stores the price of the item to be ordered
qty	int	11	-	Stores the quantity of the item to be ordered
final_price	float	-	-	Stores the final price of

				the item corresponding to the quantity

9. Table name: *branch*
   Description: This table stores the branch details of a restaurant

Fieldname	**Data type**	**Size**	**Constraint**	**Description**
branch_no	int	5	PK	Stores the branch number
restaurant_id	int	5	FK referred from restaurant	Stores the restaurant ID
branch_name	varchar	20	-	Stores the branch name
branch_city	varchar	20	-	Stores the branch city
branch_area	varchar	20	-	Stores the area where a particular branch is located
pincode	char	6		Stores the pincode of that branch

10. Table name: *delivery_address*
    Description: This table stores the delivery address details for an order delivery

Fieldname	**Data type**	**Size**	**Constraint**	**Description**
delivery_address_id	int	5	PK	Stores the delivery address ID
house_no	varchar	10	-	Stores the house number

street_no	varchar	10	-	Stores the street number
area	varchar	20	-	Stores the area of the address
city	varchar	20	-	Stores the city of the address
pincode	char	6	-	Stores the pincode of the address
customer_id	int	5	-	Stores the customer ID whose address is being stored

11. Table name:*delivery_staff*
    Description: This table stores the details of the delivery staff

Fieldname	**Data type**	**Size**	**Constraint**	**Description**
delivery_staff_id	int	5	PK	Stores the ID of the delivery staff
vehicle_no	char	10	-	Stores the registration number of the vehicle used
contact	char	10	-	Stores the contact number of the staff member

12. Table name: *payment*
    Description: This table stores the payment details corresponding to a particular bill

Fieldname	**Data type**	**Size**	**Constraint**	**Description**
customer_id	int	5	PK,FK referred from customer	Stores the customer ID of the customer making the payment
wallet_id	int	5	PK, FK	Stores the wallet ID of

			referred from wallet	the customer making the payment
bill_id	int	5	PK, FK referred from bill	Stores the bill ID of an order

13. Table name: *wallet*
    Description: This table stores the wallet details corresponding to a particular customer account

Fieldname	Data type	Size	Constraint	Description
wallet_id	int	5	PK	Stores the wallet ID of the customer
customer_id	int	5	FK referred from customer	Stores the customer ID of the customer
wallet_balance	int	11	-	Stores the wallet balance of the respective wallet

14. Table name: *food_item*
    Description: This table stores the food item details

Fieldname	Data type	Size	Constraint	Description
item_id	int	5	PK	Stores the item ID
item_name	varchar	20	-	Stores the item name
category	varchar	20	-	Stores the category of the item
price	int	11	-	Stores the price of the item
availability	int	11	-	Stores the availability of the item

15. Table name: *menu_food*
Description: This table stores the menu details corresponding to a particular restaurant

Fieldname	Data type	Size	Constraint	Description
menu_id	int	5	PK, FK referred from restaurant	Stores the menu ID
item_id	int	5	PK, FK referred from food_item	Stores the item ID

16. Table name: *feedback*
Description: This table stores the feedback details

Fieldname	Data type	Size	Constraint	Description
feedback_id	int	5	PK	Stores the feedback ID
customer_id	int	5	FK referred from customer	Stores the customer ID of the customer giving the feedback
order_no	int	5	-	Stores the order number on which feedback is given
rating	int	11	-	Stores the rating for the feedback
comment	varchar	20	-	Stores the comments on the order

17. Table name: *complaint*
Description: This table stores the complaints pertaining to a particular order

Fieldname	Data type	Size	Constraint	Description
complaint_id	int	5	PK	Stores the complaint ID

customer_id	int	5	FK referred from customer	Stores the customer ID of the customer
order_no	int	5	-	Stores the order number
complaint_type	varchar	20	-	Stores the complaint type
complaint_date	date	-	-	Stores the complaint date

18. Table name: *tracking_details*
    Description: This table stores the wallet details corresponding to a particular customer account

Fieldname	**Data type**	**Size**	**Constraint**	**Description**
tracking_id	int	5	PK	Stores the tracking ID of the order
ETA	time	-	-	Stores the estimated time of arrival of the delivery
status	varchar	10	-	Stores the status of the order delivery
order_no	int	5	FK referred from orders	Stores the order number corresponding to the tracking details

19. Table name: *offer*
    Description: This table stores the wallet details corresponding to a particular customer account

Fieldname	**Data type**	**Size**	**Constraint**	**Description**
offer_id	int	5	PK	Stores the offer ID
validity_date	date	-	-	Stores the validity date of the offer

disc_amount	int	11	-	Stores the discount amount of the offer
customer_id	int	5	FK referred from customer	Stores the customer ID of the customer who avails the offer

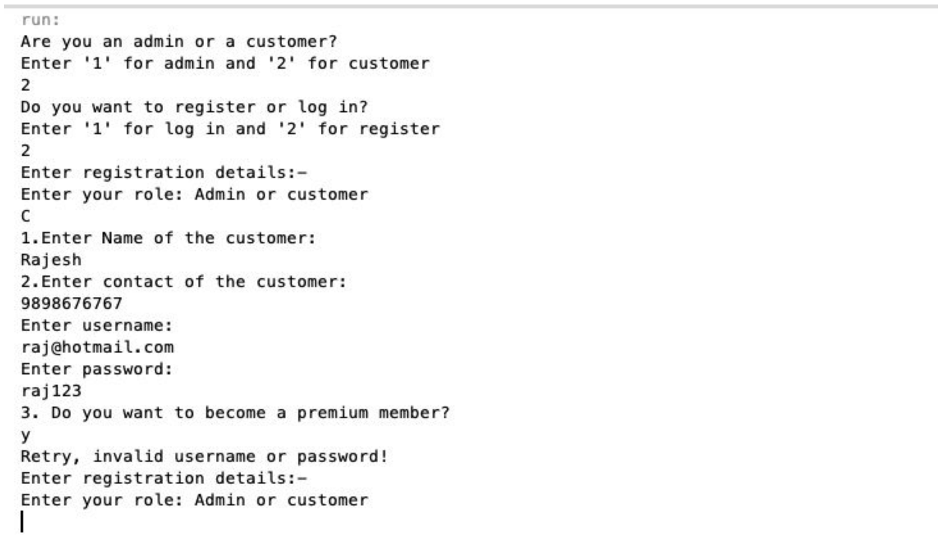
Procedures

1. **Check validity of username:** It checks for the proper format of entered username for both user and admin.

Code:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `chk_user`(IN `uname`
VARCHAR(50), OUT `flag` INT)
BEGIN
        DECLARE pos int;
   DECLARE sub varchar(50);
    DECLARE b int;
   DECLARE a varchar(50);
    DECLARE cur2 CURSOR FOR SELECT username1 FROM utable;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET b = 1;
        If (POSITION("@" IN uname) >0 or POSITION("." IN uname) >0 or POSITION("_" IN
uname) >0) THEN
        SET pos= POSITION("@" IN uname);
        SET sub= SUBSTR(uname,pos+1);
        OPEN cur2;
        SET b = 0;
        WHILE b = 0 DO
        FETCH cur2 INTO a;
            if (a=sub) THEN
                set flag=1;
            end if;
        END WHILE;
        CLOSE cur2;
    END if;

END$$
DELIMITER ;
```

Screenshot: displays invalid when done registration with invalid constraints of username



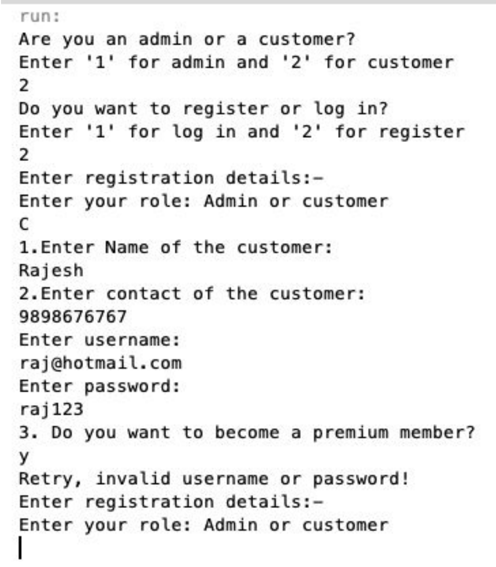
```
run:
Are you an admin or a customer?
Enter '1' for admin and '2' for customer
2
Do you want to register or log in?
Enter '1' for log in and '2' for register
2
Enter registration details:-
Enter your role: Admin or customer
C
1.Enter Name of the customer:
Rajesh
2.Enter contact of the customer:
9898676767
Enter username:
raj@hotmail.com
Enter password:
raj123
3. Do you want to become a premium member?
y
Retry, invalid username or password!
Enter registration details:-
Enter your role: Admin or customer
```

2. **Check validity of password:** It checks for the proper format of entered password for both user and admin.

Code:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `password_check`(IN `pass`
VARCHAR(30), OUT `flag` INT)
BEGIN
        if(length(pass)>=8)
        then
                if(~((strcmp(upper(pass),pass)) and (strcmp(lower(pass),pass))))
                then
                        if((pass like '%$%') or (pass like '%@%') or (pass like '%_%') or (pass
like '%&%') or (pass like '%*%') or (pass like '%#%') or (pass like '%%%') or (pass like
'%^%') or (pass like "%'%") or (pass like '%"%')) then
                                set flag = 1;
                        else
                                set flag = 0;
                end if;
                else
                set flag = 0;
                end if;
        else
                set flag = 0;
        end if;
END$$
DELIMITER ;
```

Screenshot: displays invalid when done registration with invalid constraints of password



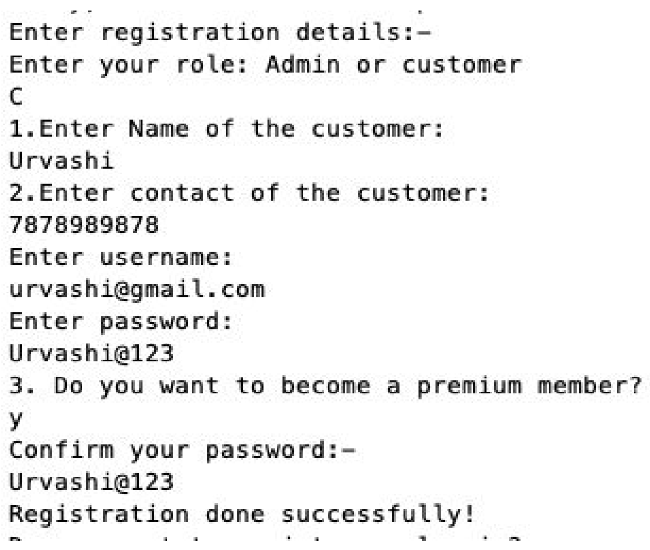
```
run:
Are you an admin or a customer?
Enter '1' for admin and '2' for customer
2
Do you want to register or log in?
Enter '1' for log in and '2' for register
2
Enter registration details:-
Enter your role: Admin or customer
C
1.Enter Name of the customer:
Rajesh
2.Enter contact of the customer:
9898676767
Enter username:
raj@hotmail.com
Enter password:
raj123
3. Do you want to become a premium member?
y
Retry, invalid username or password!
Enter registration details:-
Enter your role: Admin or customer
```

3. **Registration:** As soon as a customer registers, the details are inserted into the registration,customer and account tables. Also, the initial wallet balance is set in the wallet table.

Code:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `register1`(IN `username`
VARCHAR(50), IN `name` VARCHAR(20), IN `password1` VARCHAR(20), IN
`phone` CHAR(10), IN `premium_member1` CHAR(1), IN `role1` CHAR(1))
BEGIN
        DECLARE id int(5);
        INSERT INTO registration(username, premium_member,registration_time)
VALUES (username, premium_member1,CURRENT_TIMESTAMP);

        INSERT INTO customer(customer_name, contact, email) VALUES (name,
phone, username);

        SET id = (select customer.customer_id from customer where
customer.email=username);
        INSERT INTO wallet(customer_id,wallet_balance) values(id,100000);

        INSERT INTO account(account.username, account.password, account.role)
VALUES (username, password1, role1);

END$$
DELIMITER ;
```

Screenshot: Enter details into registration, account, customer and sets a minimum balance for wallet balance of that particular customer.



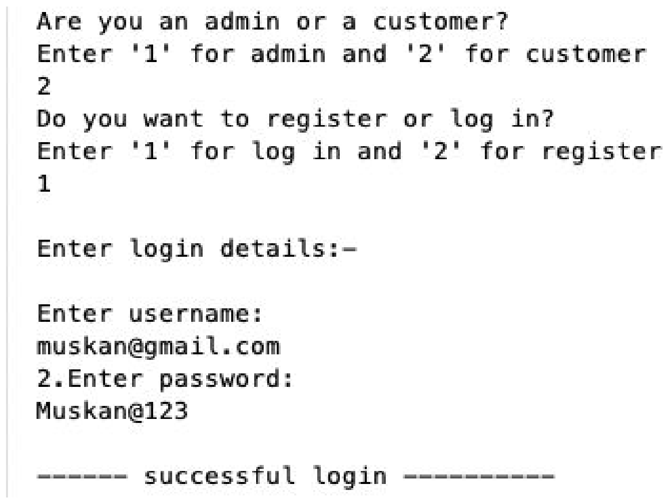
```
Enter registration details:-
Enter your role: Admin or customer
C
1.Enter Name of the customer:
Urvashi
2.Enter contact of the customer:
7878989878
Enter username:
urvashi@gmail.com
Enter password:
Urvashi@123
3. Do you want to become a premium member?
y
Confirm your password:-
Urvashi@123
Registration done successfully!
```

4. **User Login:** This procedure checks whether the entered username and password of user exist or not.

Code:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `user_login`(IN `username1`
VARCHAR(50), IN `pass` VARCHAR(30), OUT `flag` INT)
BEGIN
	DECLARE b int;
	DECLARE a varchar(50);
	   DECLARE c varchar(30);
	   DECLARE cur1 CURSOR FOR SELECT account.username,account.password
FROM account where role='C';
	DECLARE CONTINUE HANDLER FOR NOT FOUND SET b = 1;
	SET b = 0;
	SET flag = 0;
	OPEN cur1;
	FETCH cur1 INTO a,c;
	WHILE b = 0 DO
	if (a=username1) and (c=pass) then
	                    set flag = 1;

	      end if;
	FETCH cur1 INTO a,c;
	END WHILE;
	CLOSE cur1;
END$$
DELIMITER ;
```

Screenshot: It checks whether the username and password matches with the registered details and lets the user log in if it is correct.



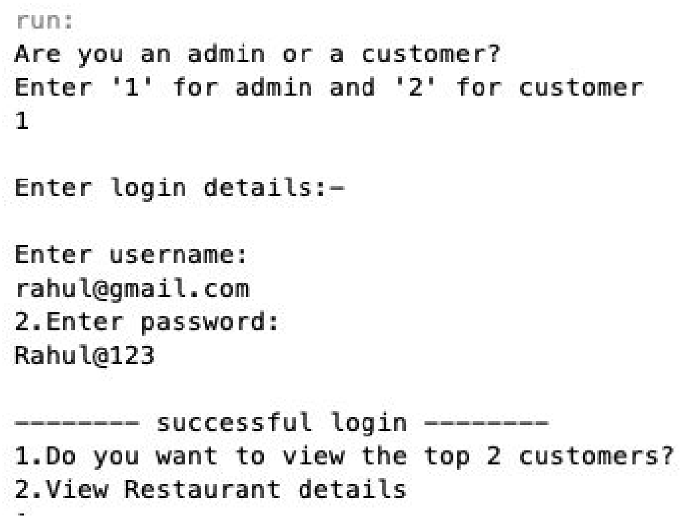
```
Are you an admin or a customer?
Enter '1' for admin and '2' for customer
2
Do you want to register or log in?
Enter '1' for log in and '2' for register
1

Enter login details:-

Enter username:
muskan@gmail.com
2.Enter password:
Muskan@123

------ successful login ----------
```

5. **Admin Login:** Checks whether the username and password of admin exist or not.

Code:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `admin_login`(IN `username1`
VARCHAR(50), IN `pass` VARCHAR(30), OUT `flag` INT)
BEGIN
        DECLARE b int;
        DECLARE a varchar(50);
           DECLARE c varchar(30);
           DECLARE cur1 CURSOR FOR SELECT account.username,account.password
FROM account where role='A';
        DECLARE CONTINUE HANDLER FOR NOT FOUND SET b = 1;
        SET b = 0;
        SET flag = 0;
        OPEN cur1;
        FETCH cur1 INTO a,c;
        WHILE b = 0 DO
        if (a=username1) and (c=pass) then
                            set flag = 1;

                end if;
        FETCH cur1 INTO a,c;
        END WHILE;
        CLOSE cur1;
END$$
DELIMITER ;
```

Screenshot: It checks whether the username and password matches with the registered details and lets the admin log in if it is correct.



```
run:
Are you an admin or a customer?
Enter '1' for admin and '2' for customer
1

Enter login details:-

Enter username:
rahul@gmail.com
2.Enter password:
Rahul@123

-------- successful login --------
1.Do you want to view the top 2 customers?
2.View Restaurant details
```

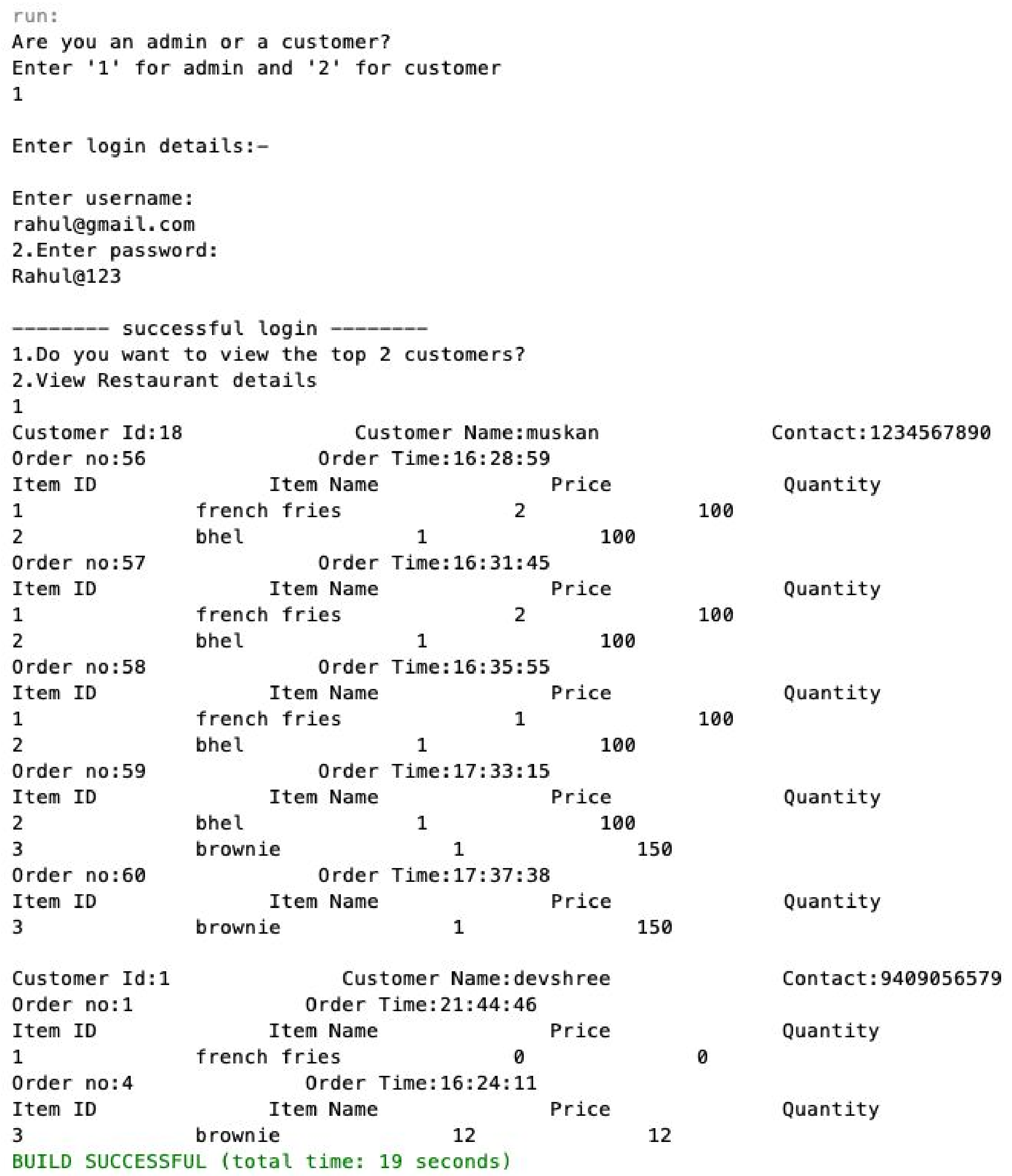
6. **Display details of the top 2 customers:** It displays order details of top 2 customers to admin based on the number of orders.

Code:

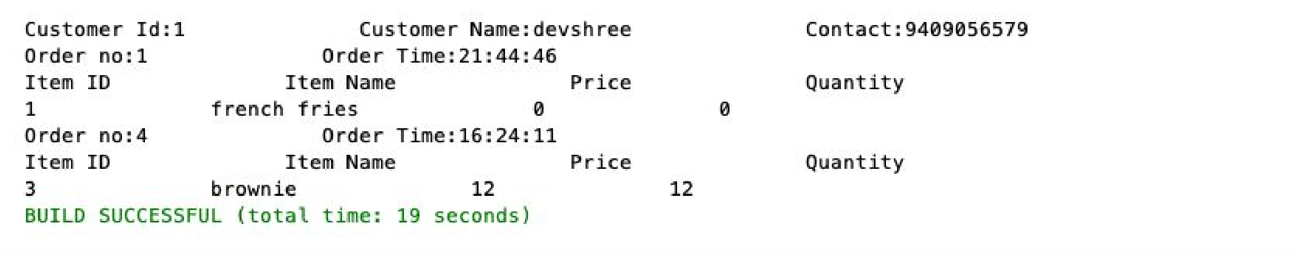
```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `top_2`()
BEGIN
    DECLARE b,r,c,qty,i,d int;
    DECLARE temp_cust_id int(5);
    DECLARE a varchar(10);
    DECLARE name varchar(20);
    DECLARE temp_order_no int(5);
    DECLARE cust_name varchar(20);
    DECLARE contact char(10);
    DECLARE order_time date;
    DECLARE price,bill_amt,disc,final_price int(11);

    BLOCK1: BEGIN
    DECLARE cur3 cursor for SELECT customer_id from orders group by customer_id order
by count(order_no) desc limit 5;
    DECLARE continue handler for not found set r = 1;
    open cur3;
         set r = 0;
    fetch cur3 into c;
    WHILE r = 0 DO
          set temp_cust_id = c;
         set cust_name = (select customer.customer_name from customer where
customer_id=temp_cust_id limit 1);
         set contact = (select customer.contact from customer where
customer_id=temp_cust_id limit 1);

         BLOCK2: BEGIN
         DECLARE cur1 CURSOR FOR SELECT order_no from orders where
orders.customer_id = temp_cust_id;
       DECLARE CONTINUE HANDLER FOR NOT FOUND SET b = 1;
       SET b = 0;
       OPEN cur1;
       SET b = 0;
       FETCH cur1 INTO a;
       WHILE b = 0 DO
                 set temp_order_no = a;
```

```
                    set order_time = (select orders.order_date from orders where order_no =
temp_order_no);

                    set bill_amt = (select bill.bill_amount from bill where order_no =
temp_order_no);

                    set disc = (select bill.discount from bill where order_no = temp_order_no);

                    set final_price = (select bill.bill_total from bill where order_no =
temp_order_no);


                    BLOCK3: BEGIN
                    DECLARE cur2 CURSOR FOR SELECT item_id from order_details where
order_details.order_no = temp_order_no;
                    DECLARE CONTINUE HANDLER FOR NOT FOUND SET c = 1;
                    SET c = 0;
          OPEN cur2;
          SET c = 0;
          FETCH cur2 INTO d;
                    WHILE c = 0 DO
                              set name = (select food_item.item_name from food_item where
item_id=d);
                              set qty = (select order_details.qty from order_details where
order_no=temp_order_no and item_id = d);
                              set price = (select order_details.price from order_details where
order_no=temp_order_no and item_id = d);

                              select
temp_cust_id,cust_name,contact,temp_order_no,order_time,d,name,qty,price,bill_amt,disc,f
inal_price;

                    fetch cur2 into d;
                    END WHILE;
                    CLOSE cur2;
                    END BLOCK3;

       FETCH cur1 INTO a;
       END WHILE;
       CLOSE cur1;
       END BLOCK2;

    fetch cur3 into c;
    END WHILE;
     CLOSE cur3;
     END BLOCK1;

END$$
DELIMITER ;
```

Screenshot: It displays top 2 customer details including name, contact and ID and their order details of each order previously ordered. It also displays item details of each order.



```
run:
Are you an admin or a customer?
Enter '1' for admin and '2' for customer
1

Enter login details:-

Enter username:
rahul@gmail.com
2.Enter password:
Rahul@123

-------- successful login --------
1.Do you want to view the top 2 customers?
2.View Restaurant details
1
Customer Id:18              Customer Name:muskan              Contact:1234567890
Order no:56             Order Time:16:28:59
Item ID             Item Name             Price             Quantity
1             french fries             2             100
2             bhel             1             100
Order no:57             Order Time:16:31:45
Item ID             Item Name             Price             Quantity
1             french fries             2             100
2             bhel             1             100
Order no:58             Order Time:16:35:55
Item ID             Item Name             Price             Quantity
1             french fries             1             100
2             bhel             1             100
Order no:59             Order Time:17:33:15
Item ID             Item Name             Price             Quantity
2             bhel             1             100
3             brownie             1             150
Order no:60             Order Time:17:37:38
Item ID             Item Name             Price             Quantity
3             brownie             1             150

Customer Id:1              Customer Name:devshree              Contact:9409056579
Order no:1             Order Time:21:44:46
Item ID             Item Name             Price             Quantity
1             french fries             0             0
Order no:4             Order Time:16:24:11
Item ID             Item Name             Price             Quantity
3             brownie             12             12
BUILD SUCCESSFUL (total time: 19 seconds)
```



```
Customer Id:1              Customer Name:devshree              Contact:9409056579
Order no:1              Order Time:21:44:46
Item ID              Item Name              Price              Quantity
1              french fries              0              0
Order no:4              Order Time:16:24:11
Item ID              Item Name              Price              Quantity
3              brownie              12              12
BUILD SUCCESSFUL (total time: 19 seconds)
```

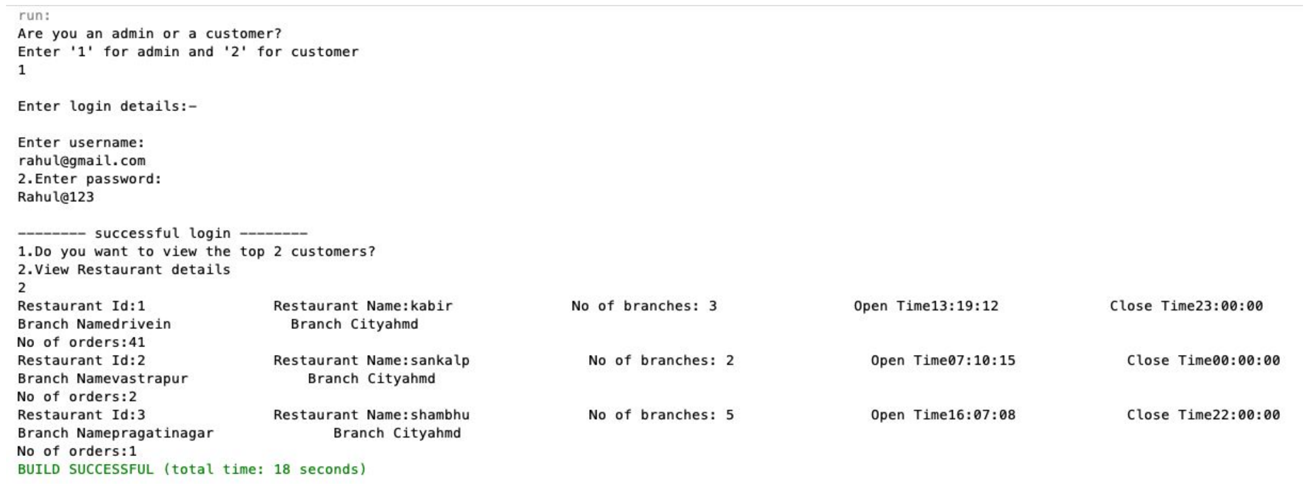
7. **Display restaurant details:** It displays the details of the restaurants to the admin.

Code:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `restaurant_details3`()
BEGIN
    DECLARE a int;
    DECLARE z int;
    DECLARE order_N int;
    DECLARE temp_order_no int;
    DECLARE temp_rest_id int;
    DECLARE b,r int;
    DECLARE c varchar(10);
    DECLARE name varchar(20);
    DECLARE city varchar(20);
    DECLARE temp_name varchar(20);
    DECLARE temp_city varchar(20);
    DECLARE d int;
    DECLARE e,f time;
    DECLARE g char(10);

    BLOCK1: BEGIN
    DECLARE cur1 cursor for select restaurant_id from restaurant;
    DECLARE continue handler for not found set b = 1;
    open cur1;
    set b=0;
    set r=0;
    set z=0;
    fetch cur1 into a;
    WHILE b = 0 DO
         set temp_rest_id = a;
         set c = (select restaurant_name from restaurant where restaurant_id=temp_rest_id);

         set d = (select no_of_branches from restaurant where restaurant_id=temp_rest_id);

      set e = (select open_time from restaurant where restaurant_id = temp_rest_id);

      set f = (select close_time from restaurant where restaurant_id = temp_rest_id);

      set g = (select contact from restaurant where restaurant_id = temp_rest_id);
```

```
        BLOCK2: BEGIN
        DECLARE cur2 cursor FOR SELECT branch_name,branch_city from branch where
branch.restaurant_id = temp_rest_id;
        DECLARE CONTINUE HANDLER FOR NOT FOUND SET z = 1;
     open cur2;
     set z=0;
     fetch cur2 into name,city;
     WHILE z = 0 DO
             set temp_name = name;
     set temp_city = city;


        BLOCK3: BEGIN
           DECLARE cur3 CURSOR FOR SELECT count(order_no) from orders where
orders.restaurant_id = temp_rest_id;
        DECLARE CONTINUE HANDLER FOR NOT FOUND SET r = 1;
        SET r = 0;
        OPEN cur3;
        SET r = 0;
        FETCH cur3 INTO order_N;
        WHILE r = 0 DO
                    set temp_order_no = order_N;
           select temp_rest_id ,c ,d ,e ,f ,g ,temp_name ,temp_city ,temp_order_no;
                    FETCH cur3 INTO order_N;
        end while;
        CLOSE cur3;
        END BLOCK3;

     FETCH cur2 INTO name, city;
        end while;
        CLOSE cur2;
        END BLOCK2;


           FETCH cur1 into a;
    end while;
           CLOSE cur1;
           END BLOCK1;
END$$
DELIMITER ;
```

Screenshot: It displays all the restaurant details with descending order of number of orders ordered in that particular restaurant. Like if 1st restaurant has got the highest number of orders till date, it will be displayed first.



```
run:
Are you an admin or a customer?
Enter '1' for admin and '2' for customer
1

Enter login details:-

Enter username:
rahul@gmail.com
2.Enter password:
Rahul@123

-------- successful login --------
1.Do you want to view the top 2 customers?
2.View Restaurant details
2
Restaurant Id:1              Restaurant Name:kabir              No of branches: 3                Open Time13:19:12             Close Time23:00:00
Branch Namedrivein             Branch Cityahmd
No of orders:41
Restaurant Id:2              Restaurant Name:sankalp              No of branches: 2                Open Time07:10:15             Close Time00:00:00
Branch Namevastrapur             Branch Cityahmd
No of orders:2
Restaurant Id:3              Restaurant Name:shambhu              No of branches: 5                Open Time16:07:08             Close Time22:00:00
Branch Namepragatinagar             Branch Cityahmd
No of orders:1
BUILD SUCCESSFUL (total time: 18 seconds)
```

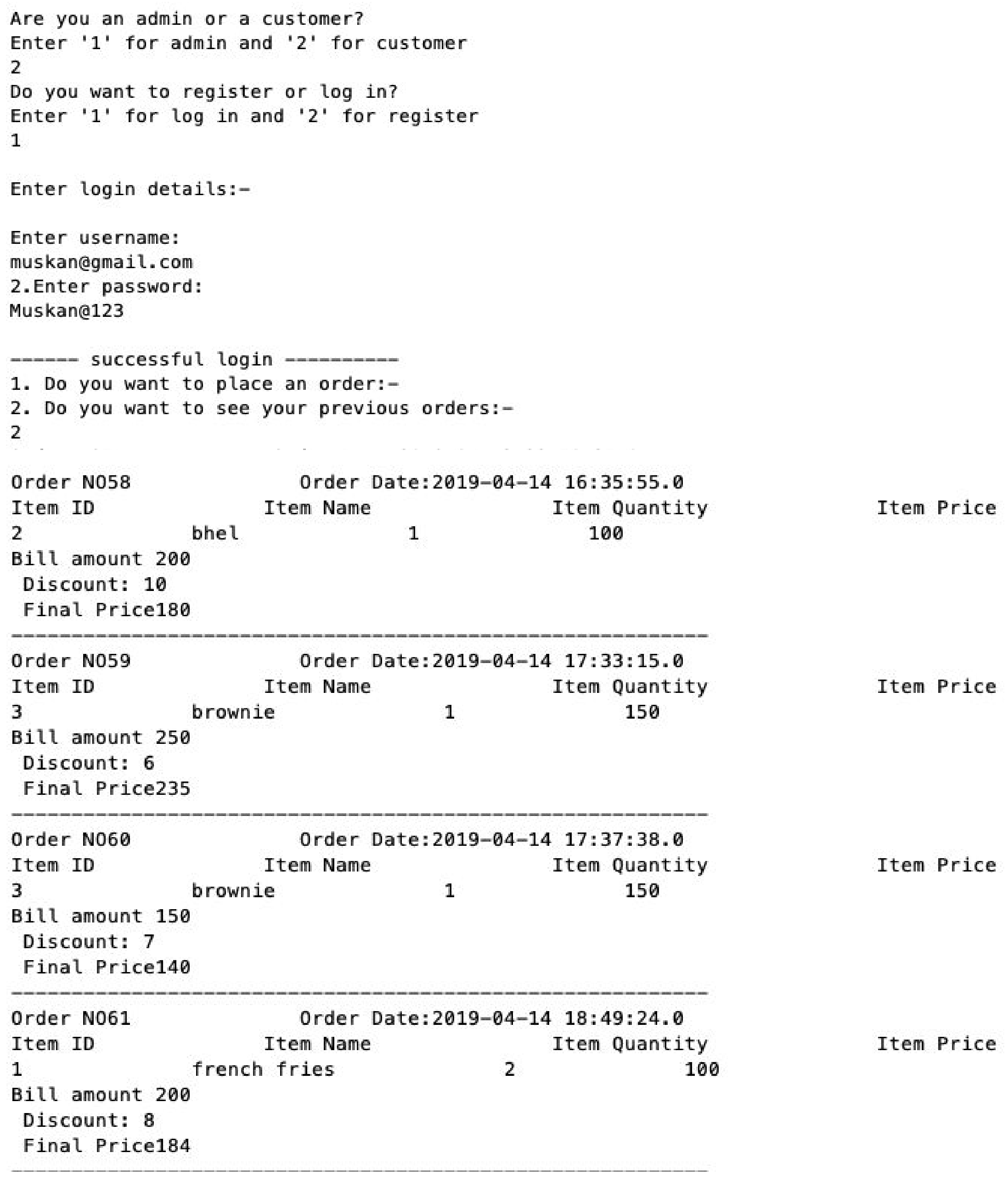
8. **Display order history:** It displays all the previous order details to the user.

Code:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `display`(IN `customer_id` INT)
BEGIN
        DECLARE name varchar(30);
        DECLARE b int;
   DECLARE qty int;
   DECLARE a int;
        DECLARE c int;
   DECLARE d int;
   DECLARE price1 int(11);
   DECLARE order_time TIMESTAMP;
   DECLARE bill_amt,disc,final_price int;
   DECLARE temp_item_id int;
        DECLARE temp_order_no int;
        BLOCK1: BEGIN
        DECLARE cur1 CURSOR FOR SELECT orders.order_no from orders where
orders.customer_id = customer_id;
      DECLARE CONTINUE HANDLER FOR NOT FOUND SET b = 1;
      OPEN cur1;
      SET b = 0;
      FETCH cur1 INTO a;
      WHILE b = 0 DO
                set temp_order_no = a;

                set order_time = (select orders.order_time from orders where
orders.order_no = temp_order_no);


                BLOCK2: BEGIN
                DECLARE cur2 CURSOR FOR SELECT order_details.item_id from
order_details where order_details.order_no = temp_order_no;
                DECLARE CONTINUE HANDLER FOR NOT FOUND SET c = 1;
        OPEN cur2;
        SET c = 0;
        FETCH cur2 INTO d;
                WHILE c = 0 DO
                        set temp_item_id = d;
                        set name = (select food_item.item_name from food_item where
food_item.item_id=d);
```

```
                                set qty = (select order_details.qty from order_details where
order_details.order_no=temp_order_no and item_id = d);

                                set price1 = (select price from order_details where
order_details.order_no=temp_order_no and item_id = d);



                      FETCH cur2 into d;
                      END WHILE;
                      CLOSE cur2;
                      END BLOCK2;

                      set bill_amt = (select bill.bill_amount from bill where bill.order_no =
temp_order_no limit 1);

                      set disc = (select bill.discount from bill where bill.order_no = temp_order_no
limit 1);

                      set final_price = (select bill.bill_total from bill where bill.order_no =
temp_order_no limit 1);

               select temp_order_no as order_no,order_time,temp_item_id as item_id,
name,qty,price1 as Price,bill_amt as bill_amount,disc as discount,final_price;

       FETCH cur1 INTO a;
       END WHILE;
       CLOSE cur1;
          END BLOCK1;
END$$
DELIMITER ;
```

Screenshot: It displays previous order details of that particular customer. For example: here user ‘muskan’ can view its previous orders as shown.



```
Are you an admin or a customer?
Enter '1' for admin and '2' for customer
2
Do you want to register or log in?
Enter '1' for log in and '2' for register
1

Enter login details:-

Enter username:
muskan@gmail.com
2.Enter password:
Muskan@123

------ successful login ----------
1. Do you want to place an order:-
2. Do you want to see your previous orders:-
2

Order NO58             Order Date:2019-04-14 16:35:55.0
Item ID            Item Name              Item Quantity                 Item Price
2             bhel              1             100
Bill amount 200
 Discount: 10
 Final Price180
----------------------------------------------------------
Order NO59             Order Date:2019-04-14 17:33:15.0
Item ID            Item Name              Item Quantity                 Item Price
3             brownie              1              150
Bill amount 250
 Discount: 6
 Final Price235
----------------------------------------------------------
Order NO60             Order Date:2019-04-14 17:37:38.0
Item ID            Item Name              Item Quantity                 Item Price
3             brownie              1              150
Bill amount 150
 Discount: 7
 Final Price140
----------------------------------------------------------
Order NO61             Order Date:2019-04-14 18:49:24.0
Item ID            Item Name              Item Quantity                 Item Price
1             french fries              2              100
Bill amount 200
 Discount: 8
 Final Price184
----------------------------------------------------------
```

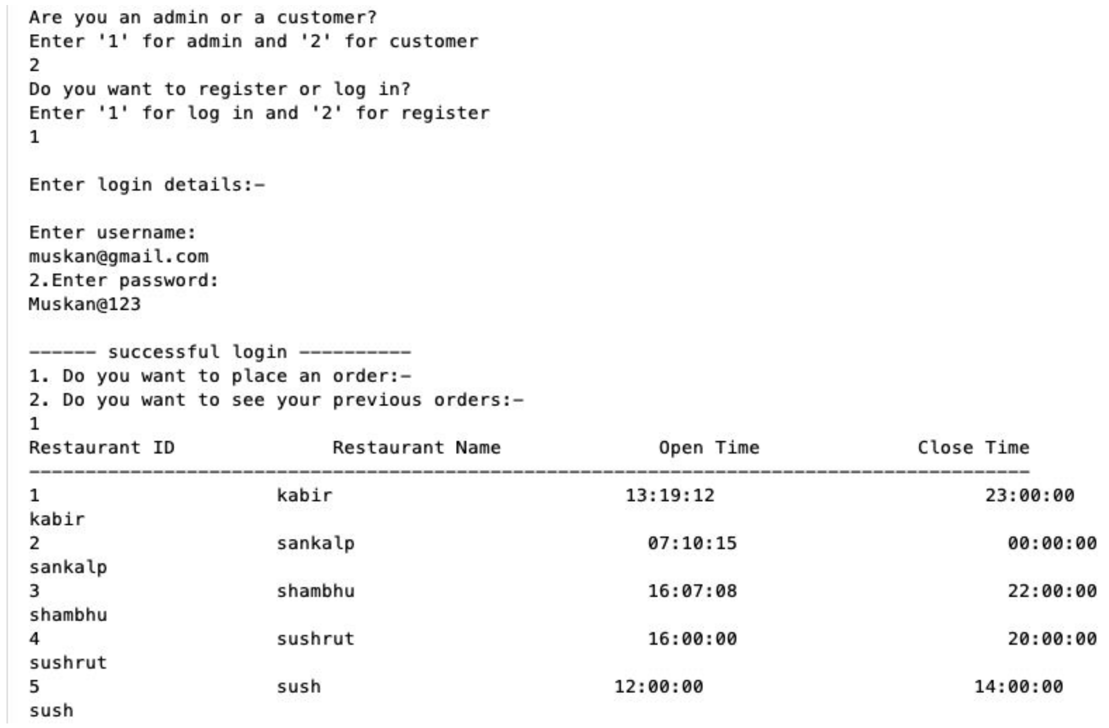
```
-----------------------------------------------------------
Order NO63              Order Date:2019-04-14 19:01:25.0
Item ID             Item Name             Item Quantity             Item Price
2              bhel             1              100
Bill amount 300
 Discount: 10
 Final Price270
-----------------------------------------------------------
Order NO64              Order Date:2019-04-14 19:04:19.0
Item ID             Item Name             Item Quantity             Item Price
2              bhel             1              100
Bill amount 0
 Discount: 0
 Final Price0
-----------------------------------------------------------
Order NO65              Order Date:2019-04-14 19:05:34.0
Item ID             Item Name             Item Quantity             Item Price
2              bhel             1              100
Bill amount 300
 Discount: 11
 Final Price267
-----------------------------------------------------------
Order NO66              Order Date:2019-04-14 21:57:32.0
Item ID             Item Name             Item Quantity             Item Price
2              bhel             1              100
Bill amount 0
 Discount: 0
 Final Price0
-----------------------------------------------------------
Order NO67              Order Date:2019-04-14 21:58:35.0
Item ID             Item Name             Item Quantity             Item Price
2              bhel             1              100
Bill amount 0
 Discount: 0
 Final Price0
```

9. **Place an order:** It inserts the details into the order table when a customer places an order.

Code:

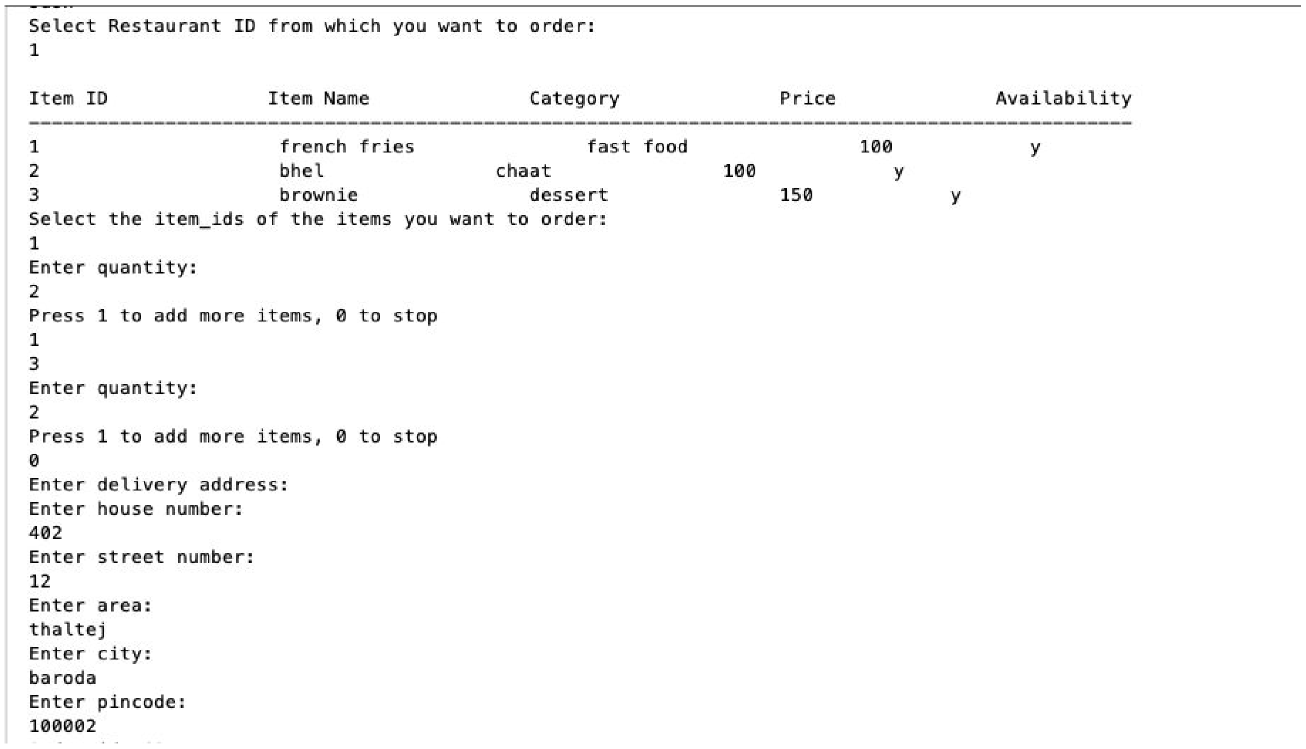
```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `orderInsert`(IN `cust_id` INT(5),              IN
`rest_id` INT(5))
BEGIN
        DECLARE deli_add_id int(5);

        set deli_add_id = (select delivery_address.delivery_address_id from delivery_address
where delivery_address.customer_id=cust_id limit 1);

        insert into orders(customer_id, delivery_address_id, restaurant_id, order_status)
values(cust_id, deli_add_id, rest_id, 'confirmed');

END$$
DELIMITER ;
```

Screenshot:-It asks user to place an order in the preferred restaurant by displaying all the restaurant details



```
Are you an admin or a customer?
Enter '1' for admin and '2' for customer
2
Do you want to register or log in?
Enter '1' for log in and '2' for register
1

Enter login details:-

Enter username:
muskan@gmail.com
2.Enter password:
Muskan@123

------ successful login ----------
1. Do you want to place an order:-
2. Do you want to see your previous orders:-
1
Restaurant ID            Restaurant Name            Open Time            Close Time
-------------------------------------------------------------------------------------------
1                    kabir                        13:19:12                        23:00:00
kabir
2                    sankalp                        07:10:15                        00:00:00
sankalp
3                    shambhu                        16:07:08                        22:00:00
shambhu
4                    sushrut                        16:00:00                        20:00:00
sushrut
5                    sush                        12:00:00                        14:00:00
sush
```

Screenshot:It asks user to choose the item to order along with the quantity

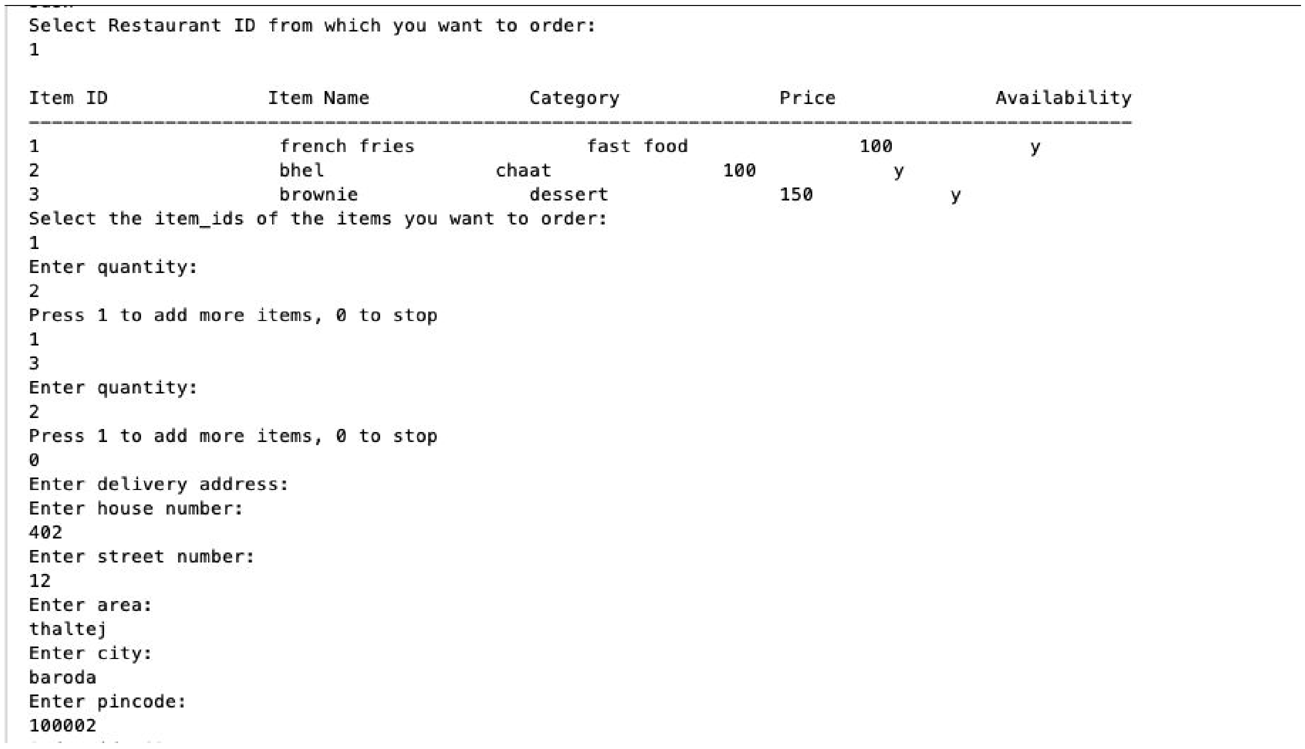


```
Select Restaurant ID from which you want to order:
1

Item ID              Item Name              Category              Price              Availability
==================================================================================================
1                     french fries               fast food                100            y
2                     bhel               chaat                100            y
3                     brownie               dessert                150            y
Select the item_ids of the items you want to order:
1
Enter quantity:
2
Press 1 to add more items, 0 to stop
1
3
Enter quantity:
2
Press 1 to add more items, 0 to stop
0
Enter delivery address:
Enter house number:
402
Enter street number:
12
Enter area:
thaltej
Enter city:
baroda
Enter pincode:
100002
```

10. **Adding items to the placed order:** It inserts the details of order like the number of items, their quantity and so on into the order_details table.

Code:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `orderDetailsInsert`(IN `cust_id` INT, IN
`qty1` INT, IN `it_id` INT)
BEGIN
        DECLARE order_id int(5);
        DECLARE pr,final_pr int(11);

        set order_id = (select max(order_no) from orders where orders.customer_id=cust_id);
        set pr = (select food_item.price from food_item where food_item.item_id=it_id);
        set final_pr = (pr*qty1);

        insert into order_details(order_no, item_id, price, qty, final_price) values(order_id, it_id, pr,
qty1, final_pr);
END$$
DELIMITER ;
```



```
Select Restaurant ID from which you want to order:
1

Item ID             Item Name              Category              Price             Availability
------------------------------------------------------------------------------------------------
1                    french fries               fast food                100           y
2                    bhel                 chaat                100           y
3                    brownie              dessert               150           y
Select the item_ids of the items you want to order:
1
Enter quantity:
2
Press 1 to add more items, 0 to stop
1
3
Enter quantity:
2
Press 1 to add more items, 0 to stop
0
Enter delivery address:
Enter house number:
402
Enter street number:
12
Enter area:
thaltej
Enter city:
baroda
Enter pincode:
100002
```

11. **Delivery Address:** The details of the delivery address are inserted via this procedure.

Code:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `deliveryAddress_insert`(IN `house_no1`
VARCHAR(10), IN `street_no1` VARCHAR(10), IN `area1` VARCHAR(20), IN `city1`
VARCHAR(20), IN `pincode1` CHAR(6), IN `customer_id` INT(5))
BEGIN
	INSERT INTO delivery_address(house_no,street_no,area,city,pincode,customer_id)
values(house_no1, street_no1, area1, city1, pincode1,customer_id);
END$$
DELIMITER ;
```

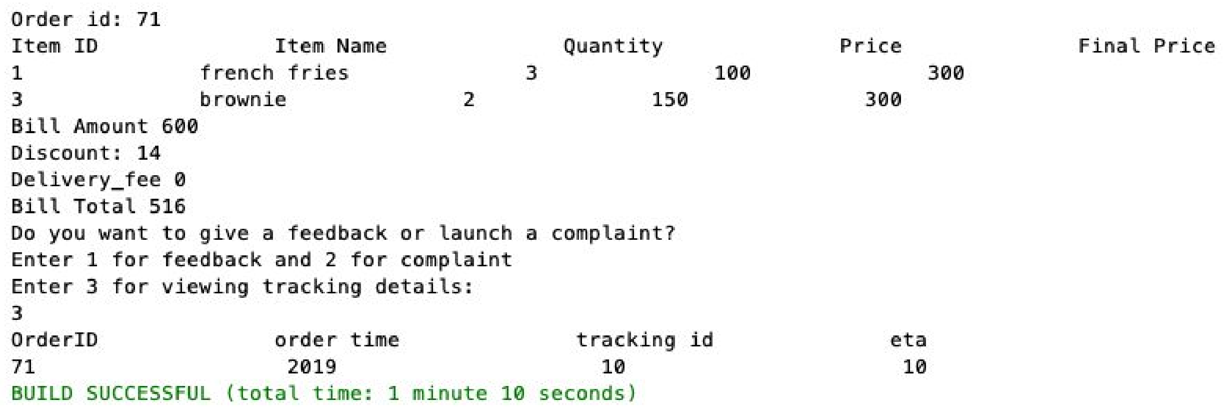
12. **Estimated Time of Arrival:** This procedure decides which branch of the requested restaurant will be delivering to the delivery address by calculating the minimum distance. Then the estimated time of arrival will be calculated based on the distance of the restaurant to the delivery address.

Code:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `eta`(IN `cust_id` INT(5), IN `restaurant_id1`
INT(5))
begin
   Declare b,min int;
   Declare a,c char(6);
   Declare eta int;
   Declare pin char(6);
   DECLARE order_id,did int(5);

   Declare cur1 cursor for select b.pincode from branch b,restaurant r where r.restaurant_id =
restaurant_id1 and b.restaurant_id = r.restaurant_id;
   Declare continue handler for not found set b = 1;

   set did = (select max(delivery_address_id) from delivery_address where d.customer_id=cust_id);

   Set pin = (select delivery_address.pincode from delivery_address where
delivery_address.delivery_address_id = did);

   set min = 5;
   set b = 0;
   set order_id = (select max(order_no) from orders where orders.customer_id=cust_id limit 1);
   open cur1;
   fetch cur1 into a;
   while b=0 Do
      if((a-c)<=min) then
       set min = (a-c);
      end if;
       fetch cur1 into a;
   end while;
   close cur1;

   if(min<=5) then
   set eta = 10;
   end if;
```

```
    if(min>5 and min<=10) then
    set eta = 20;
    end if;
    if(min>10 and min<=15) then
    set eta = 30;
    end if;

    insert into tracking_details(order_no,eta,status) values(order_id,eta,'OD');
 end$$
DELIMITER ;
```

Screenshot: It displays the tracking details of the particular order with its order_id



```
Order id: 71
Item ID              Item Name              Quantity              Price              Final Price
1              french fries              3              100              300
3              brownie              2              150              300
Bill Amount 600
Discount: 14
Delivery_fee 0
Bill Total 516
Do you want to give a feedback or launch a complaint?
Enter 1 for feedback and 2 for complaint
Enter 3 for viewing tracking details:
3
OrderID              order time              tracking id              eta
71              2019              10              10
BUILD SUCCESSFUL (total time: 1 minute 10 seconds)
```

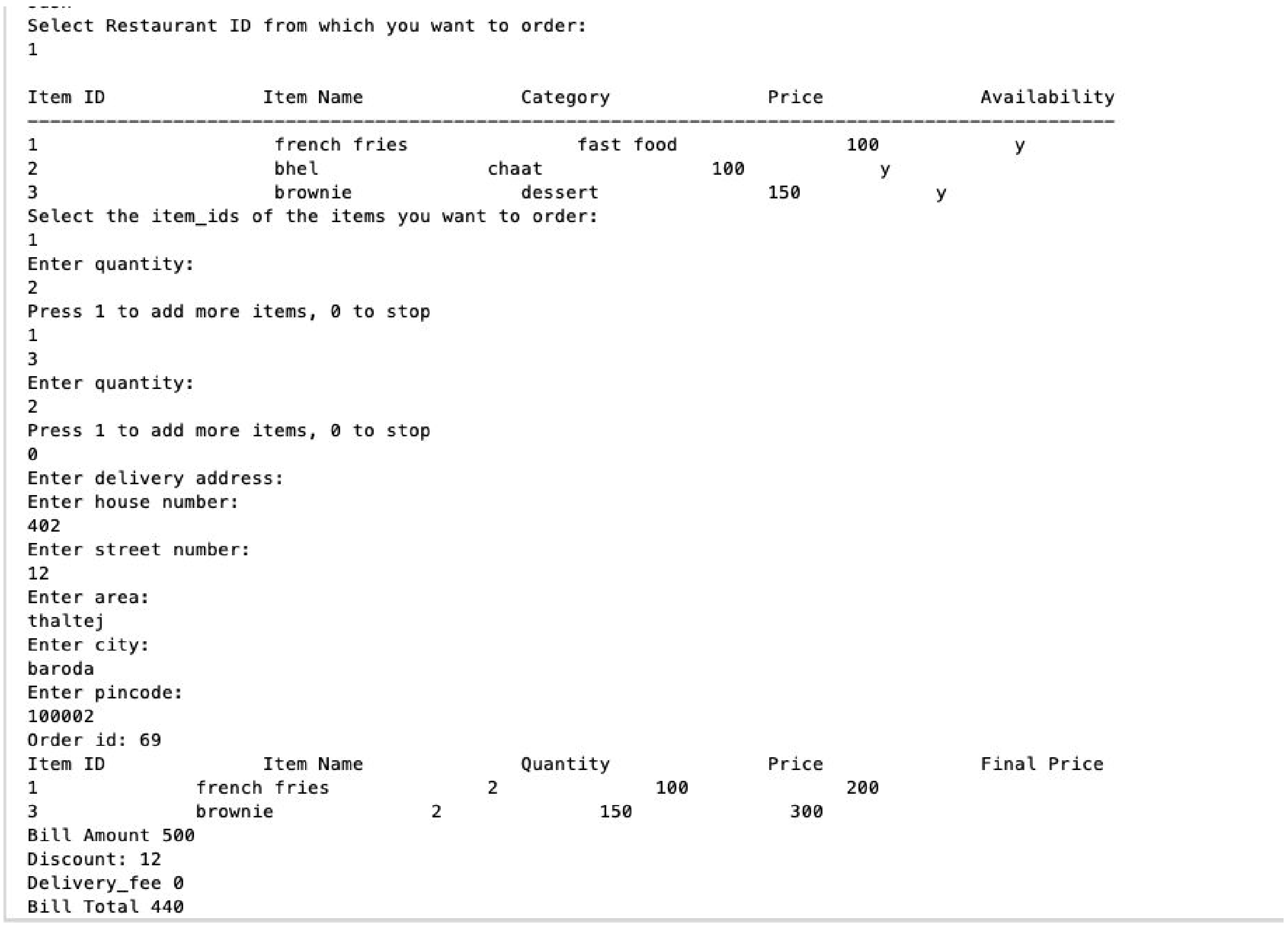
13. **Delivery Fee:** Delivery fee of the order is calculated based on the Estimated Time of Arrival (ETA). Also, if the customer is a premium member then the delivery fee is released. The fee is then inserted into the delivery table.

Code:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `delivery_fee`(IN `cust_id` INT(5))
BEGIN
    DECLARE a,x int(5);
    DECLARE d,d_id int(5);
    DECLARE c char;
    DECLARE deli_fee int;

    SET a = (SELECT max(order_no) from orders,customer where customer.customer_id =
orders.customer_id and customer.customer_id = customer_id1);

    SET c = (SELECT registration.premium_member from registration,customer where
registration.username = customer.email and customer.customer_id = customer_id1);

    SET x = (SELECT tracking_details.tracking_id from tracking_details where
tracking_details.order_no=a);

    SET d = (SELECT ETA from tracking_details where tracking_details.tracking_id = x);

    SET d_id = (SELECT max(delivery_address.delivery_address_id) from delivery_address where
delivery_address.customer_id=customer_id1);

          if (d = 10) then
      set deli_fee = 10;
      end if;

          if(d = 20) then
      set deli_fee = 20;
      end if;

      if(d = 30) then
      set deli_fee = 30;
      end if;

      if(c='y') THEN
      set deli_fee = 0;
```

```
        end if;

        insert into delivery (delivery_address_id, delivery_staff_id,  order_no,delivery_fee) values(d_id,
1, a,deli_fee);

END$$
DELIMITER ;
```

Screenshot: It displays the delivery fee as 0, because the logged in user is a premium member.Otherwise the fee is calculated according to the distance between customer's address and branch address of the restaurant.



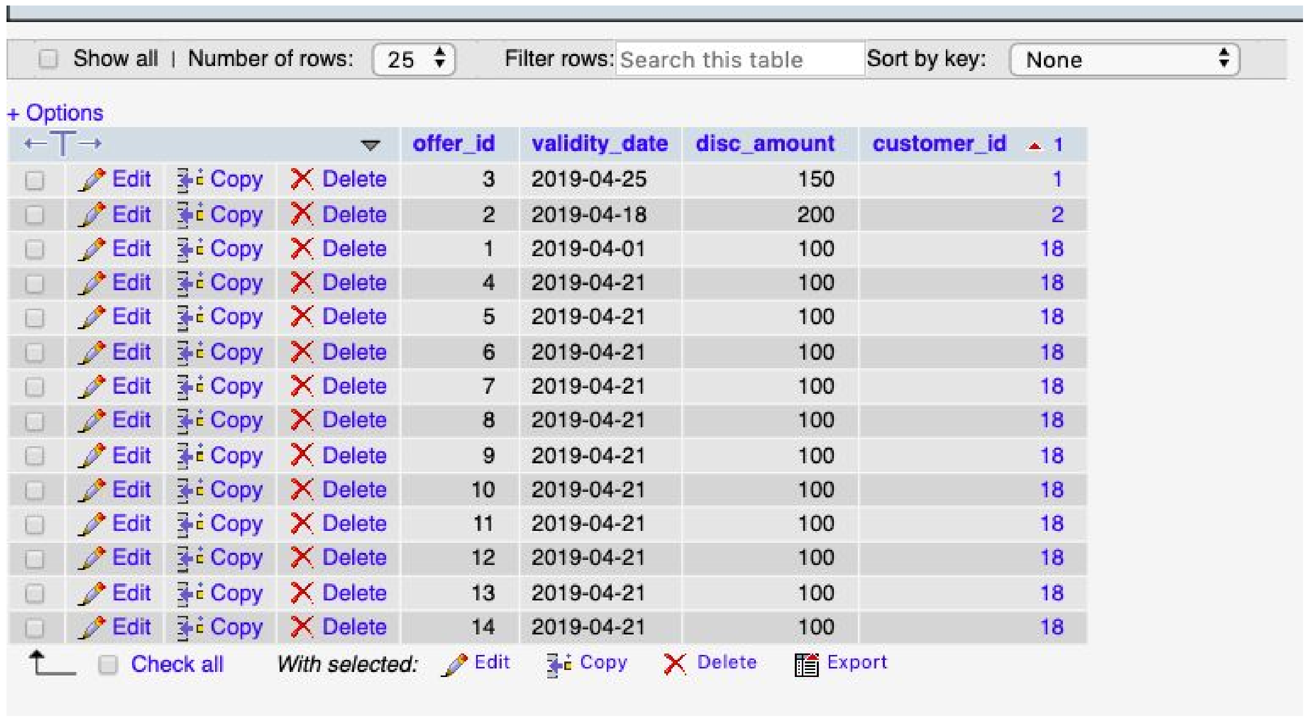
```
Select Restaurant ID from which you want to order:
1

Item ID              Item Name              Category             Price              Availability
--------------------------------------------------------------------------------------------------
1                     french fries               fast food              100           y
2                     bhel               chaat               100           y
3                     brownie               dessert               150           y
Select the item_ids of the items you want to order:
1
Enter quantity:
2
Press 1 to add more items, 0 to stop
1
3
Enter quantity:
2
Press 1 to add more items, 0 to stop
0
Enter delivery address:
Enter house number:
402
Enter street number:
12
Enter area:
thaltej
Enter city:
baroda
Enter pincode:
100002
Order id: 69
Item ID              Item Name              Quantity             Price              Final Price
1              french fries              2              100              200
3              brownie              2              150              300
Bill Amount 500
Discount: 12
Delivery_fee 0
Bill Total 440
```

14. **Provision of offer/discount:** If a customer has placed orders a given number of times then that customer will be rewarded with a discount of 100 rupees. The validity of this offer will be 7 days from the present date.

Code:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `offer_1`(IN `customer_id1` INT)
BEGIN
    DECLARE count1,date int;
    DECLARE dt date;
    DECLARE st date;
    DECLARE discount int;
    SET count1 = (SELECT count(order_no) from orders where orders.customer_id = customer_id1
group by orders.customer_id);
    if(count1>=2) then
    set discount = 100;
    set st = (select CURDATE());
    set dt = DATE_ADD(CURRENT_TIMESTAMP, INTERVAL 7 DAY);
    insert into offer(customer_id,validity_date,disc_amount) values(customer_id1,dt, discount);
    end if;
END$$
DELIMITER ;
```

Screenshot: It enters details into offer according to the previous number of orders that user has placed.



Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

	offer_id	validity_date	disc_amount	customer_id ▲ 1
Edit Copy Delete	3	2019-04-25	150	1
Edit Copy Delete	2	2019-04-18	200	2
Edit Copy Delete	1	2019-04-01	100	18
Edit Copy Delete	4	2019-04-21	100	18
Edit Copy Delete	5	2019-04-21	100	18
Edit Copy Delete	6	2019-04-21	100	18
Edit Copy Delete	7	2019-04-21	100	18
Edit Copy Delete	8	2019-04-21	100	18
Edit Copy Delete	9	2019-04-21	100	18
Edit Copy Delete	10	2019-04-21	100	18
Edit Copy Delete	11	2019-04-21	100	18
Edit Copy Delete	12	2019-04-21	100	18
Edit Copy Delete	13	2019-04-21	100	18
Edit Copy Delete	14	2019-04-21	100	18

Check all *With selected:* Edit Copy Delete Export

15. **Insertion of bill details:** The details of bill like bill amount, discount, delivery fee are inserted into the bill table.

Code:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `bill_details4`(IN `cust_id` INT(5))
BEGIN
        DECLARE fee,bill_amt,bill_total,discount int(11);
   DECLARE count1,o_no,order_id int(5);
   DECLARE q int(11);
   DECLARE p int(11);
   DECLARE b int;

   DECLARE cur CURSOR FOR SELECT price,qty FROM order_details where
order_details.order_no = (select max(orders.order_no) from orders where
orders.customer_id=cust_id limit 1);
   DECLARE CONTINUE HANDLER FOR NOT FOUND SET b = 1;
   SET bill_amt=0;
   SET fee=(SELECT delivery.delivery_fee from orders,delivery where delivery.order_no = (select
max(order_no) from orders where customer_id=cust_id) limit 1);

   OPEN cur;
   SET b = 0;
   set fee=0;
   FETCH cur INTO p,q;
   WHILE b = 0 DO

        SET bill_amt=bill_amt+(p*q);

     FETCH cur INTO p,q;
   END WHILE;
        select bill_amt;

   set order_id = (select max(orders.order_no) from orders where orders.customer_id=cust_id);

    set discount = 0;
   set discount = (select max(offer.offer_id) from offer where offer.customer_id=cust_id and
offer.validity_date>=CURRENT_DATE);

   set bill_total = bill_amt - (bill_amt*discount)/100;
   insert into bill(order_no,bill_amount,discount,bill_total,delivery_fee)
values(order_id,bill_amt,discount,bill_total,fee);
```

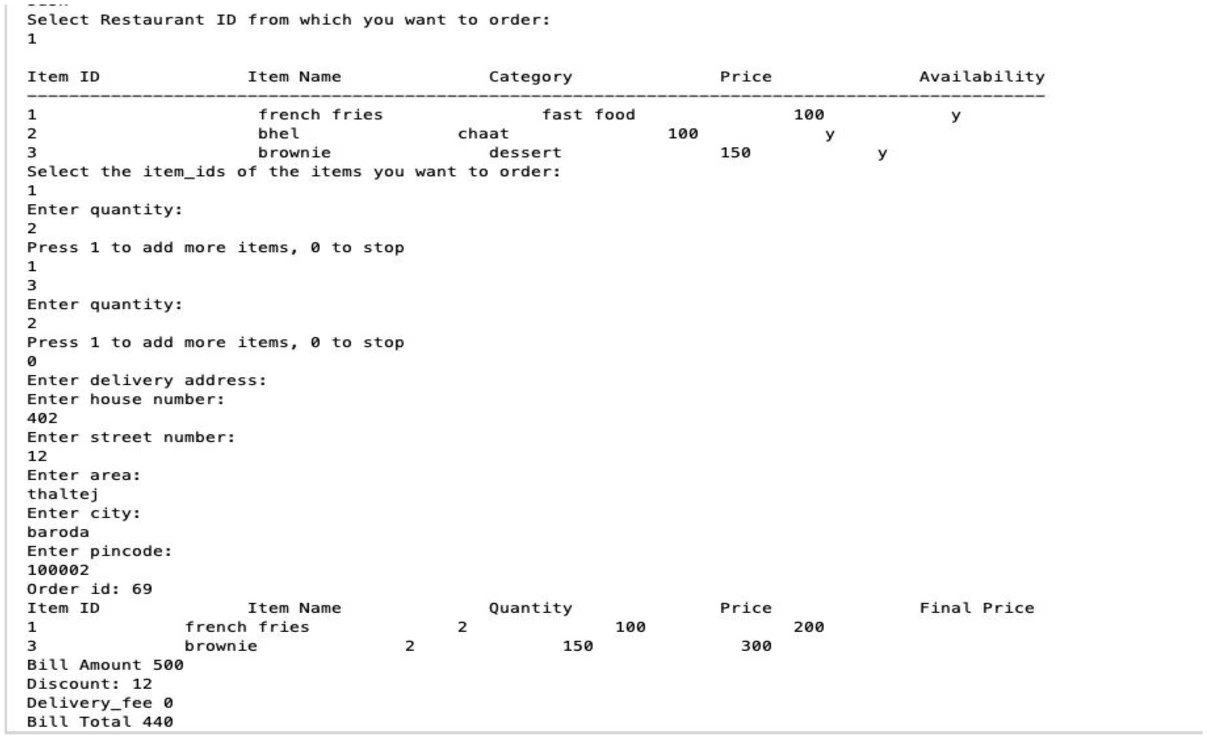
```
    CLOSE cur;
END$$
DELIMITER ;
```

16. **Display Bill details:** Details of the bill are displayed via this procedure.

Code:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `bill_procedure`(IN `cust_id` INT(5))
BEGIN
	DECLARE order_id,d int(5);
	DECLARE c int;
	DECLARE name varchar(20);
	DECLARE bill_total,qty,price,final_pr,bill_amount,discount,delivery_fee int(11);
	DECLARE cur2 CURSOR FOR SELECT order_details.item_id from order_details where
order_details.order_no = (select max(order_no) from orders where orders.customer_id=cust_id
LIMIT 1);
	DECLARE CONTINUE HANDLER FOR NOT FOUND SET c = 1;
	set order_id = (select max(order_no) from orders where orders.customer_id=cust_id limit 1);
	set bill_total = (select bill.bill_total from bill where bill.order_no=order_id limit 1);
   set bill_amount = (SELECT bill.bill_amount from bill where bill.order_no=order_id limit 1);
   set discount = (SELECT bill.discount from bill where bill.order_no=order_id limit 1);
   set delivery_fee = (SELECT bill.delivery_fee from bill where bill.order_no=order_id limit 1);
	OPEN cur2;
	SET c = 0;
	FETCH cur2 INTO d;
	WHILE c = 0 DO
		set name = (select food_item.item_name from food_item where item_id=d);
		set qty = (select order_details.qty from order_details where order_no=order_id and
item_id = d);
		set price = (select order_details.price from order_details where order_no=order_id
and item_id = d limit 1);
		set final_pr = (select final_price from order_details where order_no=order_id and
item_id = d);

		select order_id as
ord_id,d,name,qty,price,final_pr,bill_amount,discount,delivery_fee,bill_total;
      FETCH cur2 INTO d;
	END WHILE;
	CLOSE cur2;
END$$
DELIMITER ;
```

Screenshot: It displays the bill details of the order with all items he/she has ordered, discount and final bill amount.



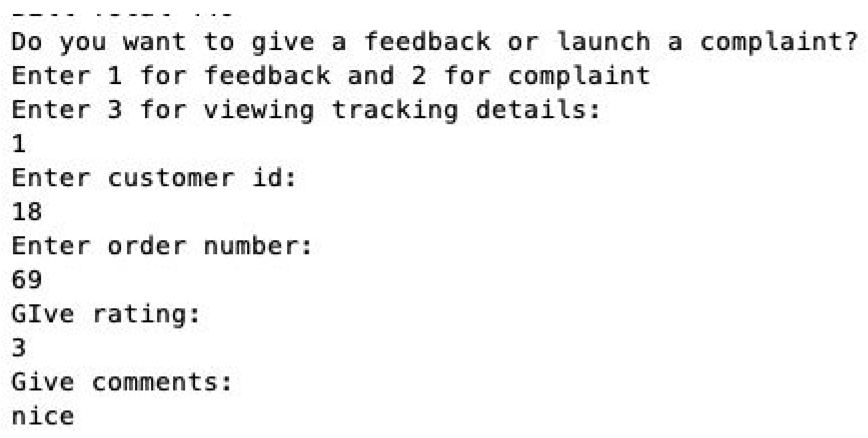
```
Select Restaurant ID from which you want to order:
1

Item ID              Item Name              Category              Price              Availability
-------------------------------------------------------------------------------------------------
1                     french fries               fast food                100            y
2                     bhel               chaat               100            y
3                     brownie              dessert               150           y
Select the item_ids of the items you want to order:
1
Enter quantity:
2
Press 1 to add more items, 0 to stop
1
3
Enter quantity:
2
Press 1 to add more items, 0 to stop
0
Enter delivery address:
Enter house number:
402
Enter street number:
12
Enter area:
thaltej
Enter city:
baroda
Enter pincode:
100002
Order id: 69
Item ID              Item Name              Quantity              Price              Final Price
1             french fries              2              100              200
3             brownie              2              150              300
Bill Amount 500
Discount: 12
Delivery_fee 0
Bill Total 440
```

17. **Feedback:** Takes in the feedback given by the customer.

Code:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `feedback1`(IN `order_no` INT(5), `cust_id`
INT(5), `rating` INT(11), `comment1` VARCHAR(20))
BEGIN
	INSERT INTO
feedback(customer_id,feedback.order_no,feedback.rating,feedback.comment)
values(cust_id,order_no,rating, comment1);
END$$
DELIMITER ;
```

Screenshot: Allows user to file a feedback.



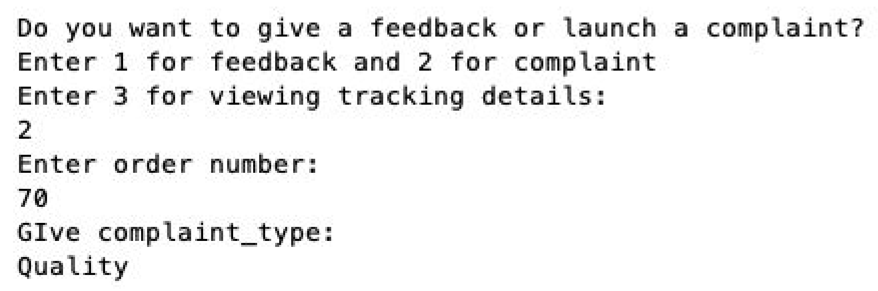
```
Do you want to give a feedback or launch a complaint?
Enter 1 for feedback and 2 for complaint
Enter 3 for viewing tracking details:
1
Enter customer id:
18
Enter order number:
69
GIve rating:
3
Give comments:
nice
```

18. **Complaint:** Inserts the complaint type and other details of complaint filed by the customer.

Code:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `complaint`(IN `order_no` INT, `cust_id` INT,
`complaint_type` VARCHAR(50), `complaint_date` DATE)
BEGIN
	INSERT INTO complaint(customer_id,order_no,complaint_type,complaint_date)
values(cust_id,order_no,complaint_type,complaint_date);
END$$
DELIMITER ;
```

Screenshot: Allows user to file a complaint.



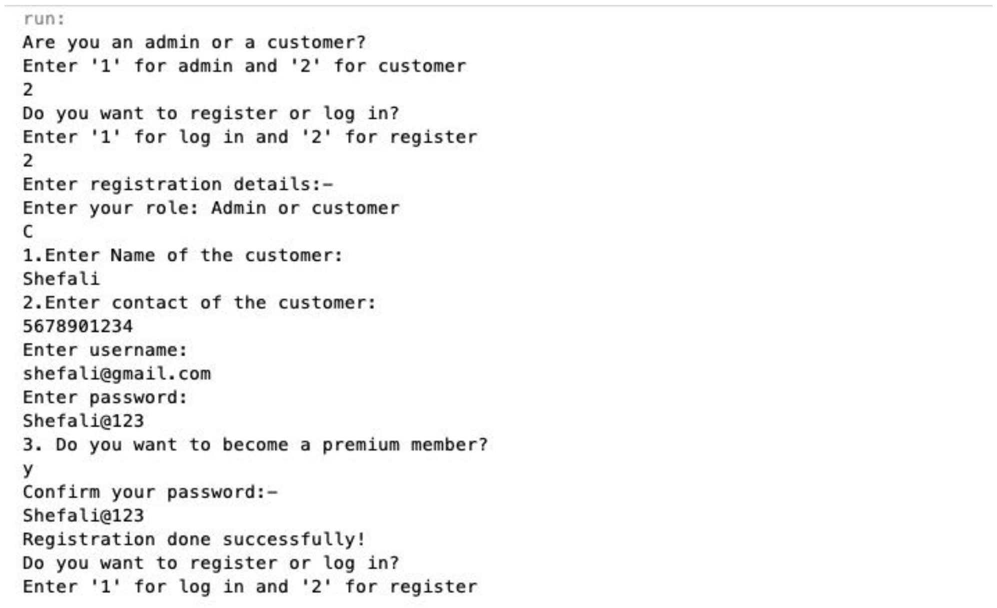
```
Do you want to give a feedback or launch a complaint?
Enter 1 for feedback and 2 for complaint
Enter 3 for viewing tracking details:
2
Enter order number:
70
GIve complaint_type:
Quality
```

19. Premium Wallet: This procedure checks if the registered customer is a premium member or not. If the customer is a premium member then an amount of 500 rupees will be deducted from the existing wallet balance.

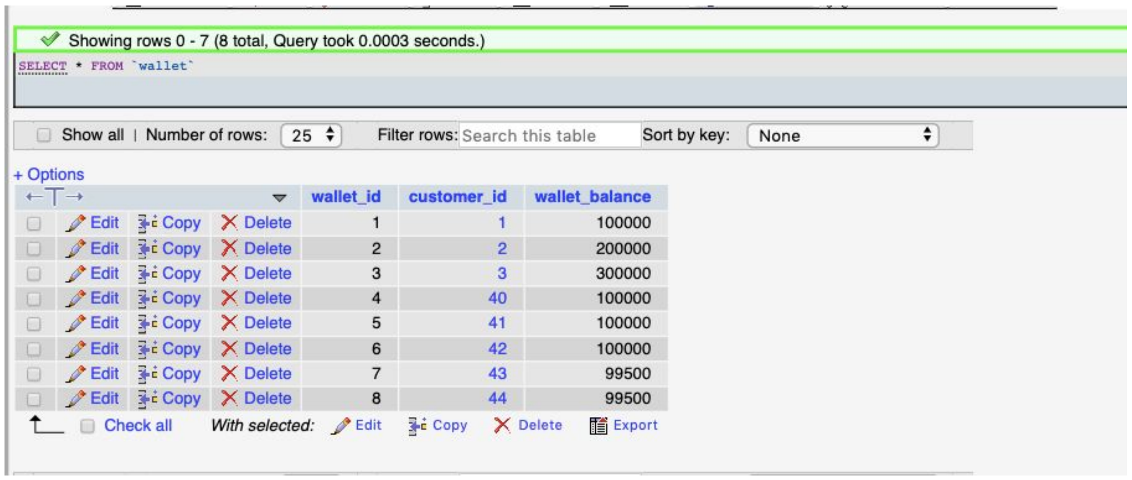
Code:

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `premium_wallet`(IN `premium` CHAR(1), IN
`username` VARCHAR(50))
BEGIN
	DECLARE account_id int(5);
	DECLARE wallet_id1 int(5);
	DECLARE premium_stat char(1);
    DECLARE c_id int(5);

    SET c_id = (SELECT customer_id from customer where customer.email=username);

	SET wallet_id1 =(SELECT wallet.wallet_id FROM wallet WHERE wallet.customer_id=c_id
limit 1);


	IF premium_stat='y' THEN
		update wallet set wallet.wallet_balance=wallet.wallet_balance-500 where
wallet.wallet_id=wallet_id1;
	END IF;

END$$
DELIMITER ;
```

Screenshot: Updates the wallet of premium member by charging 500 as the amount for premium membership after registration



```
run:
Are you an admin or a customer?
Enter '1' for admin and '2' for customer
2
Do you want to register or log in?
Enter '1' for log in and '2' for register
2
Enter registration details:-
Enter your role: Admin or customer
C
1.Enter Name of the customer:
Shefali
2.Enter contact of the customer:
5678901234
Enter username:
shefali@gmail.com
Enter password:
Shefali@123
3. Do you want to become a premium member?
y
Confirm your password:-
Shefali@123
Registration done successfully!
Do you want to register or log in?
Enter '1' for log in and '2' for register
```



Showing rows 0 - 7 (8 total, Query took 0.0003 seconds.)

```
SELECT * FROM `wallet`
```

wallet_id	customer_id	wallet_balance
1	1	100000
2	2	200000
3	3	300000
4	40	100000
5	41	100000
6	42	100000
7	43	99500
8	44	99500

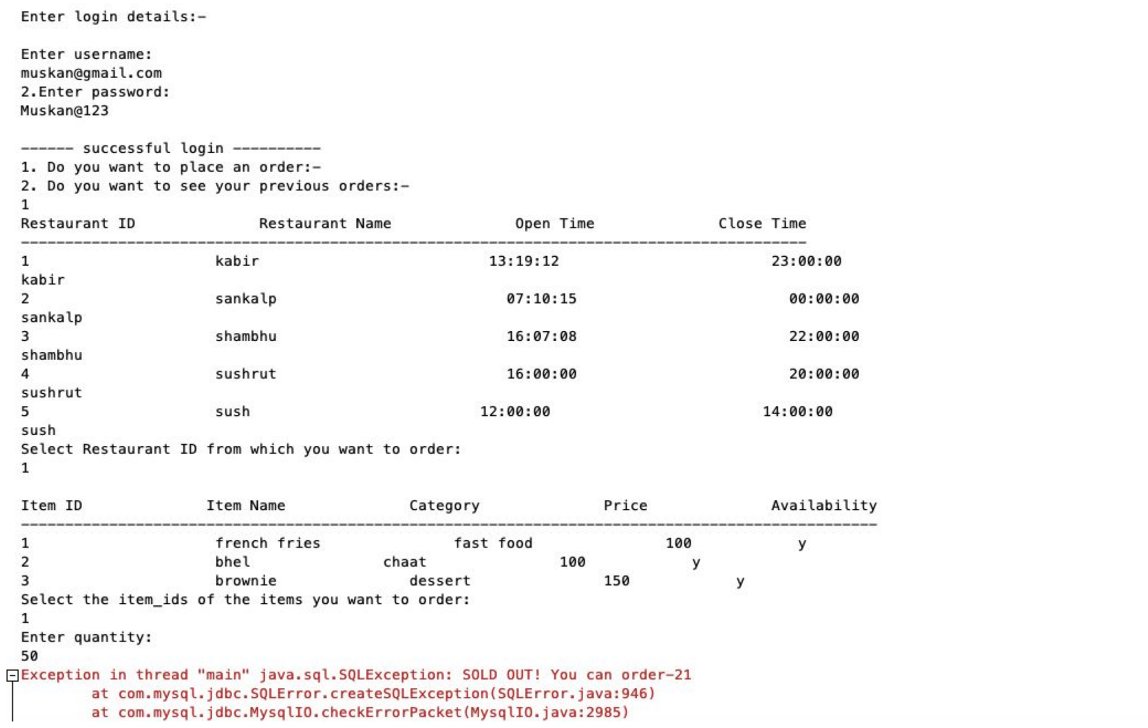
Triggers

1. **Check availability of food items:** If the ordered food item is available then the new availability status (no of items) will be updated in the food_item table. If the ordered item is not available then an error message will be displayed.

Code:

```
    CREATE TRIGGER `avail` BEFORE INSERT ON `order_details`
 FOR EACH ROW BEGIN
        DECLARE av int(11);
        DECLARE msg varchar(50);

        SET av= (SELECT food_item.availability from food_item WHERE
food_item.item_id=new.item_id);

        IF av>0 and av>new.qty THEN
                UPDATE food_item set food_item.availability =(av-new.qty) where
food_item.item_id=new.item_id;
        ELSE
                SET msg= concat('SOLD OUT! You can order',av-new.qty);
                signal sqlstate '45001' set message_text = msg;
        END IF;
END
```

Screenshot: Checks the availability of the items the customer wants to order and displays error as sold out when availability is 0.



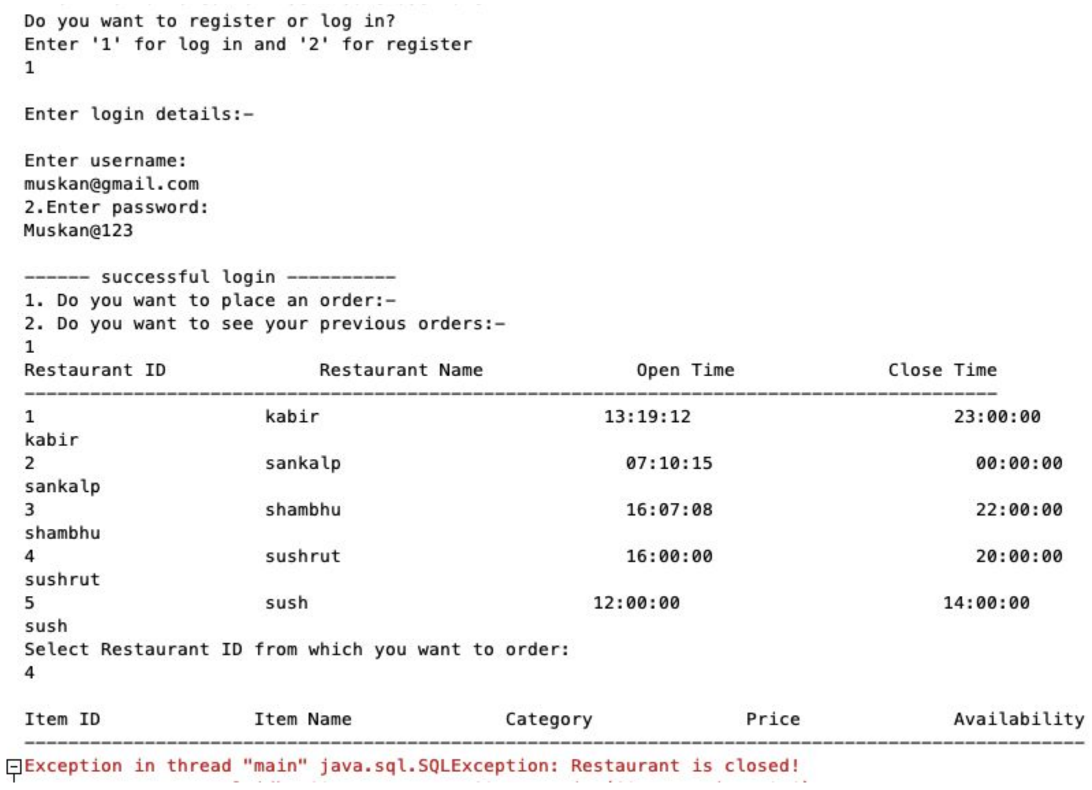
```
Enter login details:-

Enter username:
muskan@gmail.com
2.Enter password:
Muskan@123

------ successful login ----------
1. Do you want to place an order:-
2. Do you want to see your previous orders:-
1
Restaurant ID              Restaurant Name              Open Time              Close Time
-------------------------------------------------------------------------------------------
1                      kabir                          13:19:12                         23:00:00
kabir
2                      sankalp                          07:10:15                         00:00:00
sankalp
3                      shambhu                          16:07:08                         22:00:00
shambhu
4                      sushrut                          16:00:00                         20:00:00
sushrut
5                      sush                           12:00:00                         14:00:00
sush
Select Restaurant ID from which you want to order:
1

Item ID              Item Name              Category              Price              Availability
---------------------------------------------------------------------------------------------------
1                      french fries              fast food              100              y
2                      bhel              chaat              100              y
3                      brownie              dessert              150              y
Select the item_ids of the items you want to order:
1
Enter quantity:
50
Exception in thread "main" java.sql.SQLException: SOLD OUT! You can order-21
        at com.mysql.jdbc.SQLError.createSQLException(SQLError.java:946)
        at com.mysql.jdbc.MysqlIO.checkErrorPacket(MysqlIO.java:2985)
```

2. **Check availability of restaurant:** The time of placing the order is checked with the open time and close time of the restaurant. If the corresponding time falls in that range then food items can be ordered and if not then an error message will be displayed.

Code:

```
CREATE TRIGGER `check_restro` BEFORE INSERT ON `orders`
 FOR EACH ROW BEGIN
        DECLARE a,b,c time;
   DECLARE msg varchar(128);
        SET c=CURRENT_TIMESTAMP;
   SET a=(SELECT open_time from restaurant where restaurant_id=new.restaurant_id);
   SET b=(SELECT close_time from restaurant where restaurant_id=new.restaurant_id);
   IF c NOT BETWEEN CONVERT(a,TIME) AND CONVERT(b,TIME) THEN
        set msg = 'Restaurant is closed!';
      signal sqlstate '45001' set message_text = msg;
   END IF;
END
```

Screenshot: Checks if the user can place an order at a specific restaurant based on its working hours and displays appropriate message when restaurant is closed



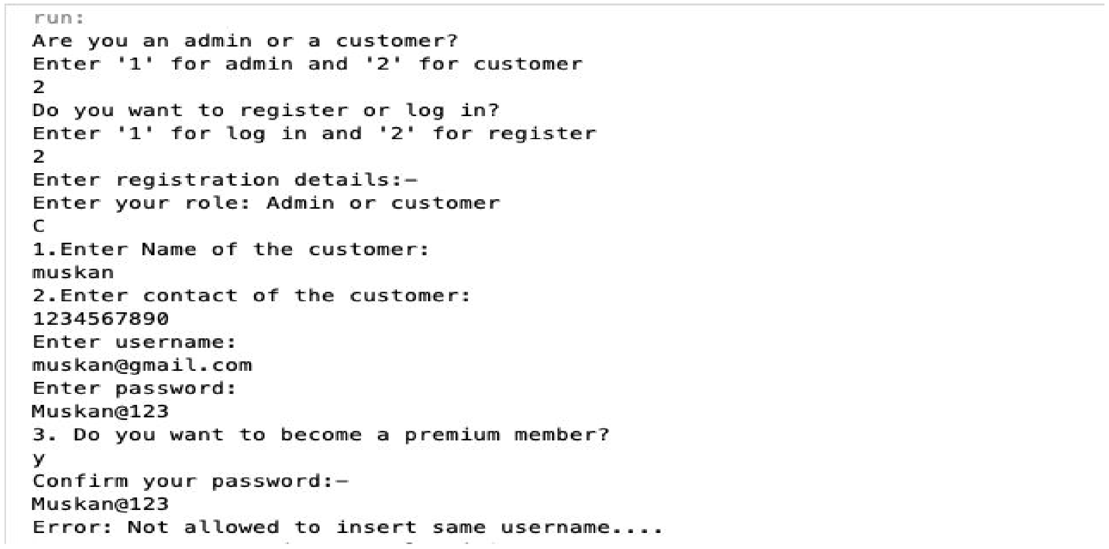
```
Do you want to register or log in?
Enter '1' for log in and '2' for register
1

Enter login details:-

Enter username:
muskan@gmail.com
2.Enter password:
Muskan@123

------ successful login ----------
1. Do you want to place an order:-
2. Do you want to see your previous orders:-
1
Restaurant ID             Restaurant Name             Open Time             Close Time
------------------------------------------------------------------------------------------
1                     kabir                         13:19:12                        23:00:00
kabir
2                     sankalp                         07:10:15                        00:00:00
sankalp
3                     shambhu                         16:07:08                        22:00:00
shambhu
4                     sushrut                         16:00:00                        20:00:00
sushrut
5                     sush                         12:00:00                        14:00:00
sush
Select Restaurant ID from which you want to order:
4

Item ID             Item Name             Category             Price             Availability
--------------------------------------------------------------------------------------------------
Exception in thread "main" java.sql.SQLException: Restaurant is closed!
```

3. **On registration:** This trigger checks for the username at the time of registration. If the newly entered username already exists then an error message will be displayed.

Code:

```
CREATE TRIGGER `trigger1` BEFORE INSERT ON `registration`
 FOR EACH ROW BEGIN
        DECLARE b int;
        DECLARE a varchar(50);
        DECLARE msg varchar(100);
        DECLARE cur CURSOR FOR SELECT username from registration;
        DECLARE CONTINUE HANDLER FOR NOT FOUND SET b = 1;
         OPEN cur;
         SET b = 0;
        FETCH cur INTO a;
          WHILE b = 0 DO
                if a=new.username then
                        set msg = 'Error: Not allowed to insert same username....';
                        signal sqlstate '45006' set message_text = msg;
                end if;
           FETCH cur INTO a;

         END WHILE;
        CLOSE cur;
END
```

Screenshot: Checks for the uniqueness of the username otherwise throws error



```
run:
Are you an admin or a customer?
Enter '1' for admin and '2' for customer
2
Do you want to register or log in?
Enter '1' for log in and '2' for register
2
Enter registration details:-
Enter your role: Admin or customer
C
1.Enter Name of the customer:
muskan
2.Enter contact of the customer:
1234567890
Enter username:
muskan@gmail.com
Enter password:
Muskan@123
3. Do you want to become a premium member?
y
Confirm your password:-
Muskan@123
Error: Not allowed to insert same username....
```