DBMS - Mini Project

Restaurant Table Booking System

Submitted by-
Name- Arushi Pandey
SRN- PES1UG20CS077
V Semester
Section - B

Short Description and Scope of the Project

Restaurant reservation systems **help manage the constant influx of reservations and customers** – allowing the customers to book their tables remotely so managers can schedule resources according to the number of bookings

This project consists of UI in which customers can –

- View the vacant tables in a restaurant.
- See reviews of the selected restaurant which will help the customer in deciding which restaurant to book a table in.
- They can make reservations in a particular restaurant for a specific date and time.
- They can view, edit and delete the reservations as well.
- View vacant tables in a particular area(For example Koramangala).
- Find the names of customers who have made atleast one reservation.

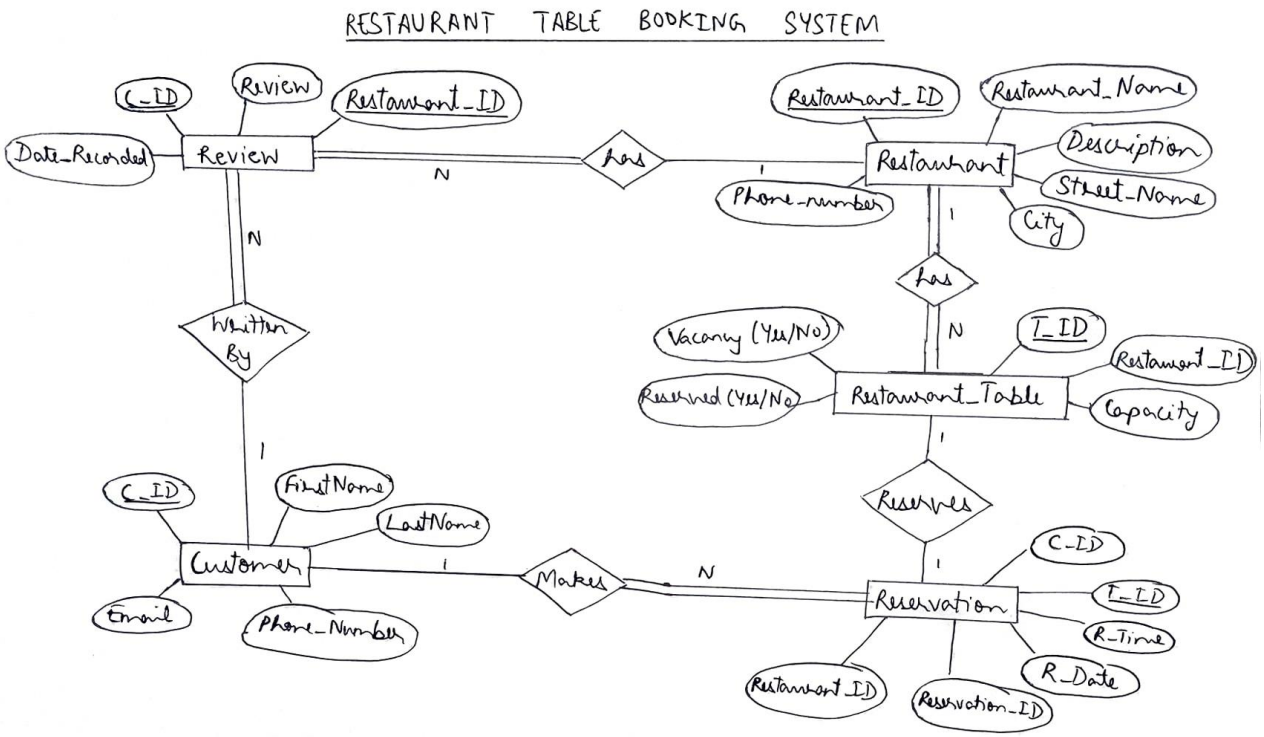
Scope of the project-

- For customers to get effective deals
- For customers to go to likeable place
- Reservations of later dining times
- Reducing wait times
- Providing options nearby
- Promoting and advertising their restaurants to get like minded or repeat buisiness

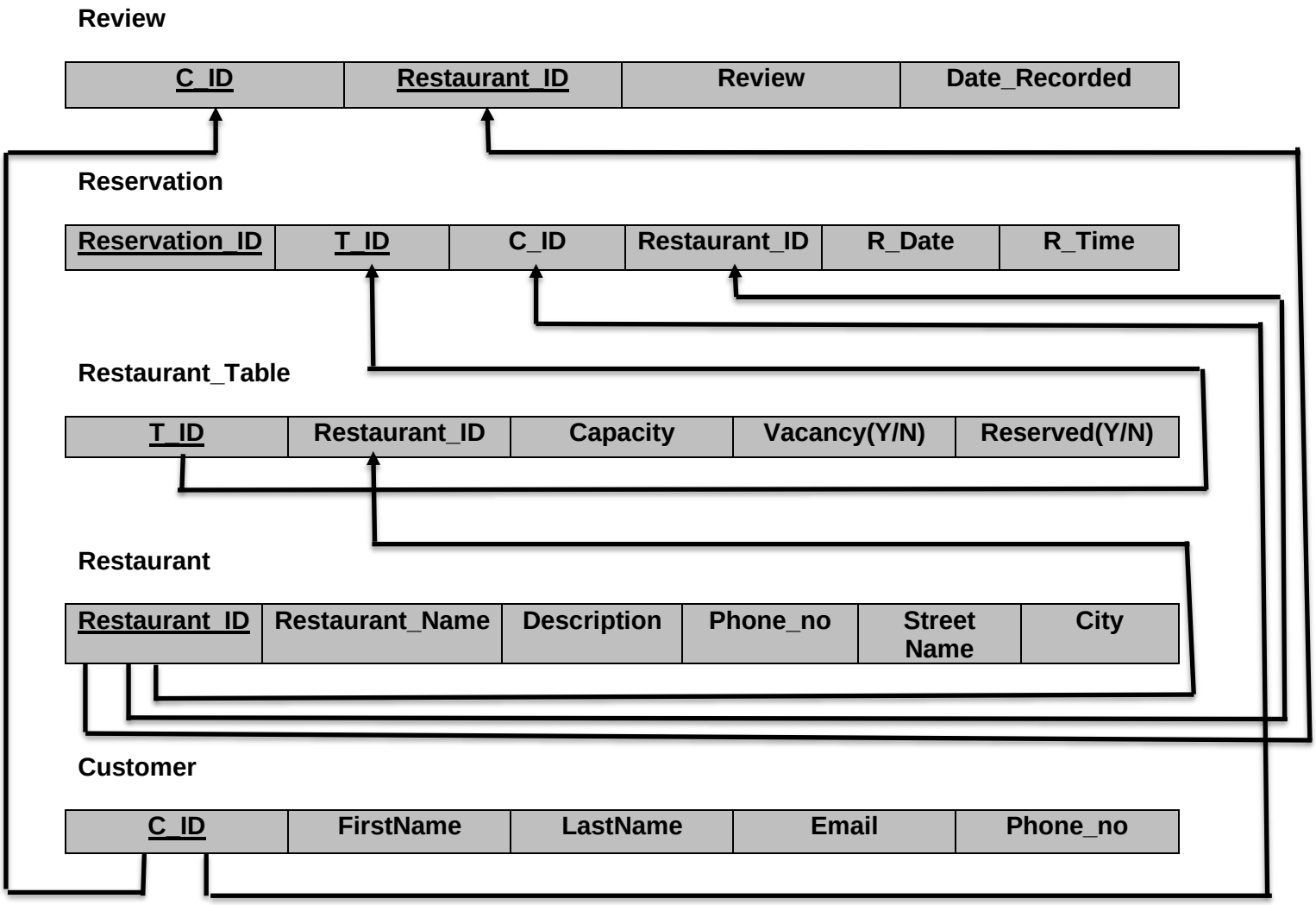
Note-

(CRUD operations have been implemented for all 5 tables.
Also we can view data for all 5 tables.
Join,procedure, trigger, set operations and aggregate operations have also been implemented in frontend.)

ER Diagram

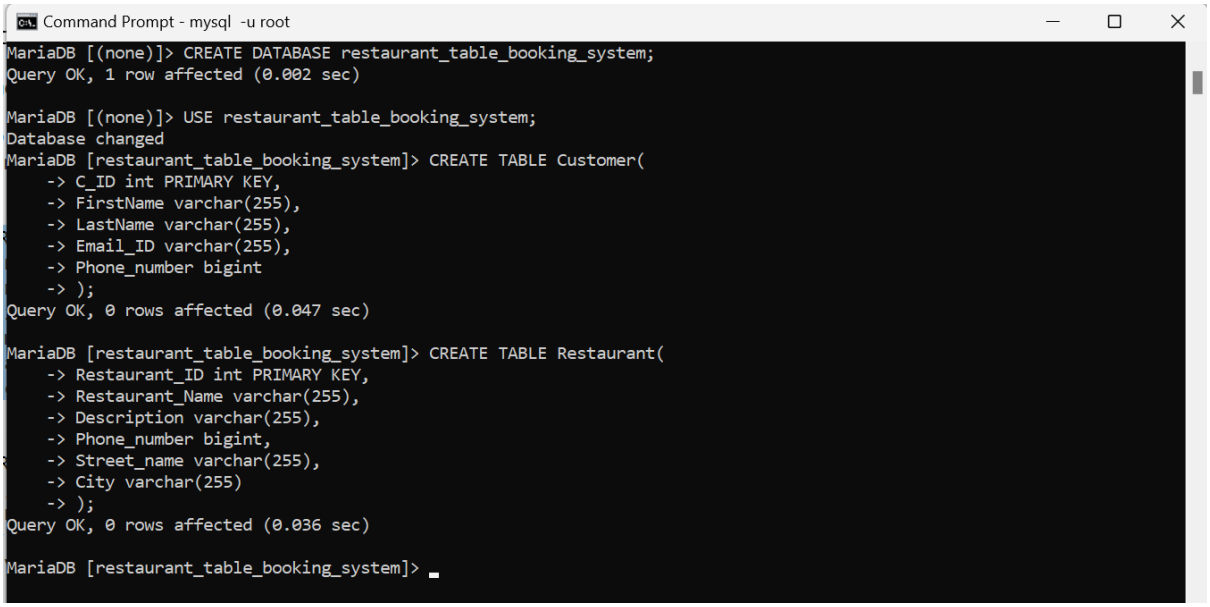

RESTAURANT TABLE BOOKING SYSTEM
C_ID
Review
Restaurant_ID
Date_Recorded
Review
N
has
1
Restaurant_ID
Restaurant_Name
Description
Street_Name
City
Phone_number
Restaurant
1
has
N
T_ID
Restaurant_ID
Capacity
Vacancy (Yes/No)
Reserved (Yes/No)
Restaurant_Table
1
Reserves
1
C_ID
T_ID
R_Time
R_Date
Reservation_ID
Restaurant_ID
Reservation
N
Makes
1
Written By
N
1
C_ID
FirstName
LastName
Email
Phone_Number
Customer

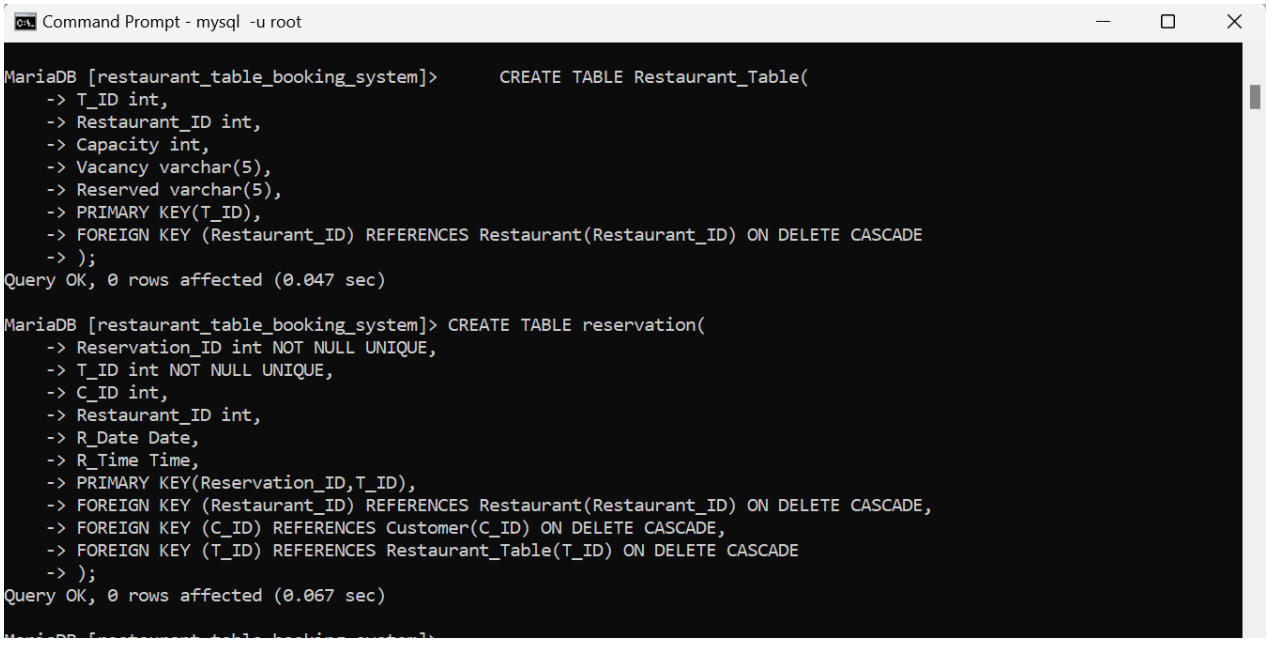
Relational Schema



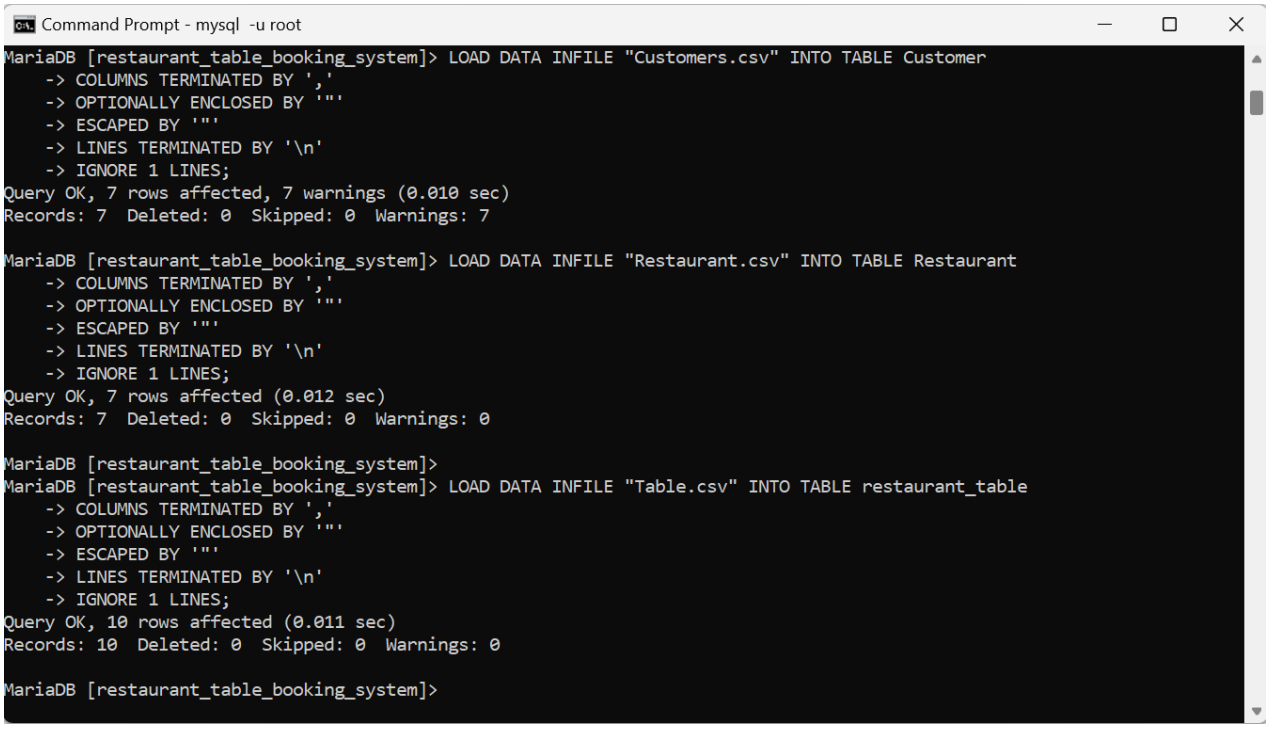
DDL statements - Building the database

Creating tables-





Populating the Database



Join Queries

Write the query in English Language, Show the equivalent SQL statement and also a screenshot of the query and the results

Find the names of customers who have made atleast 1 reservation:

SQL Query execution-

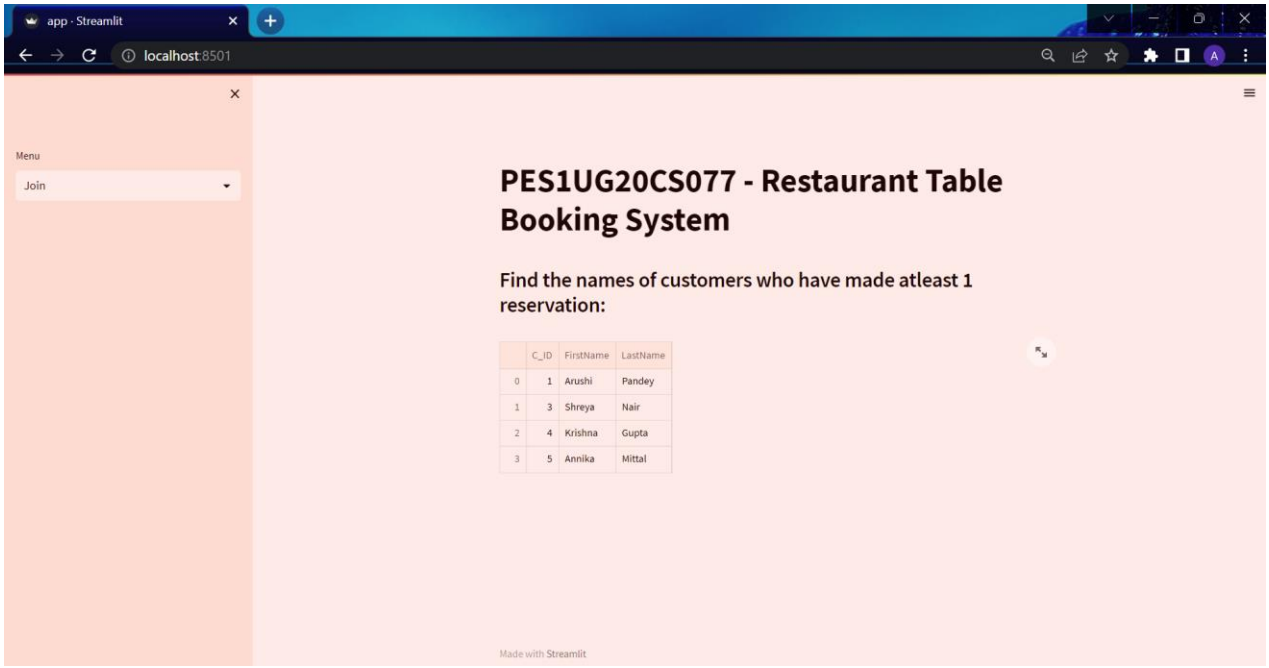
```
MariaDB [(none)]> use restaurant_table_booking_system;
Database changed
MariaDB [restaurant_table_booking_system]> SELECT C.C_ID,C.FirstName,C.LastName FROM Customer C INNER JOIN Reservation R
 ON C.C_ID=R.C_ID;
+------+-----------+----------+
| C_ID | FirstName | LastName |
+------+-----------+----------+
|    1 | Arushi    | Pandey   |
|    3 | Shreya    | Nair     |
|    4 | Krishna   | Gupta    |
|    5 | Annika    | Mittal   |
+------+-----------+----------+
4 rows in set (0.030 sec)

MariaDB [restaurant_table_booking_system]>
```

Python code-

```
def join2():
    c.execute('SELECT C.C_ID,C.FirstName,C.LastName FROM
Customer C INNER JOIN Reservation R ON C.C_ID=R.C_ID')
    data=c.fetchall()
    return data
```

UI Screenshot

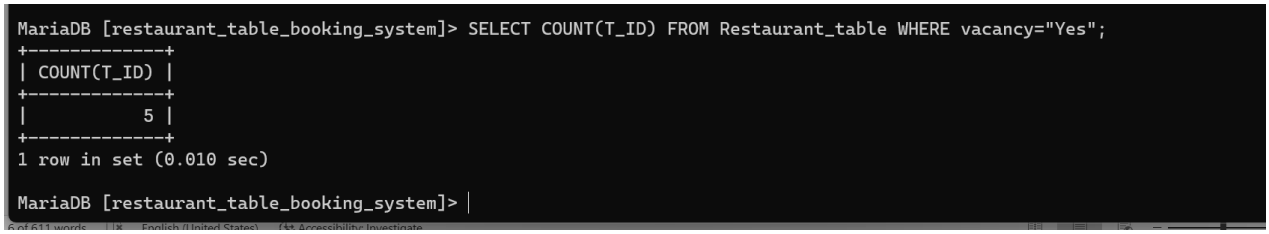


Aggregate Functions

Write the query in English Language, Show the equivalent SQL statement and also a screenshot of the query and the results

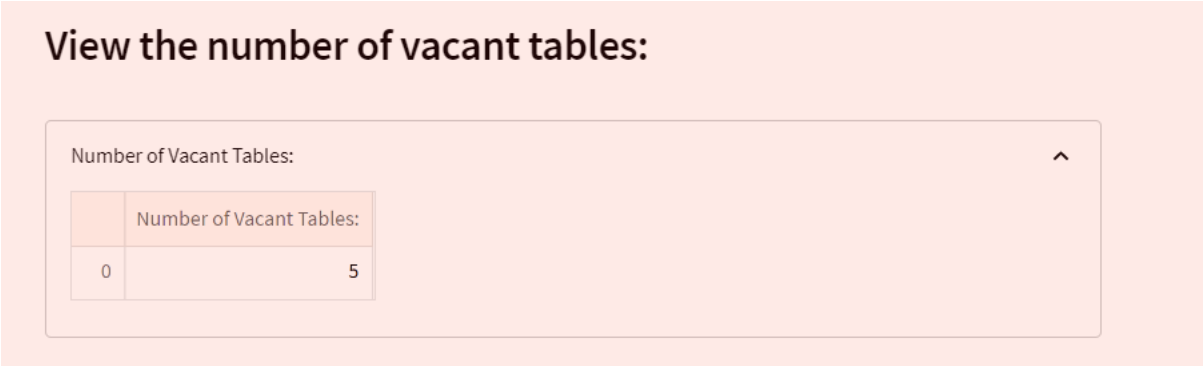
Query to find the number of vacant tables

SQL Query Execution-



Python code

```
def agg_cnt2():
    c.execute('SELECT COUNT(T_ID) FROM Restaurant_table WHERE vacancy="Yes"')
    data=c.fetchall()
    return data
```



Set Operations

Write the query in English Language, Show the equivalent SQL statement and also a screenshot of the query and the results

Find the restaurants who have vacant tables and are present in Koramangala-

SQL Query Execution-

```
MariaDB [restaurant_table_booking_system]> SELECT * FROM Restaurant WHERE Restaurant_ID IN (SELECT Restaurant_ID from Re
staurant WHERE Street_Name="Koramangala" INTERSECT SELECT Restaurant_ID from Restaurant_Table WHERE Vacancy="Yes");
+---------------+-----------------+-------------+--------------+-------------+------------+
| Restaurant_ID | Restaurant_Name | Description | Phone_number | Street_name | City       |
+---------------+-----------------+-------------+--------------+-------------+------------+
 |             3 | Mainland China  | Chinese     |   7892279326 | Koramangala | Bengaluru
 |             4 | Barbeque Nation | Buffet      |   9557522075 | Koramangala | Bengaluru
+---------------+-----------------+-------------+--------------+-------------+------------+
2 rows in set (0.004 sec)

MariaDB [restaurant_table_booking_system]>
```

Python code-

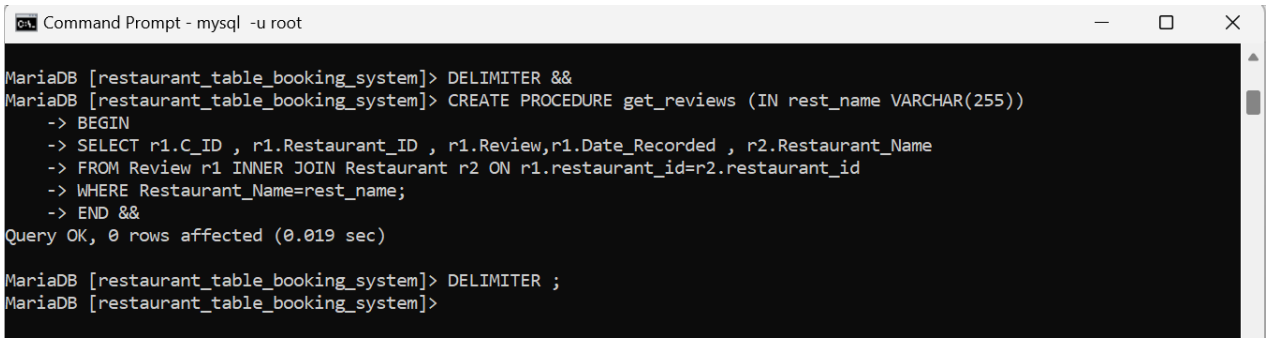
```
def set_oper2():
    c.execute('SELECT * FROM Restaurant WHERE Restaurant_ID IN
(SELECT Restaurant_ID from Restaurant WHERE
Street_Name="Koramangala" INTERSECT SELECT Restaurant_ID from
Restaurant_Table WHERE Vacancy="Yes")')
    data=c.fetchall()
    return data
```

UI Screenshot-

Functions and Procedures

Create a Function and Procedure. State the objective of the function / Procedure. Run and display the results.

Write a procedure to display the reviews given the name of a particular restaurant.

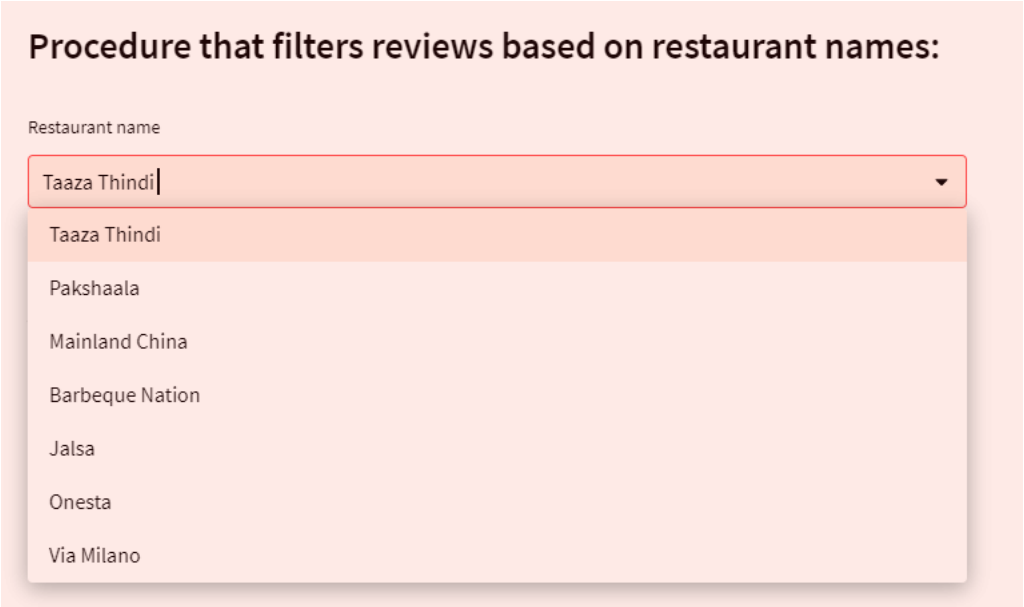


Python code-

```
def procedure2(rname):
    c.callproc('get_reviews',[rname])
    df=pd.DataFrame()
    for result in c.stored_results():
        temp_df=pd.DataFrame(result.fetchall())
        df=df.append(temp_df)
    return df
```

```
def procedure():
    list_of_r = [i[0] for i in view_only_restaurant_namess()]
    rname = st.selectbox("Restaurant name", list_of_r)
    if st.button("View reviews"):
        df=procedure2(rname)
        df.rename(columns =
{0:"C_ID",1:"Restaurant_ID",2:"Review",3:"Date_Recorded",4:"Res
taurant_Name"}, inplace = True)
        #df2=pd.DataFrame(df,columns=[ "C_ID" , "Restaurant_ID"
, "Review","Date_Recorded" , "Restaurant_Name"])
        with st.expander("View all reviews for
{}".format(rname)):
            st.dataframe(df)
```

Select the restaurant name for which you want to view the reviews

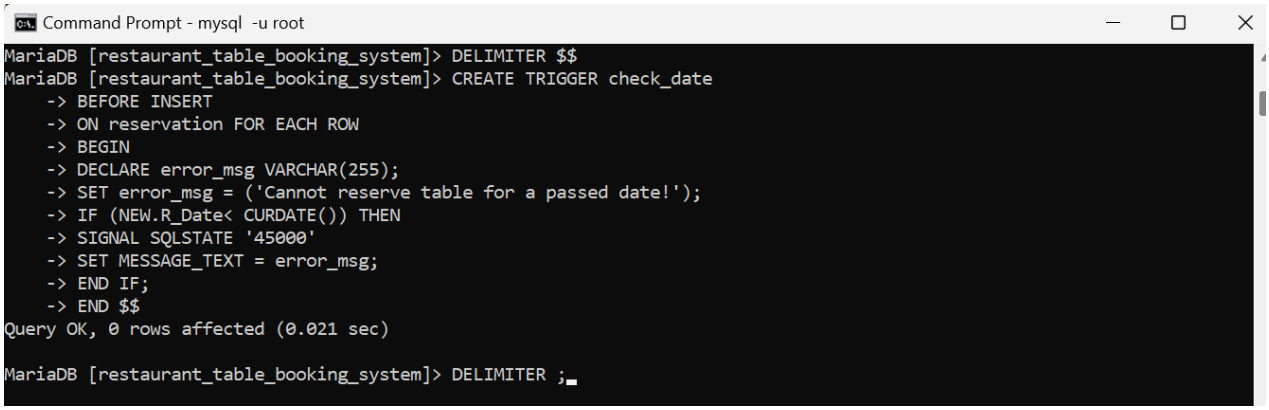


Press the “View Reviews” button to see the reviews for that particular restaurant

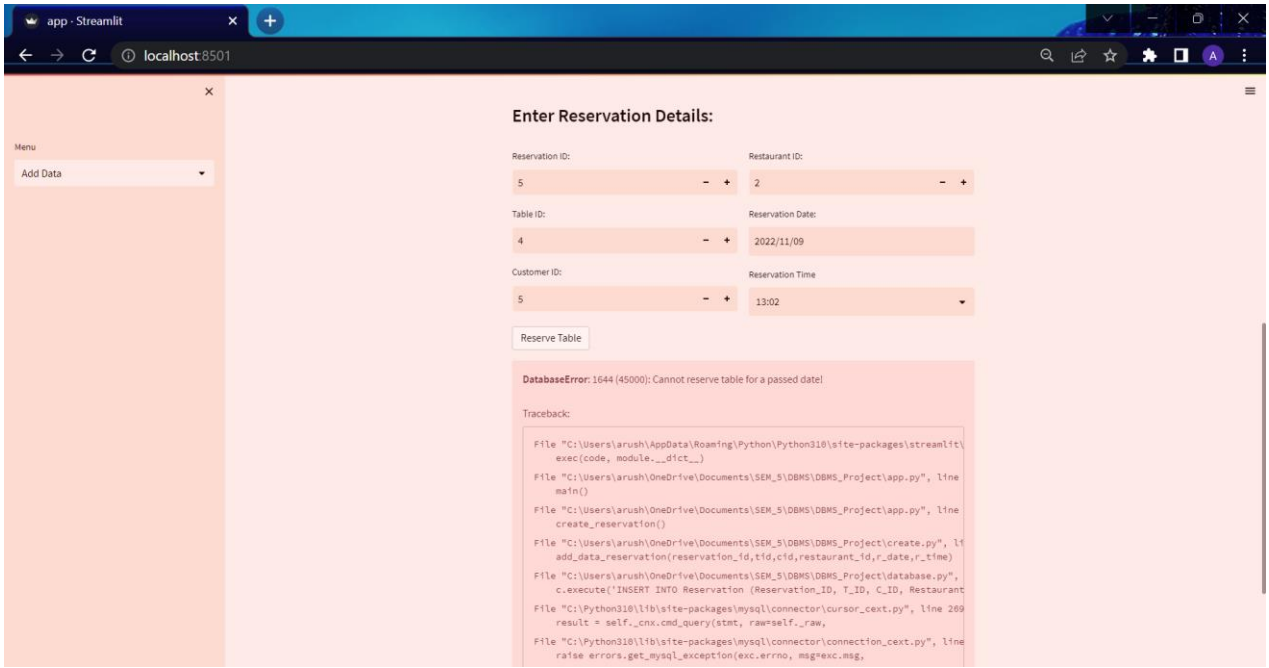
Triggers and Cursors

Create a Trigger and a Cursor. State the objective. Run and display the results.

Create a trigger to check if the reservation that is being made is not of a passed date.



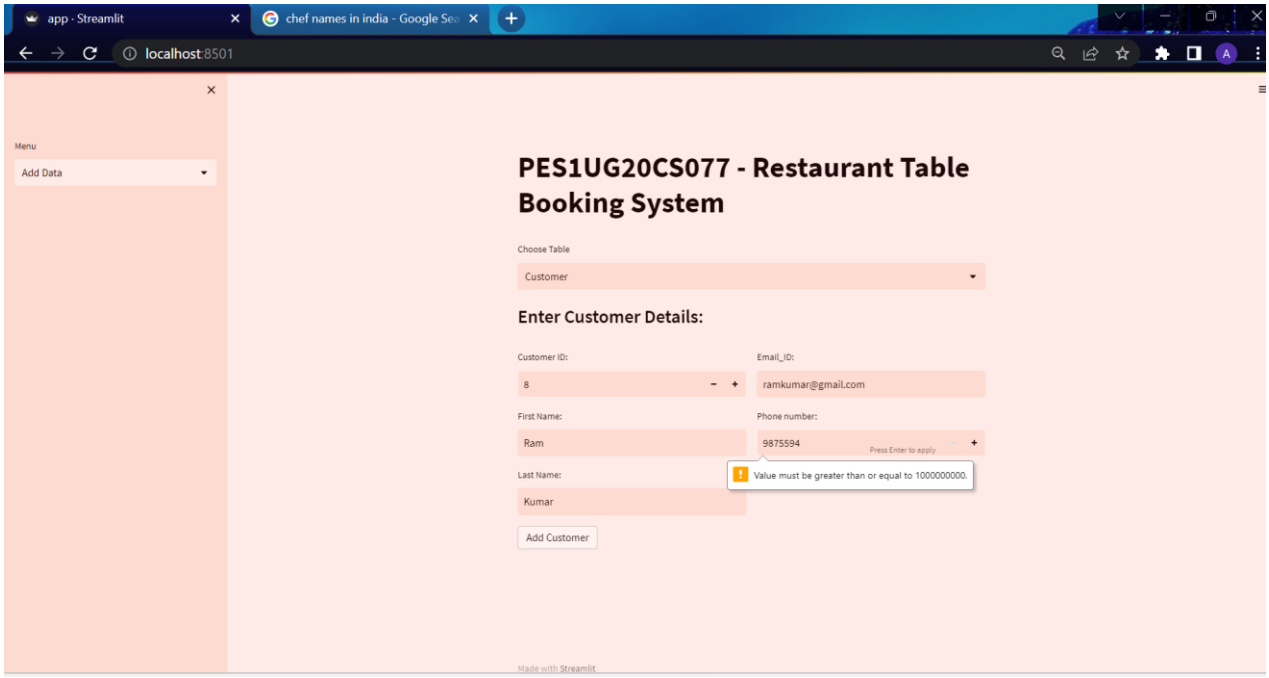
Trigger invoked when making reservation for a passed date



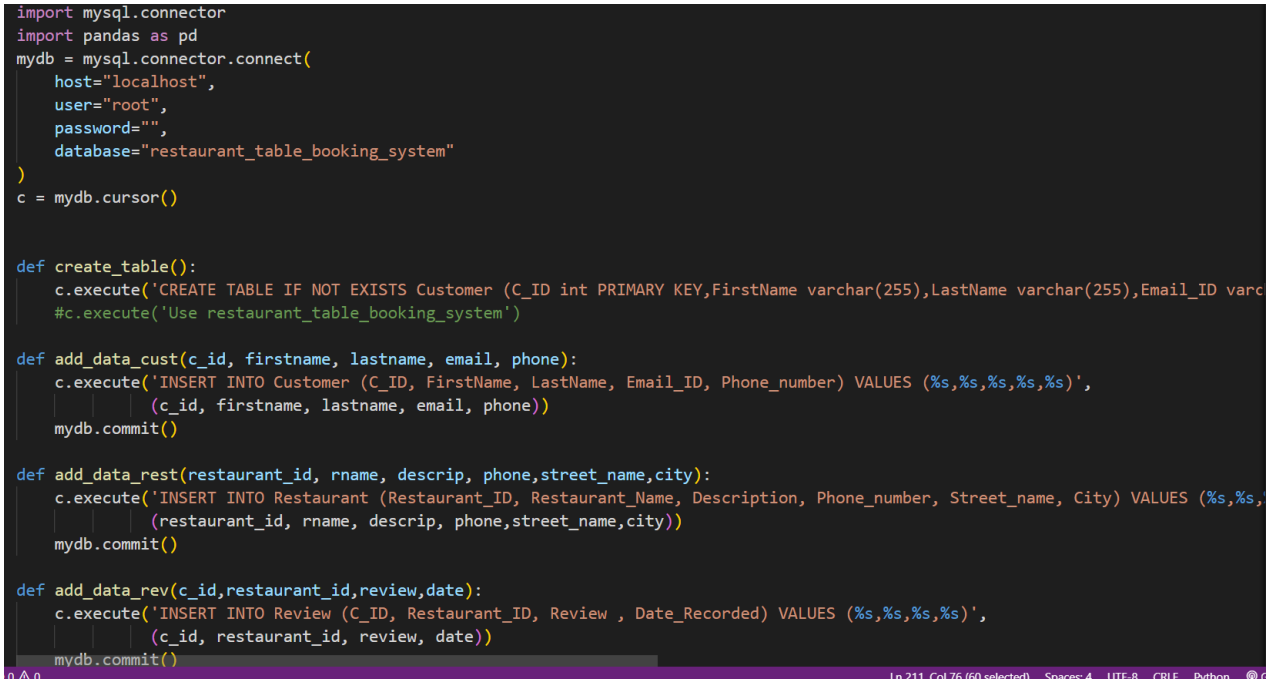
Trigger for checking if phone number is a 10 digit number

```
DELIMITER $$
CREATE TRIGGER check_phone
BEFORE INSERT
ON customer FOR EACH ROW
BEGIN
DECLARE error_msg VARCHAR(255);
SET error_msg = ('Please enter valid phone number!');
IF (NEW.Phone_Number<1000000000) THEN
SIGNAL SQLSTATE '45000'
SET MESSAGE_TEXT = error_msg;
END IF;
END $$
DELIMITER ;
```

UI Screenshot-



Cursor created in database.py file to execute various SQL queries and fetch results.



```python
import mysql.connector
import pandas as pd
mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="",
    database="restaurant_table_booking_system"
)
c = mydb.cursor()


def create_table():
    c.execute('CREATE TABLE IF NOT EXISTS Customer (C_ID int PRIMARY KEY,FirstName varchar(255),LastName varchar(255),Email_ID varc
    #c.execute('Use restaurant_table_booking_system')

def add_data_cust(c_id, firstname, lastname, email, phone):
    c.execute('INSERT INTO Customer (C_ID, FirstName, LastName, Email_ID, Phone_number) VALUES (%s,%s,%s,%s,%s)',
              (c_id, firstname, lastname, email, phone))
    mydb.commit()

def add_data_rest(restaurant_id, rname, descrip, phone,street_name,city):
    c.execute('INSERT INTO Restaurant (Restaurant_ID, Restaurant_Name, Description, Phone_number, Street_name, City) VALUES (%s,%s,
              (restaurant_id, rname, descrip, phone,street_name,city))
    mydb.commit()

def add_data_rev(c_id,restaurant_id,review,date):
    c.execute('INSERT INTO Review (C_ID, Restaurant_ID, Review , Date_Recorded) VALUES (%s,%s,%s,%s)',
              (c_id, restaurant_id, review, date))
    mydb.commit()
```

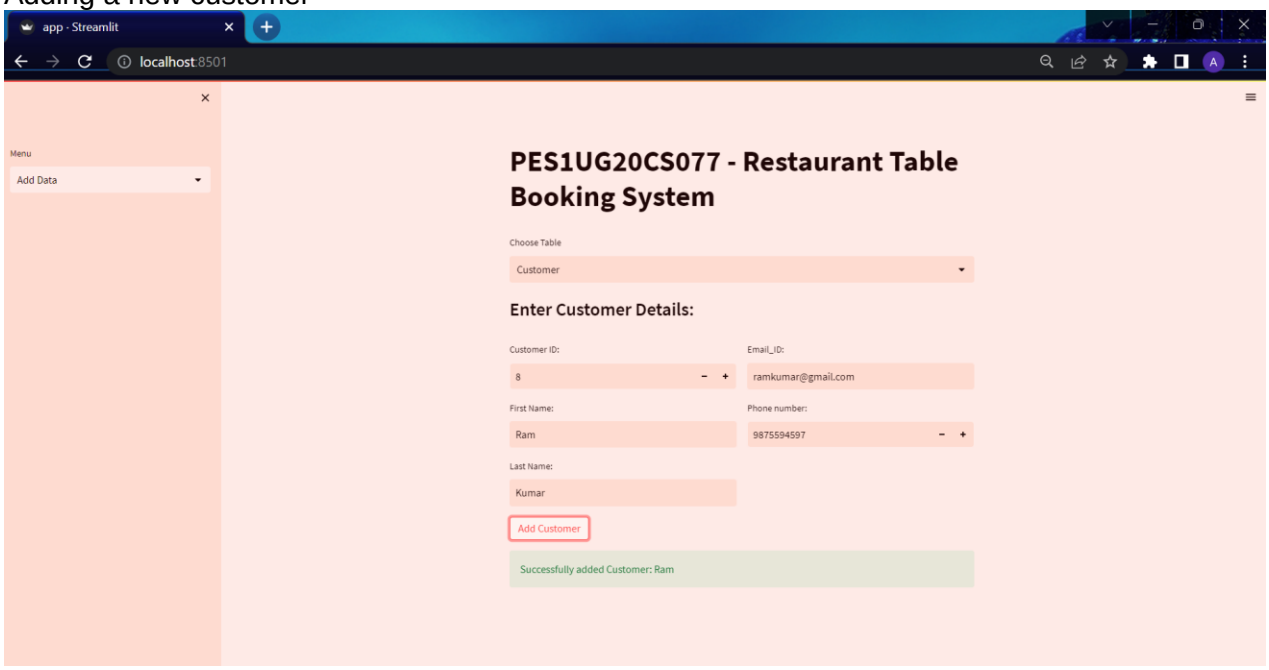
Developing a Frontend

The frontend should support

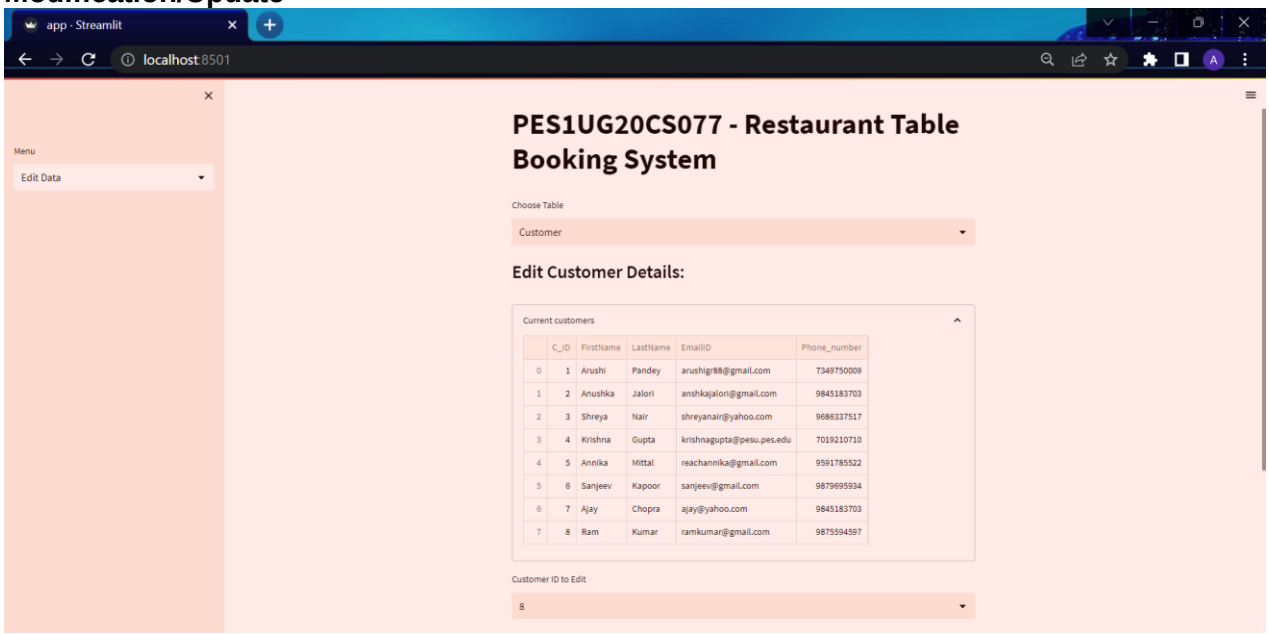
1. Addition, Modification and Deletion of records from any chosen table
2. There should be an window to accept and run any SQL statement and display the result

Addition Operation-

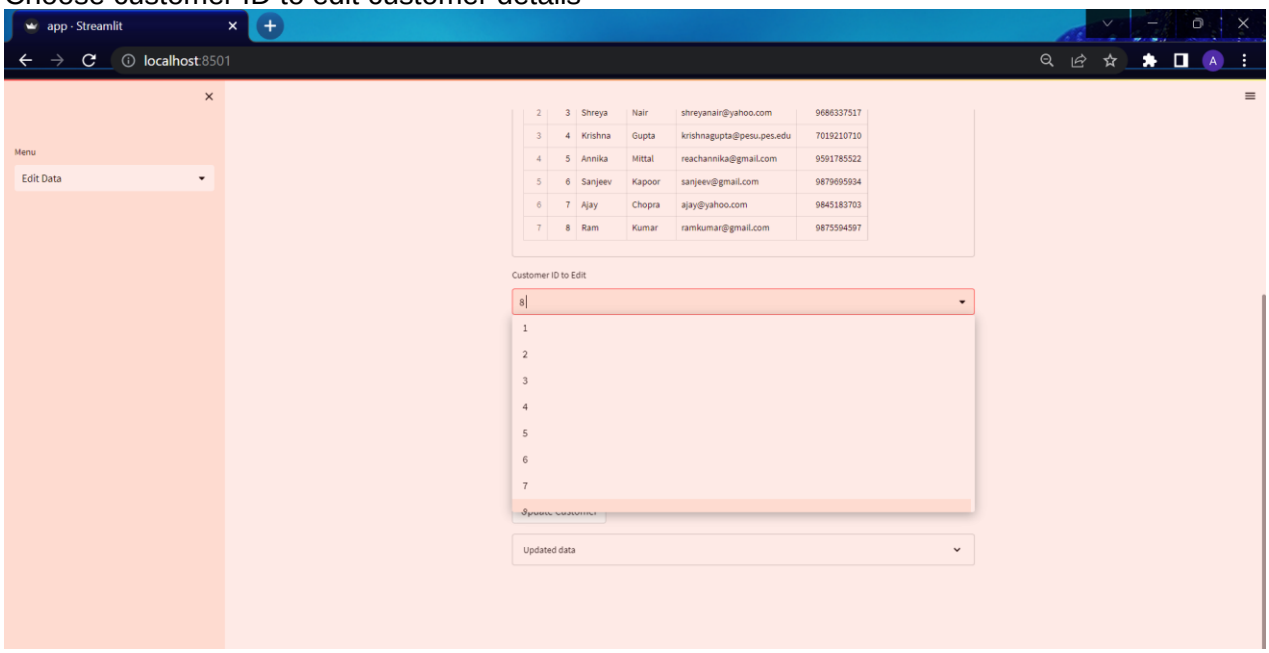
Adding a new customer



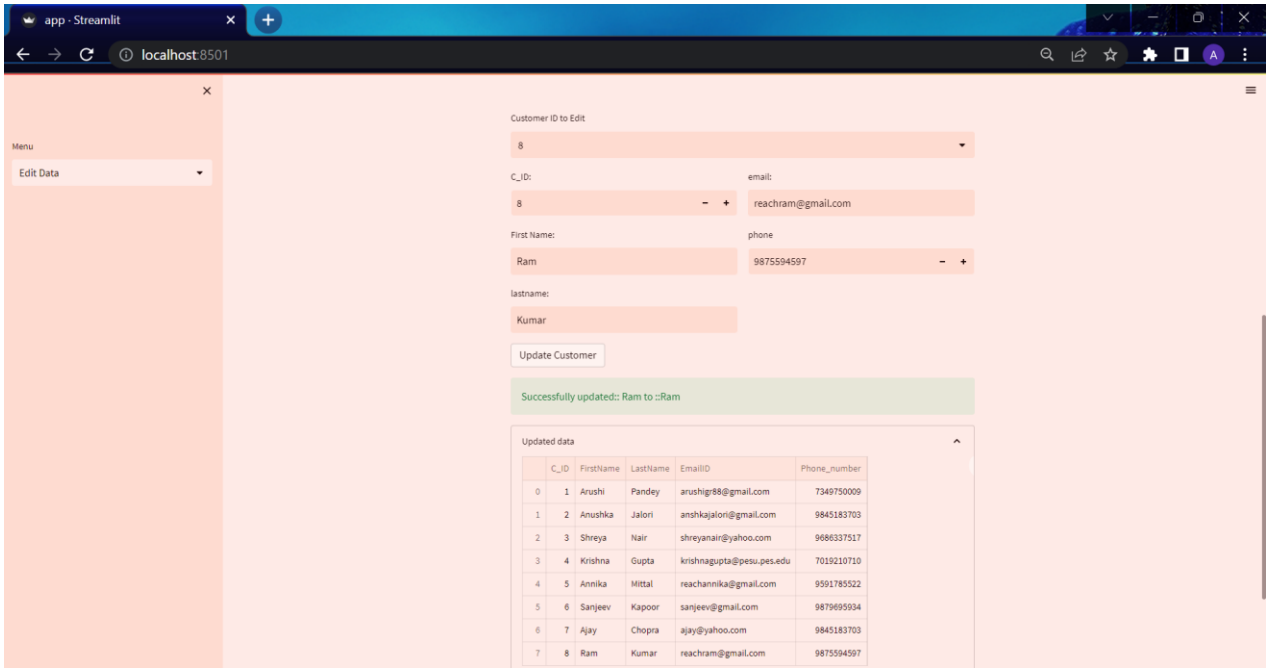
Modification/Update



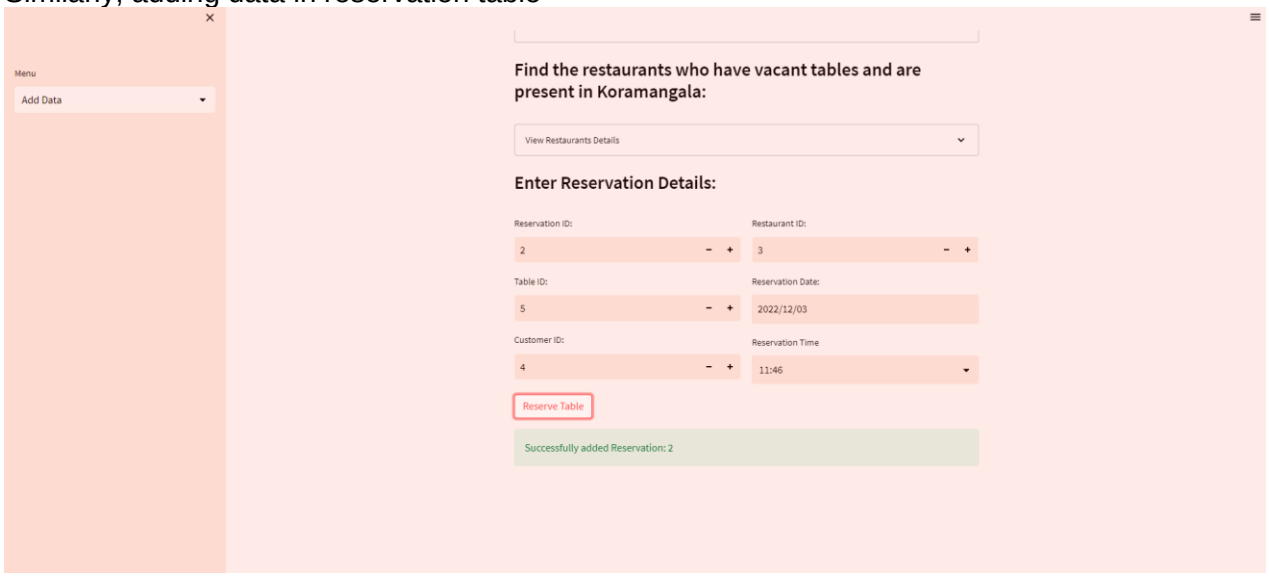
Choose customer ID to edit customer details



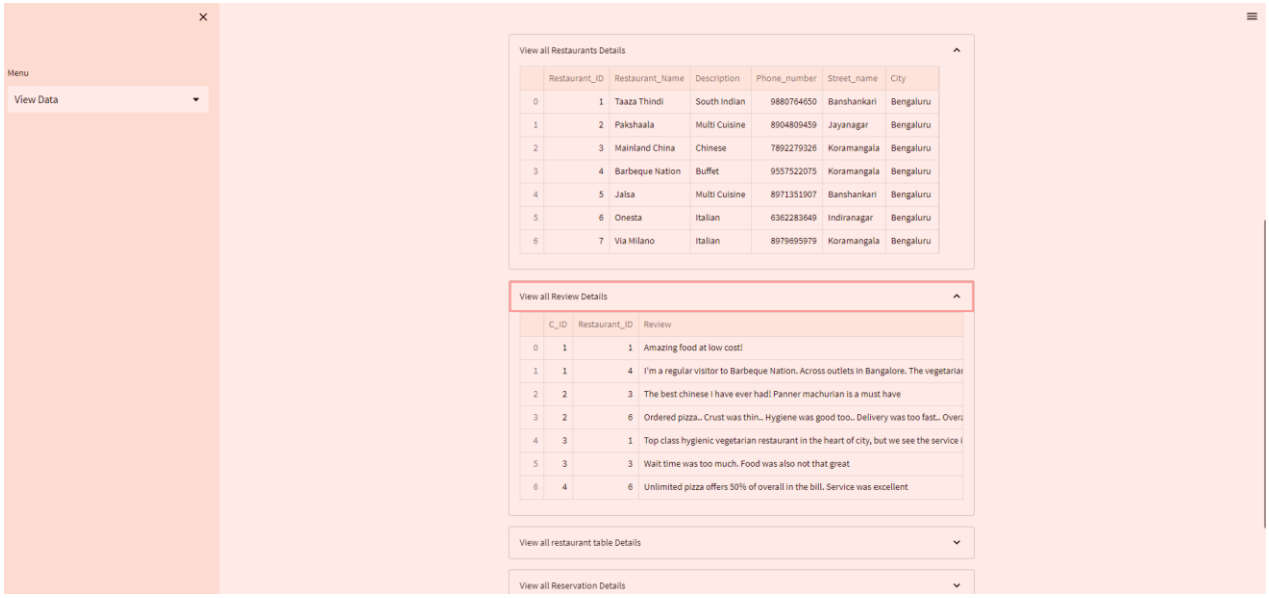
Press Update button to update the customer

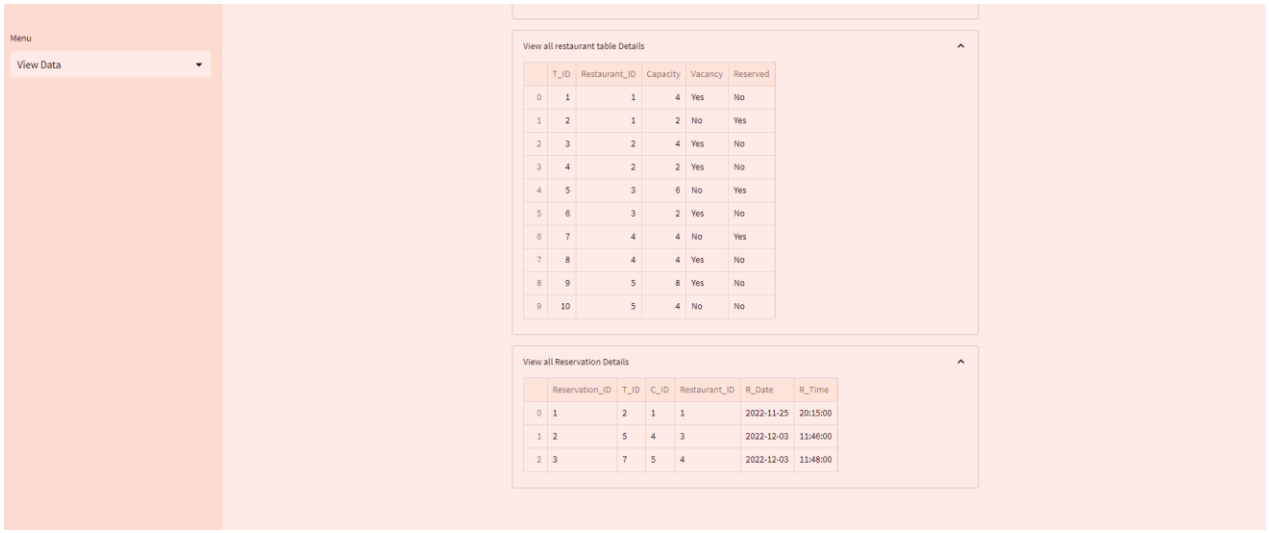


Similarly, adding data in reservation table-



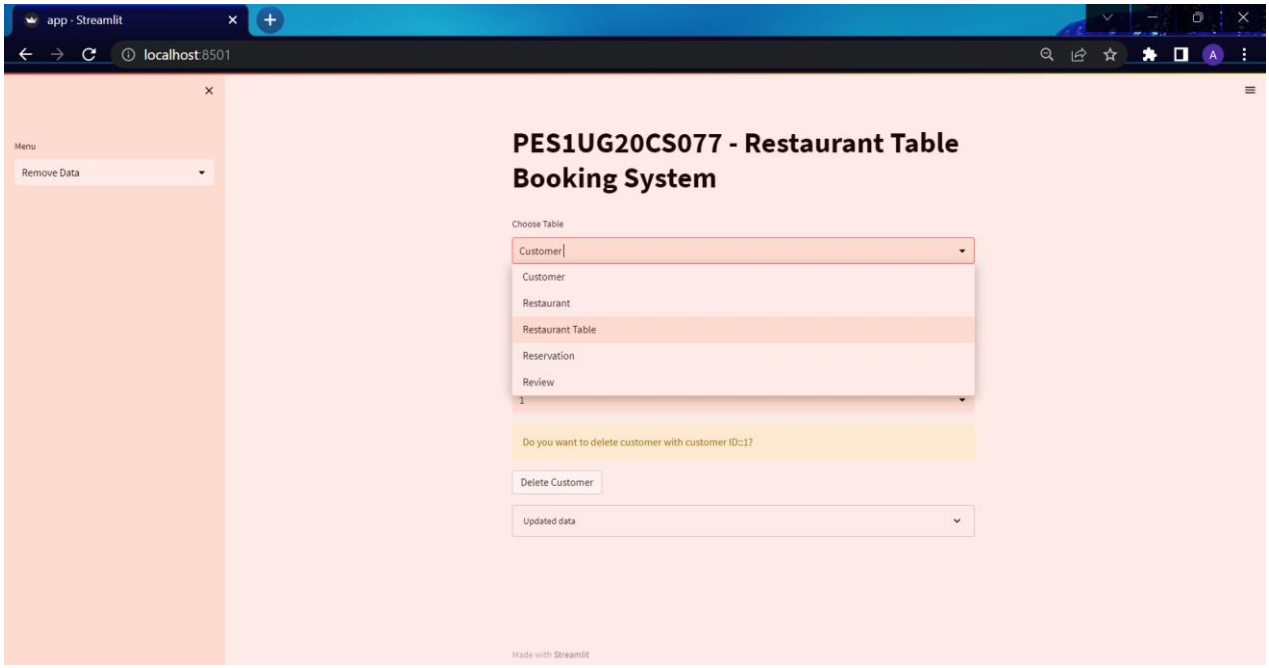
View Tables-

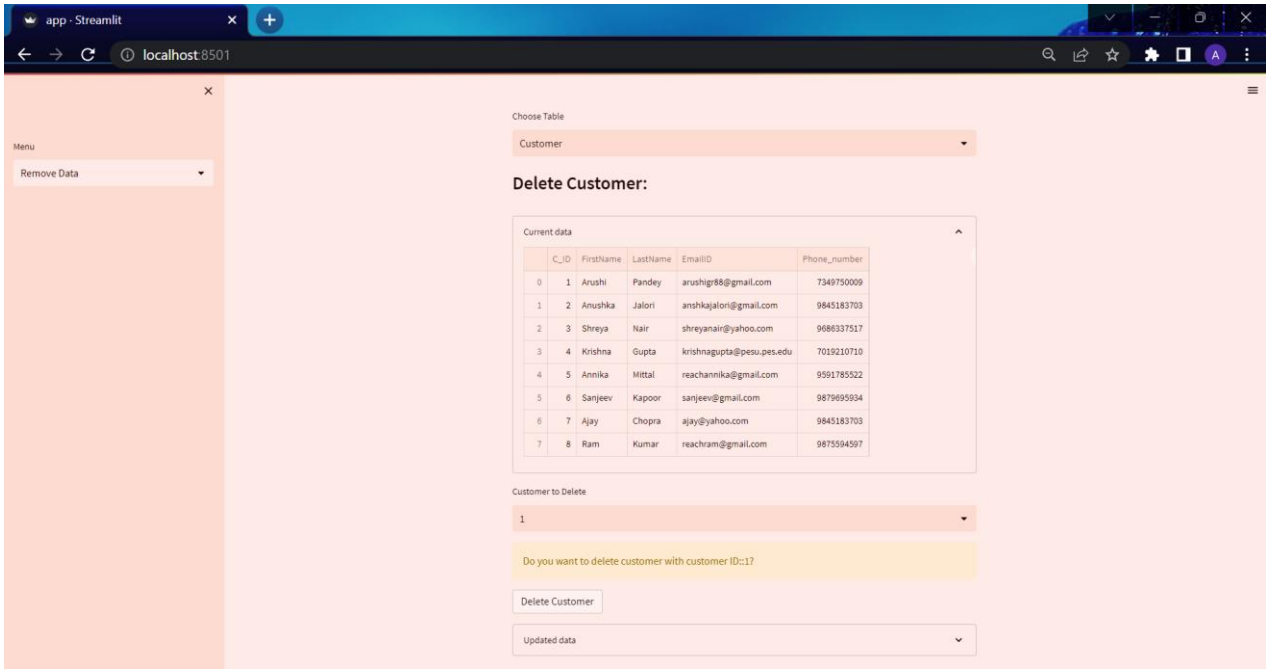




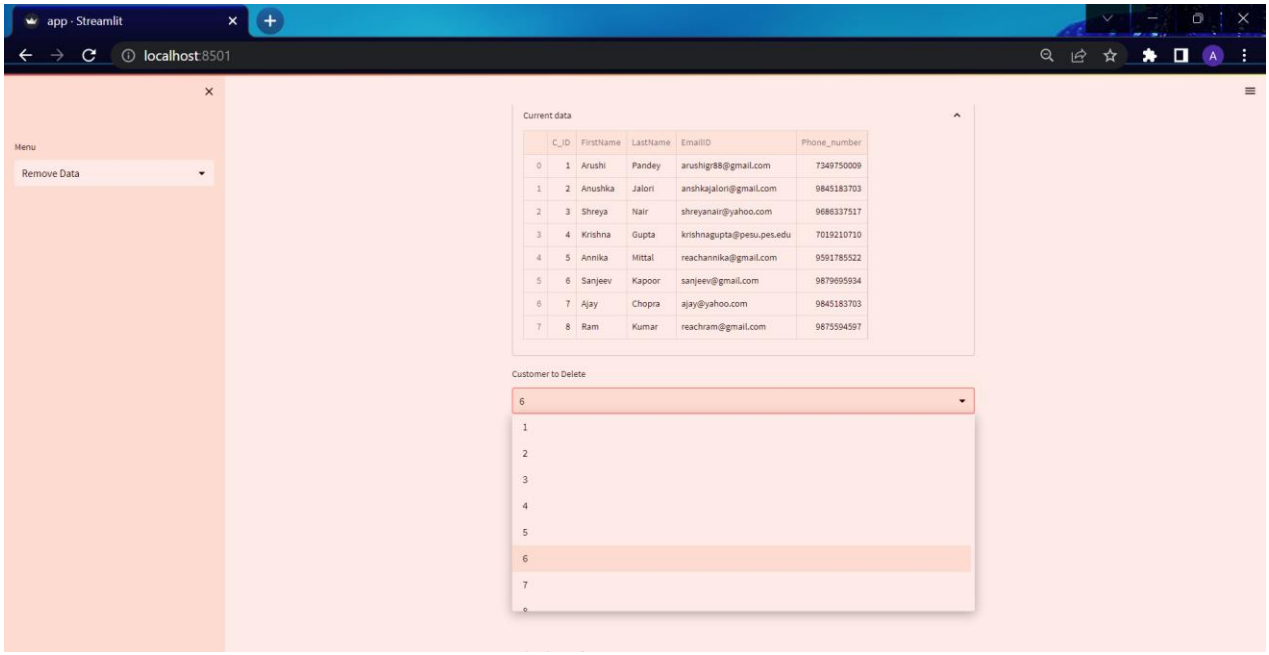
Delete-

Select table to delete from-

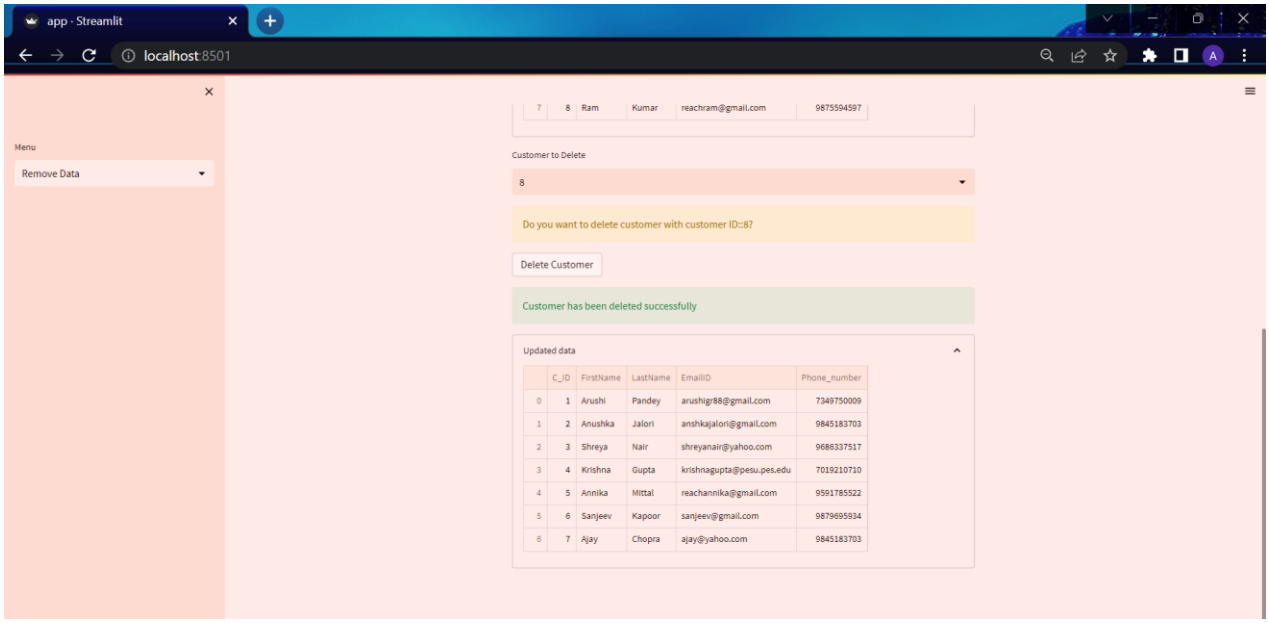




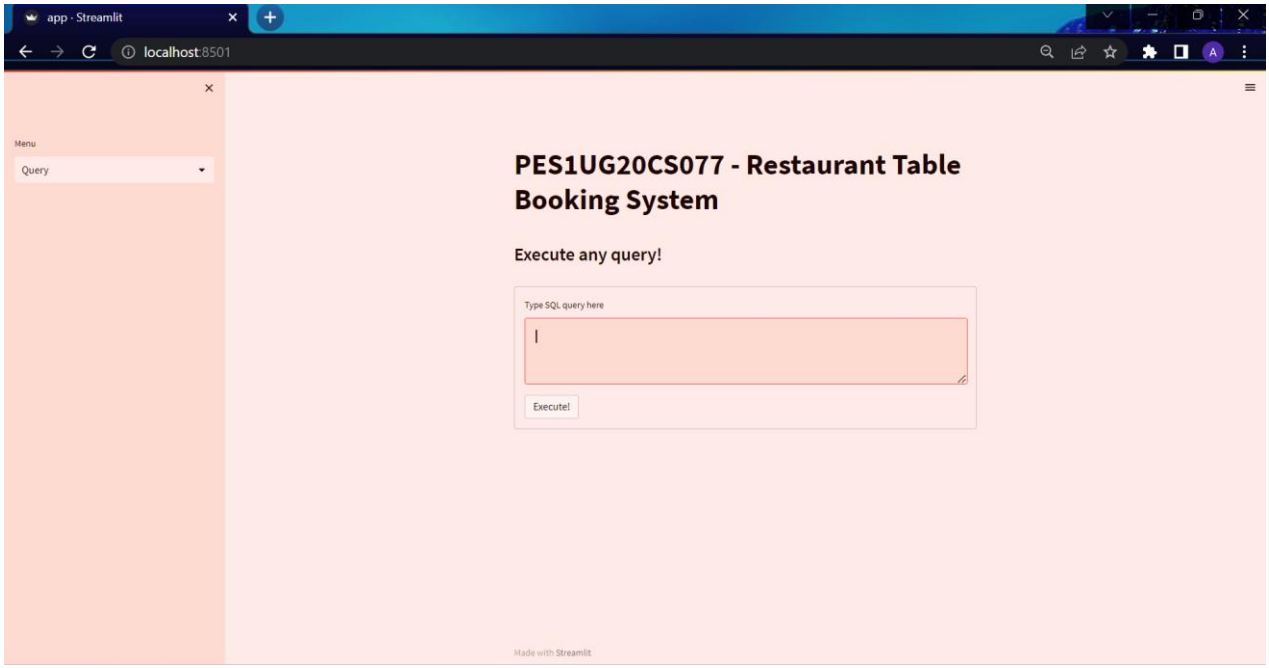
Select customer ID to delete-



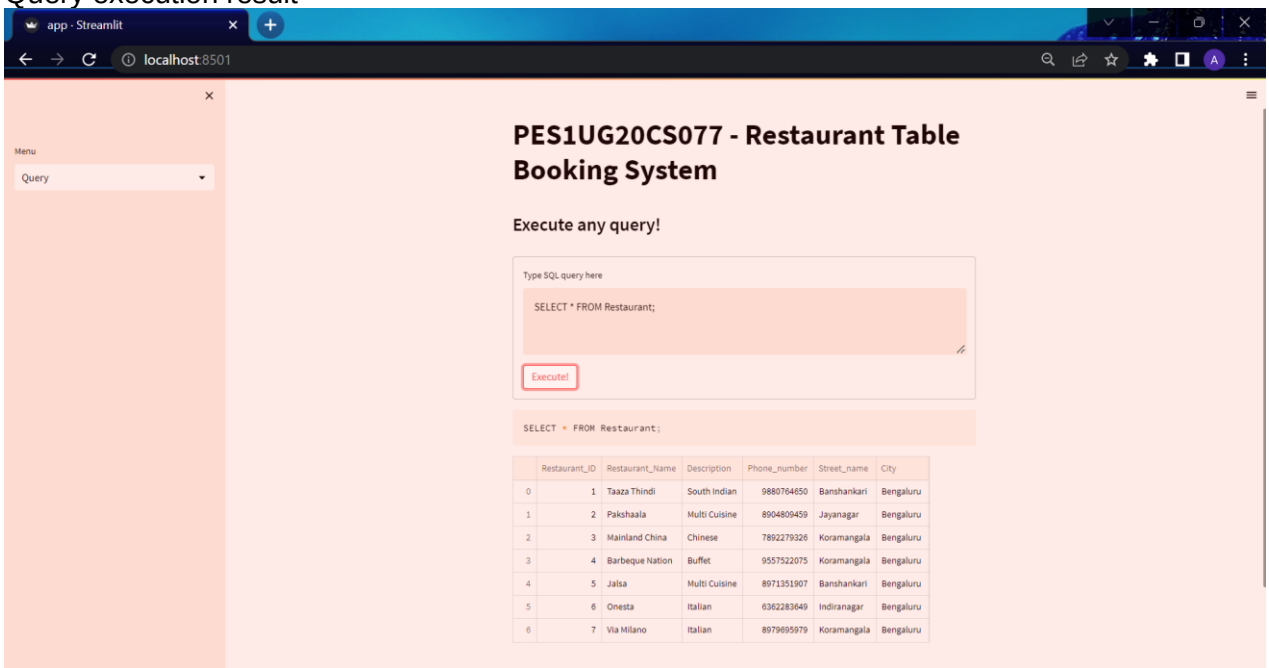
Click on delete button to delete the customer



Query in UI



Query execution result



	Restaurant_ID	Restaurant_Name	Description	Phone_number	Street_name	City
0	1	Taaza Thindi	South Indian	9880764650	Banshankari	Bengaluru
1	2	Pakshaala	Multi Cuisine	8904809459	Jayanagar	Bengaluru
2	3	Mainland China	Chinese	7892279326	Koramangala	Bengaluru
3	4	Barbeque Nation	Buffet	9557522075	Koramangala	Bengaluru
4	5	Jalsa	Multi Cuisine	8971351907	Banshankari	Bengaluru
5	6	Onesta	Italian	6362283649	Indiranagar	Bengaluru
6	7	Via Milano	Italian	8979695979	Koramangala	Bengaluru

CRUD operations have been implemented for all 5 tables.
Also we can view data for all 5 tables.
Join,procedure, trigger, set operations and aggregate operations have also been implemented in frontend.