DBMS MINI PROJECT TITLE OF THE PROJECT:

BUS BOOKING MANAGEMENT SYSTEM

## Name: Rajesh M Mysoremath SRN No: PES1UG20CS579

5th Semester, Section: J

# Description and Scope of Project:

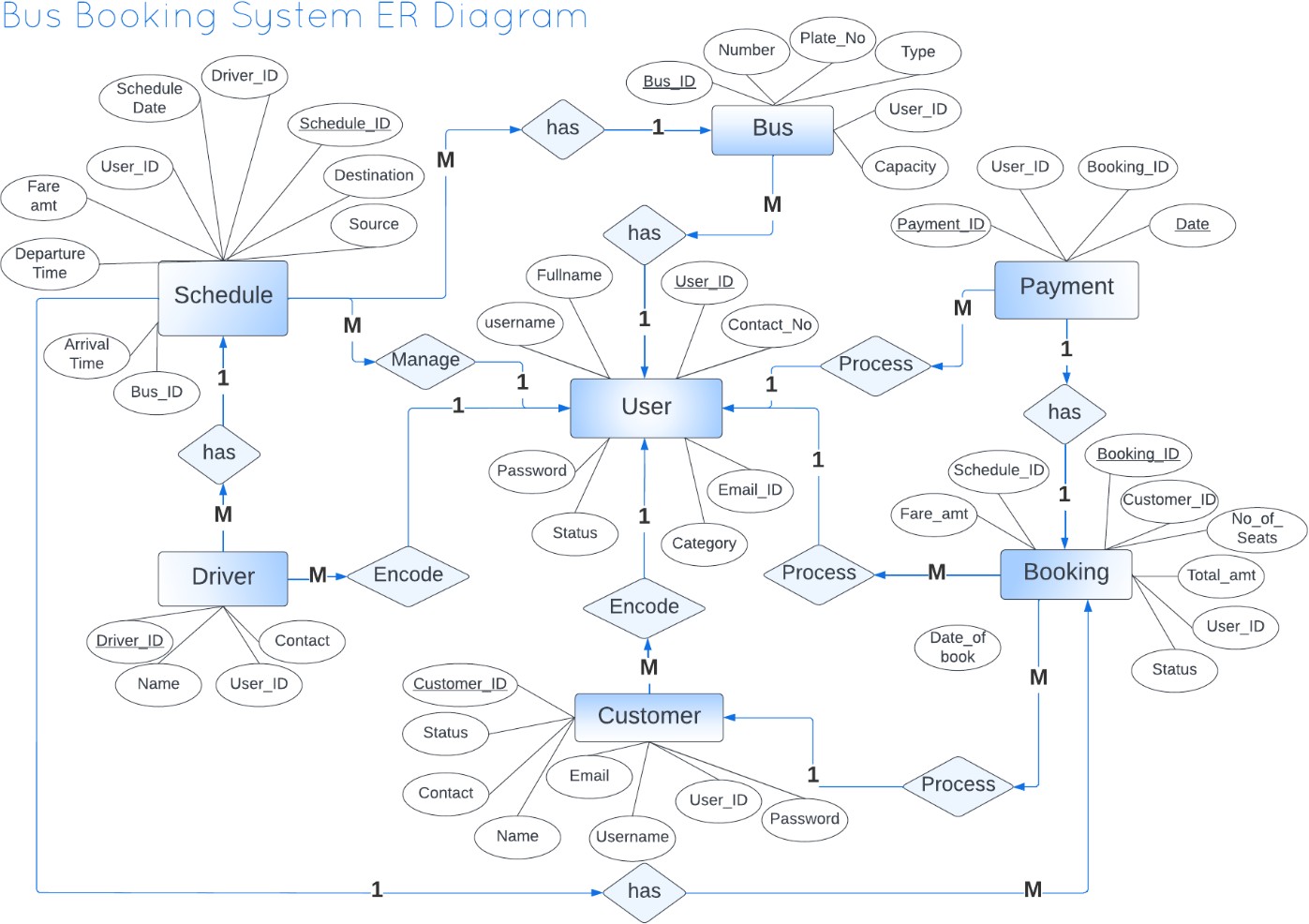
Online bus booking management system is a project which presents a portal for bus ticket reservation. This application lets in users to book bus tickets from somewhere and anytime. The customer can without difficulty e book their tickets and cancel tickets. The user can additionally view the details of the journey.

Features We can see in this project, Thread operations in

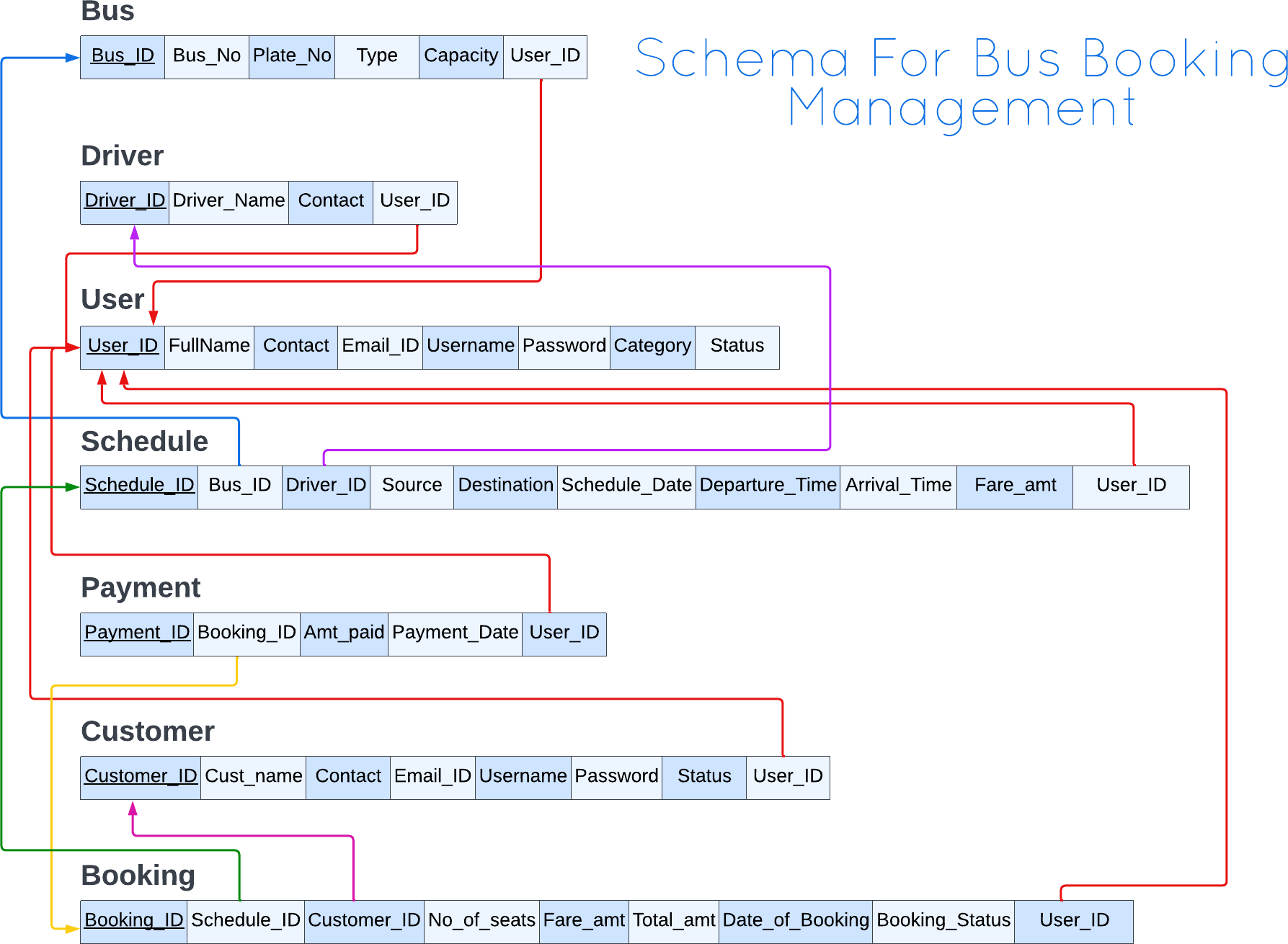
* Users
* Bus
* Driver
* Customer
* Schedule
* Booking
* Payment
* Query Box

The customer to enquire the availability of seats in a particular bus at particular date. To reserve a ticket and he can cancel a reserved ticket. The current bus booking system relies on buying tickets from the conductor for commuting to and from a location through public transportation. The task can be tedious if the number of commuters is large. Also, payment in cash can be difficult if the payable denominations are uneven. Online Bus Booking system allows the computer to either have a specific amount of money on his Android Based mobile, from which the ticket can be charged. Or, the costumer can buy the ticket on the bus.

# ER Diagram



Relational Schema



# DDL Statements

## Building The Database

1. Create User Table

CREATE TABLE `user` (

`users\_id` int(11) NOT NULL,

`full\_name` varchar(50) NOT NULL,

`contact\_no` varchar(15) NOT NULL,

`email\_address` varchar(30) NOT NULL,

`username` varchar(30) NOT NULL,

`userpassword` varchar(30) NOT NULL,

`account\_category` varchar(100) NOT NULL,

`account\_status` varchar(100) NOT NULL)

ALTER TABLE `user`

ADD PRIMARY KEY (`users\_id`);

## Create Bus Table

CREATE TABLE `bus` (

`bus\_id` int(11) NOT NULL,

`bus\_number` varchar(15) NOT NULL,

`bus\_plate\_number` varchar(15) NOT NULL,

`bus\_type` int(1) NOT NULL,

`capacity` int(3) NOT NULL,

`users\_id` int(11) NOT NULL)

ALTER TABLE `bus`

ADD PRIMARY KEY (`bus\_id`), ADD KEY `user\_id` (`users\_id`);

## Create Driver Table

CREATE TABLE `driver` (

`driver\_id` int(11) NOT NULL,

`driver\_name` varchar(50) NOT NULL,

`driver\_contact` varchar(15) NOT NULL,

`users\_id` int(11) NOT NULL)

ALTER TABLE `driver`

ADD PRIMARY KEY (`driver\_id`), ADD KEY `user\_id` (`users\_id`);

## Create Customer Table

CREATE TABLE `customer` (

`customer\_id` int(11) NOT NULL,

`customer\_name` varchar(50) NOT NULL,

`customer\_contact` varchar(15) NOT NULL,

`customer\_email` varchar(30) NOT NULL,

`username` varchar(30) NOT NULL,

`cust\_password` varchar(30) NOT NULL,

`account\_status` varchar(100) NOT NULL,

`users\_id` int(11) NOT NULL)

ALTER TABLE `customer`

ADD PRIMARY KEY (`customer\_id`), ADD KEY `user\_id` (`users\_id`);

## Create Schedule Table

CREATE TABLE `schedule` (

`schedule\_id` int(11) NOT NULL,

`bus\_id` int(11) NOT NULL,

`driver\_id` int(11) NOT NULL,

`starting\_point` varchar(30) NOT NULL,

`destination` varchar(30) NOT NULL,

`schedule\_date` date NOT NULL,

`departure\_time` varchar(100) NOT NULL,

`estimated\_arrival\_time` varchar(100) NOT NULL,

`fare\_amount` int(11) NOT NULL,

`remarks` varchar(100) NOT NULL,

`users\_id` int(11) NOT NULL)

ALTER TABLE `schedule`

ADD PRIMARY KEY (`schedule\_id`),

ADD KEY `bus\_id` (`bus\_id`,`driver\_id`,`users\_id`), ADD KEY `user\_id` (`users\_id`),

ADD KEY `driver\_id` (`driver\_id`);

## Create Booking Table

CREATE TABLE `booking` (

`booking\_id` int(11) NOT NULL,

`schedule\_id` int(11) NOT NULL,

`customer\_id` int(11) NOT NULL,

`number\_of\_seats` int(2) NOT NULL,

`fare\_amount` int(11) NOT NULL,

`total\_amount` int(11) NOT NULL,

`date\_of\_booking` date NOT NULL,

`booking\_status` varchar(100) NOT NULL,

`users\_id` int(11) NOT NULL)

ALTER TABLE `booking`

ADD PRIMARY KEY (`booking\_id`),

ADD KEY `schedule\_id` (`schedule\_id`,`customer\_id`,`users\_id`), ADD KEY `user\_id` (`users\_id`),

ADD KEY `customer\_id` (`customer\_id`);

## Create Payment Table

CREATE TABLE `payment` (

`payment\_id` int(11) NOT NULL,

`booking\_id` int(11) NOT NULL,

`amount\_paid` int(11) NOT NULL,

`payment\_date` date NOT NULL,

`users\_id` int(11) NOT NULL)

ALTER TABLE `payment`

ADD PRIMARY KEY (`payment\_id`),

ADD KEY `booking\_id` (`booking\_id`,`users\_id`), ADD KEY `user\_id` (`users\_id`);

# Populating The Database

## Populating Users

INSERT INTO `user` (`users\_id`, `full\_name`, `contact\_no`, `email\_address

`, `username`, `userpassword`, `account\_category`, `account\_status`) VA LUES ('1564679', 'Tasmai', '78945013.0', 'Tasmai123@gmail.com', 'Tasmai', ' Tasmai@123', '1', 'Active'), ('4789658', 'Sagar', '798441614.0', 'sagar123@gm ail.com', 'Sagar', 'Sagar@123', '1', 'Deactive')

## Populating Buses

INSERT INTO `bus` (`bus\_id`, `bus\_number`, `bus\_plate\_number`, `bus\_ty pe`, `capacity`, `users\_id`) VALUES ('4561187', '7896254', 'KA124579', 'Non

-AC Sleeper', '45', '4789658'), ('7862157', '156467', 'KA214357', 'Non-

AC', '57', '1564679')

## Populating Drivers

INSERT INTO `bus` (`bus\_id`, `bus\_number`, `bus\_plate\_number`, `bus\_ty pe`, `capacity`, `users\_id`) VALUES ('4561187', '7896254', 'KA124579', 'Non

-AC Sleeper', '45', '4789658'), ('7862157', '156467', 'KA214357', 'Non-

AC', '57', '1564679')

## Populating Customers

INSERT INTO `customer` (`customer\_id`, `customer\_name`, `customer\_c ontact`, `customer\_email`, `username`, `cust\_password`, `account\_statu s`, `users\_id`) VALUES ('4547987', 'hari', '79461648', 'hari123@gmail.com', 'Hari', 'Hari@123', 'Not

Done', '1564679'), ('6014064', 'Sharu', '244600747', 'sharu@gmail.com', 'Sh aru', 'Sharu@123', 'Done', '4789658')

## Populating Schedules

INSERT INTO `schedule` (`schedule\_id`, `bus\_id`, `driver\_id`, `starting\_poi nt`, `destination`, `schedule\_date`, `departure\_time`, `estimated\_arrival\_ti me`, `fare\_amount`, `remarks`, `users\_id`) VALUES ('154578', '4561187', '16

5461', 'Pune', 'Banglore', '2022-12-14', '05:45:00', '06:15:00', '2540', 'via:

Bangalore', '4789658'), ('7916548', '7862157', '781647', 'Bangalore', 'Mumba

i', '2022-11-29', '01:58:00', '01:58:00', '4578', 'via: Pune', '1564679')

## Populating Booking

INSERT INTO `booking` (`booking\_id`, `schedule\_id`, `customer\_id`, `numb er\_of\_seats`, `fare\_amount`, `total\_amount`, `date\_of\_booking`, `booking\_ status`, `users\_id`) VALUES ('4679416', '7916548', '4547987', '15', '1235', '13

00', '2022-11-

29', 'Confirmed', '4789658'), ('7841647', '154578', '6014064', '25', '4578', '460

0', '2022-12-15', 'Not Confirmed', '1564679')

## Populating Payment

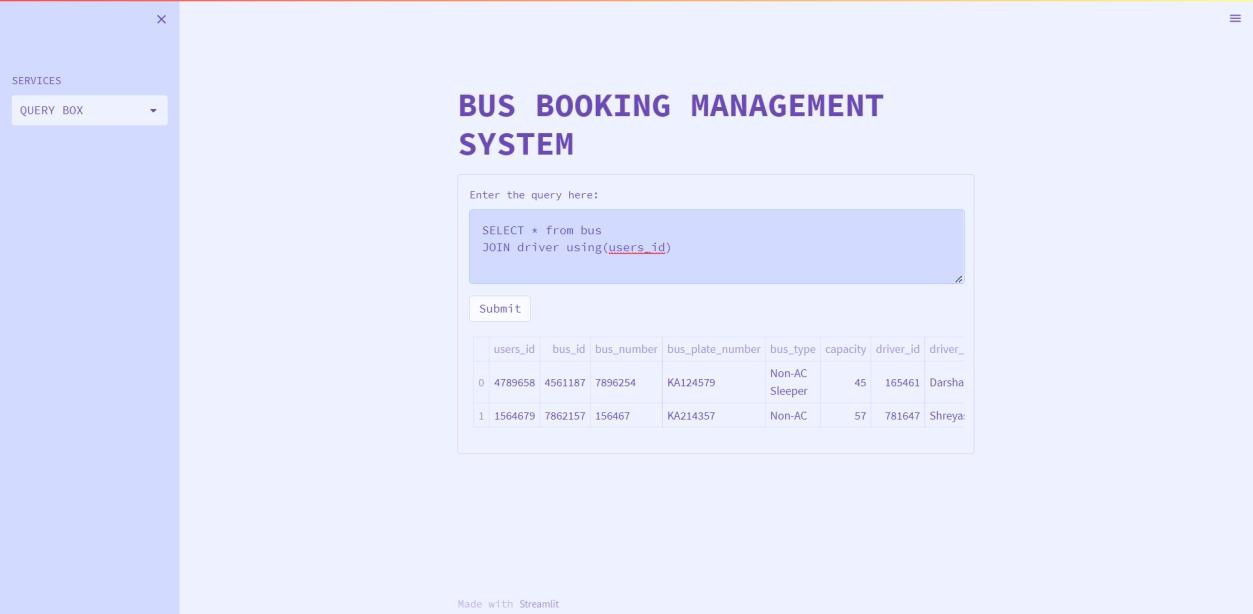
INSERT INTO `payment` (`payment\_id`, `booking\_id`, `amount\_paid`, `pay ment\_date`, `users\_id`) VALUES ('164878', '4679416', '4578', '2022-12-

22', '4789658'), ('4679167', '7841647', '1542', '2022-11-29', '1564679')

# JOIN QUERIES

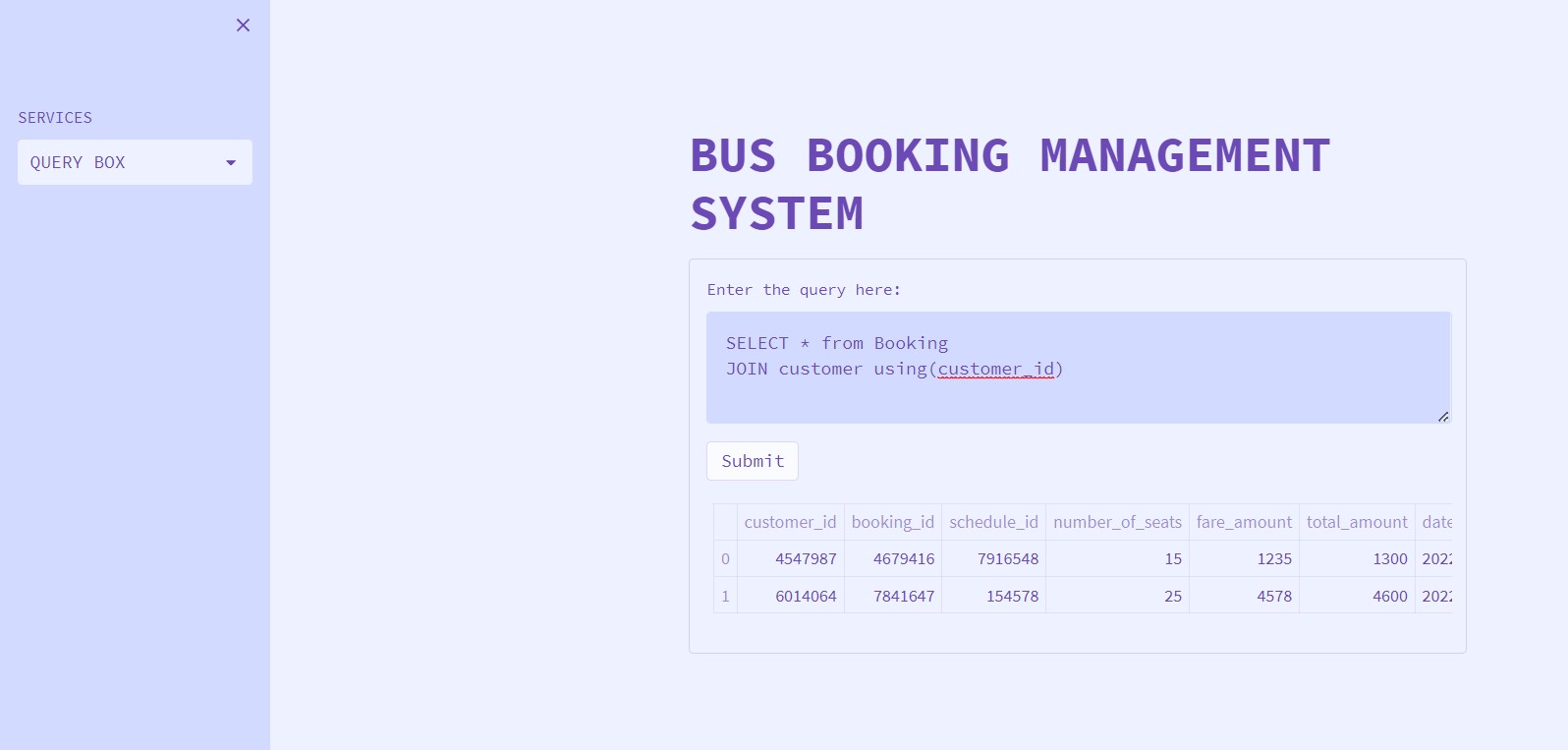
Query 1:

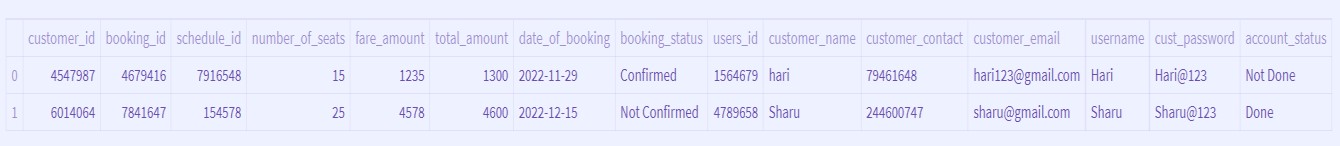
Joining bus and driver table using users\_id attribute



# Query 2:

Joining Booking table and Customer table using customer\_id attribute

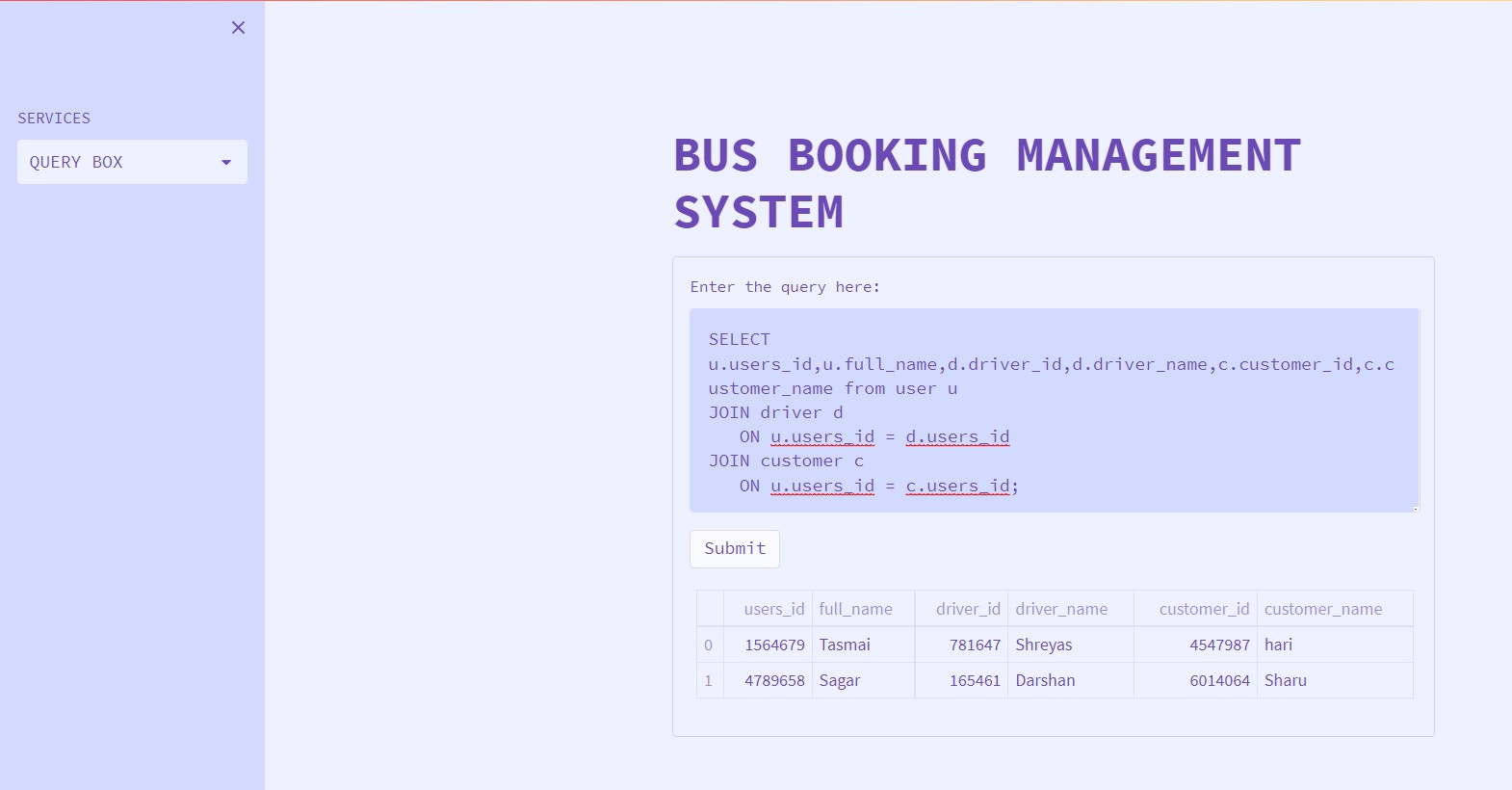




# Query 3:

### Joining 3 tables (User table, driver table, Customer table) using common attributes

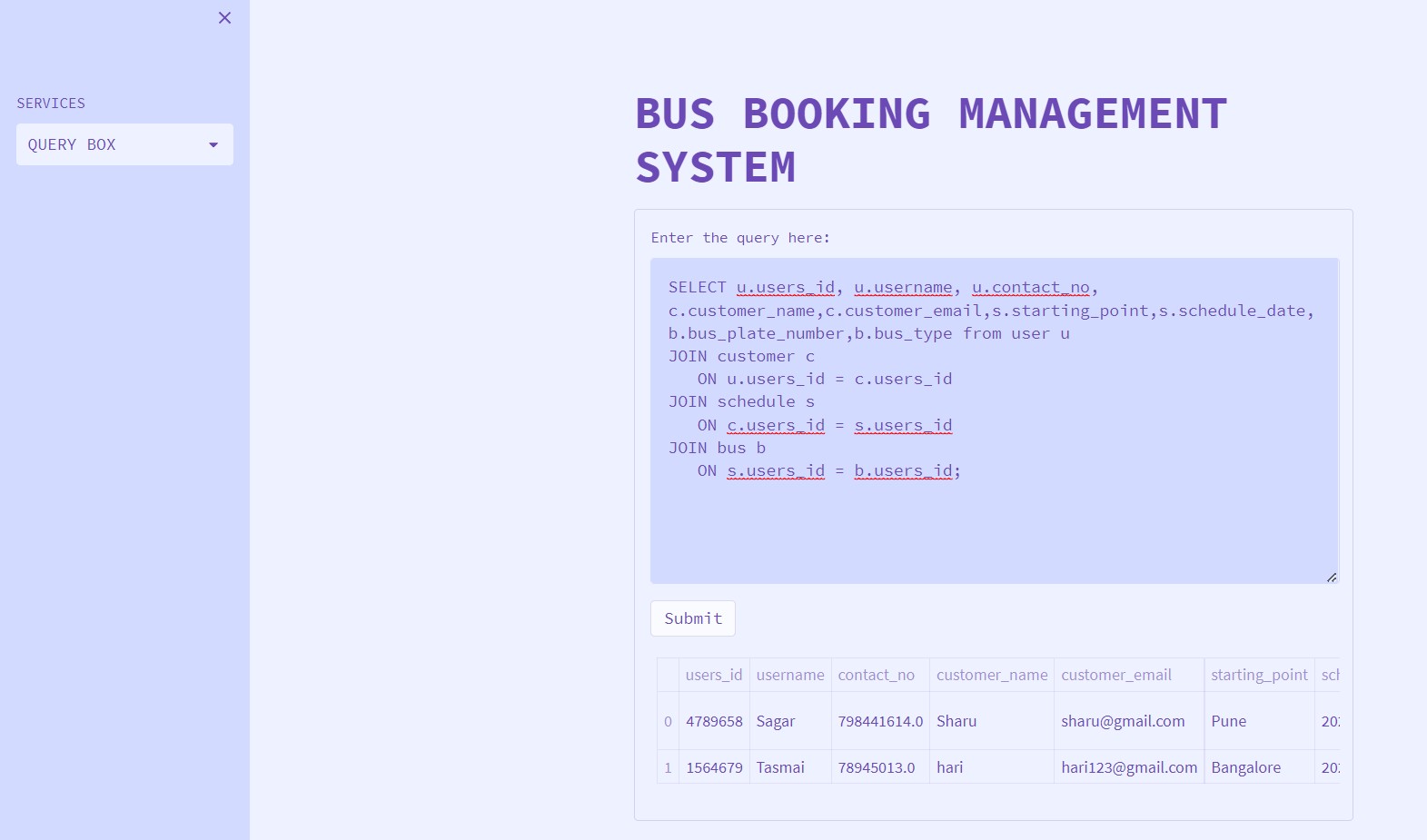
SELECľ u.useís\_id,u.full\_name,d.díiveí\_id,d.díiveí\_name, c.customeí\_id, c.c ustomeí\_name fíom useí u JOIN díiveí d ON u.useís id = d.useís id JOIN customeí c ON u.useís id = c.useís id;



# Query 4:

### Joining 4 tables (User table, Customer table, Schedule and bus table) using common attributes

SELECľ u.useís id, u.useíname, u.contact no, c.customeí\_name, c.customeí\_email,s.staíting point,s.schedule\_date, b.bus\_plate\_numbeí,b.bus\_type fíom useí u JOIN customeí c ON u.useís\_id = c.useís\_id JOIN schedule s ON c.useís id = s.useís id JOIN bus b ON s.useís id = b.useís id;

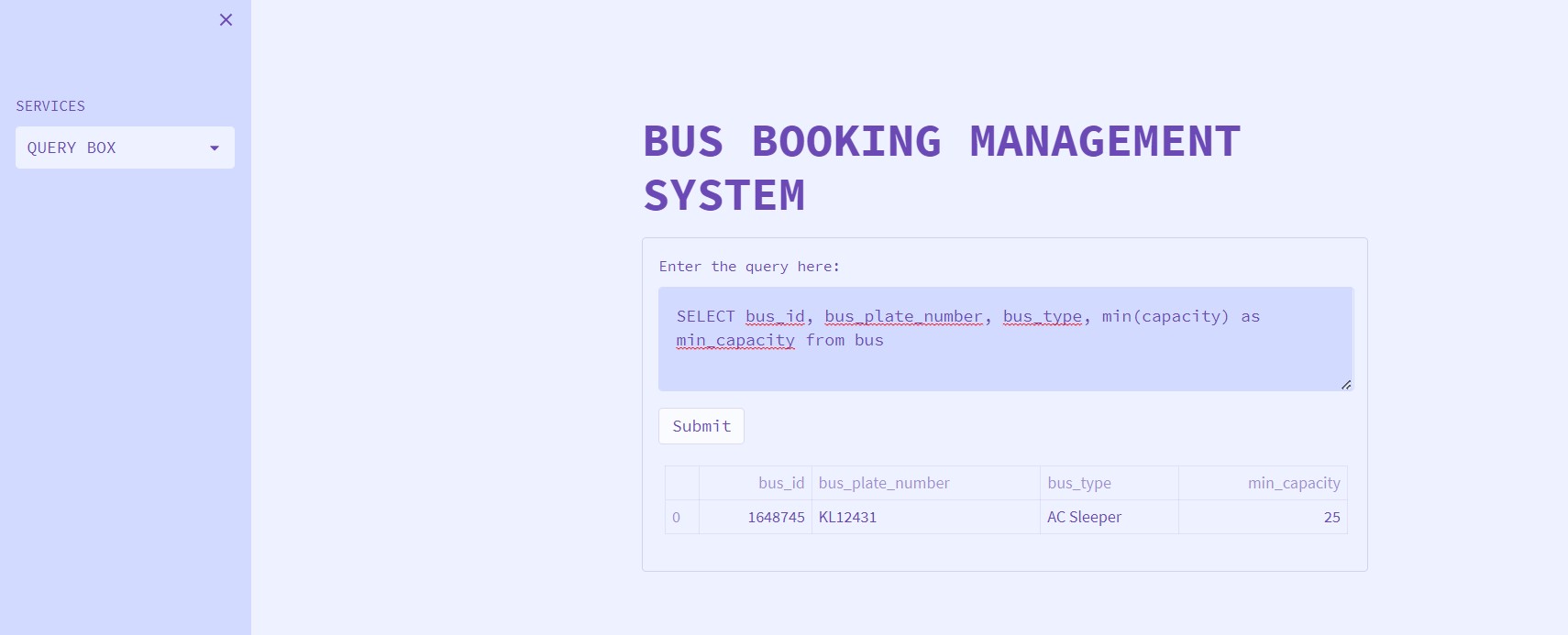




# Aggregate Functions

Query 1:

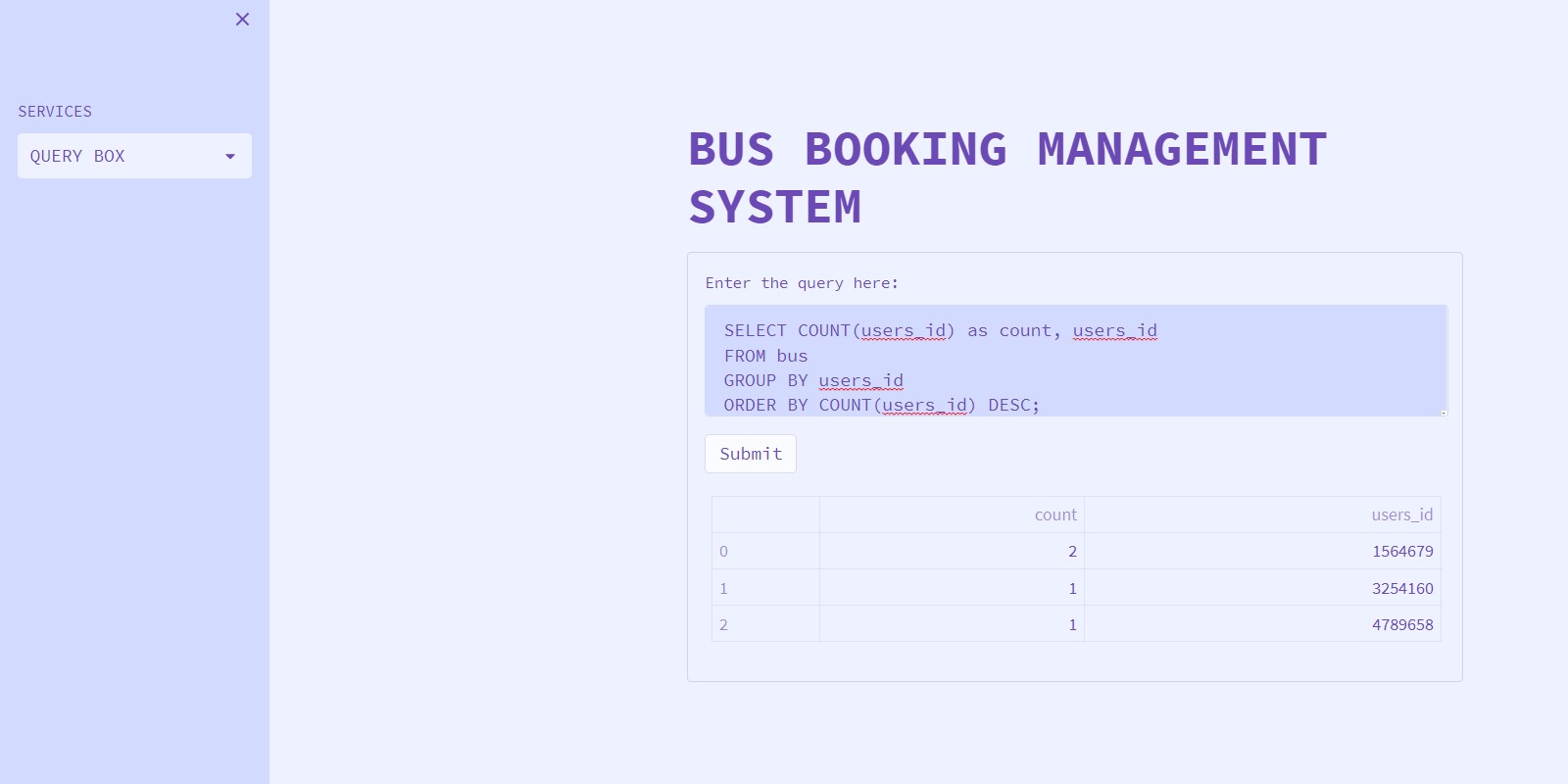
Getting the Details of minimum number of capacities in bus table



# Query 2:

## Listing Number of Users in Bus table, and sorted low to high according to user id.

SELECľ COUNľ(useís id) as count, useís id FROM bus GROUP BY useís\_id ORDER BY COUNľ (useís\_id) DESC;



# Query 3:

List the number of entries in our bus table having capacity > 46



# Query 4:

### Using min () and max () functions to display max and min no. of Capacity in bus table

SELECľ MIN(capacity) AS Minimum\_Capacity,| MAX (capacity) AS Maximum\_Capacity FROM bus



# Set Operations

Query 1:

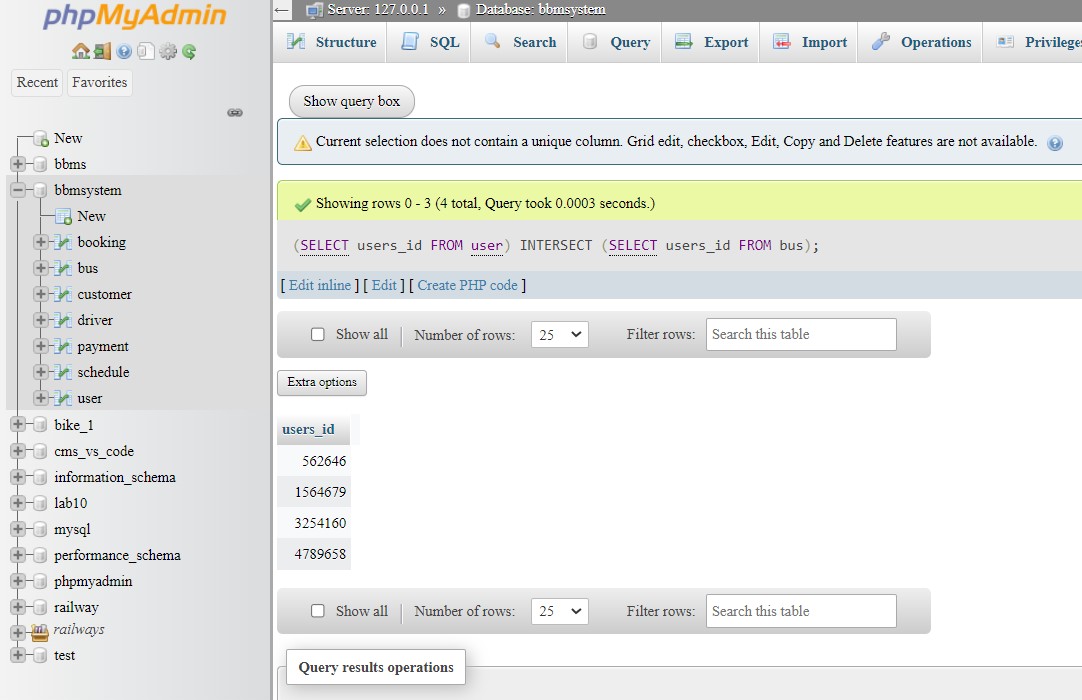
Union operation between user and Customer table of our database





# Query 2:

Intersection between user and bus table



# Query 3:

### Using union operation over bus table to differentiate users as availability low or high.

SELECT bus\_id,bus\_plate\_number,bus\_type, ‘ low’ as availability

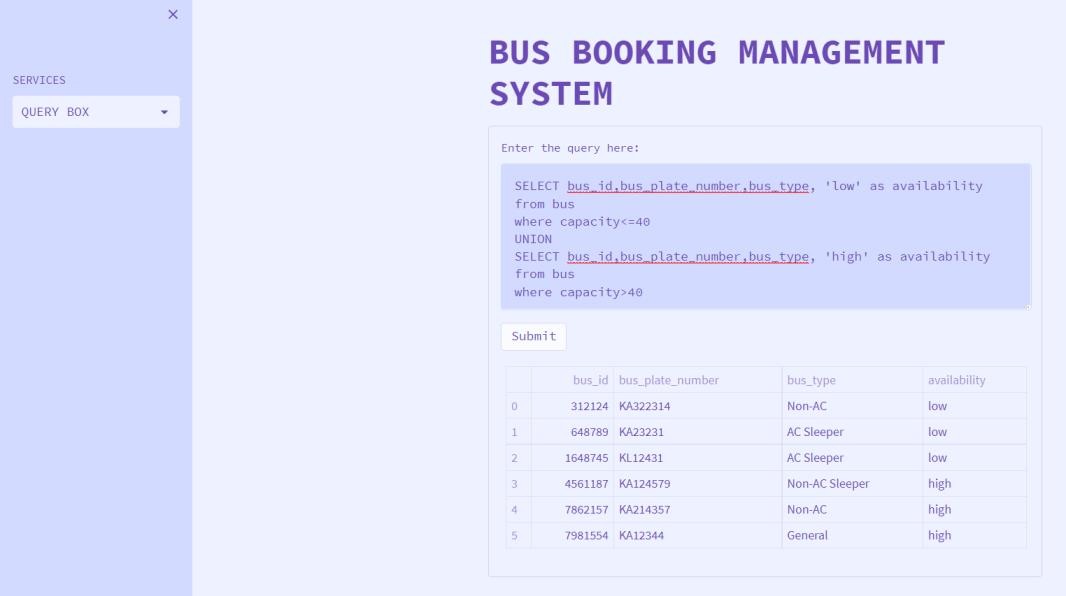
From bus

Where capacity<=40 UNION

SELECT bus\_id,bus\_plate\_number,bus\_type, ‘high’ as availability

From bus

Where capacity>=40

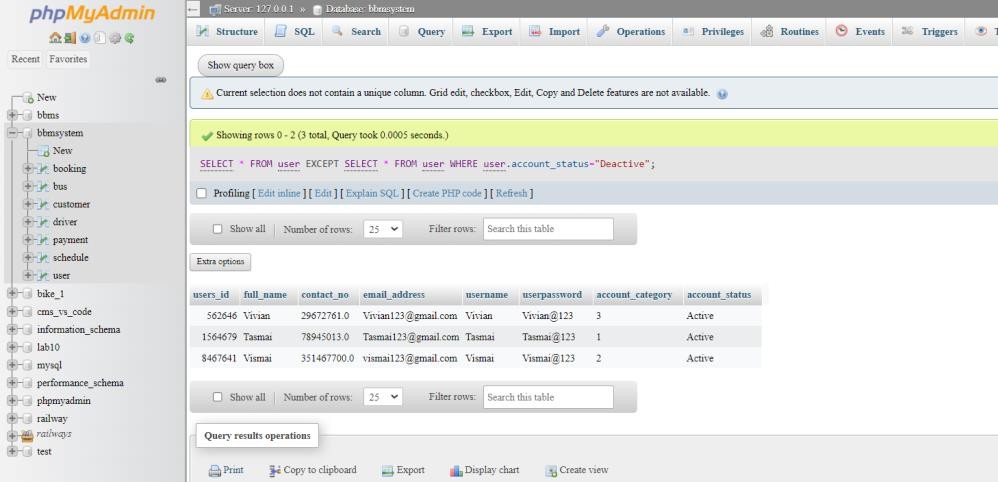


# Query 4:

### Find all users except account status is deactive.

SELECT \* FROM user EXCEPT;

SELECT \* FROM user WHERE user.acoount\_status='Deactive';



# Functions and Procedures

Query 1:

### The below query is function to get if the fare amount is less than 1600 then cheap otherwise expensive in schedule table

Function:

DELIMITER $$

CREATE DEFINER=`root`@`localhost` FUNCTION `price`(`fare\_amount` INT) RETURNS varchar(20) CHARSET utf8mb4

DETERMINISTIC BEGIN

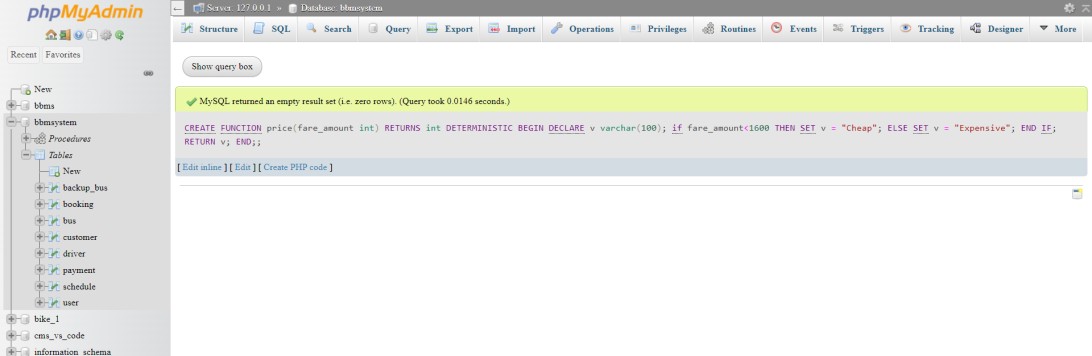
DECLARE v varchar(100); if fare\_amount<1600 THEN SET v = "Cheap";

ELSE

SET v = "Expensive"; END IF;

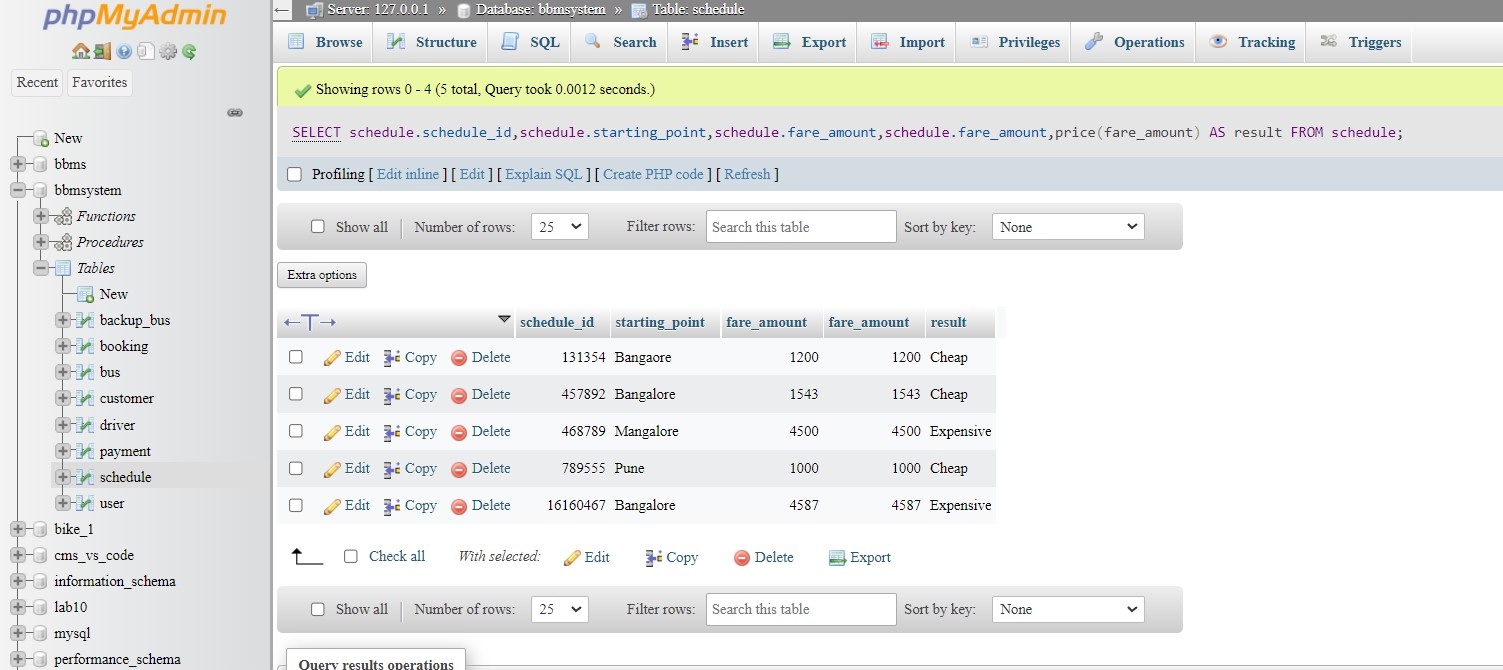
RETURN v;

END$$ DELIMITER ;



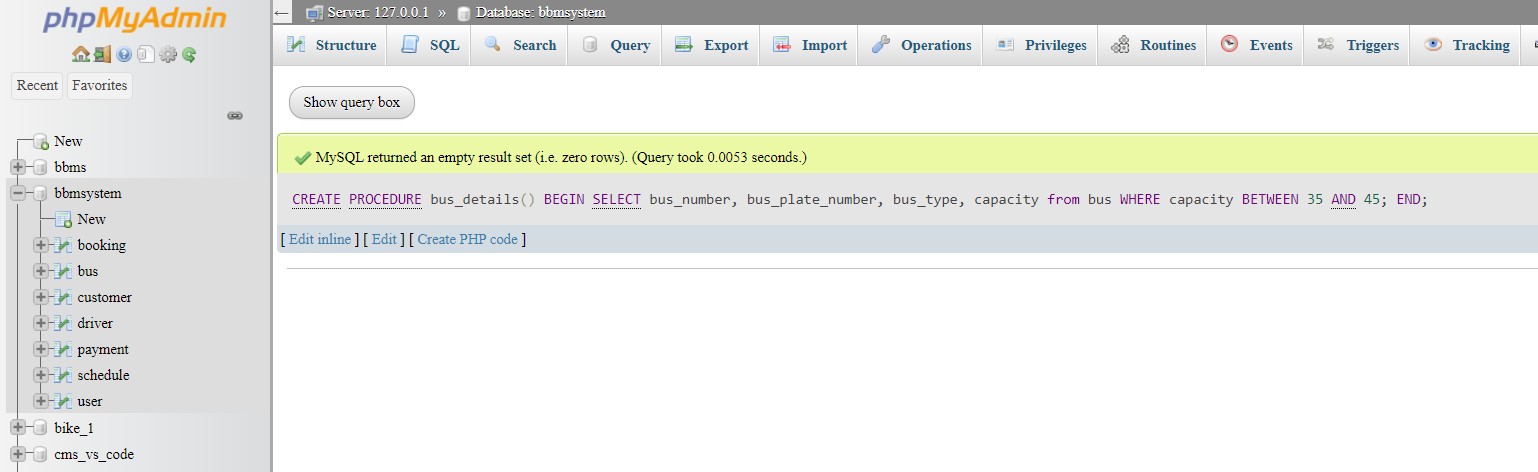
### Check Result:

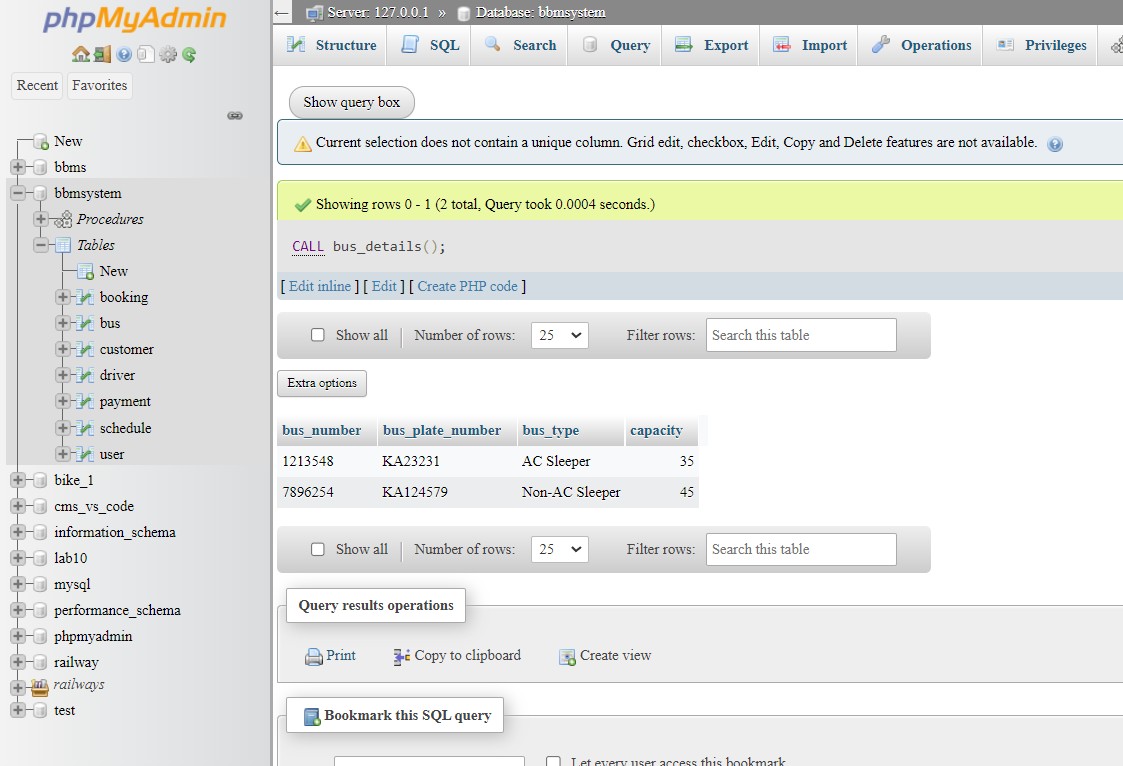
SELECT schedule.schedule\_id,schedule.starting point,schedule.fare\_amount,price(fare\_amount) AS result FROM schedule;



# Query 2:

The below query is to write a procedure to get bus number, plate number, bus types from bus table where bus capacity is between 35 and 45

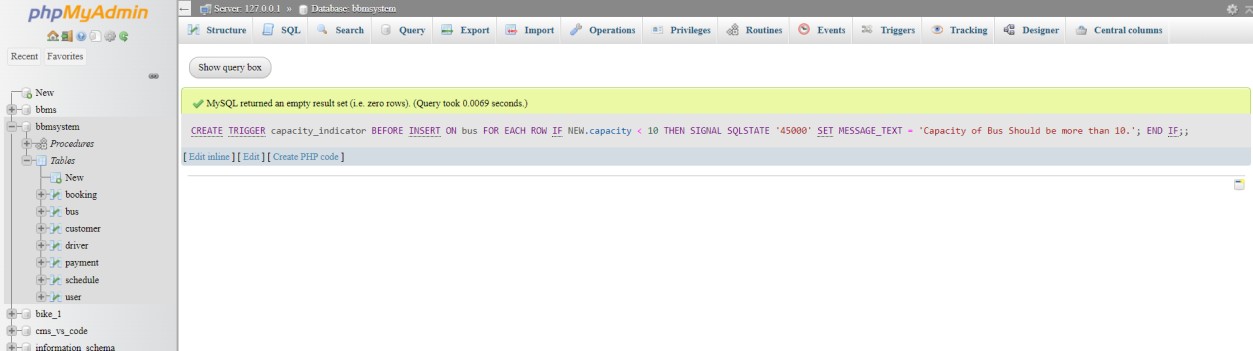




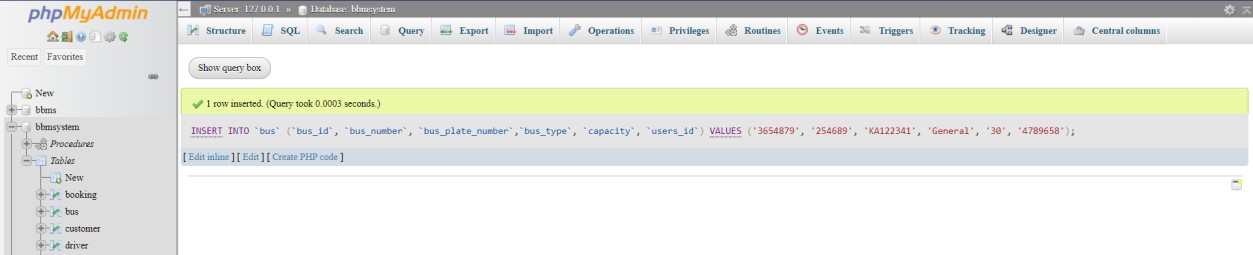
# Triggers and Cursors

Trigger:

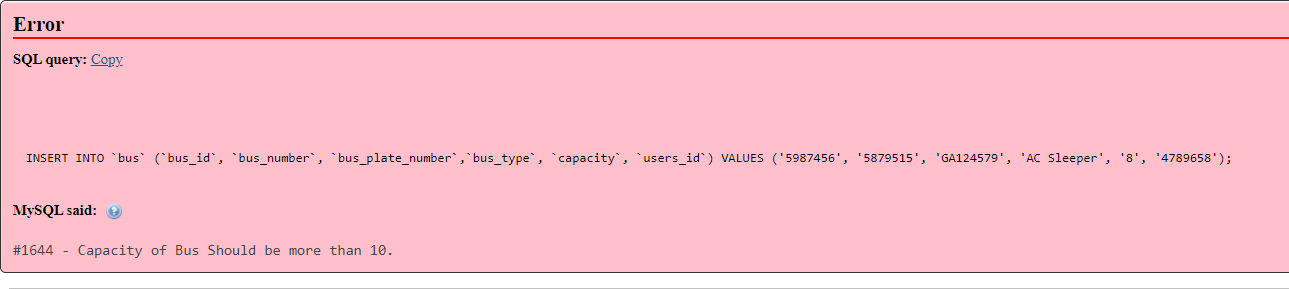
### Trigger named capacity\_indicator which is created for bus table we won’t take bus capacity which are less than 10.



When we insert capacity more than 10.



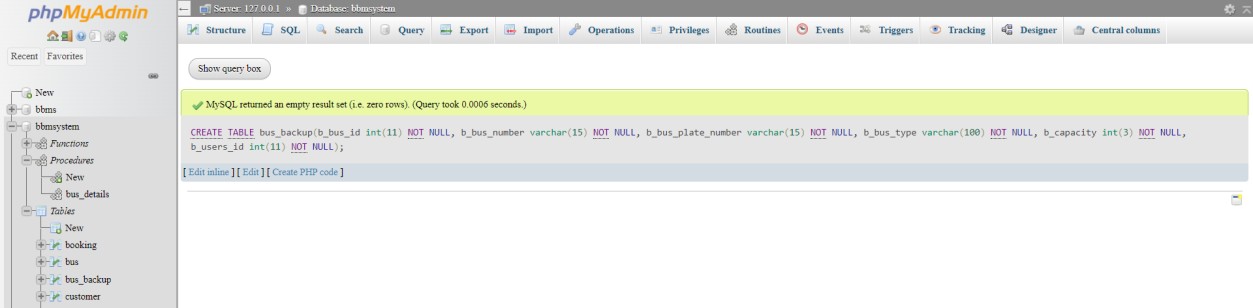
When we insert capacity less than 10.



# Cursors:

### Create a cursor to make a backup of bus table

Creating cursor table:



Procedure with cursor

DELIMITER $$

CREATE DEFINER=`root`@`localhost` PROCEDURE `backup\_bus`() BEGIN

DECLARE done int DEFAULT 0; DECLARE bus\_id int(11);

DECLARE bus\_number varchar(15); DECLARE bus\_plate\_number varchar(15); DECLARE bus\_type varchar(100); DECLARE capacity int(3);

DECLARE users\_id int(11);

DECLARE cur CURSOR FOR SELECT \* FROM bus;

DECLARE CONTINUE HANDLER FOR NOT Found SET done=1;

OPEN cur; label: LOOP

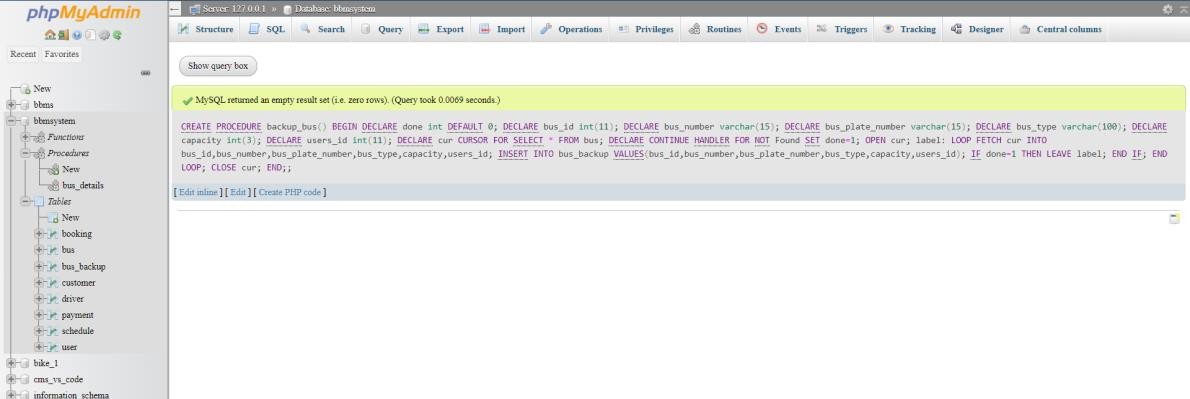
FETCH cur INTO bus\_id,bus\_number,bus\_plate\_number,bus\_type,capacity,users\_id;

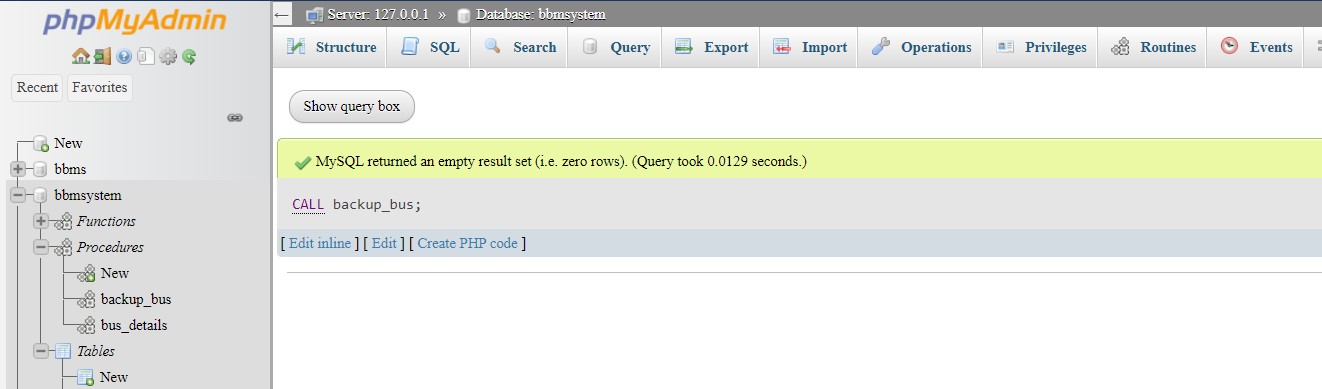
INSERT INTO bus\_backup VALUES(bus\_id,bus\_number,bus\_plate\_number,bus\_type,capacity,users\_id);

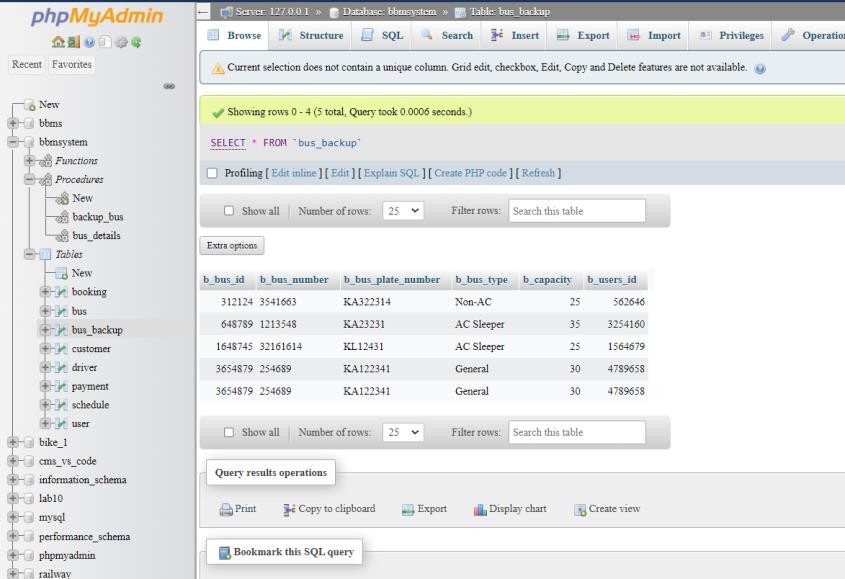
IF done=1 THEN LEAVE label;

END IF; END LOOP;

CLOSE cur; END$$ DELIMITER ;

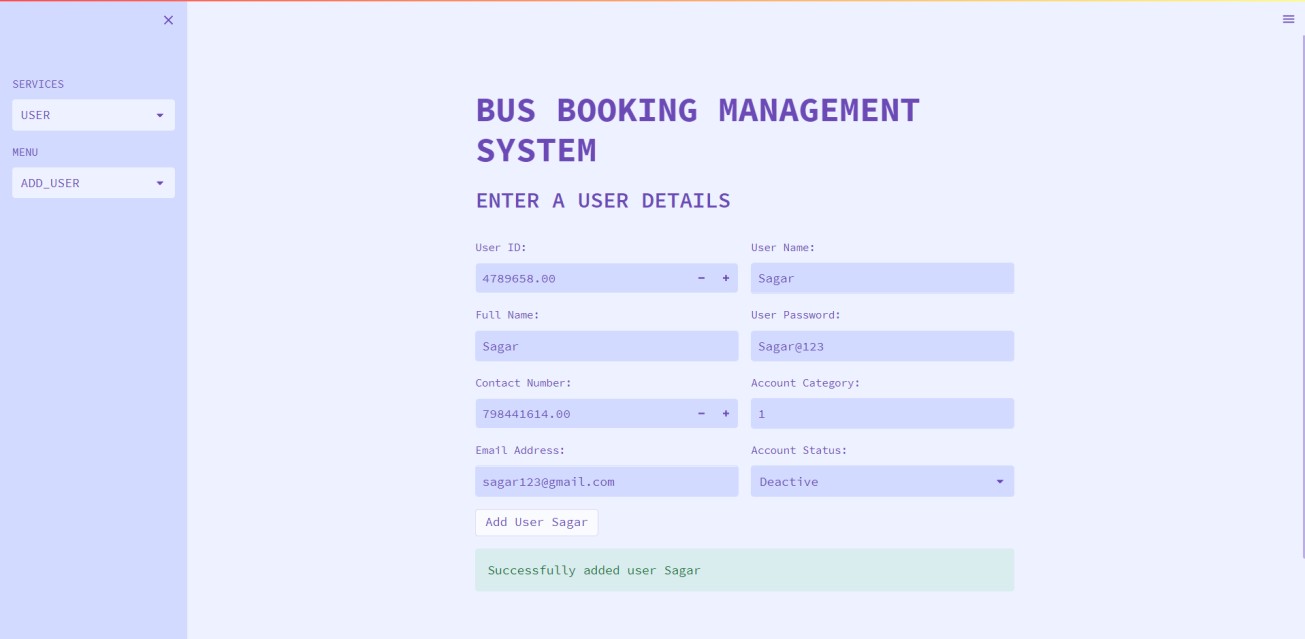






# Frontend Using Streamlit

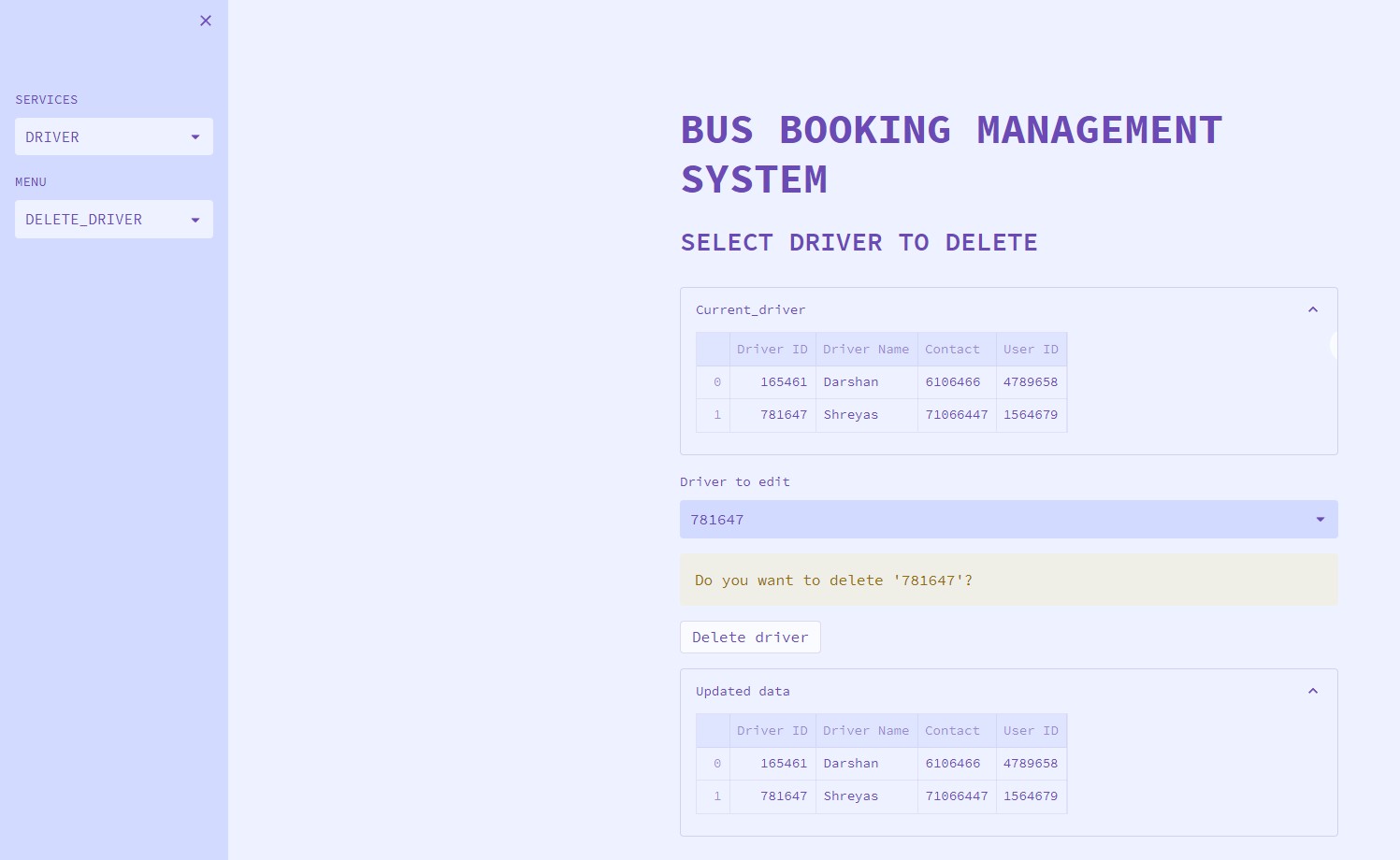
User adding Page



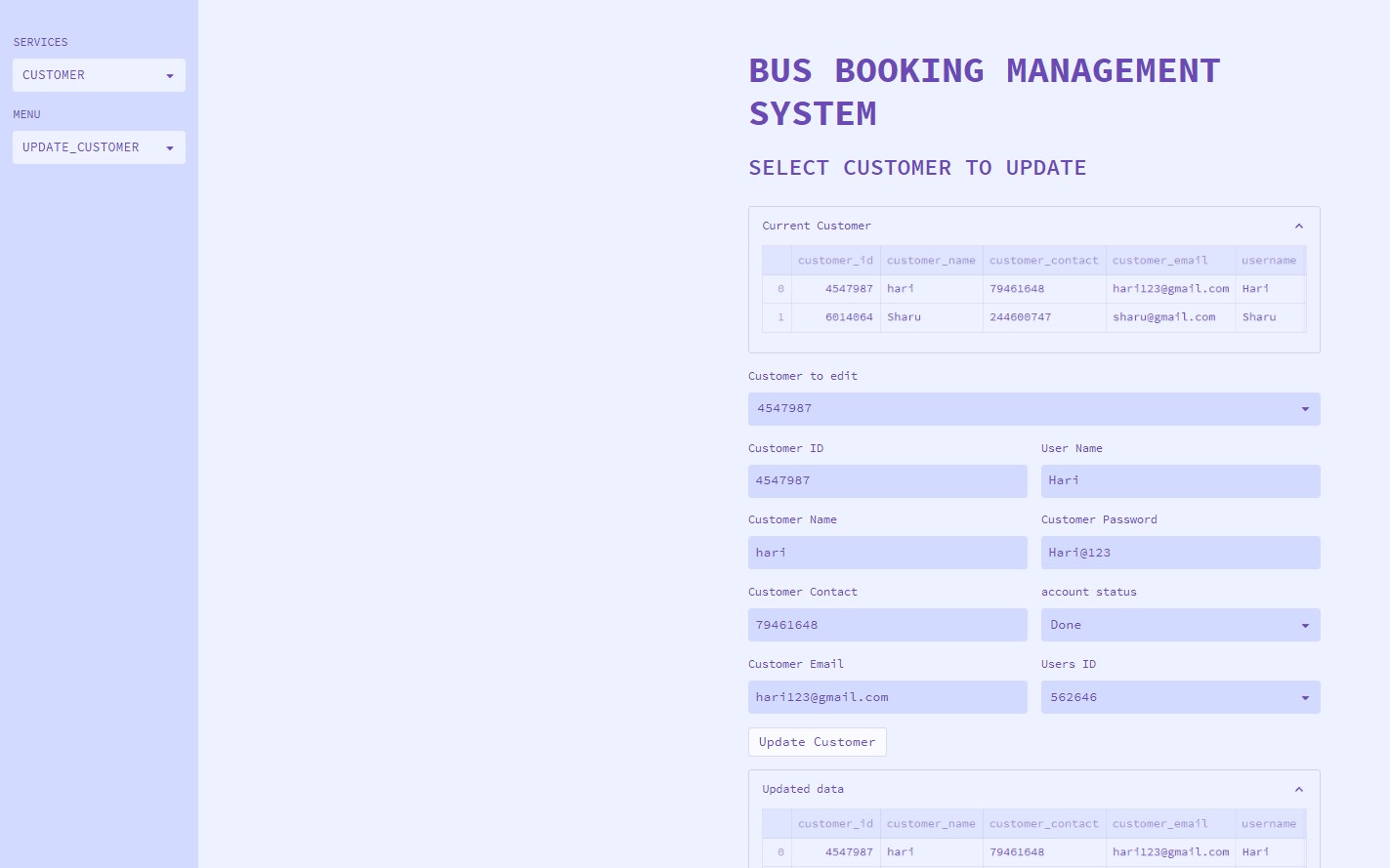
# Details of Bus Page



Driver Delete Page



# Updating Customer Page



Query Page

