Mini project report on

Grocery Store Management System

Submitted in partial fulfilment of the requirements for the award of degree of

Bachelor of Technology

in

Computer Science & Engineering

UE20CS301 – DBMS PROJECT

Submitted by:

NITHISH S **PES2UG20CS531**

Under the guidance of

Prof. Nivedita Kasturi

Assistant Professor

Designation

PES University

AUG - DEC 2022

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

FACULTY OF ENGINEERING

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)

Electronic City, Hosur Road, Bengaluru – 560 100, Karnataka, India

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
Electronic City, Hosur Road, Bengaluru – 560 100, Karnataka, India

CERTIFICATE

This is to certify that the mini project entitled

Grocery Store Management System

is a Bonafede work carried out by

NITHISH S **PES2UG20CS531**

In partial fulfilment for the completion of fifth semester DBMS Project (UE20CSS301) in the Program of Study - Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period AUG. 2022 – DEC. 2022. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The project has been approved as it satisfies the 5th semester academic requirements in respect of project work.

Signature
Prof. Nivedita Kasturi
Assistant Professor

DECLARATION

We hereby declare that the DBMS Project entitled **Grocery Store Management System** has been carried out by us under the guidance of **Prof. Nivedita Kasturi, Assistant Professor** and submitted in partial fulfilment of the course requirements for the award of degree of **Bachelor of Technology** in **Computer Science and Engineering** of **PES University, Bengaluru** during the academic semester AUG – DEC 2022.

NITHISH S **PES2UG20CS531** **<Signature>**

ACKNOWLEDGEMENT

I would like to express my gratitude to Prof. Nivedita Kasturi, Department of Computer Science and Engineering, PES University, for her continuous guidance, assistance, and encouragement throughout the development of this UE20CS301 - DBMS Project.

I take this opportunity to thank Dr. Sandesh B J, C, Professor, ChairPerson, Department of Computer Science and Engineering, PES University, for all the knowledge and support I have received from the department.

I am deeply grateful to Dr. M. R. Doreswamy, Chancellor, PES University, Prof. Jawahar Doreswamy, Pro Chancellor – PES University, Dr. Suryaprasad J, Vice-Chancellor, PES University for providing to me various opportunities and enlightenment every step of the way. Finally, this DBMS Project could not have been completed without the continual support and encouragement I have received from my family and friends.

ABSTRACT

This project is about Grocery Store Database Management system. This basically consist of management of Grocery items and employee who manges the grocery items and customers who buys these products. This project manages records of customers who buys the items. This system can be implemented in super markets so that they can easily keep the records of the grocery items for any future needs and without having to get confused.

Helps in managing customer purchases, employee can maintain the database along with the prices of the groceries. The customer can make the payment after the customer purchases the grocery items.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

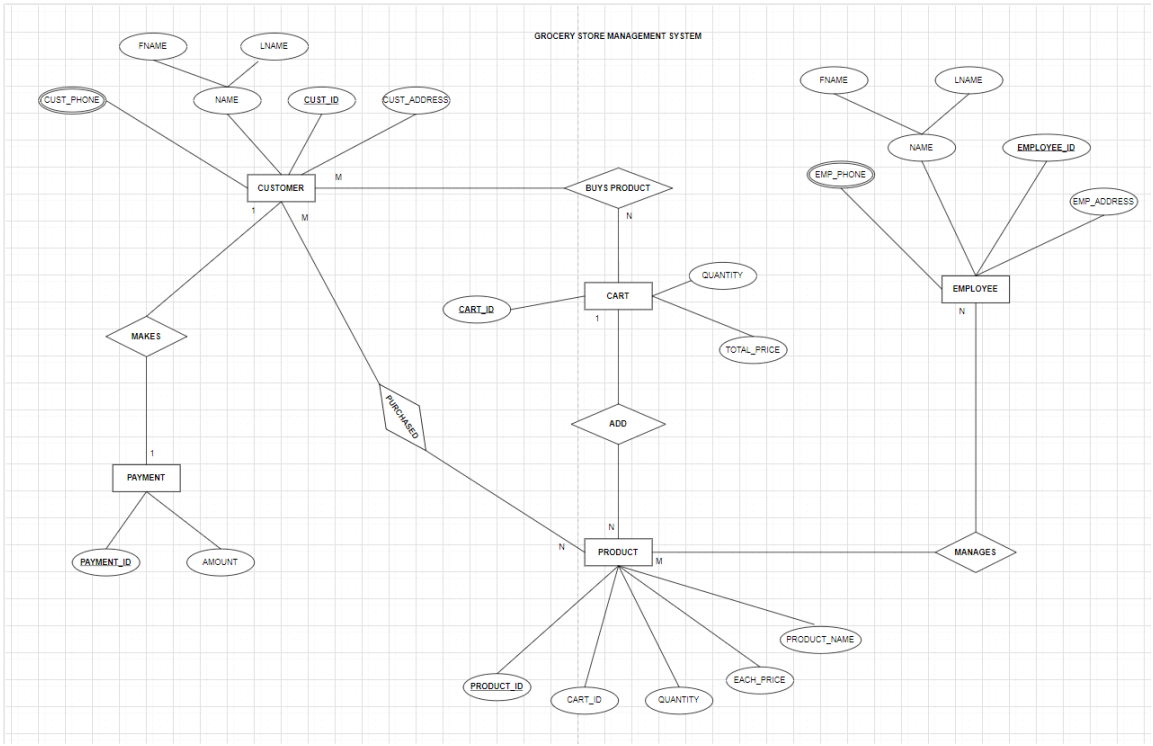
This project is about Grocery Store Database Management system. This basically consist of management of Grocery items and employee who manges the grocery items and customers who buys these products. This project manages records of customers who buys the items. This system can be implemented in super markets so that they can easily keep the records of the grocery items for any future needs and without having to get confused.

2. PROBLEM DEFINITION

Keeping track of grocery items ,employee who manages the particular item and the customers buying it is confusing and there is a need to manage these buy creating a proper database which saves all the records and use it whenever needed. The primary aim of this Grocery Store Management System is to improve accuracy and enhance safety and efficiency of tracking and keeping details of grocery items.

The MYSQL database is used as a platform. Application and the GUI are developed in HTML5, using nodejs.

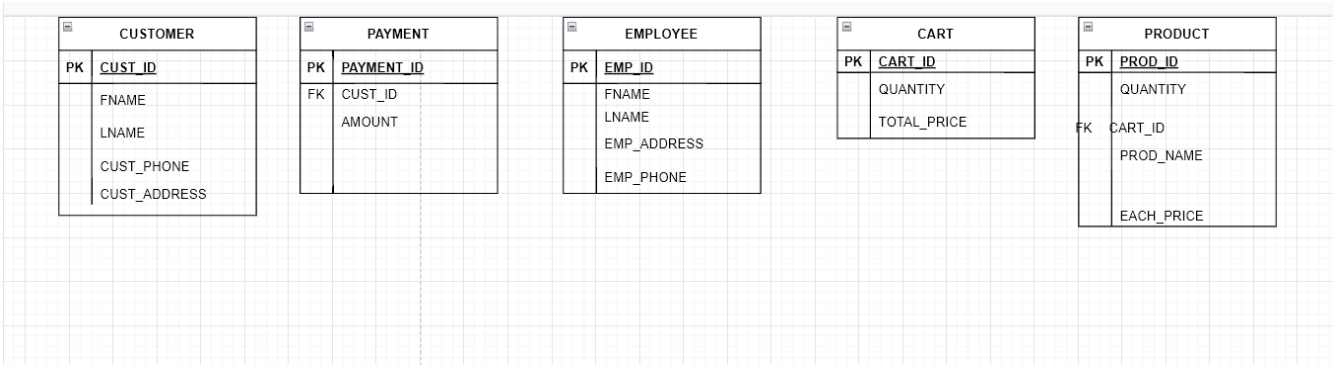
3. ER MODEL


GROCERY STORE MANAGEMENT SYSTEM
FNAME
LNAME
NAME
CUST_PHONE
CUST_ID
CUST_ADDRESS
CUSTOMER
M
1
M
BUYS PRODUCT
N
CART
QUANTITY
CART_ID
1
TOTAL_PRICE
MAKES
PURCHASED
ADD
1
PAYMENT
N
N
PRODUCT
M
PAYMENT_ID
AMOUNT
MANAGES
PRODUCT_NAME
PRODUCT_ID
CART_ID
QUANTITY
EACH_PRICE
FNAME
LNAME
NAME
EMPLOYEE_ID
EMP_PHONE
EMP_ADDRESS
EMPLOYEE
N

4. ER TO RELATIONAL MAPPING

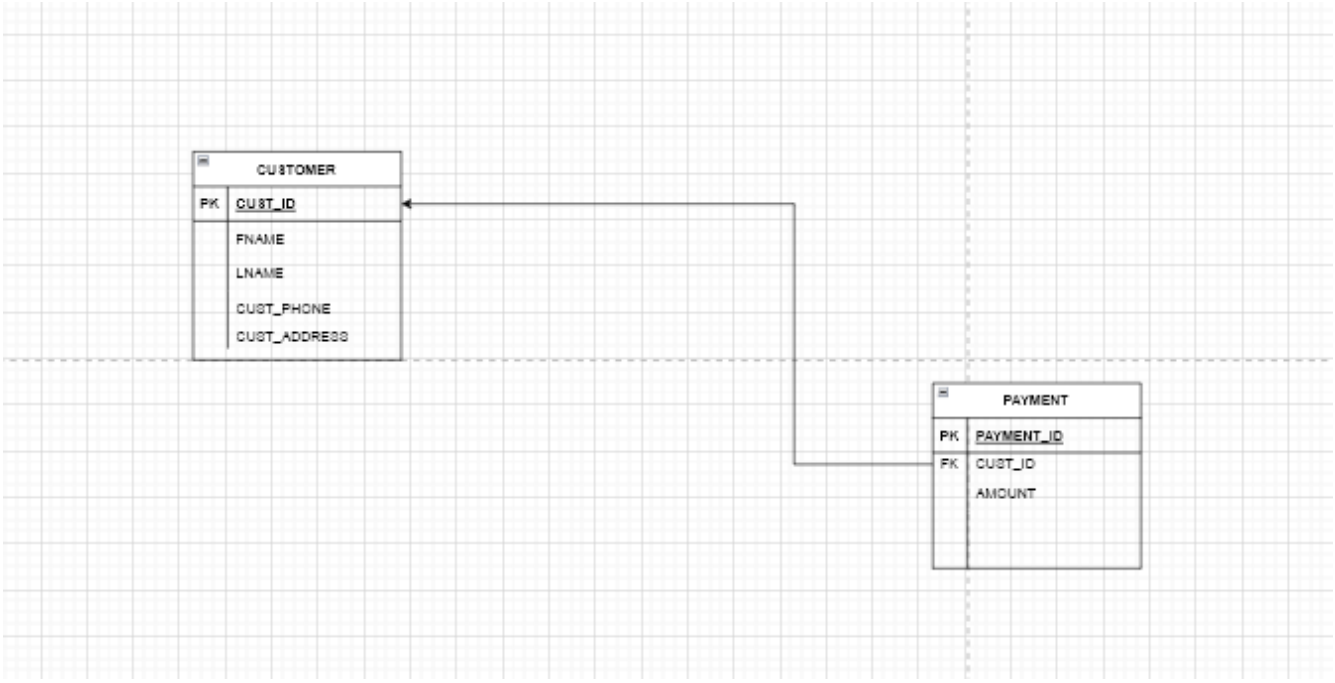
4.1 STEPS OF ALGORITHM FOR CHOOSEN PROBLEM

Step 1: Mapping of Regular Entity Types

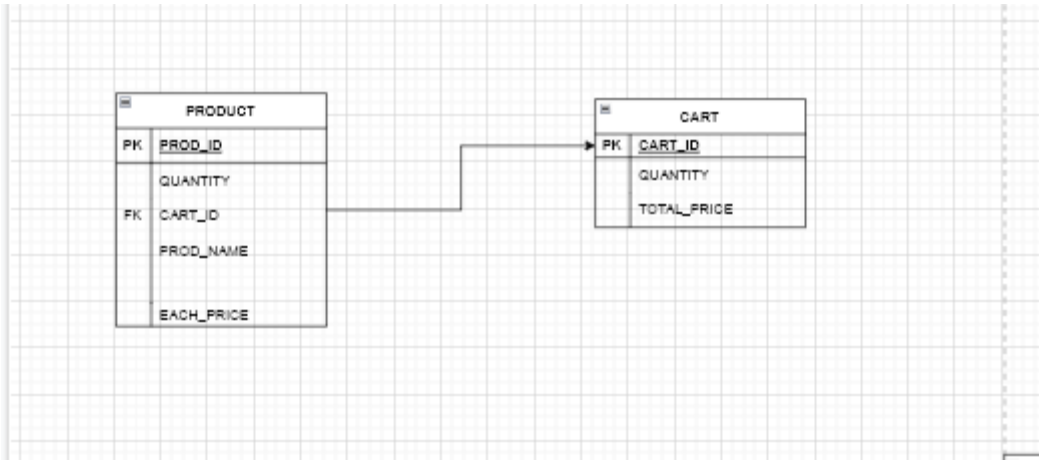


Step 2: Mapping of Weak Entity Types

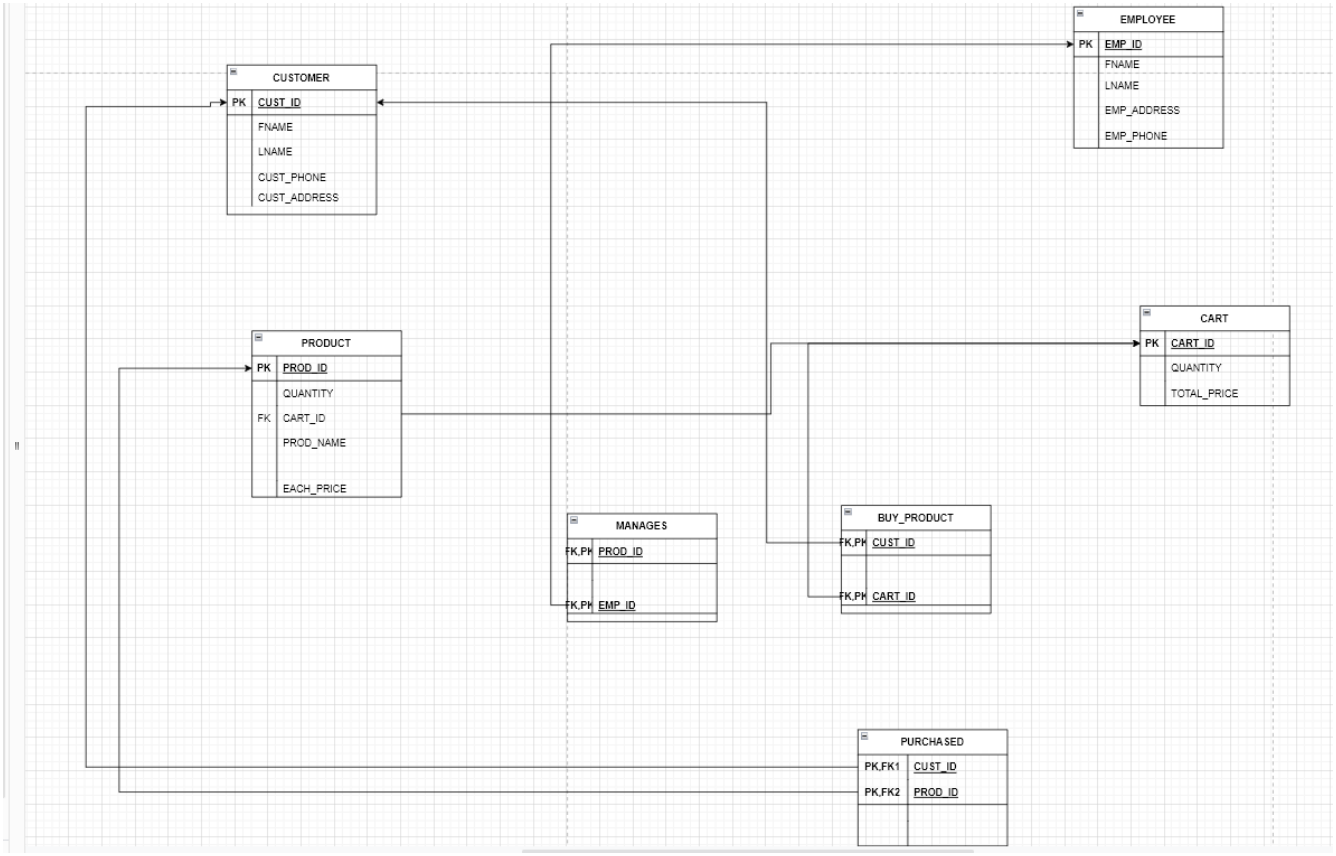
Step 3: Mapping of Binary 1:1 Relation types



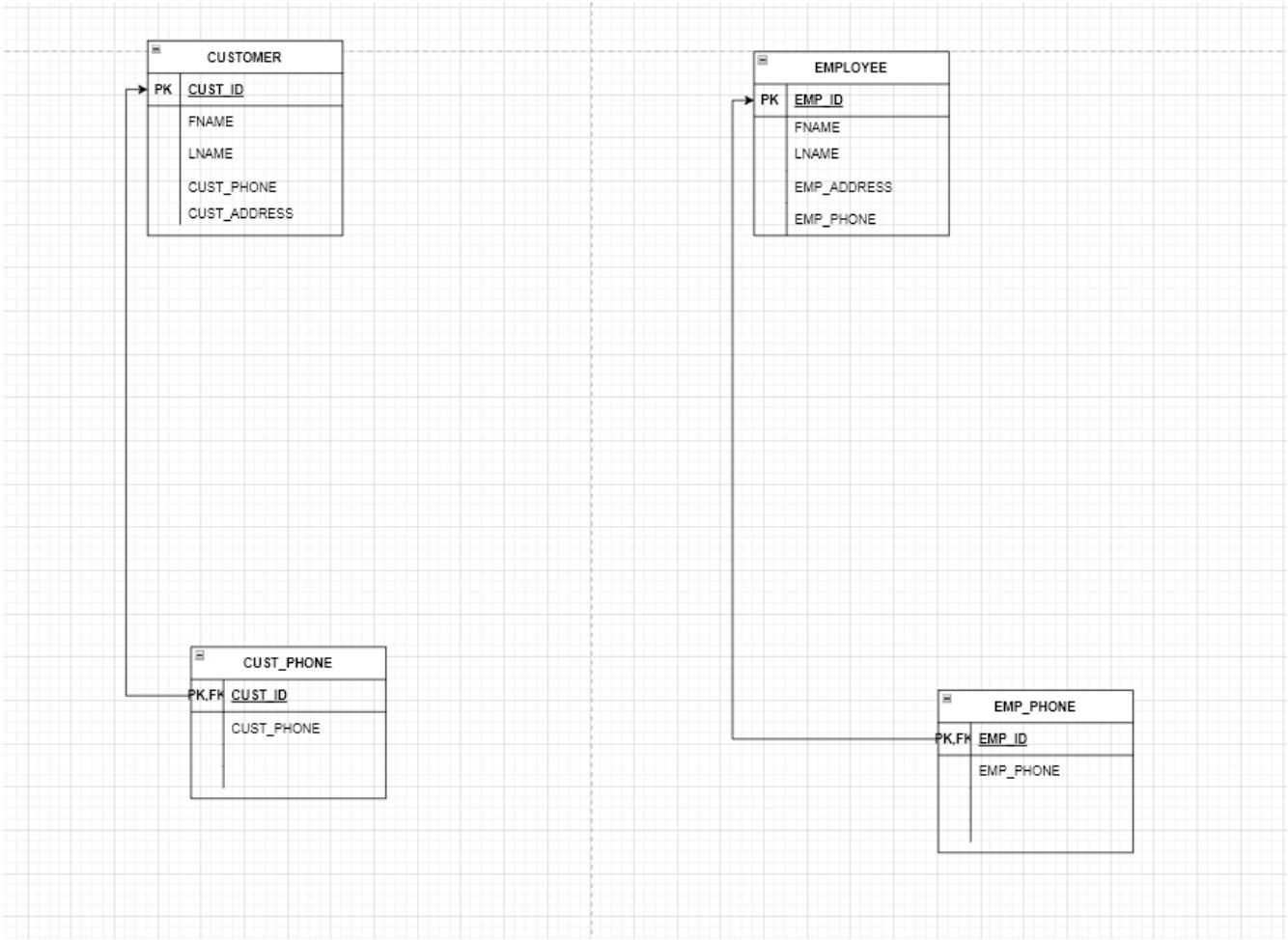
Step 4: Mapping of Binary 1:N Relationship Types.



Step 5: Mapping of Binary M:N Relationship Types.

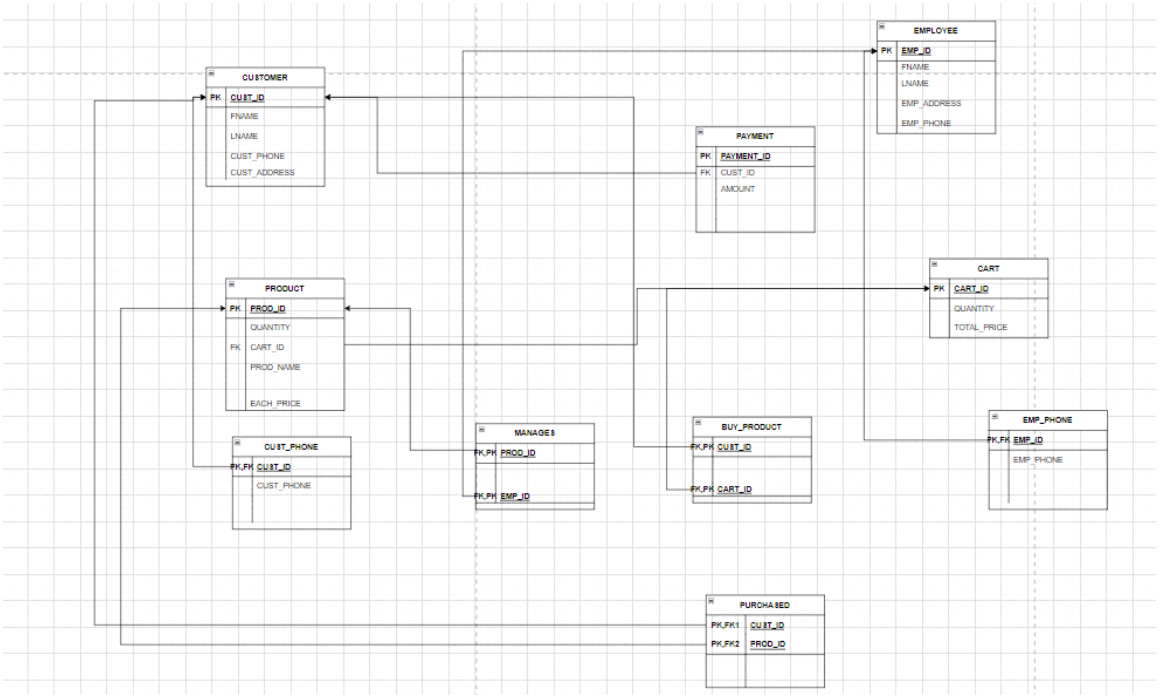


Step 6: Mapping of Multivalued attributes.



Step 7: Mapping of N-ary Relationship Types.

4.2 COMPLETE DIAGRAM OF RELATIONAL MAPPING



5. DDL STATEMENTS

STATEMENTS WITH SCREEN SHOTS OF THE TABLE CREATION

Emp_phone table:

```
create table emp_phone(emp_id int NOT NULL,emp_phone varchar(20) NOT NULL,
FOREIGN KEY (emp_id) REFERENCES employee(emp_id));
--

-- after change
```

```
mysql> desc emp_phone;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| emp_id    | int         | NO   | MUL | NULL    |       |
| emp_phone | varchar(20) | NO   |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Payment table:

```
--


--
create table payment(pay_id int NOT NULL,cust_id int NOT NULL,amount int NOT NULL,
PRIMARY KEY(pay_id),FOREIGN KEY(cust_id) REFERENCES customer(cust_id));
```

```
mysql> desc payment;
+---------+------+------+-----+---------+-------+
| Field   | Type | Null | Key | Default | Extra |
+---------+------+------+-----+---------+-------+
| pay_id  | int  | NO   | PRI | NULL    |       |
| cust_id | int  | NO   | MUL | NULL    |       |
| amount  | int  | NO   |     | NULL    |       |
+---------+------+------+-----+---------+-------+
3 rows in set (0.01 sec)
```

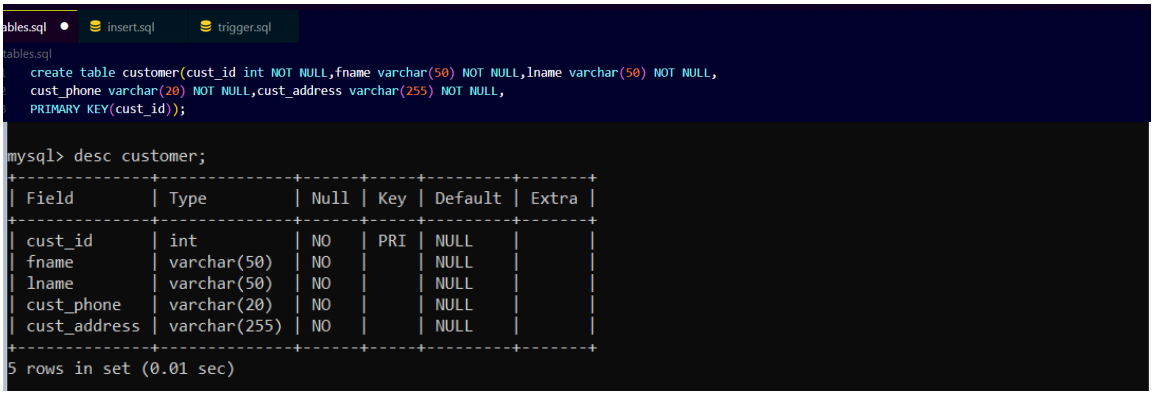
Cust_phone Table:

```
create table cust_phone(cust_id int NOT NULL,cust_phone varchar(20) NOT NULL,
FOREIGN KEY (cust_id) REFERENCES customer(cust_id));
--


--
```

```
mysql> desc cust_phone;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| cust_id    | int         | NO   | MUL | NULL    |       |
| cust_phone | varchar(20) | NO   |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Customer Table:



Manages Table:

Product table:

```
PRIMARY KEY(cust_id));

create table product(prod_id int NOT NULL,cart_id int not null,quantity int NOT NULL,
prod_name varchar(50) NOT NULL,price int NOT NULL,PRIMARY KEY (prod_id),
FOREIGN KEY (cart_id) REFERENCES cart(cart_id));
```

```
mysql> desc product;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| prod_id    | int         | NO   | PRI | NULL    |       |
| quantity   | int         | NO   |     | NULL    |       |
| prod_name  | varchar(50) | NO   |     | NULL    |       |
| each_price | int         | NO   |     | NULL    |       |
| cart_id    | int         | NO   | MUL | NULL    |       |
+------------+-------------+------+-----+---------+-------+
5 rows in set (0.01 sec)
```

Cart table:

```
create table cart(cart_id int NOT NULL,prod_id int NOT NULL,quantity int NOT NULL,
each_price int NOT NULL,total_price int NOT NULL,PRIMARY KEY(cart_id));
```

```
mysql> desc cart;
+-------------+------+------+-----+---------+-------+
| Field       | Type | Null | Key | Default | Extra |
+-------------+------+------+-----+---------+-------+
| cart_id     | int  | NO   | PRI | NULL    |       |
| quantity    | int  | NO   |     | NULL    |       |
| total_price | int  | NO   |     | NULL    |       |
| each_price  | int  | NO   |     | NULL    |       |
+-------------+------+------+-----+---------+-------+
4 rows in set (0.01 sec)
```

Employee table:

```
--
create table employee(emp_id int NOT NULL,fname varchar(50) NOT NULL,lname varchar(50) NOT NULL,
emp_address varchar(255) NOT NULL,emp_phone varchar(20) NOT NULL,PRIMARY KEY(emp_id));
```

```
mysql> desc employee;
+-------------+--------------+------+-----+---------+-------+
| Field       | Type         | Null | Key | Default | Extra |
+-------------+--------------+------+-----+---------+-------+
| emp_id      | int          | NO   | PRI | NULL    |       |
| fname       | varchar(50)  | NO   |     | NULL    |       |
| lname       | varchar(50)  | NO   |     | NULL    |       |
| emp_address | varchar(255) | NO   |     | NULL    |       |
| emp_phone   | varchar(20)  | NO   |     | NULL    |       |
+-------------+--------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

Buy_product table:

```
create table buy_product (cust_id int NOT NULL,cart_id int NOT NULL,
FOREIGN KEY (cust_id) REFERENCES customer(cust_id) on update cascade);
--
```

```
mysql> desc buy_product;
+---------+------+------+-----+---------+-------+
| Field   | Type | Null | Key | Default | Extra |
+---------+------+------+-----+---------+-------+
| cust_id | int  | NO   | MUL | NULL    |       |
| cart_id | int  | NO   | MUL | NULL    |       |
+---------+------+------+-----+---------+-------+
2 rows in set (0.01 sec)
```

Customer Table:

insert into customer

values (1,"nithish","s","8904185217","chikkaballapur"),

(2,"neeraj","gs","123456789","banglore"),

(3,"nilkant","manik","123456897","gulbarga"),

(4,"satish","s","123456879","bidar"),

(5,"allu","arjun","9945116348","hyderabad");

```
mysql> select * from customer;
+---------+---------+-------+------------+----------------+
| cust_id | fname   | lname | cust_phone | cust_address   |
+---------+---------+-------+------------+----------------+
|       1 | nithish | s     | 8904185217 | chikkaballapur |
|       2 | neeraj  | gs    | 123456789  | banglore       |
|       3 | nilkant | manik | 123456897  | gulbarga       |
|       4 | satish  | s     | 123456879  | bidar          |
|       5 | allu    | arjun | 9945116348 | hyderabad      |
+---------+---------+-------+------------+----------------+
5 rows in set (0.00 sec)
```

Product Table:

insert into product

values (1,2000,"tomatoes",50),

(2,2500,"carrot",60),

(3,2370,"brinjal",70),

(4,3700,"onion",80),

(5,5320,"garlic",90);

```
mysql> select * from product;
+---------+----------+-----------+------------+---------+
| prod_id | quantity | prod_name | each_price | cart_id |
+---------+----------+-----------+------------+---------+
|       1 |       50 | tomatoes  |         20 |       1 |
|       2 |      200 | carrot    |         50 |       2 |
|       3 |      300 | brinjal   |         60 |       3 |
|       4 |      400 | onion     |         70 |       4 |
|       5 |       50 | garlic    |         80 |       5 |
+---------+----------+-----------+------------+---------+
5 rows in set (0.01 sec)
```

Cust_phone Table:

insert into cust_phone

values(1,"8904185217"),

(2,"123456789"),

(3,"123456897"),

(4,"123456879"),

(5,"9945116348");

```
mysql> select * from cust_phone;
+---------+------------+
| cust_id | cust_phone |
+---------+------------+
|       1 | 8904185217 |
|       2 | 123456789  |
|       3 | 123456897  |
|       4 | 123456879  |
|       5 | 9945116348 |
+---------+------------+
5 rows in set (0.00 sec)
```

Manages Table :

insert into manages

values (1,1),

(2,2),

(3,3),

(4,4),

(5,5);

```
mysql> select * from manages;
+---------+--------+
| prod_id | emp_id |
+---------+--------+
|       1 |      1 |
|       2 |      2 |
|       3 |      3 |
|       4 |      4 |
|       5 |      5 |
+---------+--------+
5 rows in set (0.01 sec)
```

Payment Table :

insert into payment

values (1,2,1,2000),

(2,3,2,2500),

(3,5,3,2370),

(4,1,4,3700),

(5,4,5,5320),

(6,2,3,2370),--

(7,2,4,3700),

(8,1,1,2000),

(9,1,3,2370),

(10,3,5,5320);

```
mysql> select * from payment;
+--------+---------+--------+
| pay_id | cust_id | amount |
+--------+---------+--------+
|      1 |       2 |   2000 |
|      2 |       3 |   2500 |
|      3 |       5 |   2370 |
|      4 |       1 |   3700 |
|      5 |       4 |   5320 |
|      6 |       2 |   2370 |
|      7 |       2 |   3700 |
|      8 |       1 |   2000 |
|      9 |       1 |   2370 |
|     10 |       3 |   5320 |
+--------+---------+--------+
10 rows in set (0.00 sec)
```

Buy_product Table :

insert into buy_product
values (1,3),
(2,5),
(5,1),
(3,2),
(4,4);

```
mysql> select * from buy_product;
+---------+---------+
| cust_id | cart_id |
+---------+---------+
|       1 |       3 |
|       2 |       5 |
|       5 |       1 |
|       3 |       2 |
|       4 |       4 |
+---------+---------+
5 rows in set (0.03 sec)

mysql>
```

Employee Table :

insert into employee
values (1,"pratham","hegde","sringeri","8431218630"),
(2,"david","guetta",'nigeria','08144640489'),
(3,"john",'Even',"banglore",'07043610459'),
(4,'Mary','Evan','bidar','91359403129'),
(5,'Eastwood','Morn','bellari','6151690314');

```
mysql> select * from employee;
+--------+---------+-------+-------------+-------------+
| emp_id | fname   | lname | emp_address | emp_phone   |
+--------+---------+-------+-------------+-------------+
|      1 | pratham | hegde | sringeri    | 8431218630  |
|      2 | david   | guetta | nigeria    | 08144640489 |
|      3 | john    | Even  | banglore    | 07043610459 |
|      4 | Mary    | Evan  | bidar       | 91359403129 |
|      5 | Eastwood | Morn | bellari     | 6151690314  |
+--------+---------+-------+-------------+-------------+
5 rows in set (0.00 sec)
```

Cart Table :

insert into cart
values (1,1,15,2000),
(2,2,30,2500),
(3,3,20,2370),
(4,4,18,3700),
(5,5,26,5320);

```
mysql> select * from cart;
+---------+----------+-------------+------------+
| cart_id | quantity | total_price | each_price |
+---------+----------+-------------+------------+
|       1 |       15 |        2000 |         20 |
|       2 |       30 |        2500 |         50 |
|       3 |       20 |        2370 |         60 |
|       4 |       18 |        3700 |         70 |
|       5 |       26 |        5320 |         80 |
|       6 |       20 |        1200 |         60 |
|       7 |       10 |         700 |         70 |
|       8 |        9 |         450 |         50 |
+---------+----------+-------------+------------+
8 rows in set (0.01 sec)
```

Emp_phone Table :

insert into emp_phone
values (1,"8431218630"),
(2,'08144640489'),
(3,'07043610459'),
(4,'91359403129'),
(5,'6151690314');

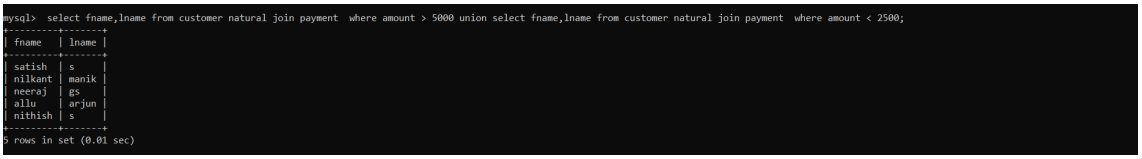
```
mysql> select * from emp_phone;
+--------+-------------+
| emp_id | emp_phone   |
+--------+-------------+
|      1 | 8431218630  |
|      2 | 08144640489 |
|      3 | 07043610459 |
|      4 | 91359403129 |
|      5 | 6151690314  |
+--------+-------------+
5 rows in set (0.00 sec)
```

QUERIES

SET OPERATION

Q1) select customer names who made payment above 5000 or less than 2500.

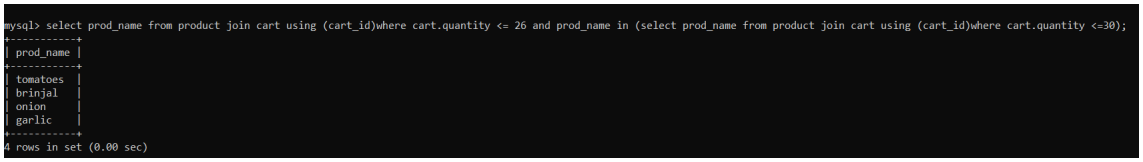
```
select fname,lname
from customer
natural join payment
where amount > 5000
union
select fname,lname
from customer
natural join payment
where amount < 2500;
```



```
mysql>  select fname,lname from customer natural join payment  where amount > 5000 union select fname,lname from customer natural join payment  where amount < 2500;
+---------+-------+
| fname   | lname |
+---------+-------+
| satish  | s     |
| nilkant | manik |
| neeraj  | gs    |
| allu    | arjun |
| nithish | s     |
+---------+-------+
5 rows in set (0.01 sec)
```

Q2) select prodcut name whose quantity in the cart is atmost 26 and atmost 30.

```
select prod_name
from product
join cart using (cart_id)
where cart.quantity <= 26 and prod_name in (
select prod_name
from product
join cart using (cart_id)
where cart.quantity <=30);
```



```
mysql> select prod_name from product join cart using (cart_id)where cart.quantity <= 26 and prod_name in (select prod_name from product join cart using (cart_id)where cart.quantity <=30);
+-----------+
| prod_name |
+-----------+
| tomatoes  |
| brinjal   |
| onion     |
| garlic    |
+-----------+
4 rows in set (0.00 sec)
```

NESTED

Select name of customer ,who has made payment for product.

```
select fname,lname
from customer
where cust_id in (select cust_id from payment);
```

```
mysql> select fname,lname from customer where cust_id in (select cust_id from payment);
+---------+-------+
| fname   | lname |
+---------+-------+
| nithish | s     |
| neeraj  | gs    |
| nilkant | manik |
| satish  | s     |
| allu    | arjun |
+---------+-------+
5 rows in set (0.00 sec)
```

CORRELATED QUERY

Select products that are present in the customers cart.

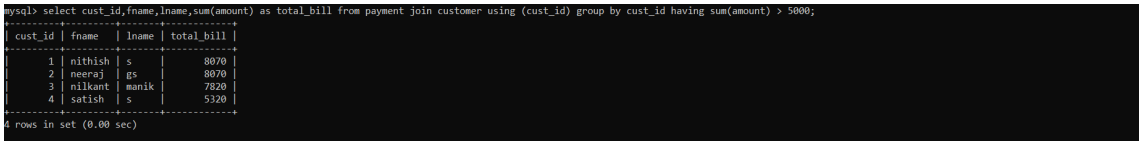
```
select prod_name
from product
where exists (select * from cart);
```

```
mysql> select prod_name from product where exists (select * from cart);
+-----------+
| prod_name |
+-----------+
| tomatoes  |
| carrot    |
| brinjal   |
| onion     |
| garlic    |
+-----------+
5 rows in set (0.02 sec)
```

AGGREGATED QUERY

1) Display the total amount paid by the each customer for all their products.

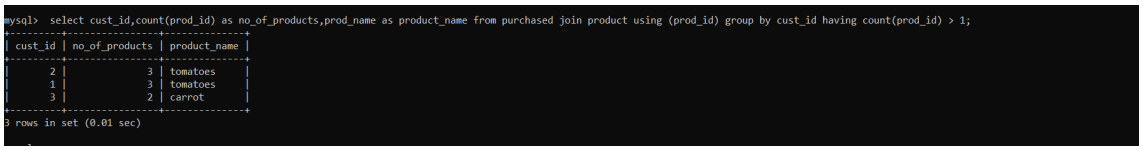
```
select cust_id,fname,lname,sum(amount) as total_bill
from payment
join customer
using (cust_id)
group by cust_id
having sum(amount) > 5000;
```



```
mysql> select cust_id,fname,lname,sum(amount) as total_bill from payment join customer using (cust_id) group by cust_id having sum(amount) > 5000;
+---------+---------+-------+------------+
| cust_id | fname   | lname | total_bill |
+---------+---------+-------+------------+
|       1 | nithish | s     |       8070 |
|       2 | neeraj  | gs    |       8070 |
|       3 | nilkant | manik |       7820 |
|       4 | satish  | s     |       5320 |
+---------+---------+-------+------------+
4 rows in set (0.00 sec)
```

2) Display the number of products and name of that product that the customer had bought.

```
select cust_id,count(prod_id) as no_of_products,prod_name as product_name
from purchased
join product using (prod_id)
group by cust_id
having count(prod_id) > 1;
```

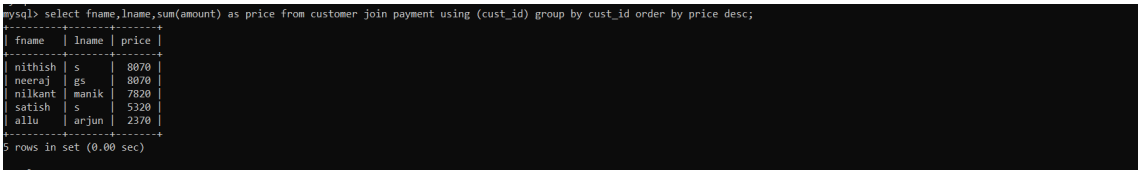


```
mysql>  select cust_id,count(prod_id) as no_of_products,prod_name as product_name from purchased join product using (prod_id) group by cust_id having count(prod_id) > 1;
+---------+---------------+--------------+
| cust_id | no_of_products | product_name |
+---------+---------------+--------------+
|       2 |             3 | tomatoes     |
|       1 |             3 | tomatoes     |
|       3 |             2 | carrot       |
+---------+---------------+--------------+
3 rows in set (0.01 sec)
```

ORDER BY

1) Display the name of customers and their total price in ascending order.

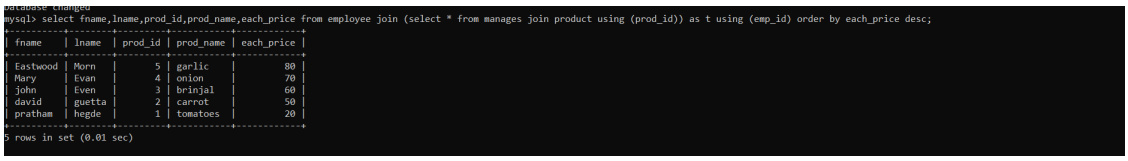
```
select fname,lname,sum(amount) as price
from customer
join payment
using (cust_id)
group by cust_id
order by price desc;
```



```
mysql> select fname,lname,sum(amount) as price from customer join payment using (cust_id) group by cust_id order by price desc;
+---------+-------+-------+
| fname   | lname | price |
+---------+-------+-------+
| nithish | s     |  8070 |
| neeraj  | gs    |  8070 |
| nilkant | manik |  7820 |
| satish  | s     |  5320 |
| allu    | arjun |  2370 |
+---------+-------+-------+
5 rows in set (0.00 sec)
```

2) Display the name of employee, who manages the stock of product of higher price along with name of product.

```
select fname,lname,prod_id,prod_name,each_price
from employee
join (select * from manages join product using (prod_id)) as t
using (emp_id)
order by each_price desc;
```



```
Database changed
mysql> select fname,lname,prod_id,prod_name,each_price from employee join (select * from manages join product using (prod_id)) as t using (emp_id) order by each_price desc;
+----------+--------+---------+-----------+------------+
| fname    | lname  | prod_id | prod_name | each_price |
+----------+--------+---------+-----------+------------+
| Eastwood | Morn   |       5 | garlic    |         80 |
| Mary     | Evan   |       4 | onion     |         70 |
| john     | Even   |       3 | brinjal   |         60 |
| david    | guetta |       2 | carrot    |         50 |
| pratham  | hegde  |       1 | tomatoes  |         20 |
+----------+--------+---------+-----------+------------+
5 rows in set (0.01 sec)
```

STORED PROCEDURES, FUCNTIONS AND TRIGGERS

FUNCTION

1) Find the total_price of each prodcut with respect to their quantities and each price of product.

```
delimiter $$
create function total_price(price int,quantity int)
returns int
deterministic
begin
return price*quantity;
end $$

--
select prod_name,total_price(each_price,quantity) as stock_price
from product;
```

```
mysql> select prod_name,total_price(each_price,quantity) as stock_price
    -> from product;
+-----------+-------------+
| prod_name | stock_price |
+-----------+-------------+
| tomatoes  |        1000 |
| carrot    |       10000 |
| brinjal   |       18000 |
| onion     |       28000 |
| garlic    |        4000 |
+-----------+-------------+
5 rows in set (0.07 sec)
```

TRIGGERS

1) Trigger is created based on when the quantities in the product table changes, then that is inserted into the new table producChanges for easy identification of change in quantities with respect to product ,before and after change of quantities.

```
use grocery;
DELIMITER $$

CREATE TRIGGER after_productChanges_update
AFTER UPDATE
ON product FOR EACH ROW
BEGIN
   IF OLD.quantity <> new.quantity THEN
      INSERT INTO productChanges(prod_id,beforeQuantity, afterQuantity)
      VALUES(old.prod_id, old.quantity, new.quantity);
   END IF;
END$$

DELIMITER ;
```

before update :

```
mysql> select * from product;
+---------+----------+-----------+------------+---------+
| prod_id | quantity | prod_name | each_price | cart_id |
+---------+----------+-----------+------------+---------+
|       1 |       50 | tomatoes  |         20 |       1 |
|       2 |      200 | carrot    |         50 |       2 |
|       3 |      300 | brinjal   |         60 |       3 |
|       4 |      400 | onion     |         70 |       4 |
|       5 |       50 | garlic    |         80 |       5 |
+---------+----------+-----------+------------+---------+
5 rows in set (0.00 sec)
```

after update :

```
mysql> select * from product;
+---------+----------+-----------+------------+---------+
| prod_id | quantity | prod_name | each_price | cart_id |
+---------+----------+-----------+------------+---------+
|       1 |       50 | tomatoes  |         20 |       1 |
|       2 |      200 | carrot    |         50 |       2 |
|       3 |      300 | brinjal   |         60 |       3 |
|       4 |      400 | onion     |         70 |       4 |
|       5 |       50 | garlic    |         80 |       5 |
+---------+----------+-----------+------------+---------+
5 rows in set (0.00 sec)
```

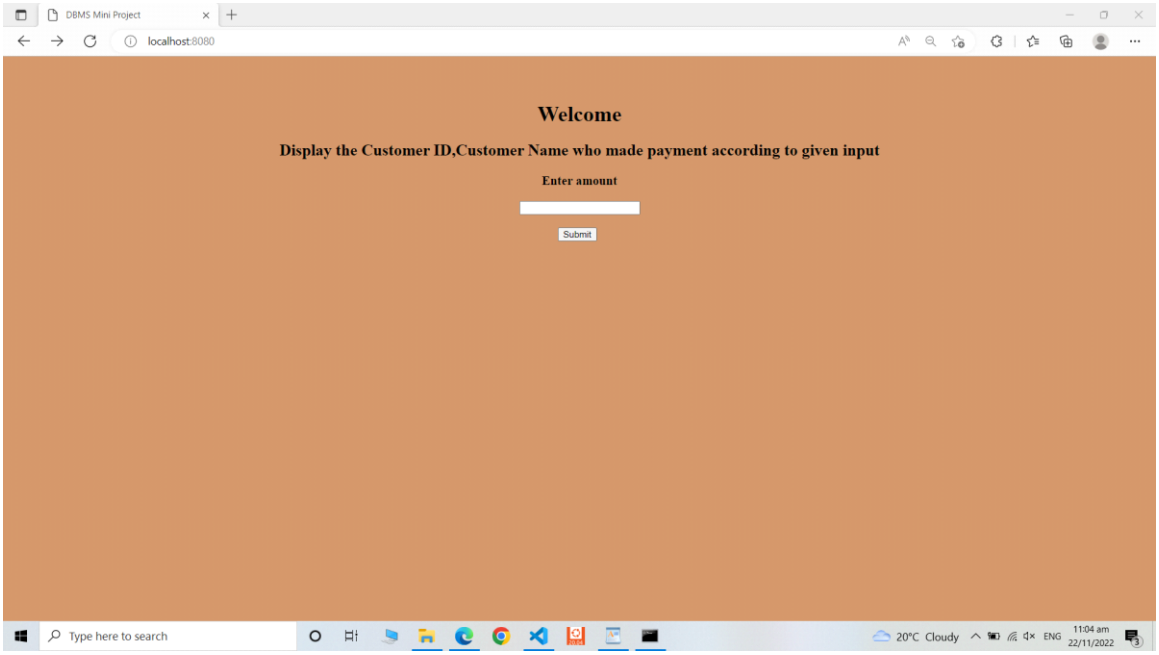
```
mysql> select * from product;
+---------+----------+-----------+------------+---------+
| prod_id | quantity | prod_name | each_price | cart_id |
+---------+----------+-----------+------------+---------+
|       1 |     1000 | tomatoes  |         20 |       1 |
|       2 |      200 | carrot    |         50 |       2 |
|       3 |      300 | brinjal   |         60 |       3 |
|       4 |      400 | onion     |         70 |       4 |
|       5 |     1000 | garlic    |         80 |       5 |
+---------+----------+-----------+------------+---------+
5 rows in set (0.00 sec)
```

productChanges Table :

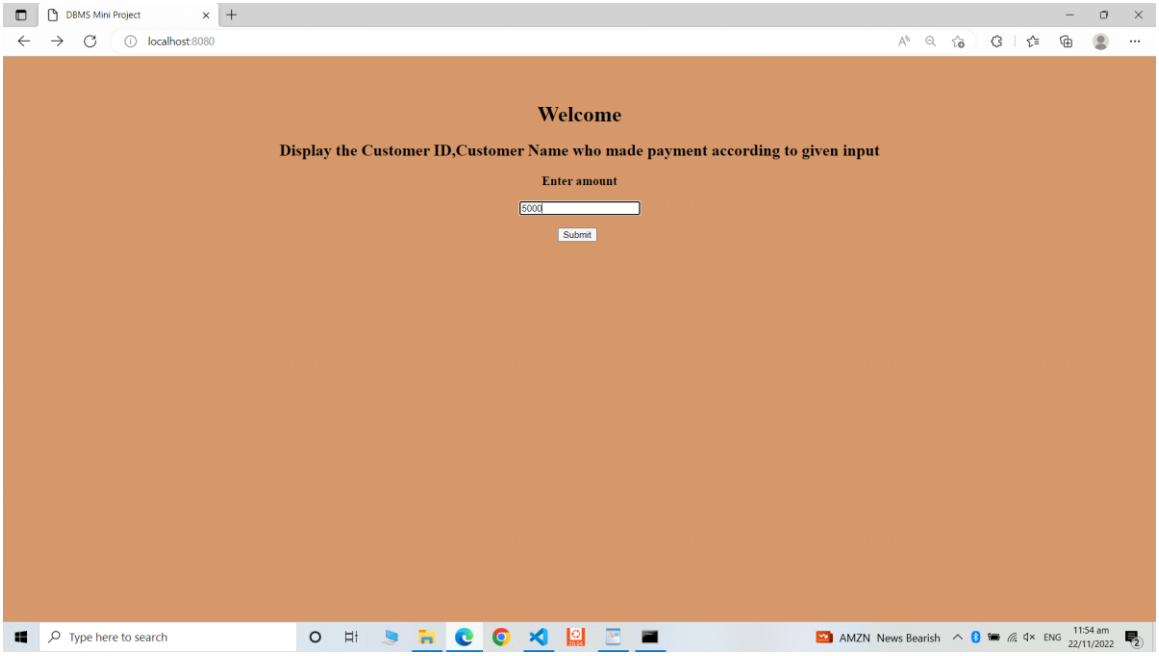
```
mysql> select * from productChanges;
+-------+---------+---------------+---------------+
| pc_id | prod_id | beforeQuantity | afterQuantity |
+-------+---------+---------------+---------------+
|     1 |       1 |           100 |            50 |
|     2 |       5 |           500 |            50 |
|     3 |       1 |            50 |          1000 |
|     4 |       5 |            50 |          1000 |
+-------+---------+---------------+---------------+
4 rows in set (0.00 sec)
```

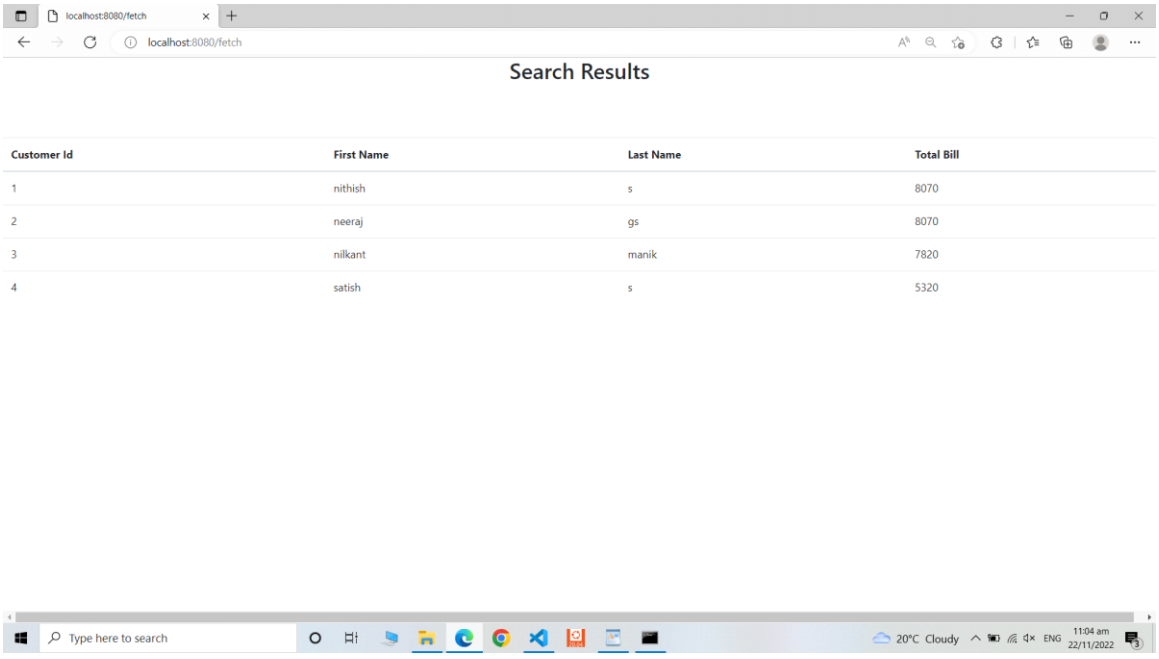
9. FRONT END DEVELOPEMNT

Front-end :

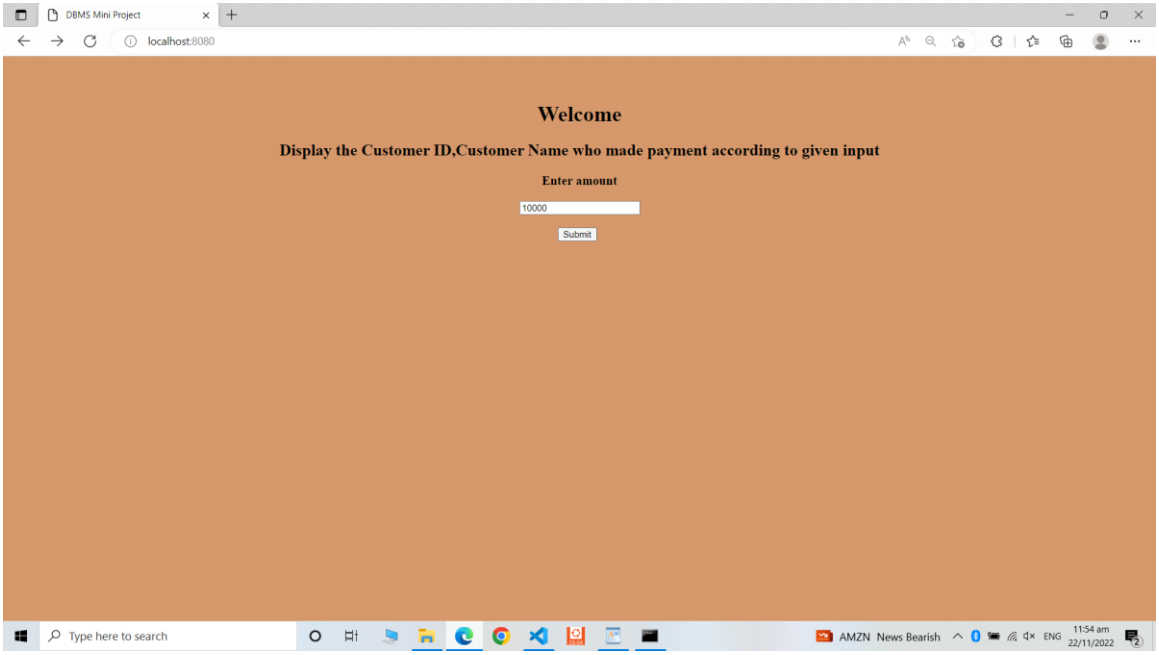


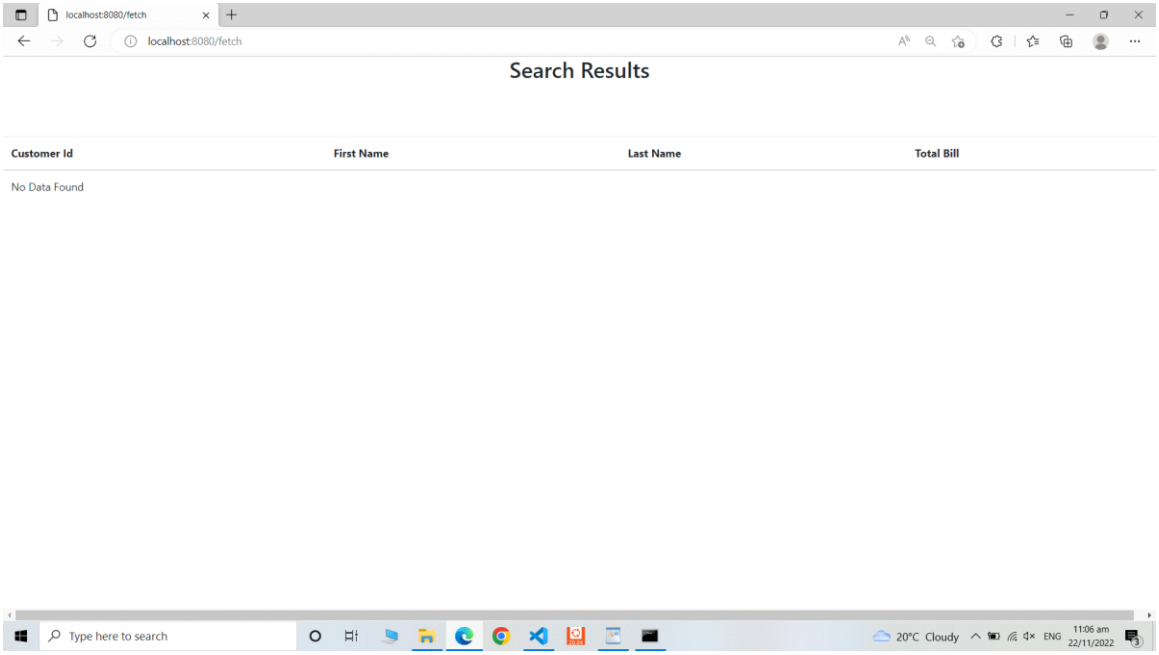
For Valid Input :





For Invalid Input :





REFERENCES

[1]