DBMS - Mini Project
Fast Food Delivery Database

Submitted By:

Name: Shreyash Chatterjee
SRN: PES1UG20CS410
Semester: 5th
Section: G

Short Description and Scope

This project focuses on creating a simple and easy to use database implementation of a Fast Food Delivery Service along with an easy to use implementation of general purpose User Interface to manipulate the Database.

This database includes many handy triggers which automatically updates prices and their corresponding discounts when added to the the cart, so as to minimise manual calculations, insertions and updations.

Be it adding or deleting item from the cart, the triggers and functions implemented handle every discount recalculation and price updation efficiently.

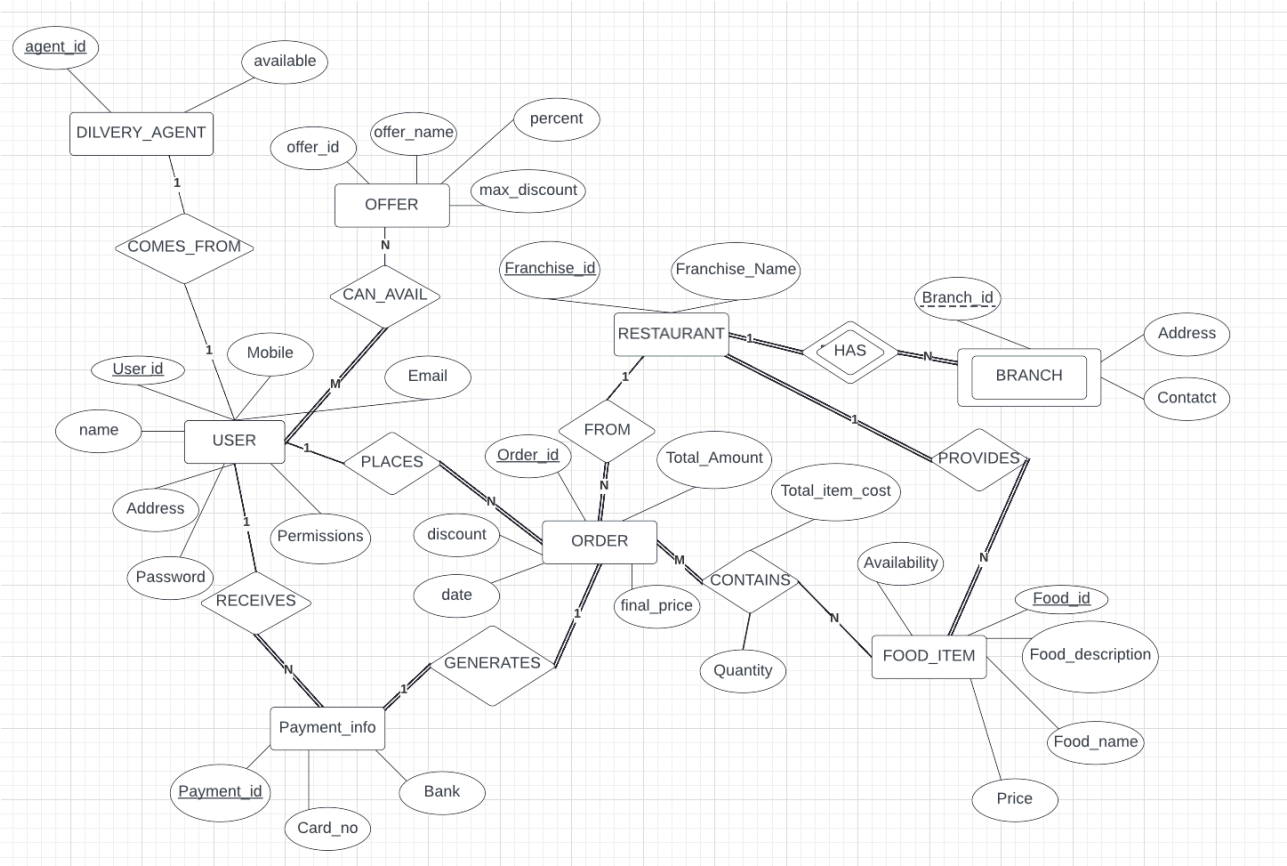
The database also includes necessary error handling required to ensure many essentials constraints are not broken, such as a user adding a food item from any other restaurant in the same order, if any updation is being done to any order which already has been ordered and paid for, to only accept payment for an order if there are delivery agents available to deliver, whether the delivery agents being added have indeed applied for being a delivery_agent and exist in the user table, and much more!

This database combines with an easy to use and feature rich front end, which has the capability to add, remove and edit any database records. Additionally it has the capability to execute any SQL command, if the user who is using it is an administrator.

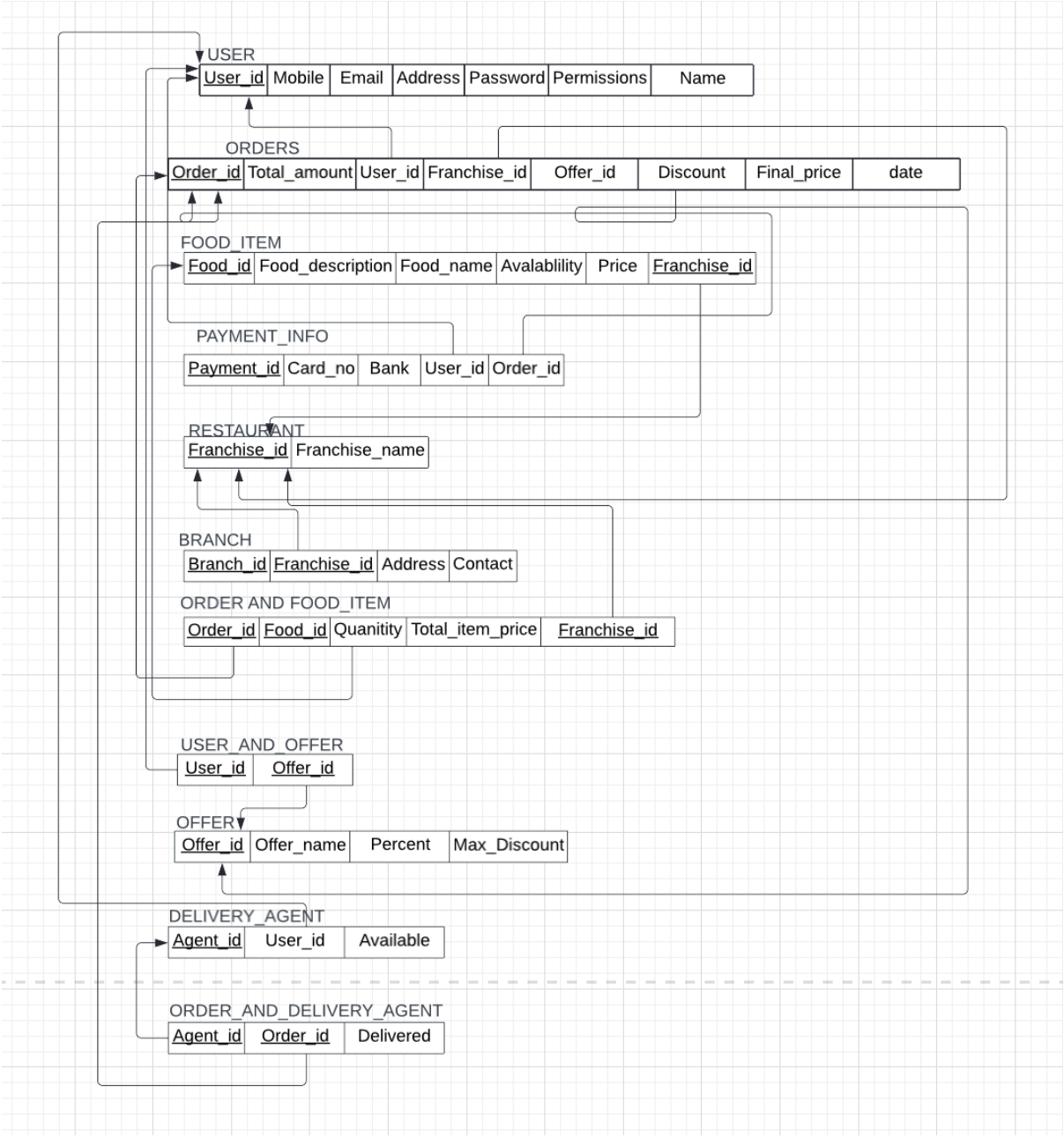
To implement this functionality of having different ‘views’ for different user, the front end also implements a login page, which fetches it’s data from the user table. Once logged in, according to the logged in user’s permissions, a different view is shown to them.

The future scope of this project can be to implement a mobile app for the same and have different types of forms for user, admin and delivery_agent to compete with popular apps like Swiggy. Integration with google maps can also be done on the app.

ER diagram


agent_id
available
DILVERY_AGENT
offer_id
offer_name
percent
max_discount
OFFER
COMES_FROM
CAN_AVAIL
Franchise_id
Franchise_Name
Branch_id
RESTAURANT
HAS
BRANCH
Address
Contatct
User id
Mobile
Email
name
USER
PLACES
FROM
Order_id
Total_Amount
PROVIDES
Total_item_cost
Address
Permissions
discount
ORDER
Availability
Password
RECEIVES
date
final_price
CONTAINS
Food_id
GENERATES
Quantity
FOOD_ITEM
Food_description
Payment_info
Food_name
Payment_id
Bank
Price
Card_no

Relational Schema



DDL Statements - Building the Database

app_copy.py | functions.py | table_creation.sql | database.py

```sql
create table USER(
    user_id varchar(255) PRIMARY KEY,
    mobile int(10),
    email varchar(255),
    address varchar(255),
    password varchar(255),
    permissions varchar(255),
    name varchar(255)
);

create table RESTAURANT(
    franchise_id int(10) PRIMARY KEY,
    franchise_name varchar(255)
);

create table ORDERS(
    order_id int(10) PRIMARY KEY,
    total_amount int(10),
    user_id varchar(255),
    franchise_id int(10),
    offer_id int (10),
    discount int(10) default 0,
    final_price int(10) default 0,
    date varchar(255),
    FOREIGN KEY (user_id) REFERENCES USER(user_id),
    FOREIGN KEY (franchise_id) REFERENCES RESTAURANT(franchise_id),
    FOREIGN KEY (offer_id) REFERENCES OFFER(offer_id)
);

create table PAYMENT_INFO(
    payment_id int(10) PRIMARY KEY,
    order_id int(10) UNIQUE,
    bank varchar(255),
    user_id varchar(255),
    card_no varchar(16),
    FOREIGN KEY (user_id) REFERENCES USER(user_id),
    FOREIGN KEY (order_id) REFERENCES ORDERS(order_id)
);
```

```sql
create table FOOD_ITEM(
    food_id int(10),
    food_description varchar(255),
    food_name varchar(255),
    franchise_id int(10),
    availability varchar(255),
    price int(10),
    FOREIGN KEY (franchise_id) REFERENCES RESTAURANT(franchise_id),
    PRIMARY KEY (food_id, franchise_id)
);

create table BRANCH(
    branch_id int(10),
    franchise_id int(10),
    address varchar(255),
    contact varchar(10),
    PRIMARY KEY(branch_id, franchise_id),
    FOREIGN KEY (franchise_id) references RESTAURANT(franchise_id)
);

create table ORDER_AND_FOOD_ITEM(
    order_id int(10),
    food_id int(10),
    franchise_id int(10),
    quantity int(10),
    total_item_price int(10),
    FOREIGN KEY (order_id) references ORDERS(order_id),
    FOREIGN KEY (food_id) references FOOD_ITEM(food_id),
    FOREIGN KEY (franchise_id) references RESTAURANT(franchise_id),
    PRIMARY KEY (order_id, food_id, franchise_id)
);

create table OFFER(
    offer_id int(10) PRIMARY KEY,
    offer_name varchar(255),
    percent float,
    max_discount int(10)
);
```

```sql
create table USER_AND_OFFER(
    user_id varchar(255),
    offer_id int(10),
    PRIMARY KEY (user_id, offer_id),
    FOREIGN KEY (user_id) references USER(user_id),
    FOREIGN KEY (offer_id) references OFFER(offer_id)
);

create table DELIVERY_AGENT(
    agent_id int(10),
    user_id varchar(255) UNIQUE,
    available int(10) default 1,
    PRIMARY KEY (agent_id),
    FOREIGN KEY (user_id) references user(user_id)
);

create table ORDER_AND_DELIVERY_AGENT(
    order_id int(10),
    agent_id int(10),
    delivered int(10) default 0,
    PRIMARY KEY (order_id, agent_id),
    FOREIGN KEY (agent_id) references DELIVERY_AGENT(agent_id),
    FOREIGN KEY (order_id) references ORDERS(order_id)
);
```

Populating the database

```
MariaDB [fast_food]> insert into order_and_foot_item (order_id, food_id, franchise_id, quantity) values (100, 1, 2, 3);
ERROR 1146 (42S02): Table 'fast_food.order_and_foot_item' doesn't exist
MariaDB [fast_food]> insert into order_and_food_item (order_id, food_id, franchise_id, quantity) values (100, 1, 2, 3);
Query OK, 1 row affected (0.008 sec)

MariaDB [fast_food]> insert into order_and_food_item (order_id, food_id, franchise_id, quantity) values (100, 2, 2, 2);
Query OK, 1 row affected (0.003 sec)

MariaDB [fast_food]> insert into order_and_food_item (order_id, food_id, franchise_id, quantity) values (100, 3, 2, 1);
Query OK, 1 row affected (0.003 sec)

MariaDB [fast_food]> insert into order_and_food_item (order_id, food_id, franchise_id, quantity) values (101, 3, 4, 2);
Query OK, 1 row affected (0.003 sec)

MariaDB [fast_food]> insert into order_and_food_item (order_id, food_id, franchise_id, quantity) values (101, 1, 4, 1);
Query OK, 1 row affected (0.003 sec)

MariaDB [fast_food]> insert into order_and_food_item (order_id, food_id, franchise_id, quantity) values (102, 2, 9, 2);
Query OK, 1 row affected (0.003 sec)

MariaDB [fast_food]> insert into order_and_food_item (order_id, food_id, franchise_id, quantity) values (103, 4, 7, 1);
Query OK, 1 row affected (0.003 sec)

MariaDB [fast_food]> insert into order_and_food_item (order_id, food_id, franchise_id, quantity) values (103, 5, 7, 2);
Query OK, 1 row affected (0.003 sec)

MariaDB [fast_food]> insert into order_and_food_item (order_id, food_id, franchise_id, quantity) values (104, 5, 1, 4);
Query OK, 1 row affected (0.003 sec)

MariaDB [fast_food]> insert into order_and_food_item (order_id, food_id, franchise_id, quantity) values (104, 1, 1, 2);
Query OK, 1 row affected (0.003 sec)

MariaDB [fast_food]> insert into order_and_food_item (order_id, food_id, franchise_id, quantity) values (105, 2, 3, 1);
Query OK, 1 row affected (0.004 sec)

MariaDB [fast_food]> insert into order_and_food_item (order_id, food_id, franchise_id, quantity) values (106, 3, 1, 2);
Query OK, 1 row affected (0.003 sec)

MariaDB [fast_food]> insert into order_and_food_item (order_id, food_id, franchise_id, quantity) values (106, 2, 1, 2);
Query OK, 1 row affected (0.004 sec)

MariaDB [fast_food]> insert into order_and_food_item (order_id, food_id, franchise_id, quantity) values (107, 2, 8, 1);
Query OK, 1 row affected (0.002 sec)
```

The following were added by importing in the table -

food_item

1	**Signature burger made with a crunchy chicken fillet and veggies & a delicious mayo sauce**	**Classic Zinger Burger**	**1**	**1**	**190**
2	Signature veg burger with crispy patties and veggies & a tangy sauce	Veg Zinger Burger	1	1	180
3	Hyderabadi style Biryani rice served with 1 pc Hot & Crispy & Spicy Gravy	Classic Chicken Biryani Bucket	1	1	230
4	Hyderabadi style Biryani rice served with Veg Patty & Spicy Gravy	Veg Biryani Bucket	1	1	189
5	Soft Chocolate cake with a gooey center- perfect chocolaty end to every meal	Chocolate Lava Cake	1	1	109
1	Tender and juicy boneless chicken thigh meat coated in crispy batter with a kick of spice topped with a creamy sauce and crispy shredded lettuce will have you craving for more!	McSpicy Chicken Burger	2	1	185
2	Rich and filling cottage cheese patty coated in spicy and crispy batter topeed with a creamy sauce and crispy shredded lettuce will have you craving for more	Mcspicy Paneer Burger	2	1	189
3	The McChicken is a delightfully crispy chicken sandwich with a crispy chicken fillet topped with mayonnaise and shredded iceberg lettuce and served on a perfectly toasty bun	McChicken Burger	2	1	131
4	A delectable patty made of green goodness and potatoes and peas and carrots and selection of Indian spices. Topped with crispy lettuce and mayonnaise and packed into sesame toasted buns	Mcveggie Burger	2	1	145
5	The crisp and craveable and fan favorite. World Famous Fries.Our French fries are born from premium potatoes such as the Russet Burbank and the Shepody. With 0g of trans fat per labeled serving these epic fries are crisp	Regular French Fries	2	1	109
1	Crispy hashbrowns	Hashbrown	3	1	39
2	Our Best-Selling Burger With Crispy Veg Double Patty and Fresh Onion And Our Signature Sauce and Crispy Veg Double Patty Burger	Crispy Veg Double Patty Burger	3	1	90
3	Our Big Burger With Flame Grilled Smoky Chicken Patty & Loads Of Sauces Makes It A Grill Treat and Big Smoky Grill Chicken Burger	Big Smoky Grill Chicken Burger	3	1	159
4	Our tribute to classic american taste. BK veg patty and garden fresh crispy lettuce and juicy tomato(seasonal)and Cheese slice & our signature sauce	Bk Classic Veg with cheese	3	1	119
5	Our Signature Whopper With 7 Layers Between The Buns In A Convenient Size. Flame Grilled Chicken Patty and Fresh Onion and Crispy Lettuce and Juicy Tomatoes(seasonal) and Tangy Gherkins and Creamy And Smoky S	Lite Whopper Jr Chicken	3	1	139
1	Classic delight with 100% real mozzarella cheese	Margherita	4	1	299
2	Grilled chicken rashers and peri-peri chicken and onion & capsicum and a complete fiesta	Chicken Fiesta	4	1	629
3	Black olives and capsicum and onion and grilled mushroom and corn and tomato and jalapeno & extra cheese	Veg Extravaganza	4	1	629
1	Nothing but cheese!	Margherita	5	1	169
2	Cheese & sweet corn	Corn & Cheese	5	1	219
3	Herbed onion and green capsicum and sweet corn	Veggie Feast	5	1	259
4	Schezwan chicken meatball & onion	Spiced Chicken Meatballs	5	1	319
1	A strong shot of Italian styled espresso evened out with steamed and foamed milk. And Serving Size(gm/ml) - 250 and Energy (kcal) - 128.9 Contains Milk	Cappuccino	6	1	199
2	Stretched espresso with goodness of honey and cinnamon. Serving Size(gm/ml) - 180 and Energy (kcal) - 55.3	Honey Cinnamon Coffee	6	1	189
3	For the health conscious and loaded with the antioxidants of green coffee & freshness of lemon and served cold. Serving Size(gm/ml) - 210 and Energy (kcal) - 85.27	Lemmon Green Coffee	6	1	189
4	Single shot of espresso and ice cold water make an addictive beverage to beat the heat. Serving Size(gm/ml) - 300 and Energy (kcal) - 52.48	Iced Americano	6	1	199
1	The Soft taco has a warm and flour tortilla and served with Mexican paneer & lava sauce	Soft Taco Mexican Paneer	7	1	105
2	The soft taco has a warm and flour tortilla and served with Grilled chicken & lava sauce	Soft Taco Grilled Chicken	7	1	109
3	It's crunchy. It's delicious! Our signature product served with Fajita veggies & zesty ranch sauce	Crunchy Taco Fajita Veg	7	1	105
4	The Soft taco has a warm and flour tortilla and served with Mexican Chicken & lava sauce	Soft Taco Mexican Chicken	7	1	109
5	Sriracha Burrito is a soft tortilla roll with delicious Fajita Veg filling topped with Tangy and Spicy Sriracha Sauce	Sriracha Burrito	7	1	109
1	Introducing the new small size from Starbucks - Picco Cappuccino Dark and rich espresso lies in wait under a smoothed and stretched layer of thick foam. It's truly the height of our baristas' craft	Picco Cappuccino	8	1	175
2	Rich and full-bodied espresso with hot water in true European style	Espresso	8	1	190
1	Delicious Combination Of Fresh Lettuce and Tomatoes and Green Peppers and Onions and Olives And Pickles. Served On A Freshly Baked Bread	Veg Extravaganza Sub	9	1	196
2	Experience Authentic Flavours With A Kebab Made Of Lentils And Enriched With Mouth-Watering Hints Of Garlic And Onion Accompanied By Nutritious Veggies In Your Favourite Bread	Veg Shammi Sub	9	1	234
3	Authentic Indian Meatballs Perfectly Seasoned And Spiced. Served Along With Wholesome Veggies And Packed In Your Choice Of Freshly Baked Bread	Chicken Kofta Sub	9	0	253
1	Our Signature Doughnut which is Light & Fluffy Dough With Our Classic Glaze	Original Glazed	10	1	90
2	A fun and fruity ring doughnut and coated with white ganache and topped with mango compound and a mango jam drizzle	Mango Masti	10	0	88
3	A plain ring doughnut dipped in and coated with a perfect combination of white ganache and dark chocolate chips	Vanilla Choco Chip	10	1	88

Join Queries

1) Find out all the food items every delivery agent has ever delivered along with their agent_id

```
MariaDB [fast_food]> select d.agent_id, f.food_name from food_item f, order_and_
food_item o, order_and_delivery_agent d where o.franchise_id = f.franchise_id an
d o.food_id = f.food_id and o.order_id = d.order_id;
+----------+--------------------------+
| agent_id | food_name                |
+----------+--------------------------+
|     1001 | McSpicy Chicken Burger   |
|     1001 | Mcspicy Paneer Burger    |
|     1001 | McChicken Burger         |
|     1002 | Margherita               |
|     1002 | Veg Extravaganza         |
|     1003 | Veg Shammi Sub           |
|     1004 | Soft Taco Mexican Chicken |
|     1004 | Sriracha Burrito         |
|     1005 | Classic Zinger Burger    |
|     1005 | Chocolate Lava Cake      |
+----------+--------------------------+
10 rows in set (0.007 sec)
```

2)Find all the order_id's which have Diwali Bonanza as their offers added -

```
MariaDB [fast_food]> select order_id from orders natural join offer where offer_
name = 'Diwali Bonanza';
+----------+
| order_id |
+----------+
|      100 |
|      106 |
|      108 |
+----------+
3 rows in set (0.003 sec)
```

3)Find all the restaurant names from which the user Claude Monet has placed order or is ordering-

```
Database changed
MariaDB [Fast_Food]> select franchise_name from restaurant where franchise_id in
 (select franchise_id from restaurant natural join orders where user_id = 'Claud
e Monet');
+----------------+
| franchise_name |
+----------------+
| Dominos        |
| KFC            |
+----------------+
2 rows in set (0.036 sec)
```

4)Find all the names of the food_items the user George Washington has ever ordered -

```
MariaDB [Fast_Food]> select food_name from order_and_food_item natural join food
_item where order_id in (select order_id from orders where user_id = 'George Was
hington');
+-----------------------+
| food_name             |
+-----------------------+
| McSpicy Chicken Burger |
| Mcspicy Paneer Burger  |
| McChicken Burger       |
| Classic Zinger Burger  |
| Chocolate Lava Cake    |
+-----------------------+
5 rows in set (0.017 sec)
```

Aggregate Queries

1)Find the count of all the orders which has been placed from KFC -

```
MariaDB [Fast_Food]> select count(*) from orders where franchise_id = (select fr
anchise_id from restaurant where franchise_name = 'KFC');
+----------+
| count(*) |
+----------+
|        2 |
+----------+
1 row in set (0.032 sec)
```

2)Find the user who has ordered or is ordering the most expensive meal -

```
MariaDB [Fast_Food]> select user_id from orders where final_price = (select max(
final_price) from orders);
+--------------+
| user_id      |
+--------------+
| Claude Monet |
+--------------+
1 row in set (0.009 sec)
```

3)Find the user who has ordered or ordering the cheapest meal -

```
MariaDB [fast_food]> select user_id from orders where final_price = (select min(final_price) from orders);
+------------+
| user_id    |
+------------+
| Piet Kraak |
+------------+
1 row in set (0.052 sec)
```

4)Find the the count of every food item ever ordered -

```
MariaDB [Fast_Food]> select food_name, franchise_id, count(*) as Count from order_and_food_item natural join food_item  group by franchise_id,
food_id;
+------------------------------+--------------+-------+
| food_name                    | franchise_id | Count |
+------------------------------+--------------+-------+
| Classic Zinger Burger        |            1 |     2 |
| Veg Zinger Burger            |            1 |     1 |
| Classic Chicken Biryani Bucket |            1 |     1 |
| Chocolate Lava Cake          |            1 |     1 |
| McSpicy Chicken Burger       |            2 |     1 |
| Mcspicy Paneer Burger        |            2 |     1 |
| McChicken Burger             |            2 |     1 |
| Crispy Veg Double Patty Burger |            3 |     1 |
| Bk Classic Veg with cheese   |            3 |     1 |
| Margherita                   |            4 |     1 |
| Veg Extravaganza             |            4 |     1 |
| Margherita                   |            5 |     1 |
| Corn & Cheese                |            5 |     1 |
| Soft Taco Mexican Chicken    |            7 |     1 |
| Sriracha Burrito             |            7 |     1 |
| Espresso                     |            8 |     1 |
| Veg Shammi Sub               |            9 |     1 |
+------------------------------+--------------+-------+
17 rows in set (0.002 sec)
```

Set Queries

1)Find all the users who have bought Classic Zinger Burger and also have paid for their order -

```
MariaDB [Fast_Food]> select user_id from payment_info where order_id = ((select
order_id from payment_info) intersect (select order_id from order_and_food_item
where food_id = 1 and franchise_id = 1));
+------------------+
| user_id          |
+------------------+
| George Washington |
+------------------+
1 row in set (0.005 sec)
```

2)Find all the users who have bought the first item from franchise 1 or franchise 5 -

```
MariaDB [Fast_Food]> select user_id from orders where order_id in ((select order
_id from order_and_food_item where food_id = 1 and franchise_id = 1) union (sele
ct order_id from order_and_food_item where food_id = 1 and franchise_id = 5));
+------------------+
| user_id          |
+------------------+
| Claude Monet     |
| George Washington |
| Georges Brassens  |
+------------------+
3 rows in set (0.009 sec)
```

3)Find all the users who have bought an item with quantity 2 or with quantity 3 -

```
MariaDB [Fast_Food]> select distinct user_id from orders where order_id in((sele
ct order_id from order_and_food_item where quantity = 2) union all (select order
_id from order_and_food_item where quantity = 3));
+------------------+
| user_id          |
+------------------+
| Claude Monet     |
| El Greco         |
| George Washington |
| Georges Brassens  |
| Piet Kraak       |
| Salvador Allende  |
+------------------+
6 rows in set (0.003 sec)
```

4)Find all the users who have paid for an order from either KFC or McDonald's except those who have paid through ICICI Bank -

```
MariaDB [Fast_Food]> select distinct user_id from orders where order_id in((sele
ct order_id from order_and_food_item where franchise_id = 1 or franchise_id = 2
and order_id in (select order_id from payment_info)) except (select order_id fro
m payment_info where bank = 'ICICI'));
+-------------+
| user_id     |
+-------------+
| Claude Monet |
+-------------+
1 row in set (0.003 sec)
```

Function

To find the discount on the food items the user is planning to purchase. This is used inside a trigger, which is invoked when a food_item is added or deleted.

```
delimiter $$
create function calculate_discount(max_discount int, percent float, total_amount int)
returns int
deterministic
begin
    declare new_discount int;
    if (total_amount * percent) > max_discount
          then
          set new_discount = max_discount;
          else
          set new_discount = total_amount * percent;
          end if;

    return new_discount;
end $$
delimiter ;
```

```
MariaDB [fast_food]> select * from orders;
+----------+--------------+-------------------+-------------+----------+----------+-------------+
| order_id | total_amount | user_id           | franchise_id | offer_id | discount | final_price |
+----------+--------------+-------------------+-------------+----------+----------+-------------+
|      100 |         2128 | George Washington |            2 |        1 |      100 |        2028 |
|      101 |         3114 | Claude Monet      |            4 |        0 |        0 |        3114 |
|      102 |          936 | Piet Kraak        |            9 |        0 |        0 |         936 |
|      103 |          654 | Salvador Allende  |            7 |        2 |      150 |         504 |
|      104 |         1632 | George Washington |            1 |        0 |        0 |        1632 |
|      105 |          180 | Linda Lovelace    |            3 |        2 |       72 |         108 |
|      106 |         2020 | Claude Monet      |            1 |        1 |      100 |        1920 |
|      109 |          168 | El Greco          |            3 |        3 |       42 |         126 |
|      110 |            0 | Piet Kraak        |            5 |        0 |        0 |           0 |
+----------+--------------+-------------------+-------------+----------+----------+-------------+
9 rows in set (0.001 sec)

MariaDB [fast_food]> insert into order_and_food_item values(109, 3, 3, 2, 0);
Query OK, 1 row affected (0.003 sec)

MariaDB [fast_food]> select * from orders;
+----------+--------------+-------------------+-------------+----------+----------+-------------+
| order_id | total_amount | user_id           | franchise_id | offer_id | discount | final_price |
+----------+--------------+-------------------+-------------+----------+----------+-------------+
|      100 |         2128 | George Washington |            2 |        1 |      100 |        2028 |
|      101 |         3114 | Claude Monet      |            4 |        0 |        0 |        3114 |
|      102 |          936 | Piet Kraak        |            9 |        0 |        0 |         936 |
|      103 |          654 | Salvador Allende  |            7 |        2 |      150 |         504 |
|      104 |         1632 | George Washington |            1 |        0 |        0 |        1632 |
|      105 |          180 | Linda Lovelace    |            3 |        2 |       72 |         108 |
|      106 |         2020 | Claude Monet      |            1 |        1 |      100 |        1920 |
|      109 |          486 | El Greco          |            3 |        3 |      122 |         364 |
|      110 |            0 | Piet Kraak        |            5 |        0 |        0 |           0 |
+----------+--------------+-------------------+-------------+----------+----------+-------------+
9 rows in set (0.001 sec)
```

Procedure

Procedure for food_delivery which sets the delivered attribute of order_and_food_delivery to 1 and makes the current deliver_agent available again to take more orders.

```
delimiter $$
create procedure food_delivered(in cur_order_id int, in cur_agent_id int)
begin
        update order_and_delivery_agent set delivered = 1 where order_id = cur_order_id and
agent_id = cur_agent_id;
        update delivery_agent set available = 1 where agent_id = cur_agent_id;
end $$
delimiter;
```

```
MariaDB [fast_food]> select * from order_and_delivery_agent;
+----------+----------+-----------+
| order_id | agent_id | delivered |
+----------+----------+-----------+
|      100 |     1001 |         1 |
|      101 |     1002 |         0 |
|      102 |     1005 |         0 |
|      103 |     1004 |         0 |
|      104 |     1003 |         0 |
|      105 |     1001 |         0 |
+----------+----------+-----------+
6 rows in set (0.001 sec)

MariaDB [fast_food]> select * from delivery_agent;
+----------+--------------------+-----------+
| agent_id | agent_name         | available |
+----------+--------------------+-----------+
|     1001 | Aniruddha Kulkarni |         0 |
|     1002 | Tushar Kumar       |         0 |
|     1003 | Atharv Tiwari      |         0 |
|     1004 | Shubangi Saxena    |         0 |
|     1005 | Avanish Bhat       |         0 |
+----------+--------------------+-----------+
5 rows in set (0.002 sec)

MariaDB [fast_food]> call food_delivered(101, 1002);
Query OK, 2 rows affected (0.006 sec)

MariaDB [fast_food]> select * from order_and_delivery_agent;
+----------+----------+-----------+
| order_id | agent_id | delivered |
+----------+----------+-----------+
|      100 |     1001 |         1 |
|      101 |     1002 |         1 |
|      102 |     1005 |         0 |
|      103 |     1004 |         0 |
|      104 |     1003 |         0 |
|      105 |     1001 |         0 |
+----------+----------+-----------+
6 rows in set (0.001 sec)

MariaDB [fast_food]> select * from delivery_agent;
+----------+--------------------+-----------+
| agent_id | agent_name         | available |
+----------+--------------------+-----------+
|     1001 | Aniruddha Kulkarni |         0 |
|     1002 | Tushar Kumar       |         1 |
|     1003 | Atharv Tiwari      |         0 |
|     1004 | Shubangi Saxena    |         0 |
|     1005 | Avanish Bhat       |         0 |
+----------+--------------------+-----------+
5 rows in set (0.000 sec)

MariaDB [fast_food]>
```

Trigger

Adds total if payment has not been done yet and also calculates item_total and also checks if the order has already been paid for, also checks if the restaurant from which the food is being ordered from is same as the original order_id-

```
delimiter $$
create trigger add_food
before insert
on
order_and_food_item
for each row begin
set @actual_franchise = (select franchise_id from orders where order_id = new.order_id);
if(@actual_franchise <> new.franchise_id)
then
signal sqlstate '45000'
set message_text="This order is not from this franchise...";
end if;


set @pri = (select price from food_item where food_id = new.food_id and franchise_id =
new.franchise_id);
set new.total_item_price = @pri * new.quantity;
if not exists (select * from payment_info where order_id = new.order_id)
then
update orders set total_amount = total_amount + new.total_item_price
where order_id = new.order_id;

else
signal sqlstate '45000'
set message_text="Cannot add food to an order which has been completed and paid for...";
end if;
END $$
delimiter ;
```

```
MariaDB [fast_Food]> select * from orders;
+----------+--------------+--------------------+--------------+----------+----------+-------------+
| order_id | total_amount | user_id            | franchise_id | offer_id | discount | final_price |
+----------+--------------+--------------------+--------------+----------+----------+-------------+
|      100 |         2128 | George Washington  |            2 |        1 |      100 |        2028 |
|      101 |         3114 | Claude Monet       |            4 |        0 |        0 |        3114 |
|      102 |          936 | Piet Kraak         |            9 |        0 |        0 |         936 |
|      103 |          654 | Salvador Allende   |            7 |        2 |      150 |         504 |
|      104 |         1632 | George Washington  |            1 |        0 |        0 |        1632 |
|      105 |          180 | Linda Lovelace     |            3 |        2 |       72 |         108 |
|      106 |         2020 | Claude Monet       |            1 |        1 |      100 |        1920 |
|      109 |            0 | El Greco           |            3 |        3 |        0 |           0 |
|      110 |            0 | Piet Kraak         |            5 |        0 |        0 |           0 |
+----------+--------------+--------------------+--------------+----------+----------+-------------+
9 rows in set (0.001 sec)

MariaDB [fast_Food]> insert into order_and_food_item values(109, 2, 3, 2, 0);
Query OK, 1 row affected (0.022 sec)

MariaDB [fast_Food]> select * from orders;
+----------+--------------+--------------------+--------------+----------+----------+-------------+
| order_id | total_amount | user_id            | franchise_id | offer_id | discount | final_price |
+----------+--------------+--------------------+--------------+----------+----------+-------------+
|      100 |         2128 | George Washington  |            2 |        1 |      100 |        2028 |
|      101 |         3114 | Claude Monet       |            4 |        0 |        0 |        3114 |
|      102 |          936 | Piet Kraak         |            9 |        0 |        0 |         936 |
|      103 |          654 | Salvador Allende   |            7 |        2 |      150 |         504 |
|      104 |         1632 | George Washington  |            1 |        0 |        0 |        1632 |
|      105 |          180 | Linda Lovelace     |            3 |        2 |       72 |         108 |
|      106 |         2020 | Claude Monet       |            1 |        1 |      100 |        1920 |
|      109 |          180 | El Greco           |            3 |        3 |       45 |         135 |
|      110 |            0 | Piet Kraak         |            5 |        0 |        0 |           0 |
+----------+--------------+--------------------+--------------+----------+----------+-------------+
9 rows in set (0.001 sec)

MariaDB [fast_Food]>
```

Here trying to insert more food_items to an order which has already been paid for -

```
MariaDB [fast_Food]> insert into order_and_food_item values(100, 4, 2, 1);
ERROR 1136 (21S01): Column count doesn't match value count at row 1
MariaDB [fast_Food]> select * from orders;
+----------+--------------+-------------------+--------------+----------+----------+-------------+
| order_id | total_amount | user_id           | franchise_id | offer_id | discount | final_price |
+----------+--------------+-------------------+--------------+----------+----------+-------------+
|      100 |         2128 | George Washington |            2 |        1 |      100 |        2028 |
|      101 |         3114 | Claude Monet      |            4 |        0 |        0 |        3114 |
|      102 |          936 | Piet Kraak        |            9 |        0 |        0 |         936 |
|      103 |          654 | Salvador Allende  |            7 |        2 |      150 |         504 |
|      104 |         1632 | George Washington |            1 |        0 |        0 |        1632 |
|      105 |          180 | Linda Lovelace    |            3 |        2 |       72 |         108 |
|      106 |         2020 | Claude Monet      |            1 |        1 |      100 |        1920 |
|      109 |          180 | El Greco          |            3 |        3 |       45 |         135 |
|      110 |            0 | Piet Kraak        |            5 |        0 |        0 |           0 |
+----------+--------------+-------------------+--------------+----------+----------+-------------+
9 rows in set (0.001 sec)

MariaDB [fast_Food]> insert into order_and_food_item values(100, 4, 2, 1, 0);
ERROR 1644 (45000): Cannot add food to an order which has been completed and paid for...
```

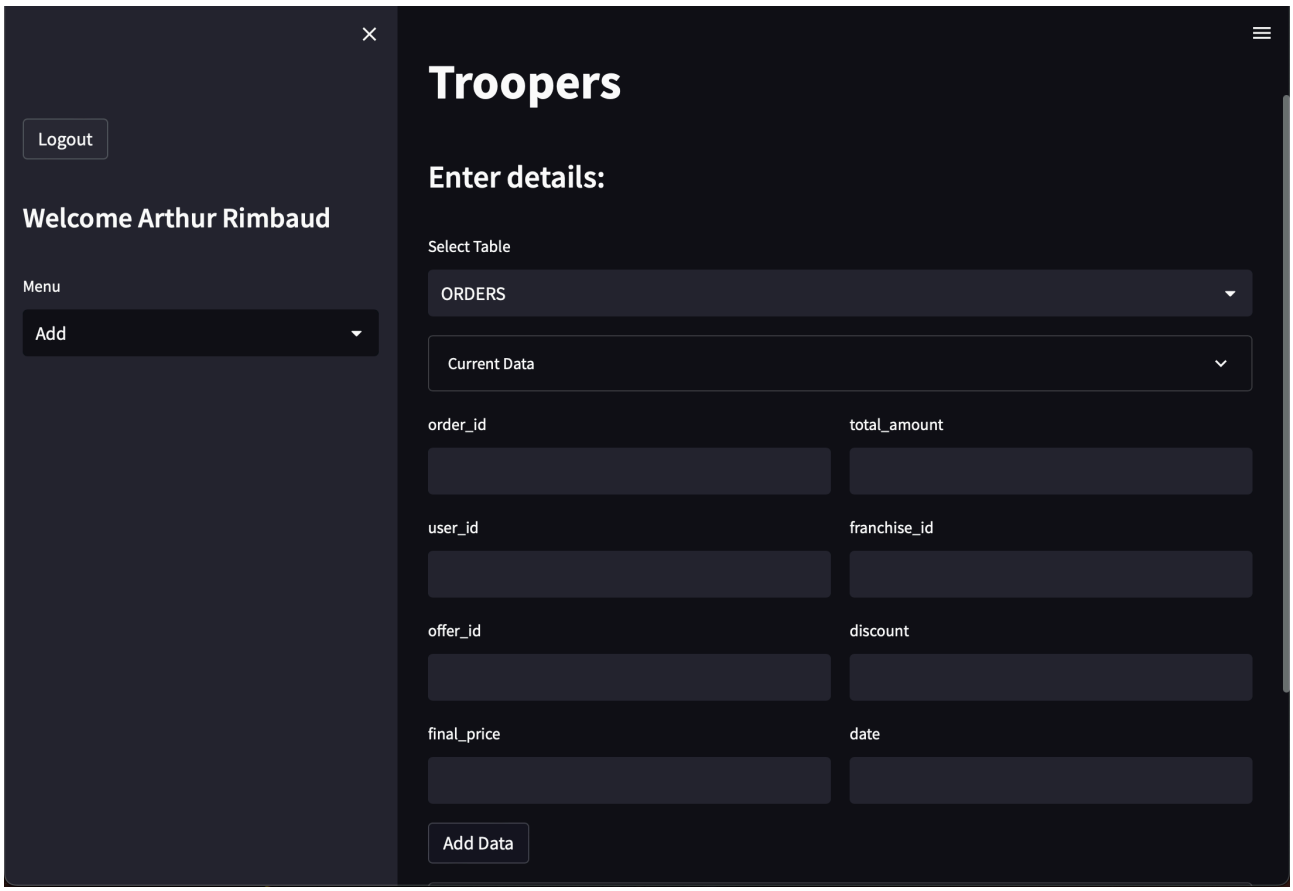
Cursor

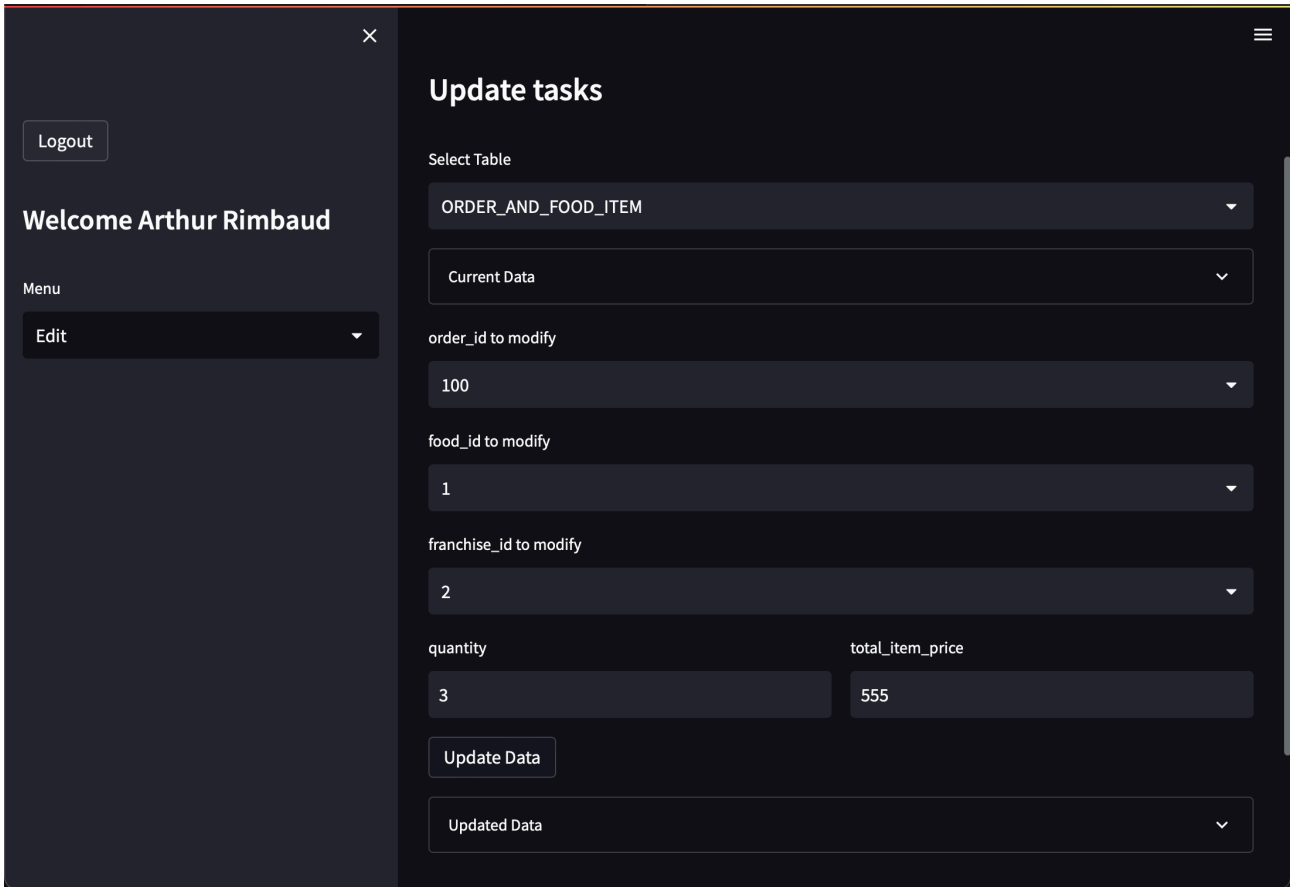
The following cursors gets the user_id and names of those users who have applied for the post of delivery_agent but have not yet been assigned an agent_id by the admins -

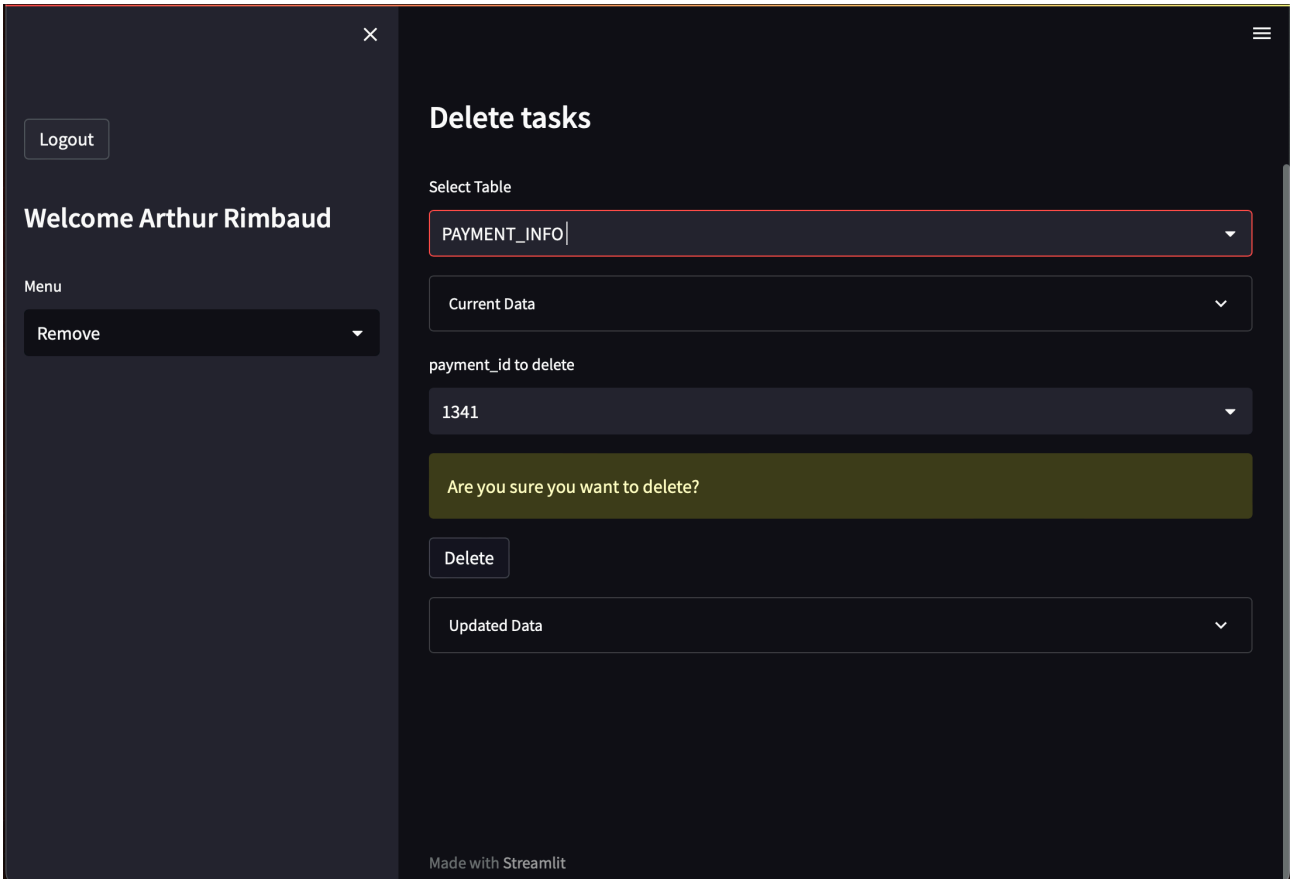
```
delimiter $$
create procedure unassigned_delivery_agents()
begin
   declare done int default 0;
   declare usr varchar(255);
   declare nam varchar(255);
   declare cur cursor for select user_id, name from user where permissions = 'delivery_agent';
   declare continue handler for not found set done = 1;
   open cur;
   label:
      loop
         fetch cur into usr, nam;
         if done = 1 then leave label;
         end if;
         if not exists(select * from delivery_agent where user_id = usr)
         then
         select usr as User_ID, nam as User_Name;
         end if;
      end loop;
   close cur;
end $$
delimiter ;
```

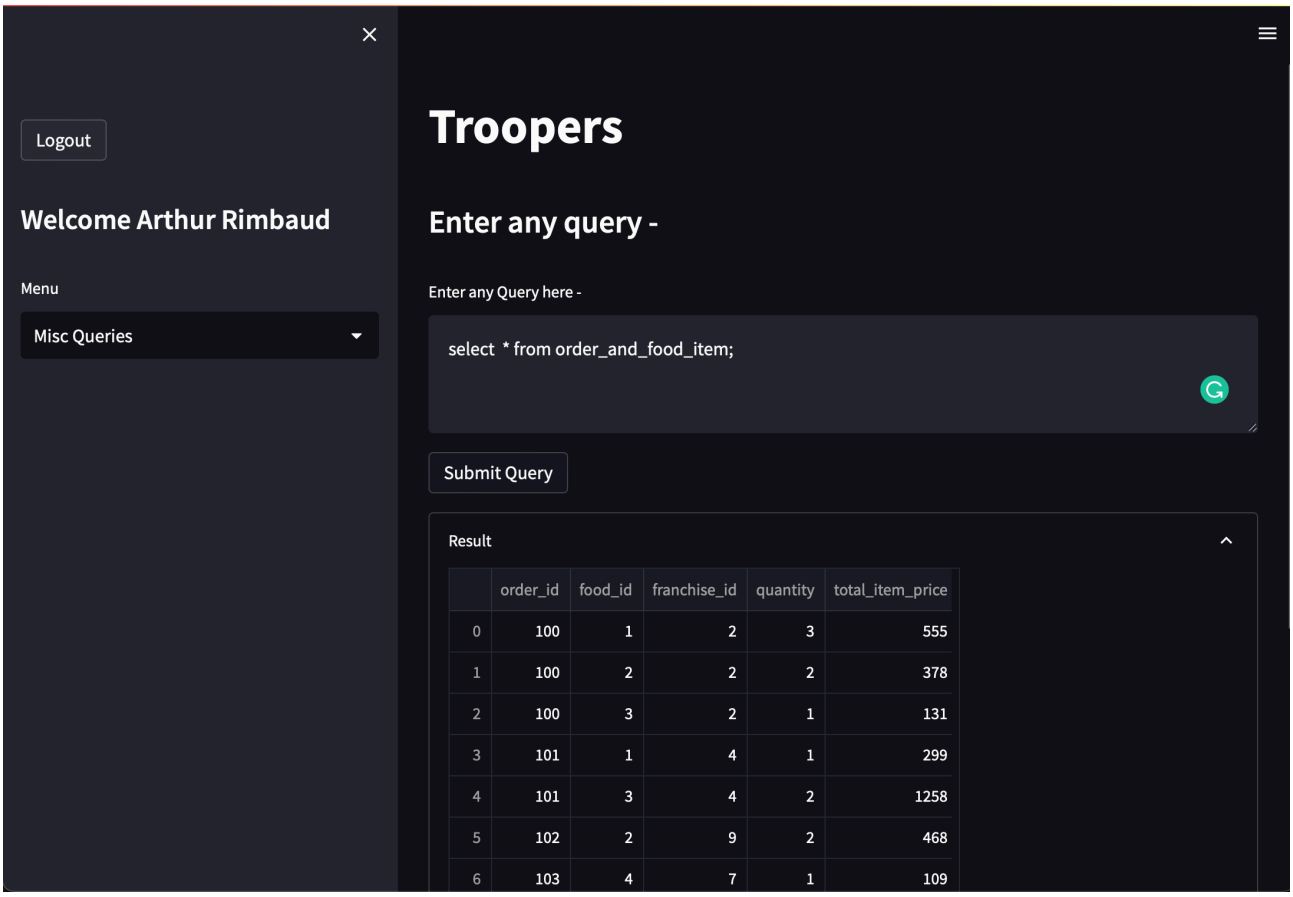
```
MariaDB [fast_Food]> call unassigned_delivery_agents();
+--------------+--------------+
| User_ID      | User_Name    |
+--------------+--------------+
| Karthik Nair | Karthik Nair |
+--------------+--------------+
1 row in set (0.039 sec)

Query OK, 0 rows affected (0.040 sec)
```

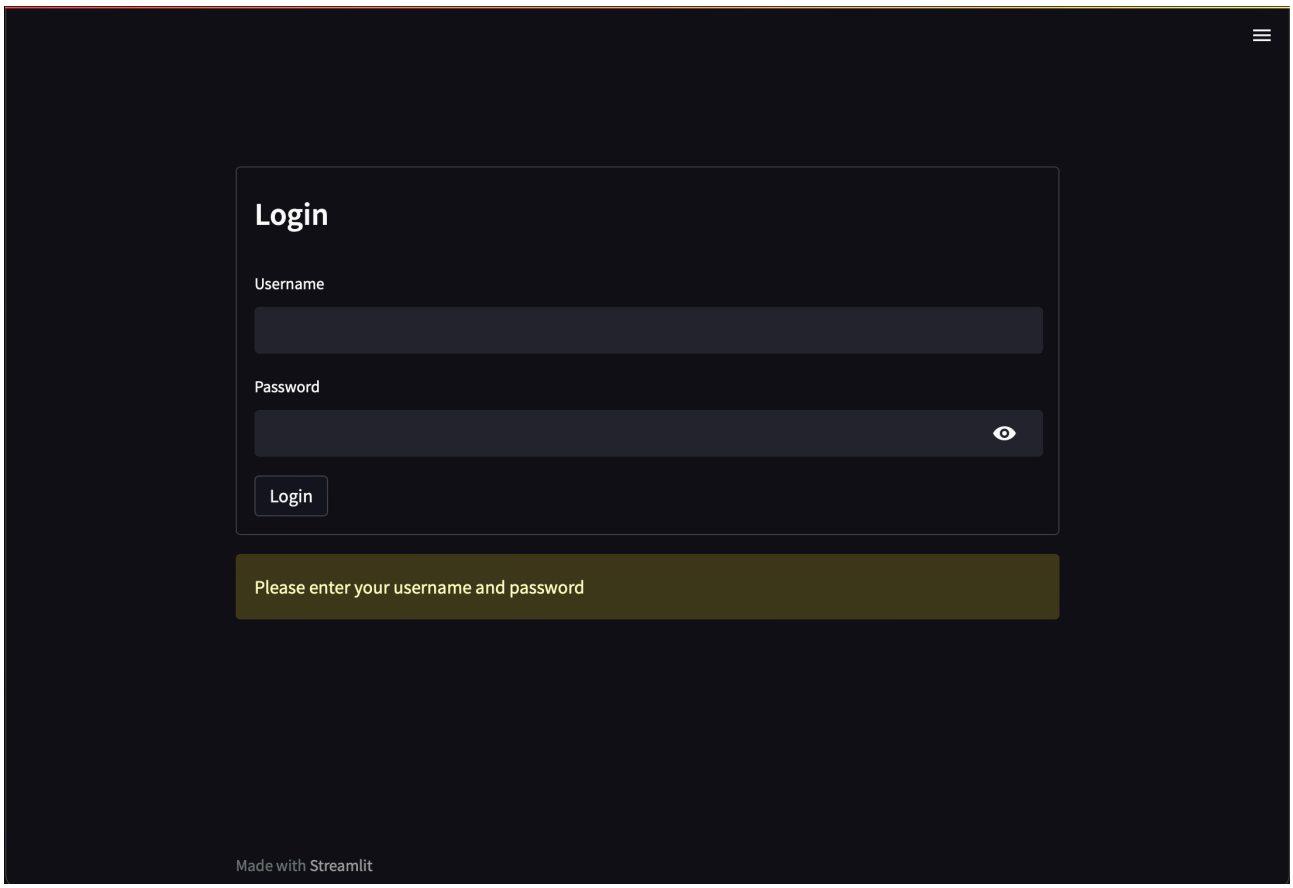
Front-End


Logout
Welcome Arthur Rimbaud
Menu
Add
Troopers
Enter details:
Select Table
ORDERS
Current Data
order_id
total_amount
user_id
franchise_id
offer_id
discount
final_price
date
Add Data


Logout
Welcome Arthur Rimbaud
Menu
Edit
Update tasks
Select Table
ORDER_AND_FOOD_ITEM
Current Data
order_id to modify
100
food_id to modify
1
franchise_id to modify
2
quantity
3
total_item_price
555
Update Data
Updated Data


Logout
Welcome Arthur Rimbaud
Menu
Remove
Delete tasks
Select Table
PAYMENT_INFO
Current Data
payment_id to delete
1341
Are you sure you want to delete?
Delete
Updated Data
Made with Streamlit


Logout
Welcome Arthur Rimbaud
Menu
Misc Queries
Troopers
Enter any query -
Enter any Query here -
select * from order_and_food_item;
Submit Query
Result
order_id food_id franchise_id quantity total_item_price
0 100 1 2 3 555
1 100 2 2 2 378
2 100 3 2 1 131
3 101 1 4 1 299
4 101 3 4 2 1258
5 102 2 9 2 468
6 103 4 7 1 109


Login
Username
Password
Login
Please enter your username and password
Made with Streamlit