CS 432: Databases

Assignment 3: Implementing a Web App using MySQL

Indian Institute of Technology Gandhinagar

Transport Management System

CS432 Assignment 3
GitHub: https://github.com/ayushmodi12/transport-management-system.git
4th March, 2024

Moon Macrosystems: Group Members

Mithil Pechimuthu, 21110129
Shreesh Agarwal, 21110198
Ayush Modi, 21110039
Vedant Kumbhar, 21110234
Anushk Bhana, 21110031

1. Problem Statement & Requirements

2. Tasks

2.1 Responsibility of G1:

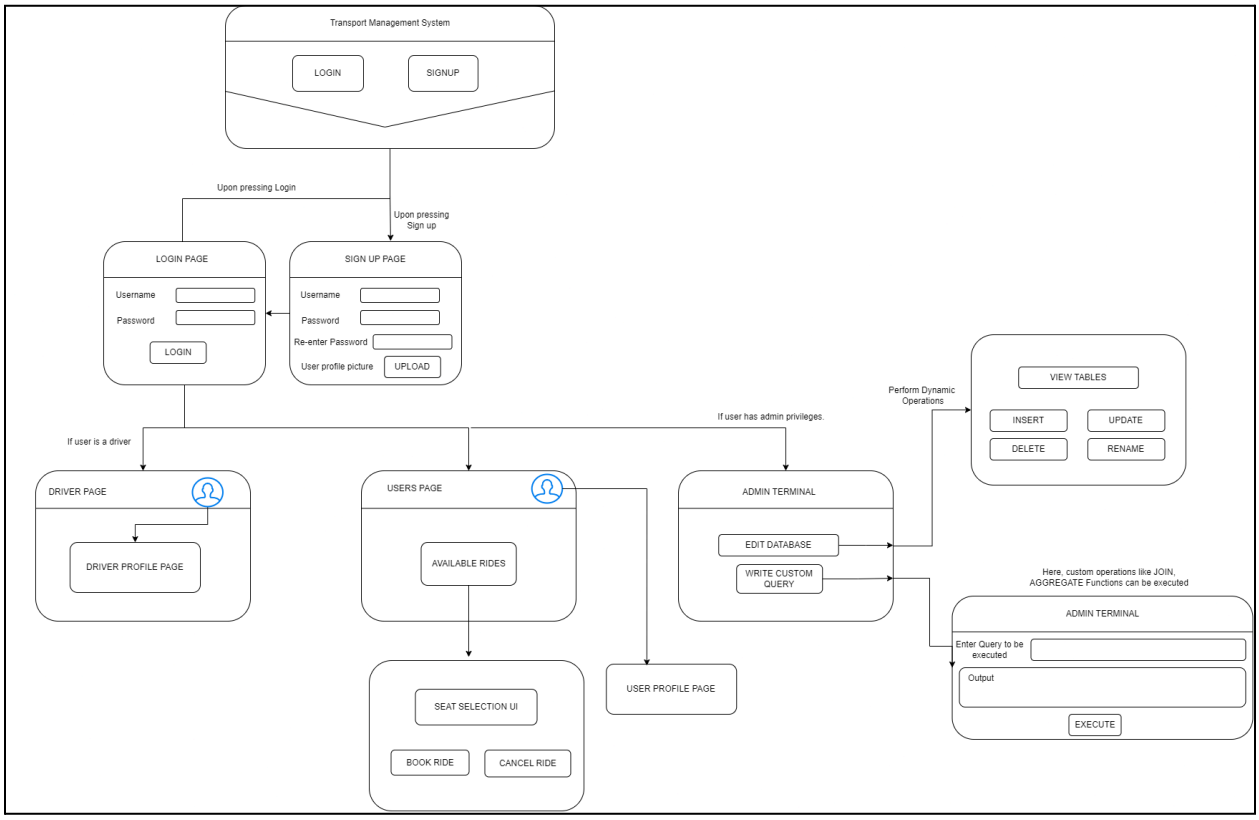
The code is submitted and uploaded to the repository as per the instructions.

2.2 Responsibility of G2:

The code is submitted and uploaded to the repository as per the instructions.

2.3 Responsibility of G1 & G2:

Design of the Web Application:



1:

The database of the designed logical schema is created along with inserting dummy values into it. The code for this is present in the repository. The transport_management_system.sql file present in the DatabaseFiles folder in the repository consists of the sql file for creating the database and inserting dummy values into it. The steps for setting up the database and running the website are provided in the following sections. All the tables of the database can be viewed on the website through the admin terminal.

2:

The web app supports the dynamic execution of all the operations including INSERT, UPDATE, DELETE, RENAME, and WHERE clause operations. We have shown the screenshots of the changes being reflected both on the main database and also on the web page, by displaying the Updated and Before Tables on the web page. The admin terminal is used to perform all these operations for viewing and modifying the database. Along with this, we have also provided support for the execution of custom queries like JOIN and AGGREGATE Functions in our web app.

3: (a)

Steps for running our web app:

Setting up the Server
First change the working directory to transport-management-system after cloning this repository.

```
git clone https://github.com/ayushmodi12/transport-management-system.git
```

Installing all the dependencies
Please run the following command with the cloned repository as the current working directory to install all the required modules and packages.

```
pip install -r requirements.txt
```

Create Database & Launch Web Server
Please run the setup.py to instantiate the database and launch the web server.
Make sure that you don't have the database before running setup.py. Else directly launch the web server by executing tms.py.

Setting up the Client:

Add DNS Entry

Go to the hosts file.
It can be typically found here C:\Windows\System32\drivers\etc\hosts.
Add the following translation 127.0.0.1 tms.iitgn.ac.in if the client and server are hosted on the same machine. Else add the public of the server machine. After that your hosts file should look like this.

```
hosts
# Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#      102.54.94.97     rhino.acme.com          # source server
#       38.25.63.10     x.acme.com              # x client host

# localhost name resolution is handled within DNS itself.
#   127.0.0.1       localhost
#   ::1             localhost

127.0.0.1   tms.iitgn.ac.in
```

Screenshots of the web pages:

After Running the setup.py file:

```
PS D:\mithil\IIT GN\Year_3\Semester 2\Databases\CS432_Assignments\Assignment_3\transport-management-system> & "C:/Program Files/Python Software/Python3_1
1_2/python.exe" "d:/mithil/IIT GN/Year_3/Semester 2/Databases/CS432_Assignments/Assignment_3/transport-management-system/setup.py"
<<<<<<<<<<<<<<<<<<<<<<< SETTING UP DATABASE >>>>>>>>>>>>>>>>>>>>>>>
Enter the database host: localhost
Enter the database user: root
Enter the database password (The password will be hidden as you type):
mysql: [Warning] Using a password on the command line interface can be insecure.
Database and tables created successfully.
<<<<<<<<<<<<<<<<<<<<<<< DATABASE SETUP COMPLETE >>>>>>>>>>>>>>>>>>>>>>>
<<<<<<<<<<<<<<<<<<<<<<< RUNNING FLASK APPLICATION >>>>>>>>>>>>>>>>>>>>>>>
D:\mithil\IIT GN\Year_3\Semester 2\Databases\CS432_Assignments\Assignment_3\transport-management-system\tms.py:2: DeprecationWarning: 'imghdr' is depreca
ted and slated for removal in Python 3.13
  import imghdr
 * Serving Flask app 'tms'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
D:\mithil\IIT GN\Year_3\Semester 2\Databases\CS432_Assignments\Assignment_3\transport-management-system\tms.py:2: DeprecationWarning: 'imghdr' is depreca
ted and slated for removal in Python 3.13
  import imghdr
 * Debugger is active!
 * Debugger PIN: 628-036-309
```

Following is the overview of our web app, and a guide on how to access the different aspects of the web app:

1. **Admin Login**:

With the admin login one can access the admin terminal. The admin terminal can be used to execute the various dynamic operations also with custom queries directly from the input space given on the terminal. Below are the credentials for one of the admin users:

Username - kumaramit@iitgn.ac.in
Password - Amit@123

After logging in we will be redirected to the admin portal, where we will be provided with two choices, namely, edit database and write custom query. One can choose whichever and may proceed with his operations (The images for the admin portal and its working are mentioned below).

2. **User Login**:

When normal users log in, they see the main page of the Transport Management System. Here, they have two choices: booking a bus or checking their user profile. There's also a small logo at the top right that takes them to their profile that they can later edit.

If they choose to book a bus, they're taken to a page where they can pick the date, time, and route for their journey. After they're done, they click 'Book'. Then, they move to a page where they can choose their seat from the available ones or cancel a previous selection.

Once they've picked their seat and confirmed everything, they'll be sent an email to their registered mail-id confirming their booking. It lets them know their travel plans are set. Similar mails are also sent for booking cancellations.

To make a user login, first do a signup which will redirect you to the login page. Then login using the details you just signed up with. You will be logged in as a user.

3.**Driver Login**:

Whenever we login using the credentials of the driver, they are directly directed to the page where they are shown all their details which are not editable.

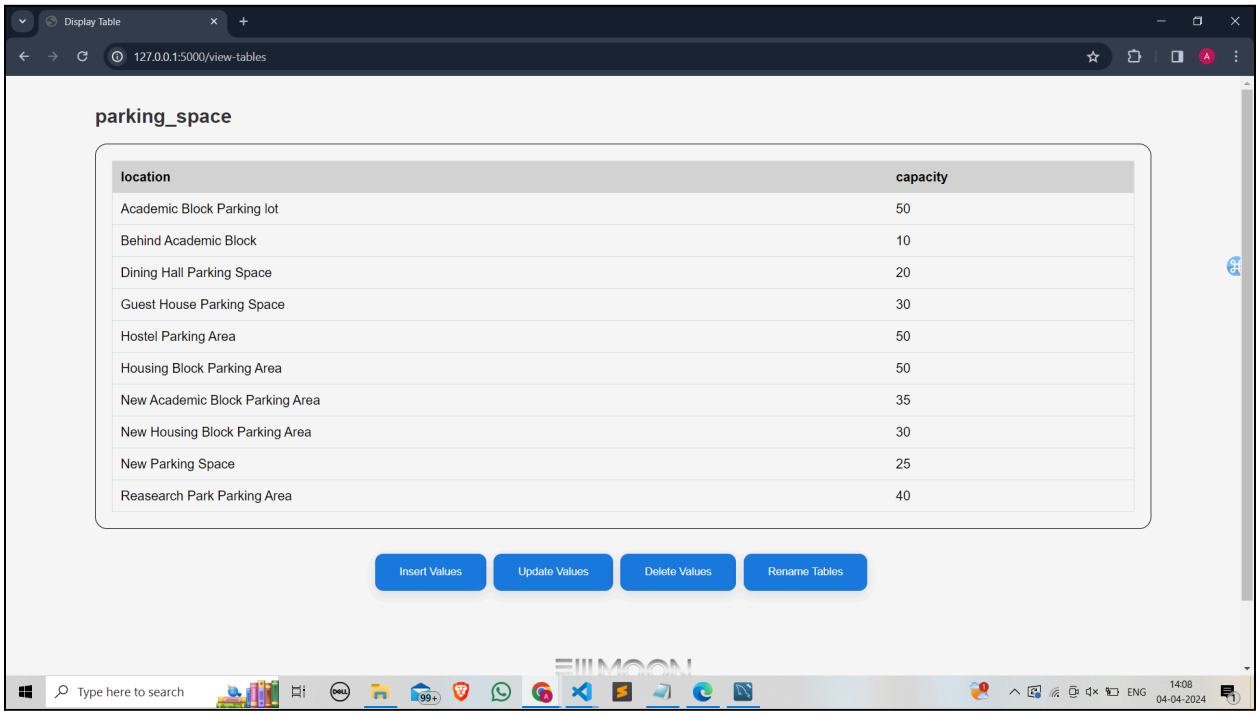
Below are the credentials for one of the driver login:

Username - john_doe@iitgn.ac.in
Password - John@123

(b):

<u>Screenshots of successful execution of the dynamic operations:</u>

Viewing the table parking_space:



location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Dining Hall Parking Space	20
Guest House Parking Space	30
Hostel Parking Area	50
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking Space	25
Reasearch Park Parking Area	40

1. **Performing the INSERT Operation:**
   Before inserting, viewing the table on workbench:

```
use transportmanagement;

select * from parking_space
```

Result Grid | Filter Rows:

location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Dining Hall Parking Space	20
Guest House Parking Space	30
Hostel Parking Area	50
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking Space	25
Reasearch Park Parking Area	40
NULL	NULL

Now, inserting a new value in the table:

Insert Form

127.0.0.1:5000/insert-values

Insert Values into parking_space

location: location

capacity: capacity

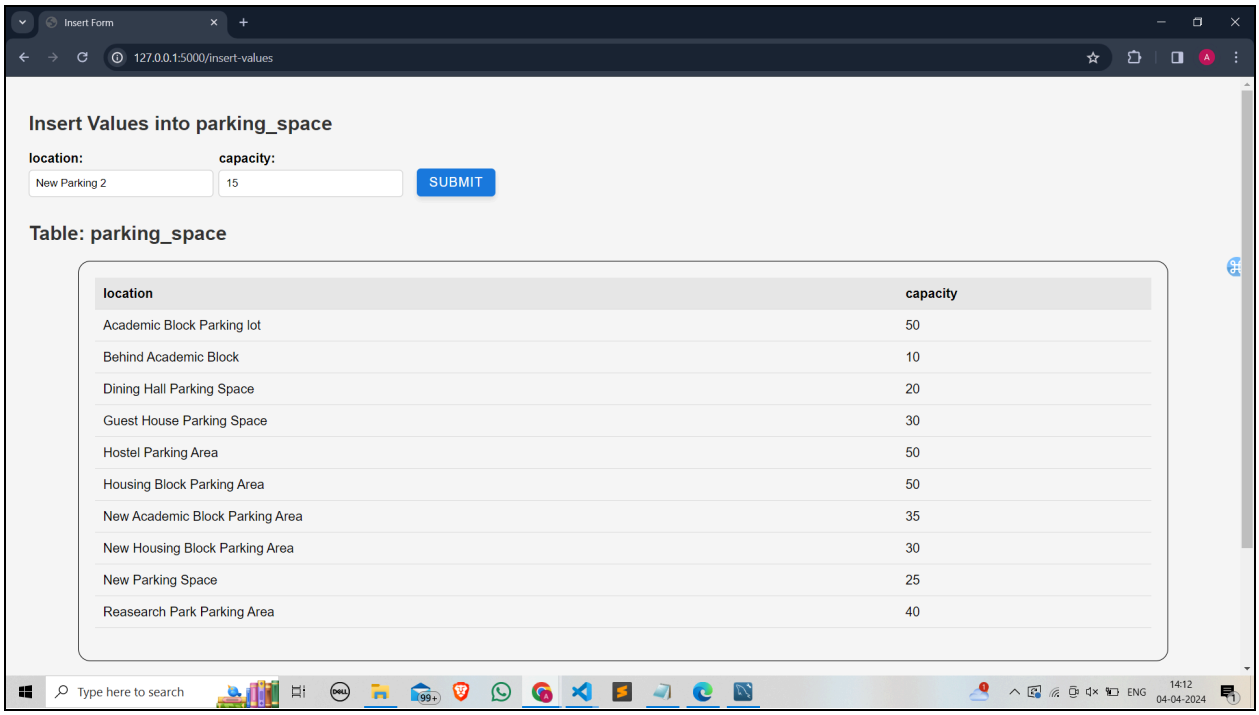
SUBMIT

Table: parking_space

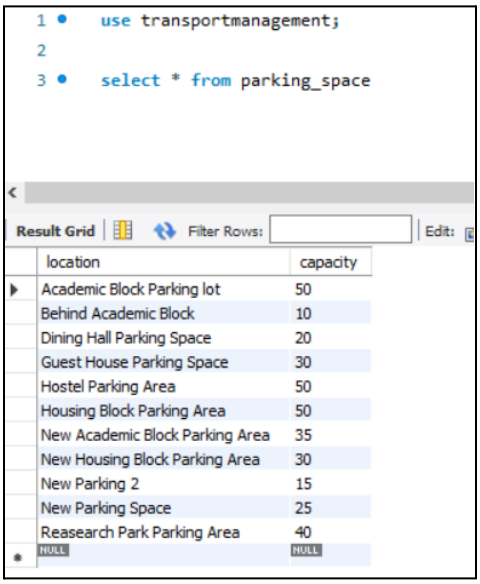
location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Dining Hall Parking Space	20
Guest House Parking Space	30
Hostel Parking Area	50
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking Space	25
Reasearch Park Parking Area	40

Type here to search

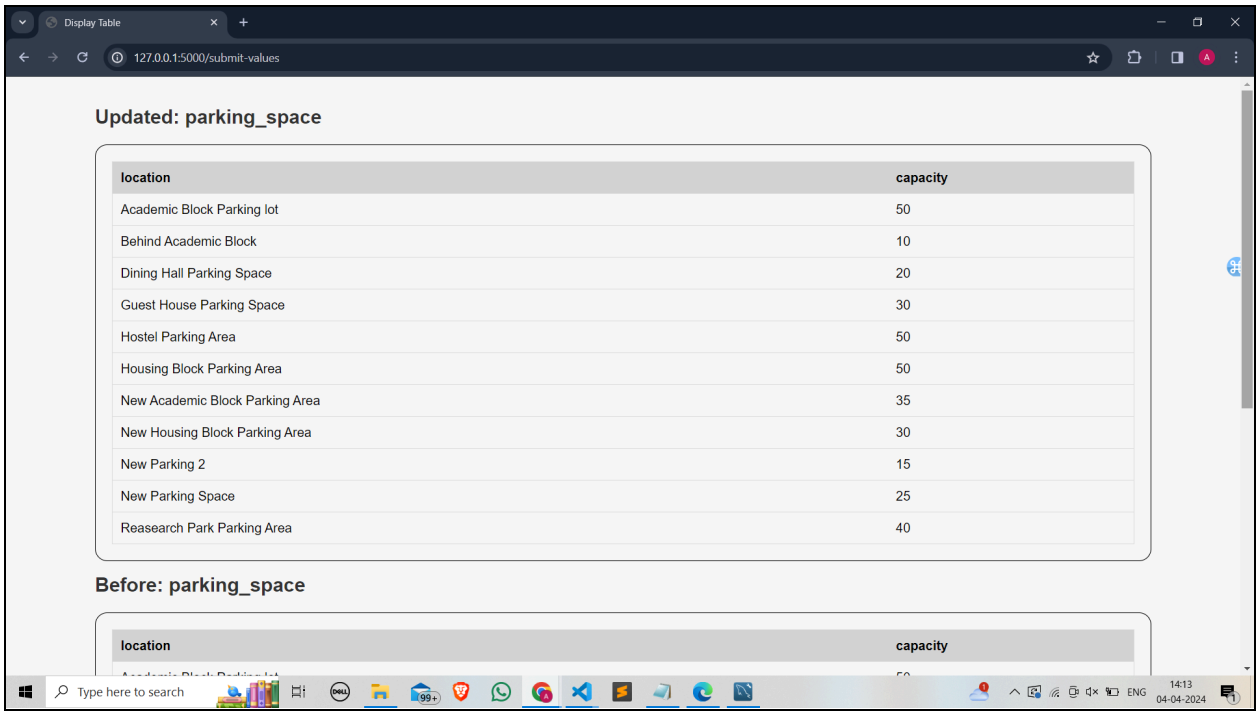
ENG 14:11 04-04-2024



New value inserted can be seen on workbench:



Showing the Updated and before state of the table on the website:

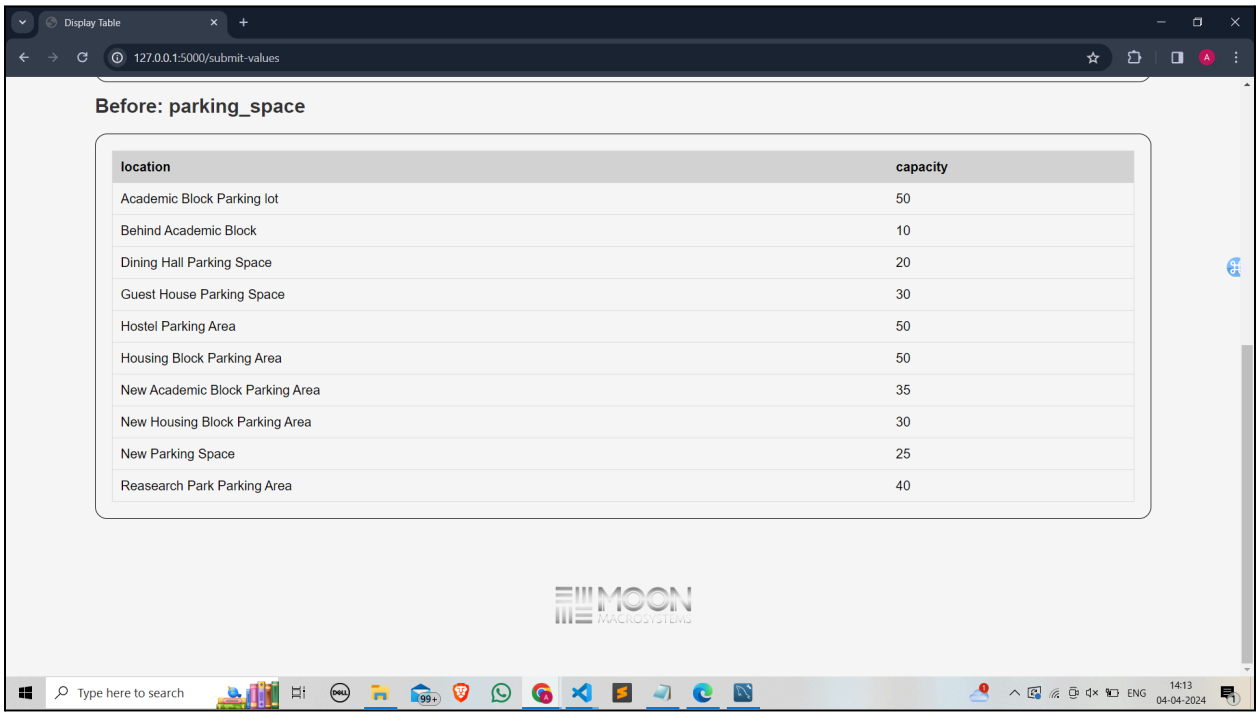


Updated: parking_space

location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Dining Hall Parking Space	20
Guest House Parking Space	30
Hostel Parking Area	50
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking 2	15
New Parking Space	25
Reasearch Park Parking Area	40

Before: parking_space

location	capacity



Before: parking_space

location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Dining Hall Parking Space	20
Guest House Parking Space	30
Hostel Parking Area	50
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking Space	25
Reasearch Park Parking Area	40

2. **Performing the UPDATE Operation:**
   Before updating, viewing the table on workbench:

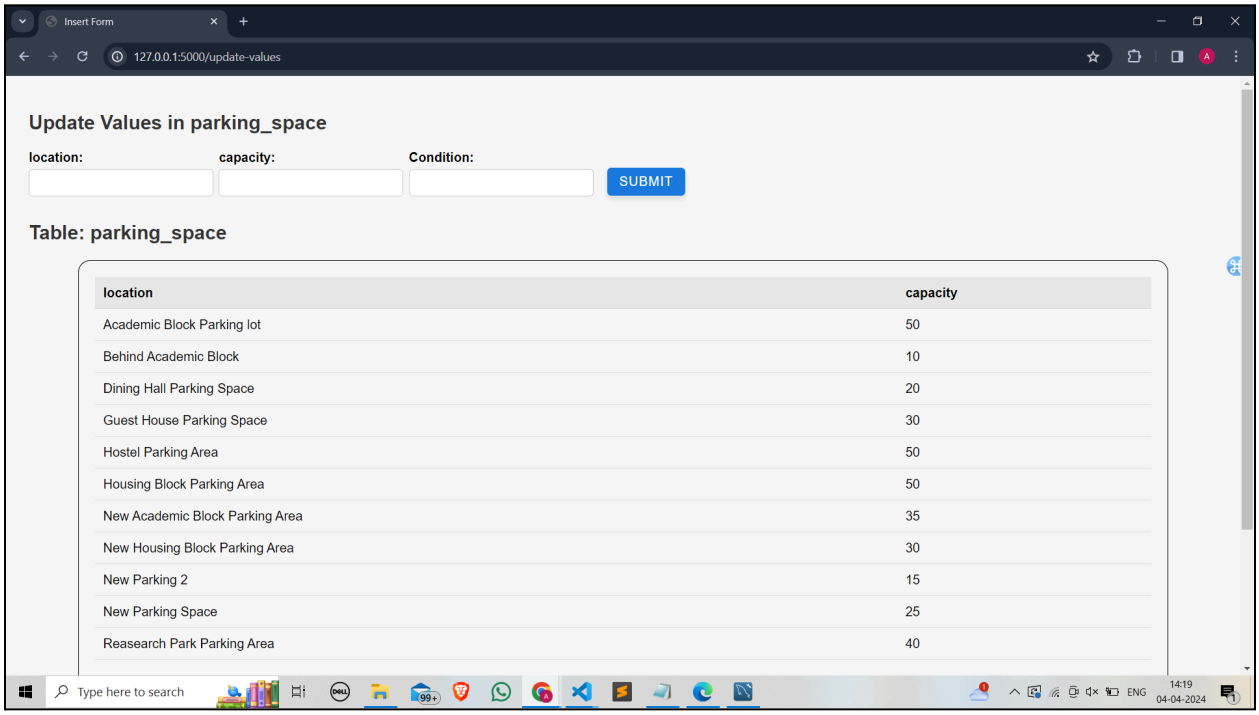
```
use transportmanagement;
select * from parking_space
```

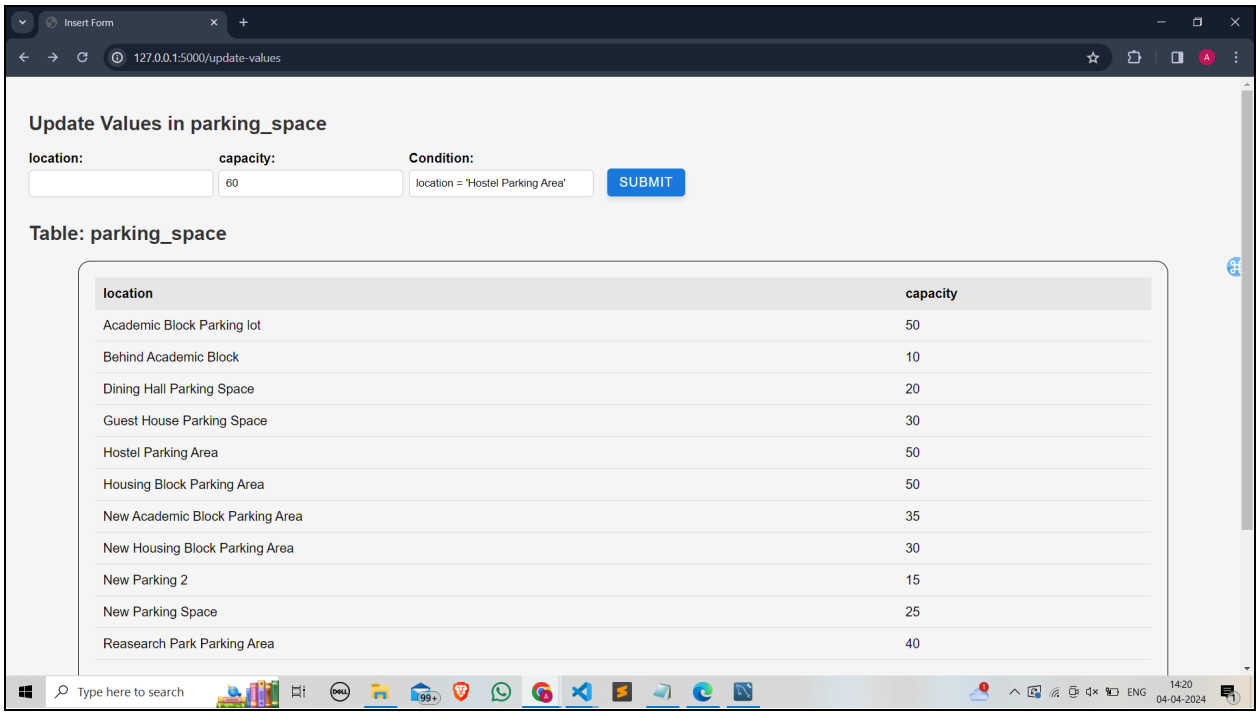
Result Grid | Filter Rows:

location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Dining Hall Parking Space	20
Guest House Parking Space	30
Hostel Parking Area	50
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking 2	15
New Parking Space	25
Reasearch Park Parking Area	40
NULL	NULL

Now, updating the capacity to 60 in the row where location="Hostel Parking Area":

NOTE: Only those input fields are to be filled for which we want to update the values, if the field is left empty, the value for that column will not be updated.





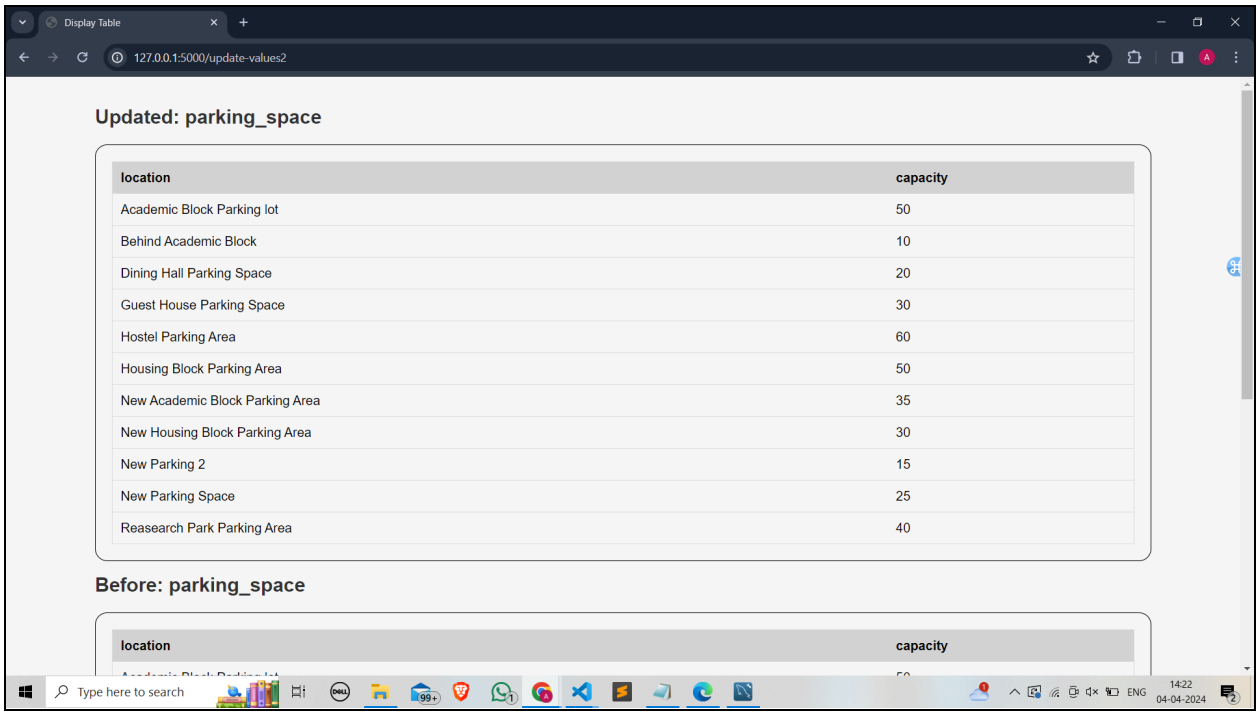
This updated row can be seen in the table on workbench:

```
use transportmanagement;
select * from parking_space
```

Result Grid Filter Rows:

location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Dining Hall Parking Space	20
Guest House Parking Space	30
Hostel Parking Area	60
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking 2	15
New Parking Space	25
Reasearch Park Parking Area	40
NULL	NULL

Showing the Updated and before states of the table on the website:

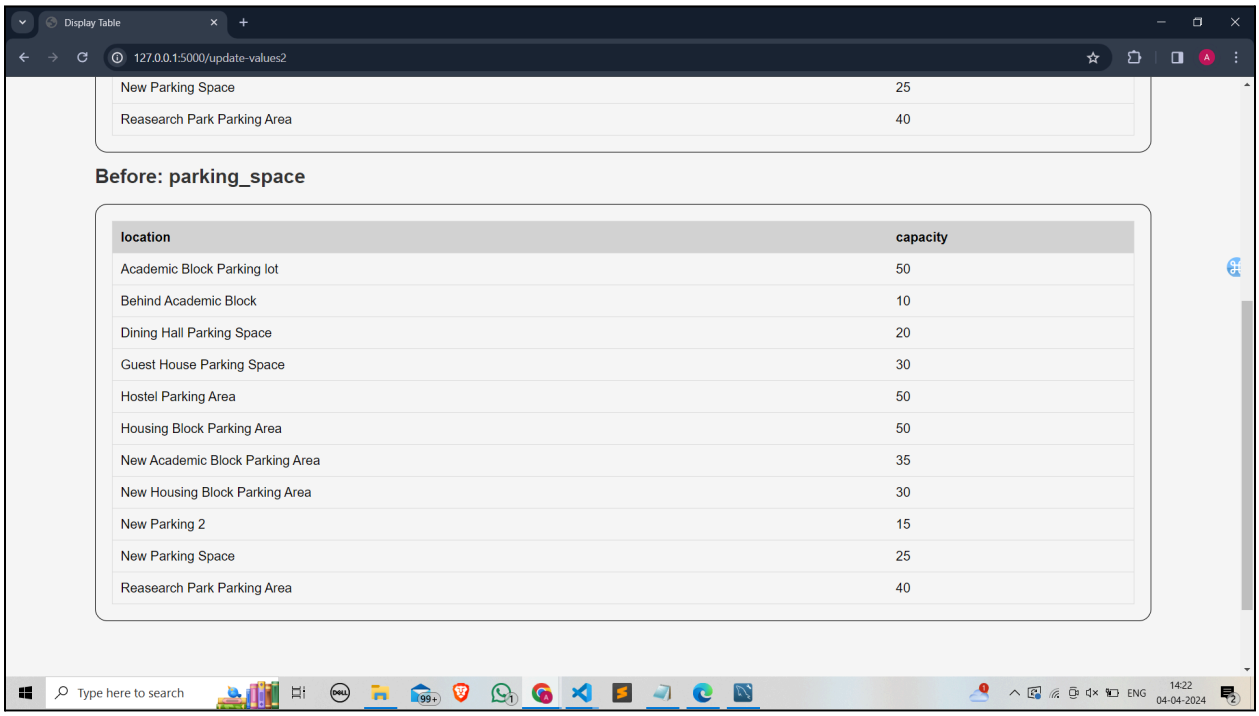


Updated: parking_space

location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Dining Hall Parking Space	20
Guest House Parking Space	30
Hostel Parking Area	60
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking 2	15
New Parking Space	25
Reasearch Park Parking Area	40

Before: parking_space

location	capacity



New Parking Space	25
Reasearch Park Parking Area	40

Before: parking_space

location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Dining Hall Parking Space	20
Guest House Parking Space	30
Hostel Parking Area	50
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking 2	15
New Parking Space	25
Reasearch Park Parking Area	40

3. Performing the DELETE Operation:

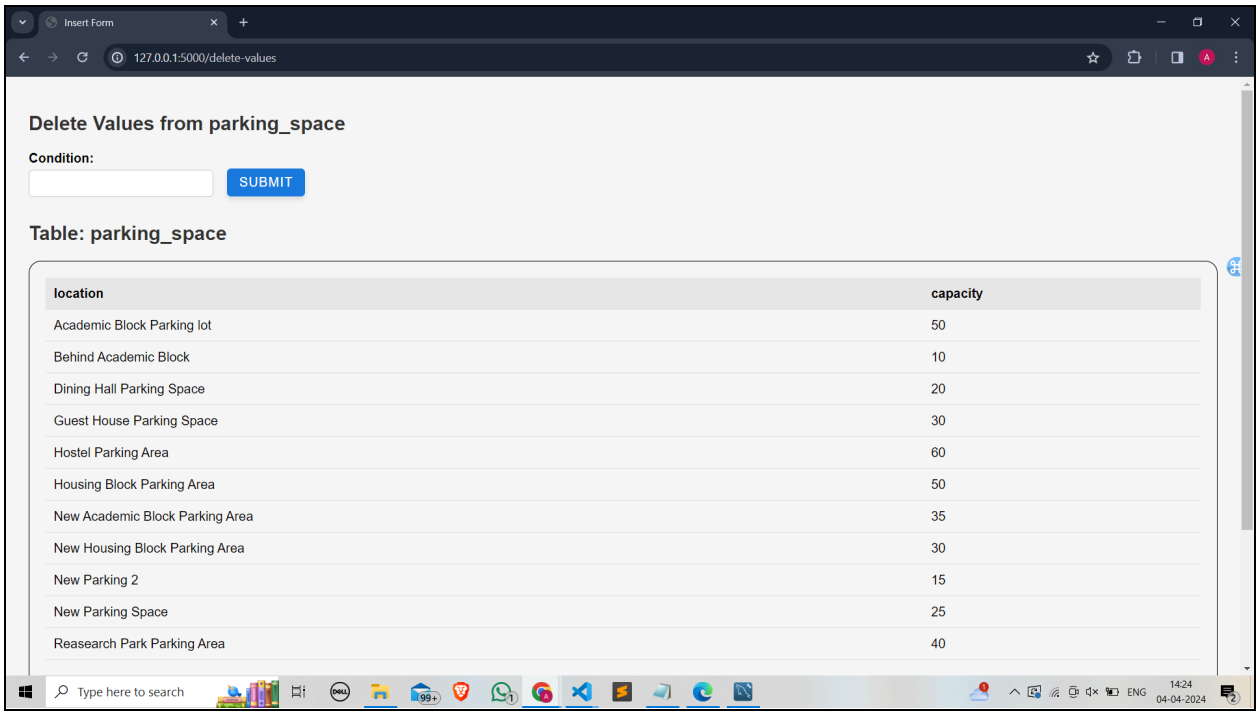
Before Deleting, viewing the table on workbench:

```
use transportmanagement;
select * from parking_space
```

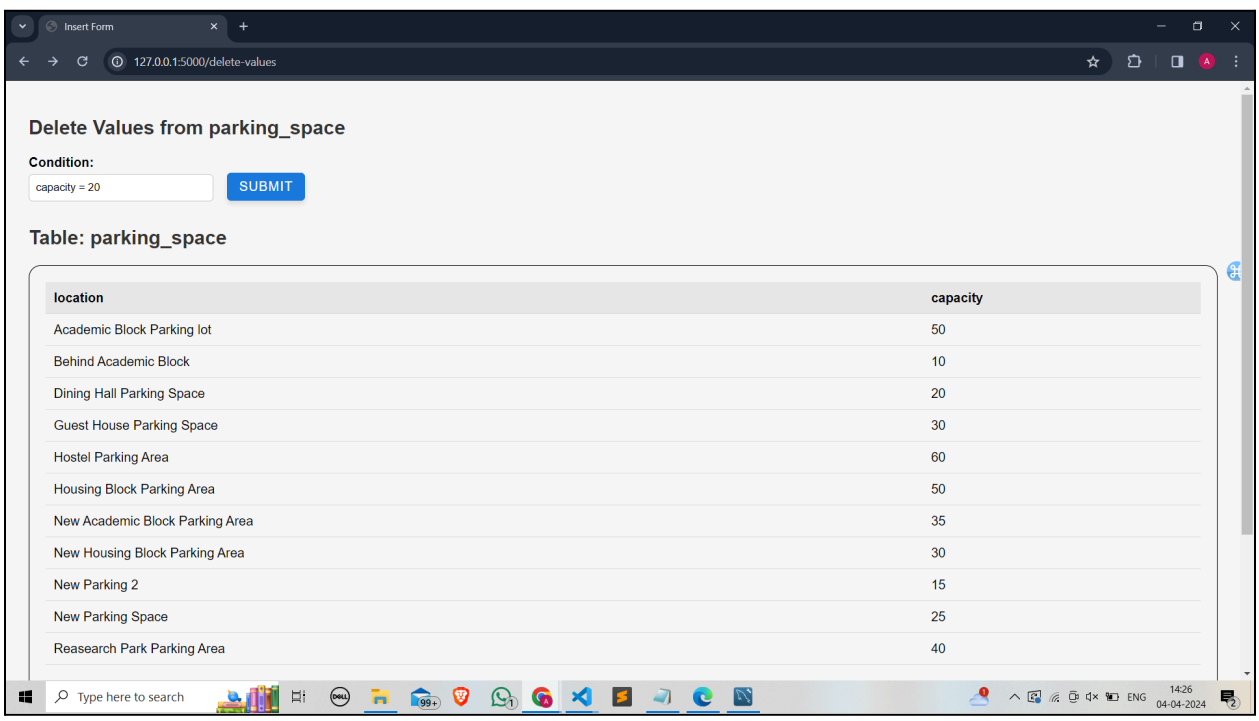
Result Grid | Filter Rows: | Edit:

location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Dining Hall Parking Space	20
Guest House Parking Space	30
Hostel Parking Area	60
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking 2	15
New Parking Space	25
Reasearch Park Parking Area	40
NULL	NULL

Now, deleting the row where capacity = 20:



location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Dining Hall Parking Space	20
Guest House Parking Space	30
Hostel Parking Area	60
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking 2	15
New Parking Space	25
Reasearch Park Parking Area	40



location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Dining Hall Parking Space	20
Guest House Parking Space	30
Hostel Parking Area	60
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking 2	15
New Parking Space	25
Reasearch Park Parking Area	40

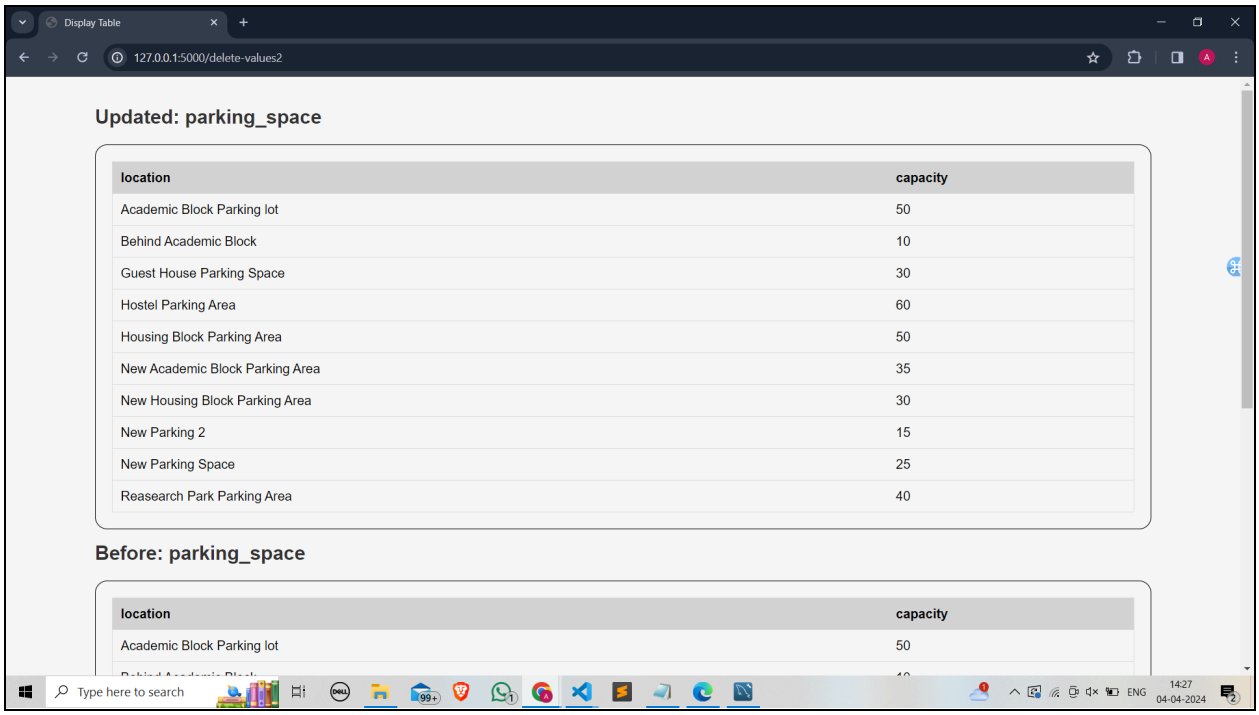
We can see on workbench, that the row having capacity = 20 is deleted:

```
use transportmanagement;
select * from parking_space
```

Result Grid | Filter Rows: | Edi

location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Guest House Parking Space	30
Hostel Parking Area	60
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking 2	15
New Parking Space	25
Reasearch Park Parking Area	40
NULL	NULL

Showing the Updated and before states of the table on website:

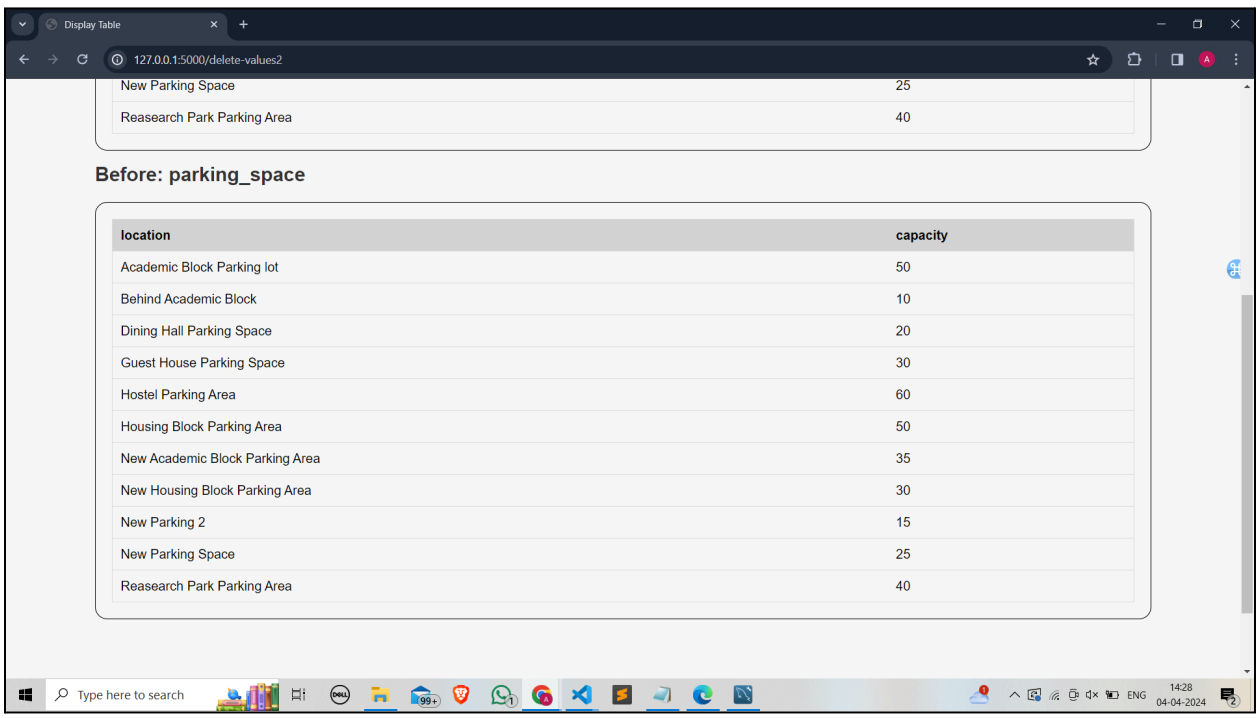


Updated: parking_space

location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Guest House Parking Space	30
Hostel Parking Area	60
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking 2	15
New Parking Space	25
Reasearch Park Parking Area	40

Before: parking_space

location	capacity
Academic Block Parking lot	50



New Parking Space	25
Reasearch Park Parking Area	40

Before: parking_space

location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Dining Hall Parking Space	20
Guest House Parking Space	30
Hostel Parking Area	60
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking 2	15
New Parking Space	25
Reasearch Park Parking Area	40

4. Performing the RENAME Operation:

Before renaming, viewing the current table on workbench:

```
use transportmanagement;
select * from parking_space
```

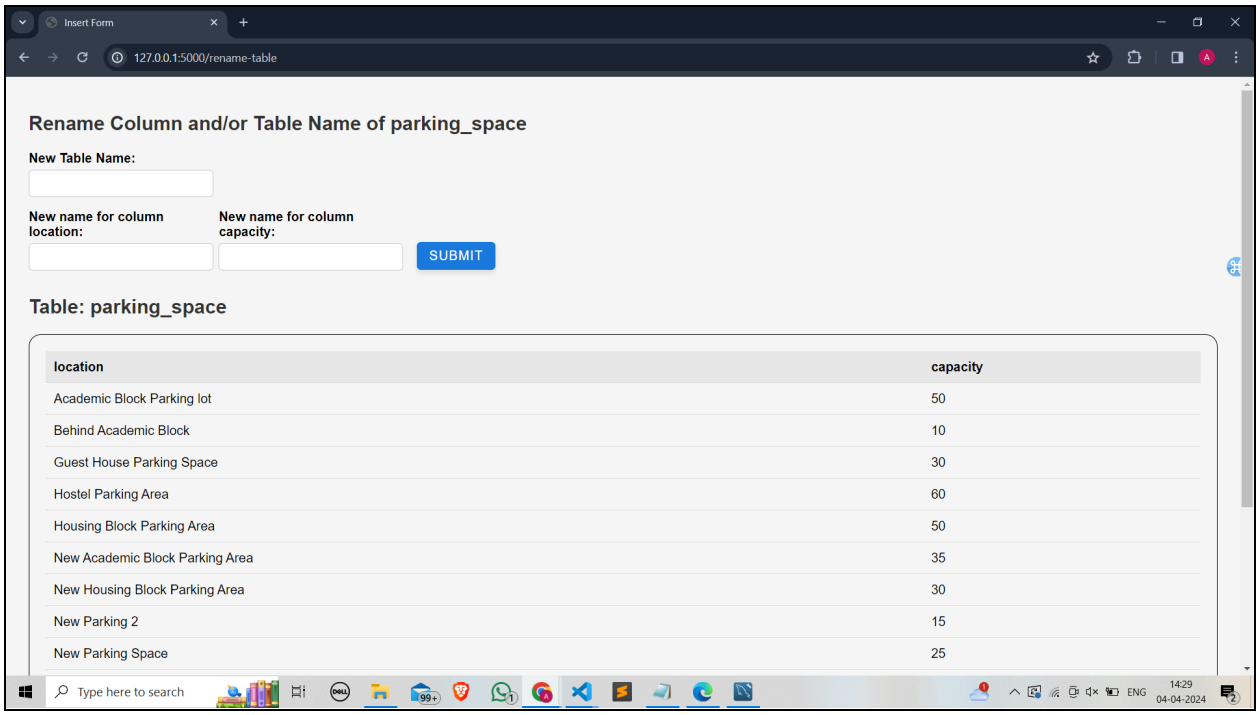
Result Grid | Filter Rows:

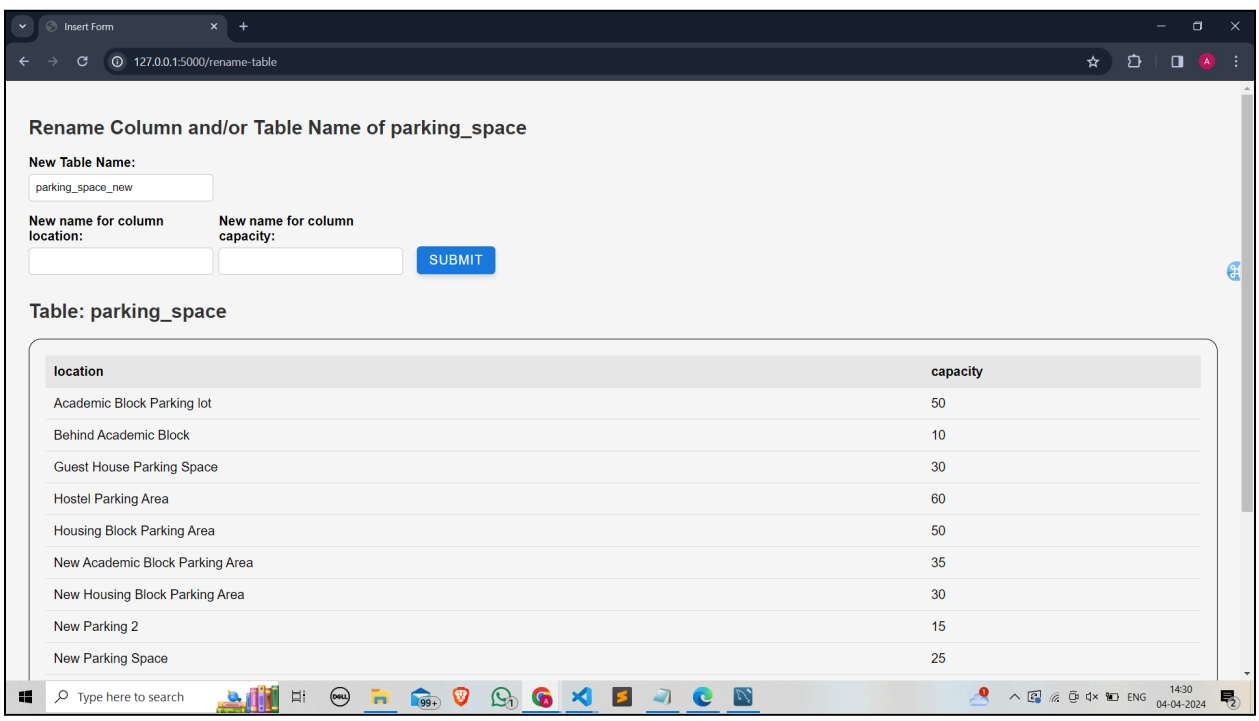
location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Guest House Parking Space	30
Hostel Parking Area	60
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking 2	15
New Parking Space	25
Reasearch Park Parking Area	40
NULL	NULL

S

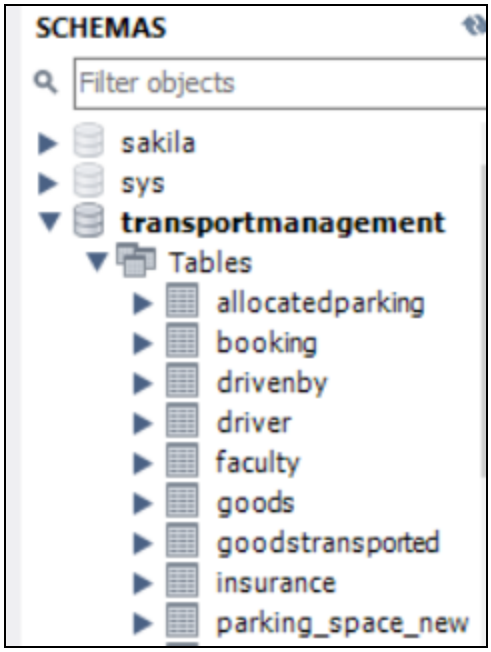
Now, updating the table name to parking_space_new:

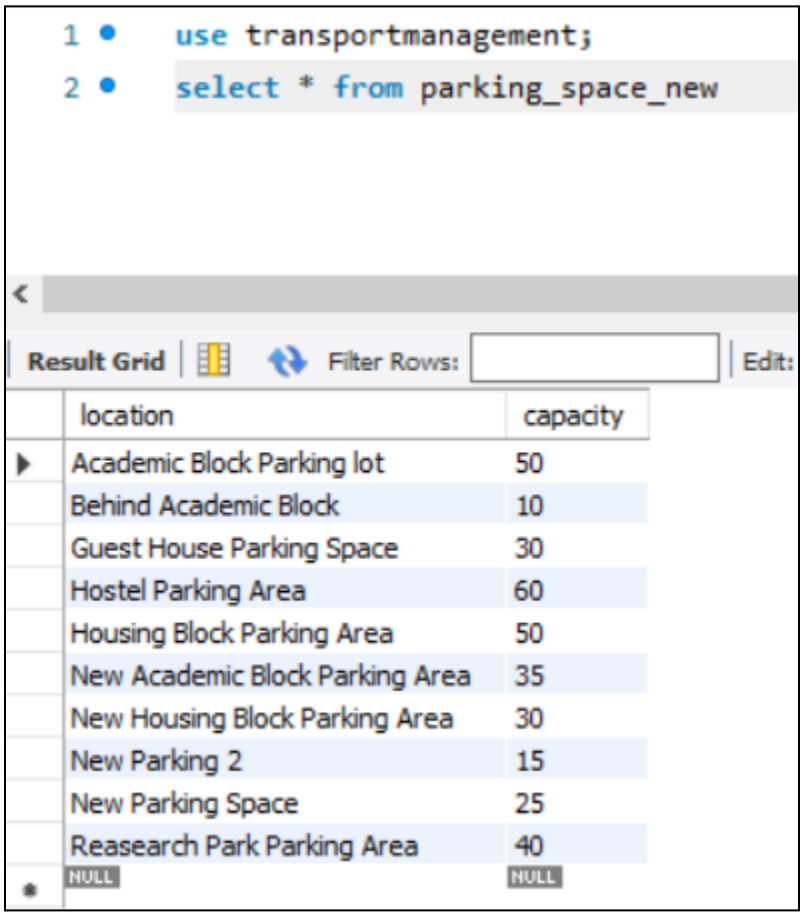
NOTE: Only those input fields are to be filled for which we want to rename, for rest others, leave the input field empty:





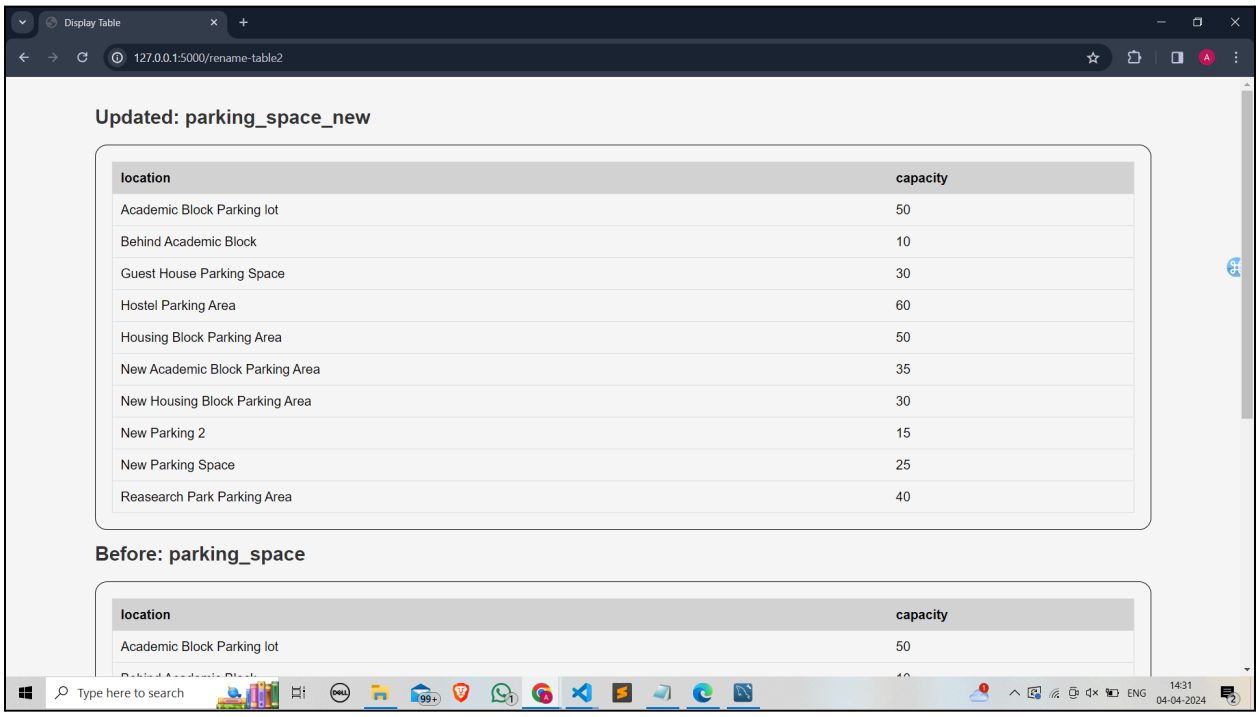
As we can see on workbench, the table name has been updated to parking_space_new:

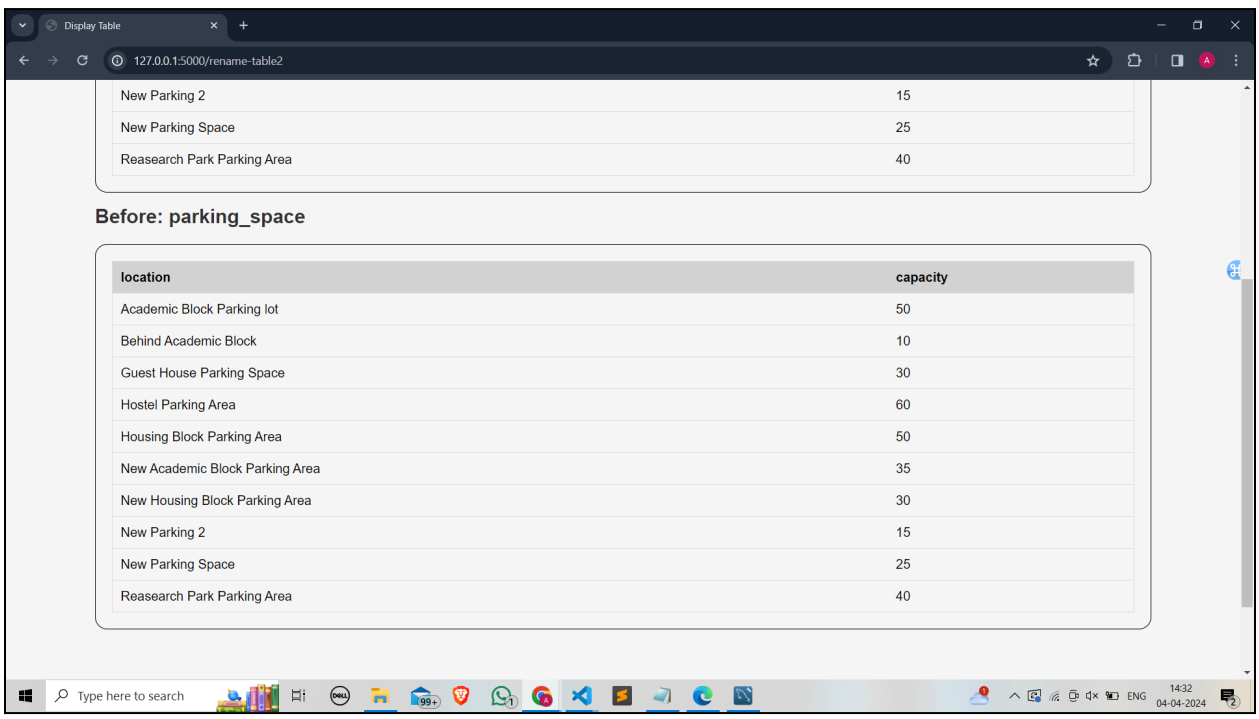




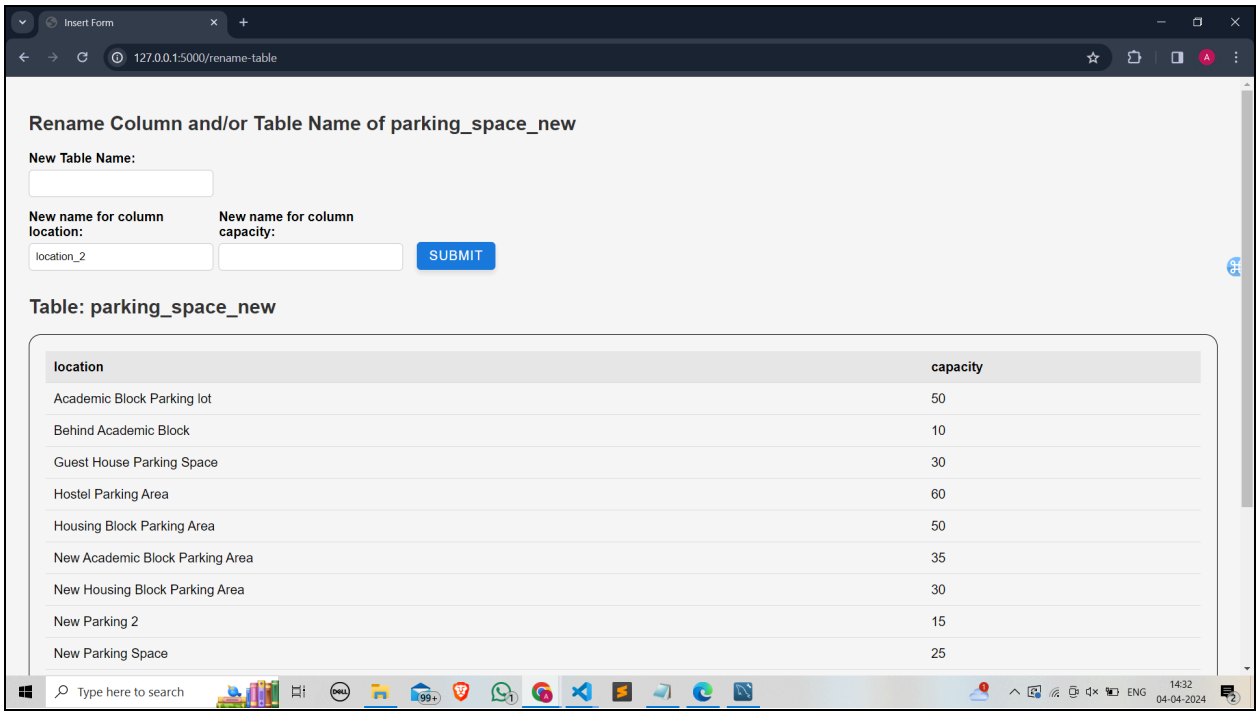
location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Guest House Parking Space	30
Hostel Parking Area	60
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking 2	15
New Parking Space	25
Reasearch Park Parking Area	40
NULL	NULL

Showing the Updated and before states of the table on website:





Now, renaming the column location to location_2:



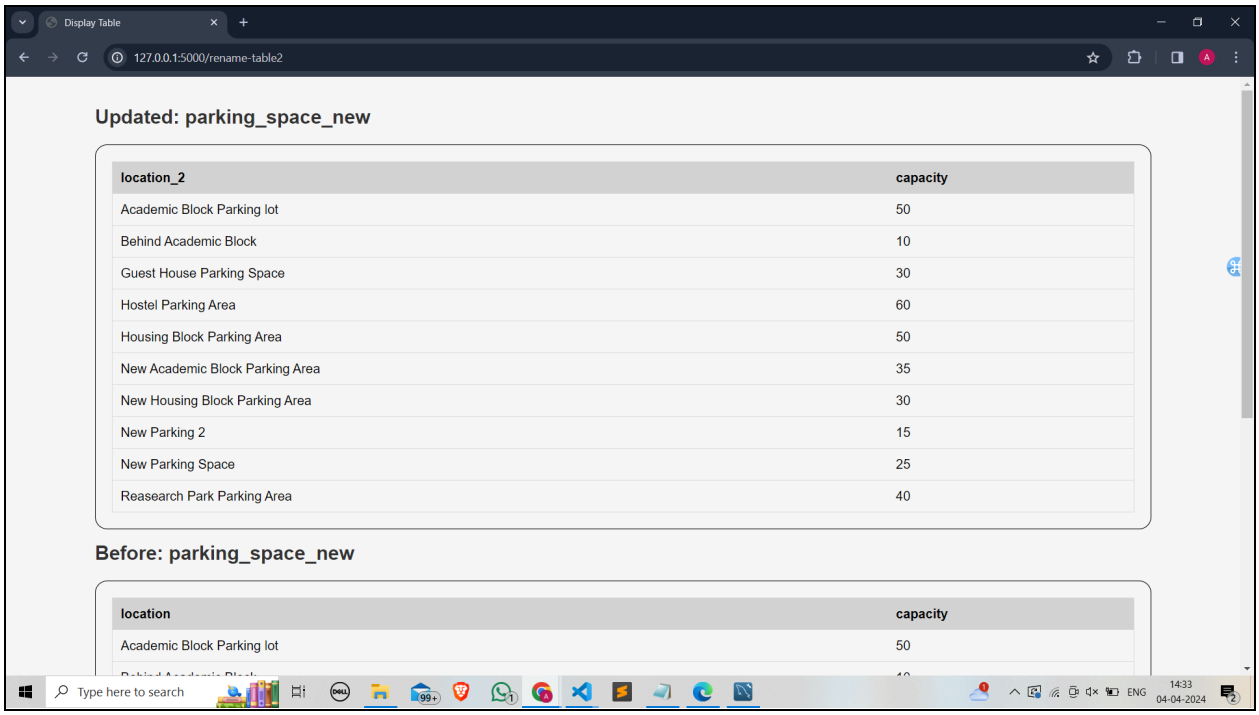
We can see in the workbench, the table name of location has been changed:

```
use transportmanagement;
select * from parking_space_new
```

Result Grid | Filter Rows: | Edit:

location_2	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Guest House Parking Space	30
Hostel Parking Area	60
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking 2	15
New Parking Space	25
Reasearch Park Parking Area	40
NULL	NULL

Showing the Updated and before states of the table on the website:

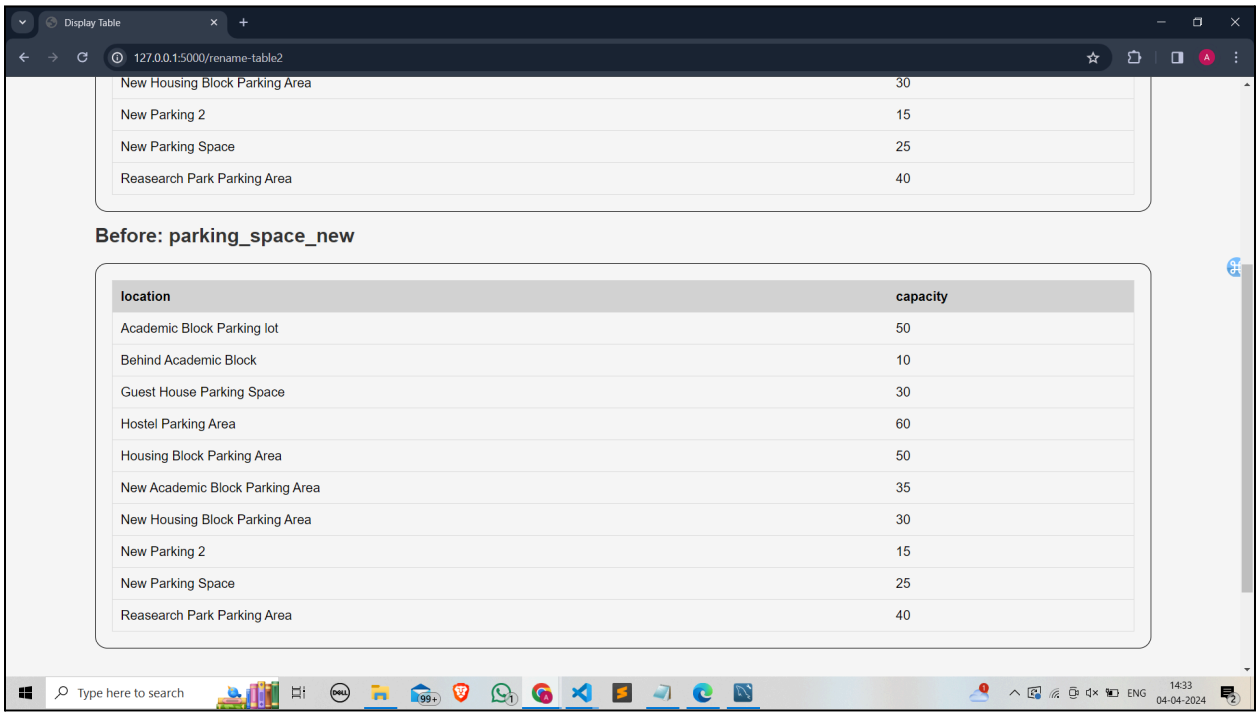


Updated: parking_space_new

location_2	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Guest House Parking Space	30
Hostel Parking Area	60
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking 2	15
New Parking Space	25
Reasearch Park Parking Area	40

Before: parking_space_new

location	capacity
Academic Block Parking lot	50



New Housing Block Parking Area	30
New Parking 2	15
New Parking Space	25
Reasearch Park Parking Area	40

Before: parking_space_new

location	capacity
Academic Block Parking lot	50
Behind Academic Block	10
Guest House Parking Space	30
Hostel Parking Area	60
Housing Block Parking Area	50
New Academic Block Parking Area	35
New Housing Block Parking Area	30
New Parking 2	15
New Parking Space	25
Reasearch Park Parking Area	40

5. Performing Operations using WHERE Clause:

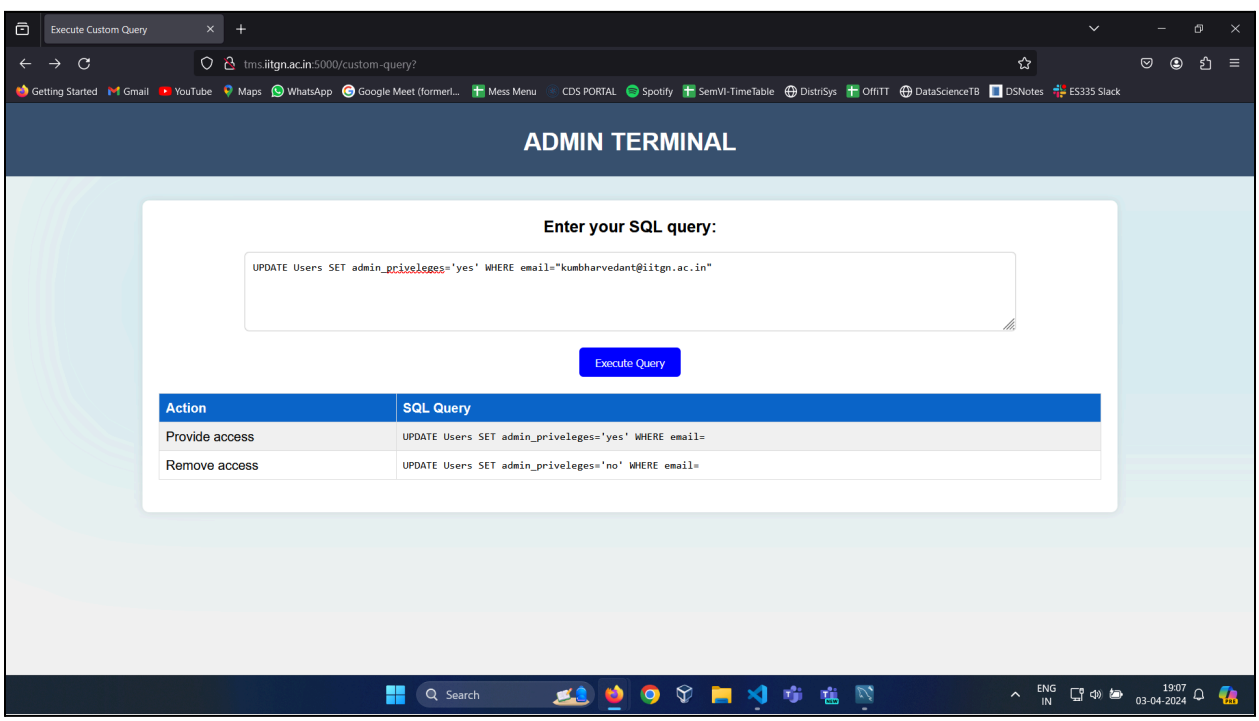
Now, the WHERE clause has already been used in the above operations where we have performed UPDATE and DELETE.
The input field “Condition” in UPDATE and DELETE Operations takes the WHERE Clause as input and accordingly performs the operations. The screenshots for the operations have thus already been shown in the above sections. Below is the snippet of the code for the /update-values api, that creates the sql query for UPDATE operation using WHERE Clause:

```
values = {key: request.form[key] for key in request.form if (key != 'tableName' and key != 'whereCondition' and request.form[key]!='')}

where_condition = ''

if (request.form['whereCondition']!=''):
    where_condition = request.form['whereCondition']

set_clause = ", ".join([f"{col} = %s" for col in values.keys()])

sql = ''

if (where_condition!=''):
    sql = f"UPDATE {table_name} SET {set_clause} WHERE {where_condition}"
else:
    sql = f"UPDATE {table_name} SET {set_clause}"
```

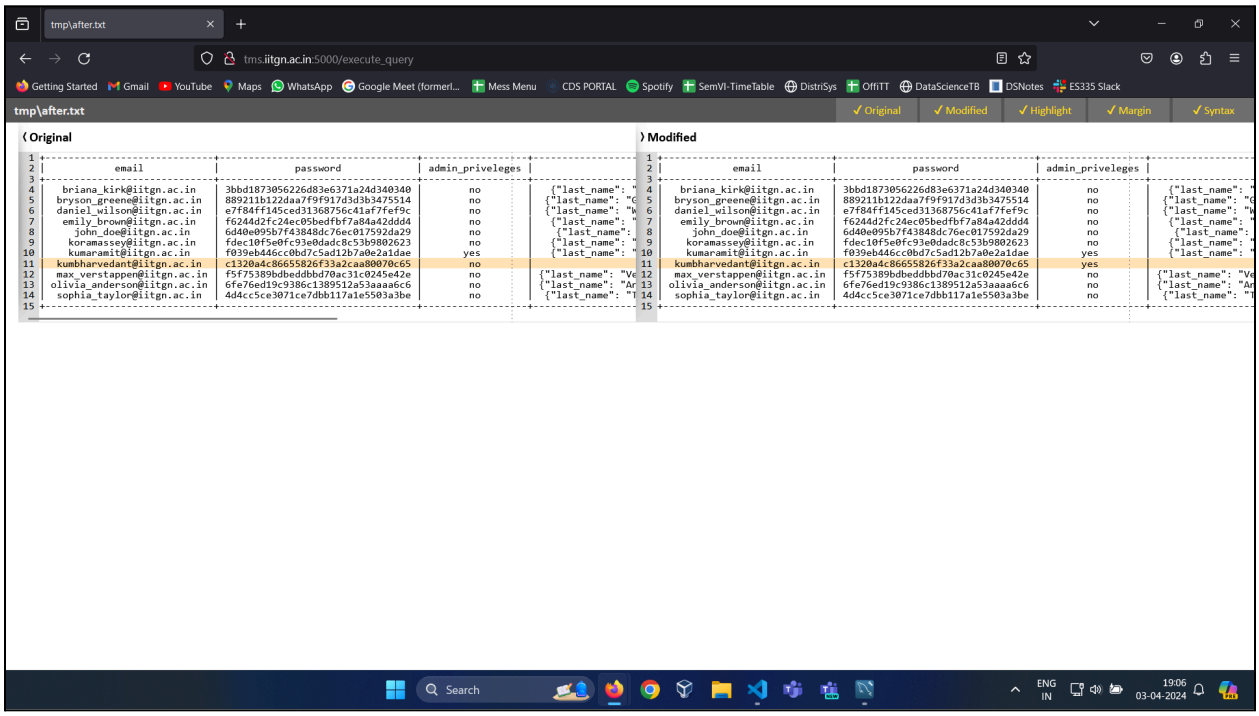
As we have already added screenshots above in UPDATE and DELETE, which also contain the WHERE Clause, we are not adding screenshots here.

Below are the screenshots for execution of custom queries:

Making a certain user as admin:

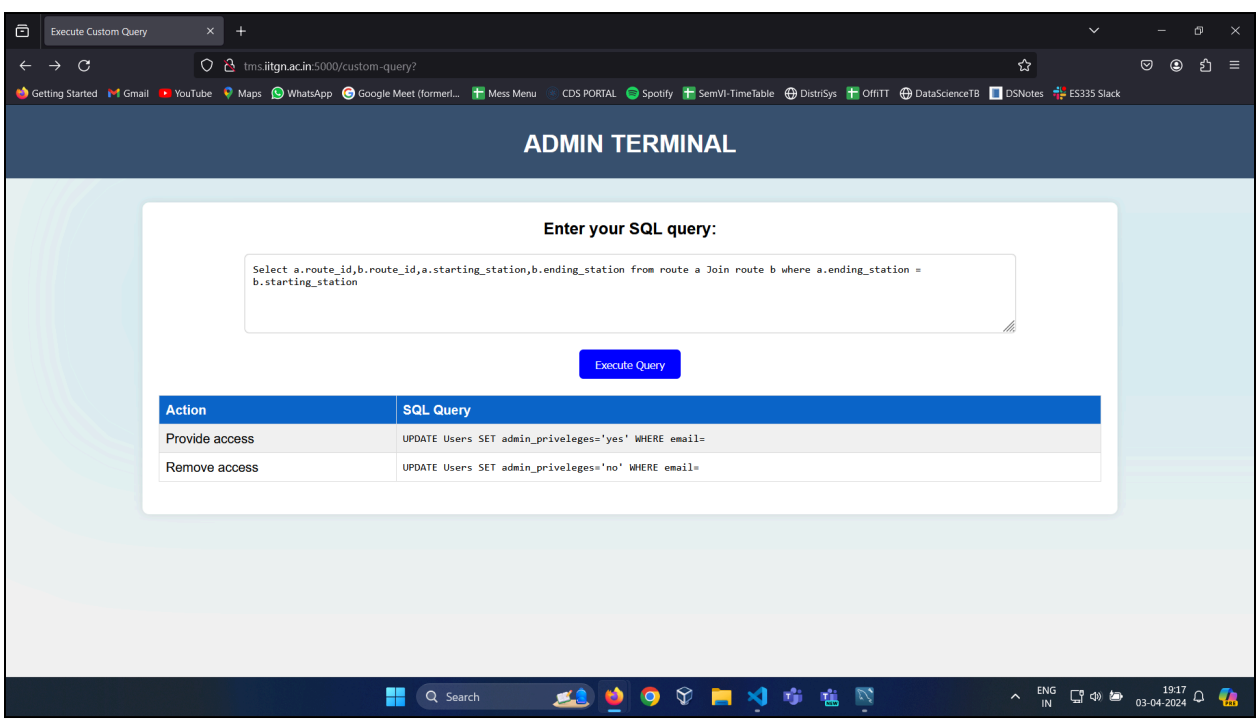


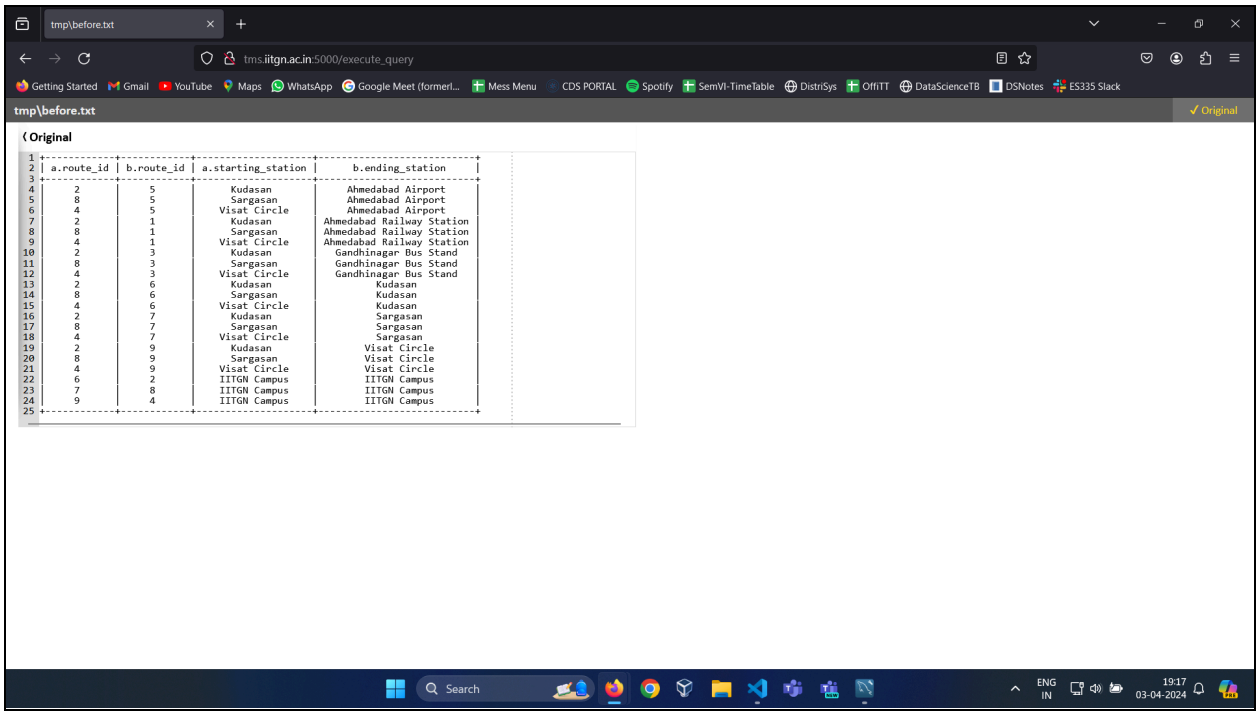
Output:



Custom Queries:

- **Use of WHERE clause:** Some custom query using JOIN operation which may be used for a two-way trip.



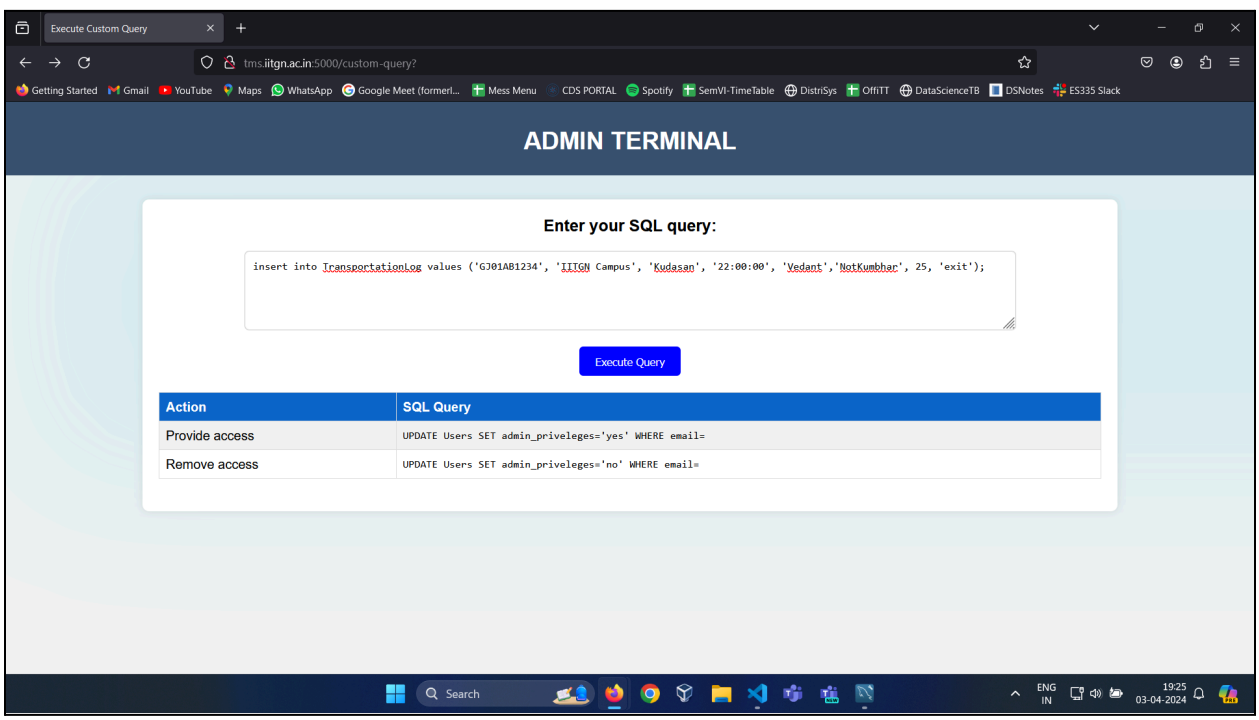


tmp\before.txt

Original

```
+------------+------------+------------------+-------------------------+
| a.route_id | b.route_id | a.starting_station |      b.ending_station     |
+------------+------------+------------------+-------------------------+
|     2      |     5      |     Kudasan      |     Ahmedabad Airport     |
|     8      |     5      |     Sargasan     |     Ahmedabad Airport     |
|     4      |     5      |   Visat Circle   |     Ahmedabad Airport     |
|     2      |     1      |     Kudasan      | Ahmedabad Railway Station |
|     8      |     1      |     Sargasan     | Ahmedabad Railway Station |
|     4      |     1      |   Visat Circle   | Ahmedabad Railway Station |
|     2      |     3      |     Kudasan      |  Gandhinagar Bus Stand    |
|     8      |     3      |     Sargasan     |  Gandhinagar Bus Stand    |
|     4      |     3      |   Visat Circle   |  Gandhinagar Bus Stand    |
|     2      |     6      |     Kudasan      |          Kudasan          |
|     8      |     6      |     Sargasan     |          Kudasan          |
|     4      |     6      |   Visat Circle   |          Kudasan          |
|     2      |     7      |     Kudasan      |          Sargasan         |
|     8      |     7      |     Sargasan     |          Sargasan         |
|     4      |     7      |   Visat Circle   |          Sargasan         |
|     2      |     9      |     Kudasan      |        Visat Circle       |
|     8      |     9      |     Sargasan     |        Visat Circle       |
|     4      |     9      |   Visat Circle   |        Visat Circle       |
|     6      |     2      |   IITGN Campus   |        IITGN Campus       |
|     7      |     8      |   IITGN Campus   |        IITGN Campus       |
|     9      |     4      |   IITGN Campus   |        IITGN Campus       |
+------------+------------+------------------+-------------------------+
```

- INSERT Operation:



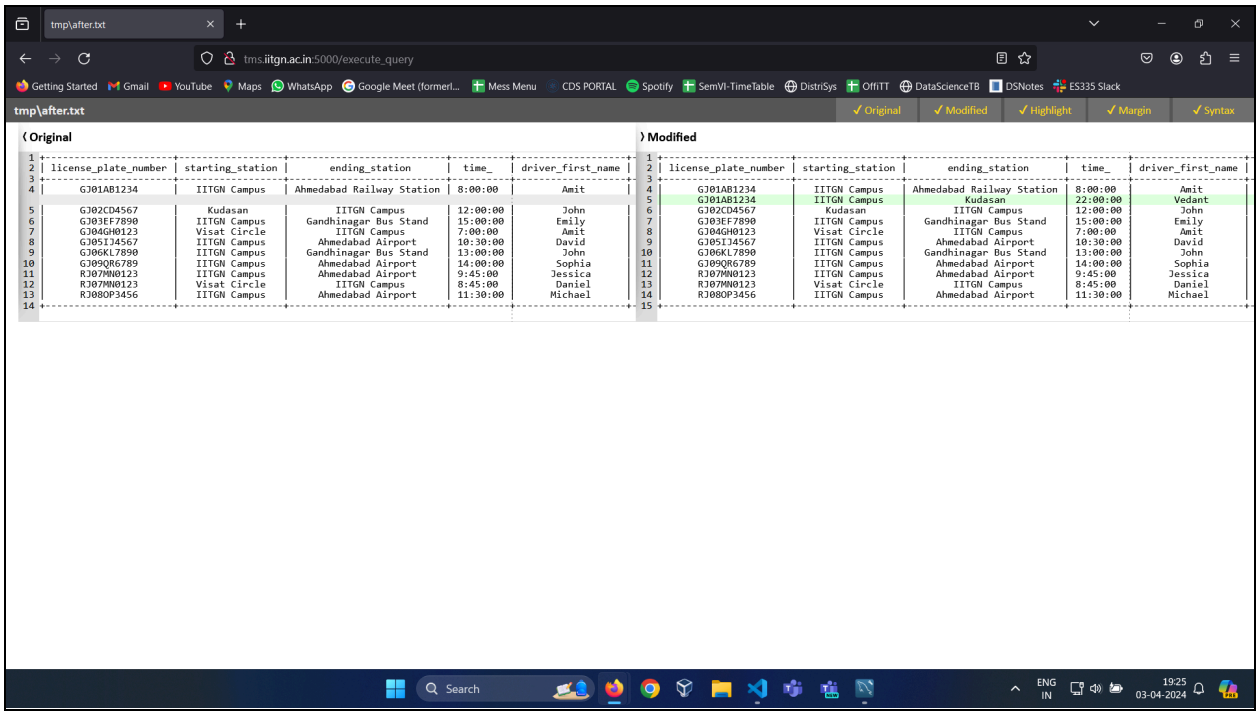
ADMIN TERMINAL

Enter your SQL query:

```
insert into TransportationLog values ('GJ01AB1234', 'IITGN Campus', 'Kudasan', '22:00:00', 'Vedant','NotKumbhar', 25, 'exit');
```

Execute Query

Action	SQL Query
Provide access	UPDATE Users SET admin_priveleges='yes' WHERE email=
Remove access	UPDATE Users SET admin_priveleges='no' WHERE email=



tmp\after.txt

tms.iitgn.ac.in:5000/execute_query

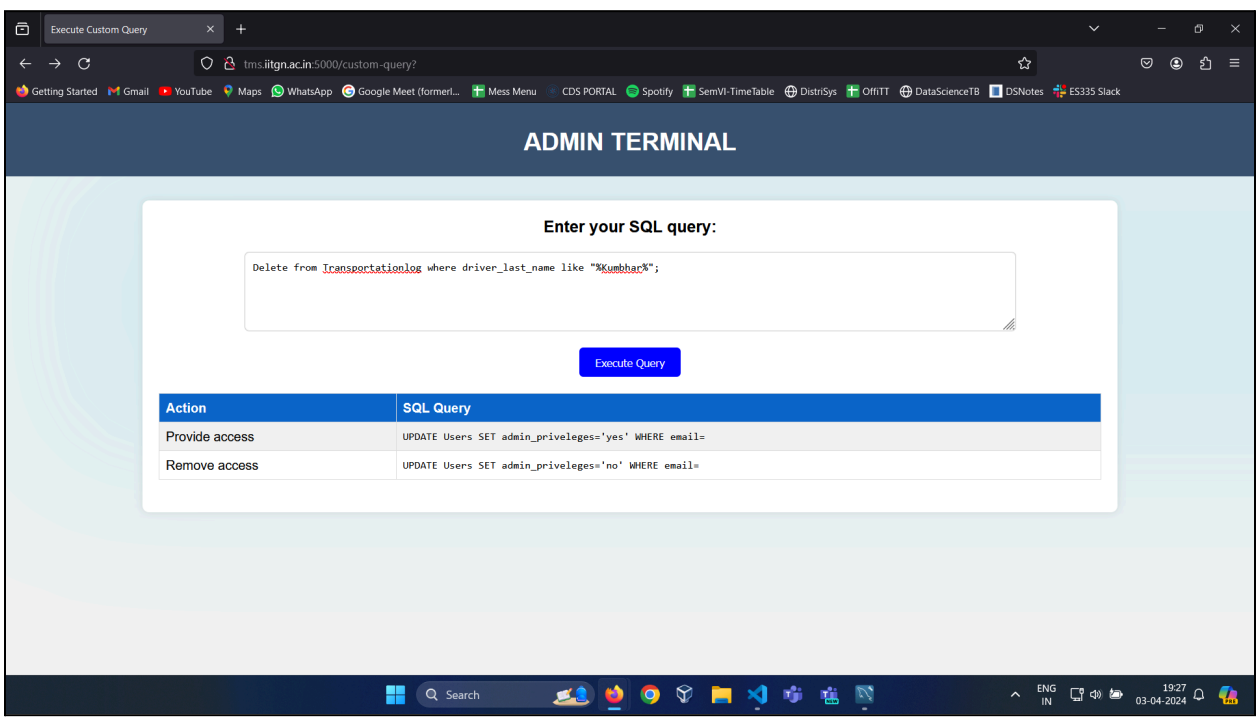
Original

license_plate_number	starting_station	ending_station	time_	driver_first_name
GJ01AB1234	IITGN Campus	Ahmedabad Railway Station	8:00:00	Amit
GJ02CD4567	Kudasan	IITGN Campus	12:00:00	John
GJ03EF7890	IITGN Campus	Gandhinagar Bus Stand	15:00:00	Emily
GJ04GH0123	Visat Circle	IITGN Campus	7:00:00	Amit
GJ05IJ4567	IITGN Campus	Ahmedabad Airport	10:30:00	David
GJ06KL7890	IITGN Campus	Gandhinagar Bus Stand	13:00:00	John
GJ09QR6789	IITGN Campus	Ahmedabad Airport	14:00:00	Sophia
RJ07MN0123	IITGN Campus	Ahmedabad Airport	9:45:00	Jessica
RJ07MN0123	Visat Circle	IITGN Campus	8:45:00	Daniel
RJ08OP3456	IITGN Campus	Ahmedabad Airport	11:30:00	Michael

Modified

license_plate_number	starting_station	ending_station	time_	driver_first_name
GJ01AB1234	IITGN Campus	Ahmedabad Railway Station	8:00:00	Amit
GJ01AB1234	IITGN Campus	Kudasan	22:00:00	Vedant
GJ02CD4567	Kudasan	IITGN Campus	12:00:00	John
GJ03EF7890	IITGN Campus	Gandhinagar Bus Stand	15:00:00	Emily
GJ04GH0123	Visat Circle	IITGN Campus	7:00:00	Amit
GJ05IJ4567	IITGN Campus	Ahmedabad Airport	10:30:00	David
GJ06KL7890	IITGN Campus	Gandhinagar Bus Stand	13:00:00	John
GJ09QR6789	IITGN Campus	Ahmedabad Airport	14:00:00	Sophia
RJ07MN0123	IITGN Campus	Ahmedabad Airport	9:45:00	Jessica
RJ07MN0123	Visat Circle	IITGN Campus	8:45:00	Daniel
RJ08OP3456	IITGN Campus	Ahmedabad Airport	11:30:00	Michael

- DELETE Operation:



Execute Custom Query

tms.iitgn.ac.in:5000/custom-query?

ADMIN TERMINAL

Enter your SQL query:

```
Delete from Transportationlog where driver_last_name like "%Kumbhar%";
```

Execute Query

Action	SQL Query
Provide access	UPDATE Users SET admin_priveleges='yes' WHERE email=
Remove access	UPDATE Users SET admin_priveleges='no' WHERE email=

tmp\after.txt

tms.iitgn.ac.in:5000/execute_query

Getting Started | Gmail | YouTube | Maps | WhatsApp | Google Meet (formerl... | Mess Menu | CDS PORTAL | Spotify | SemVI-TimeTable | DistriSys | OffiTT | DataScienceTB | DSNotes | ES335 Slack

tmp\after.txt

✓ Original ✓ Modified ✓ Highlight ✓ Margin ✓ Syntax

Original

license_plate_number	starting_station	ending_station	time_	driver_first_name
GJ01AB1234	IITGN Campus	Ahmedabad Railway Station	8:00:00	Amit
GJ01AB1234	IITGN Campus	Kudasan	22:00:00	Vedant
GJ02CD4567	Kudasan	IITGN Campus	12:00:00	John
GJ03EF7890	IITGN Campus	Gandhinagar Bus Stand	15:00:00	Emily
GJ04GH0123	Visat Circle	IITGN Campus	7:00:00	Amit
GJ05IJ4567	IITGN Campus	Ahmedabad Airport	10:30:00	David
GJ06KL7890	IITGN Campus	Gandhinagar Bus Stand	13:00:00	John
GJ09QR6789	IITGN Campus	Ahmedabad Airport	14:00:00	Sophia
RJ07MN0123	IITGN Campus	Ahmedabad Airport	9:45:00	Jessica
RJ07MN0123	Visat Circle	IITGN Campus	8:45:00	Daniel
RJ08OP3456	IITGN Campus	Ahmedabad Airport	11:30:00	Michael

Modified

license_plate_number	starting_station	ending_station	time_	driver_first_name
GJ01AB1234	IITGN Campus	Ahmedabad Railway Station	8:00:00	Amit
GJ02CD4567	Kudasan	IITGN Campus	12:00:00	John
GJ03EF7890	IITGN Campus	Gandhinagar Bus Stand	15:00:00	Emily
GJ04GH0123	Visat Circle	IITGN Campus	7:00:00	Amit
GJ05IJ4567	IITGN Campus	Ahmedabad Airport	10:30:00	David
GJ06KL7890	IITGN Campus	Gandhinagar Bus Stand	13:00:00	John
GJ09QR6789	IITGN Campus	Ahmedabad Airport	14:00:00	Sophia
RJ07MN0123	IITGN Campus	Ahmedabad Airport	9:45:00	Jessica
RJ07MN0123	Visat Circle	IITGN Campus	8:45:00	Daniel
RJ08OP3456	IITGN Campus	Ahmedabad Airport	11:30:00	Michael

Search

ENG IN 19:27 03-04-2024

Execute Custom Query

127.0.0.1:5000/custom-query?

ADMIN TERMINAL

Enter your SQL query:

```
SELECT AVG(capacity) from parking_space_new
```

Execute Query

Action	SQL Query
Provide access	UPDATE Users SET admin_priveleges='yes' WHERE email=
Remove access	UPDATE Users SET admin_priveleges='no' WHERE email=

Type here to search

ENG 15:06 04-04-2024

tmp\before.txt

127.0.0.1:5000/execute_query

tmp\before.txt

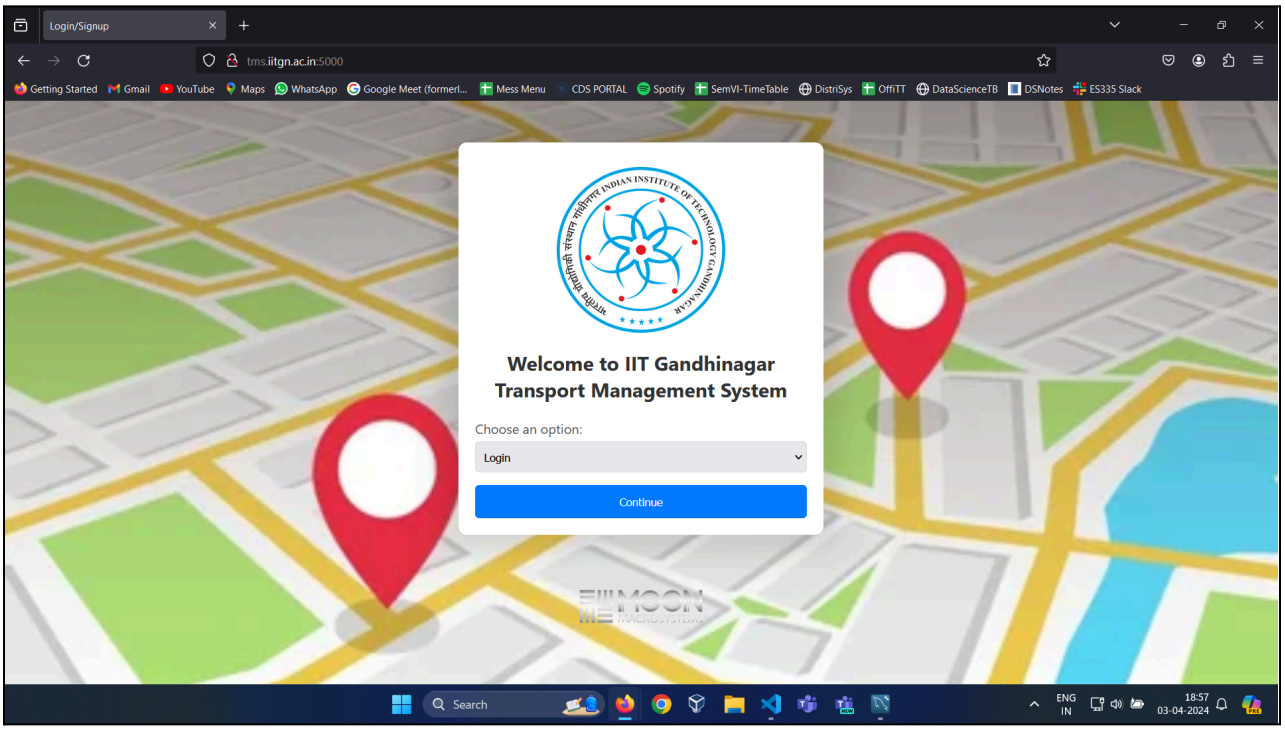
✓ Original

Original

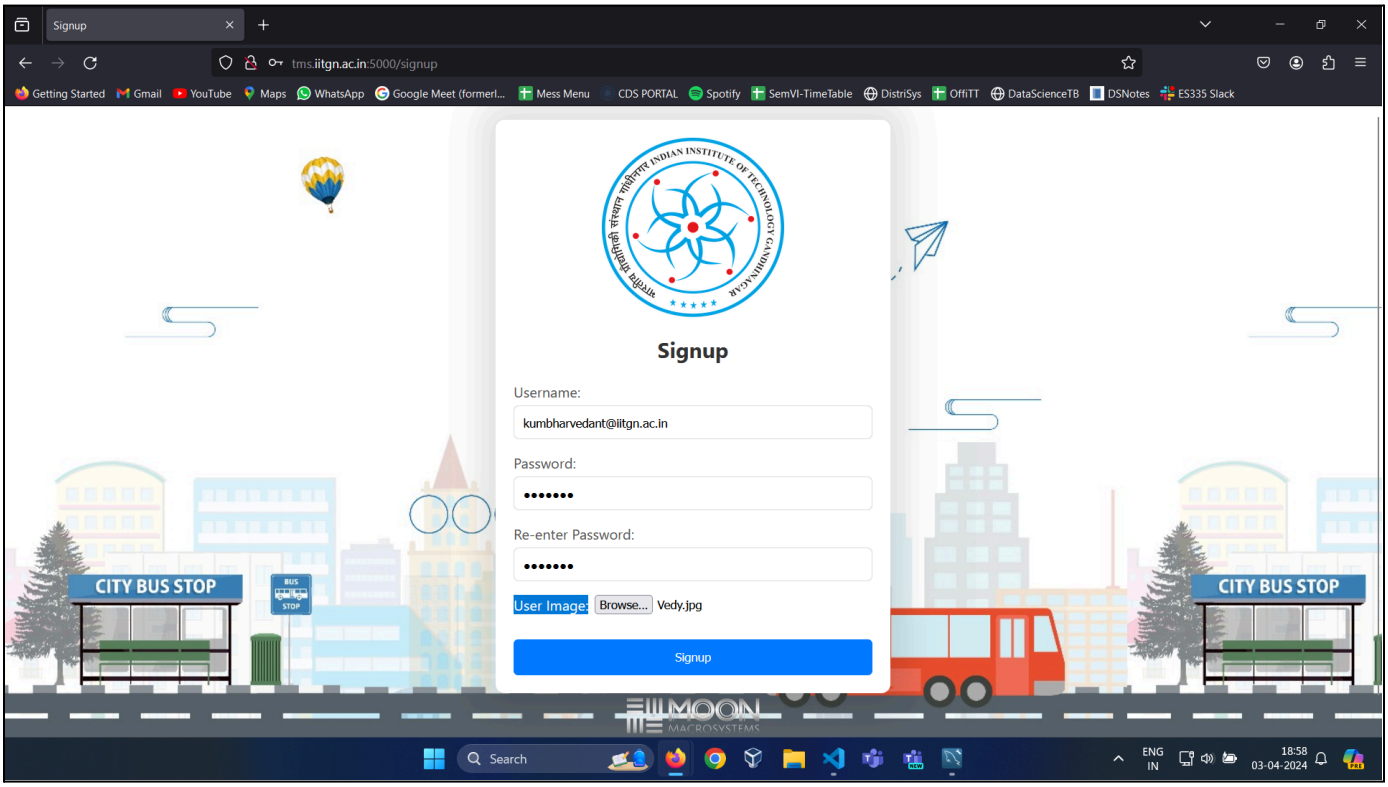
AVG(capacity)
34.5000

Components of the website:

<u>Login/SignUp Page:</u> This is the Login/SignUp Page for all the users (including **admins**). One can select whether they want to Login/Signup and proceed further.

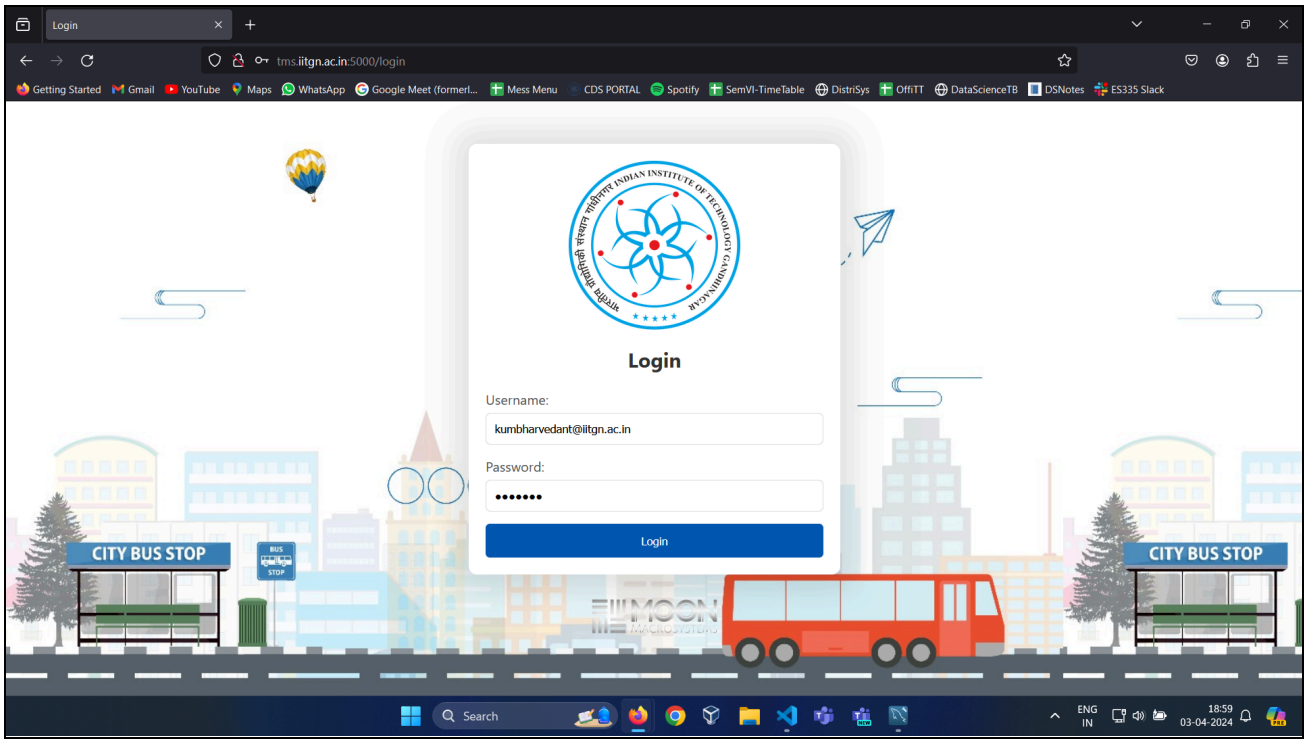


Signing up a dummy user:

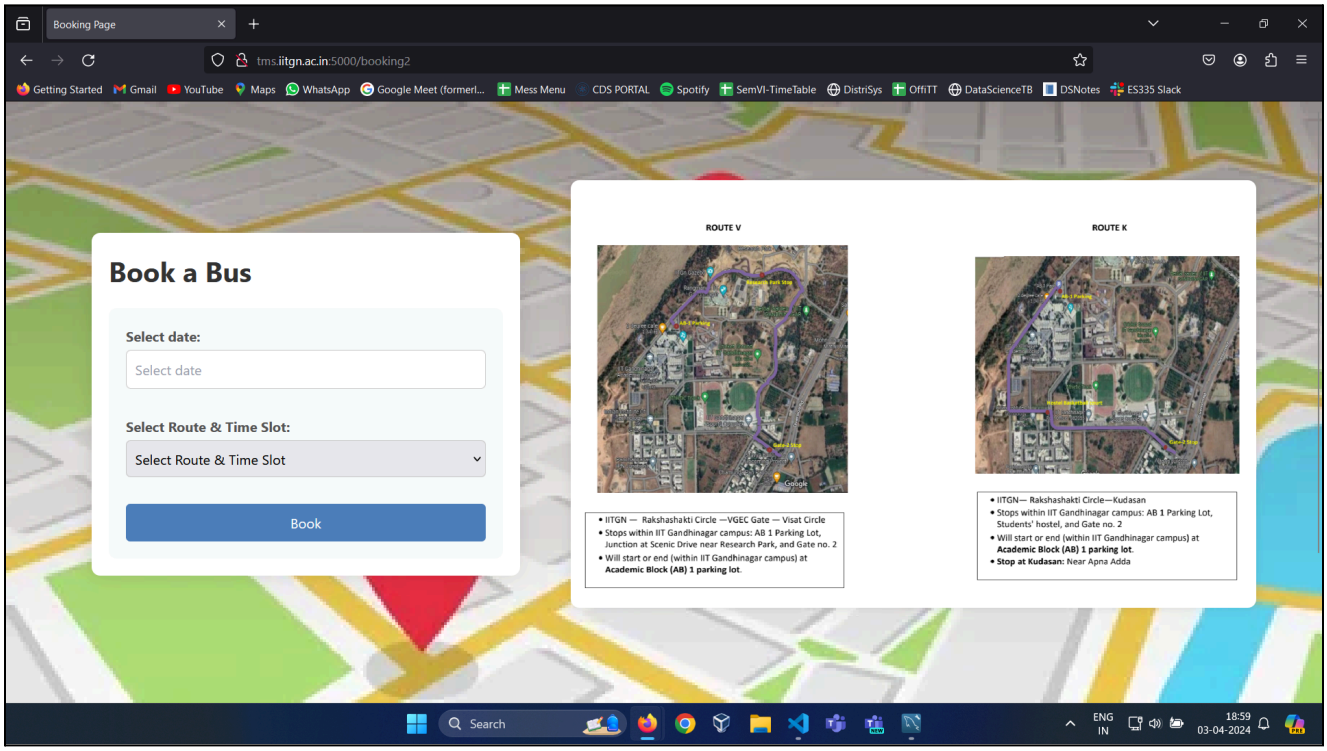


Now the user has Signed Up. After signing up is successful, they are redirected to the the login page.

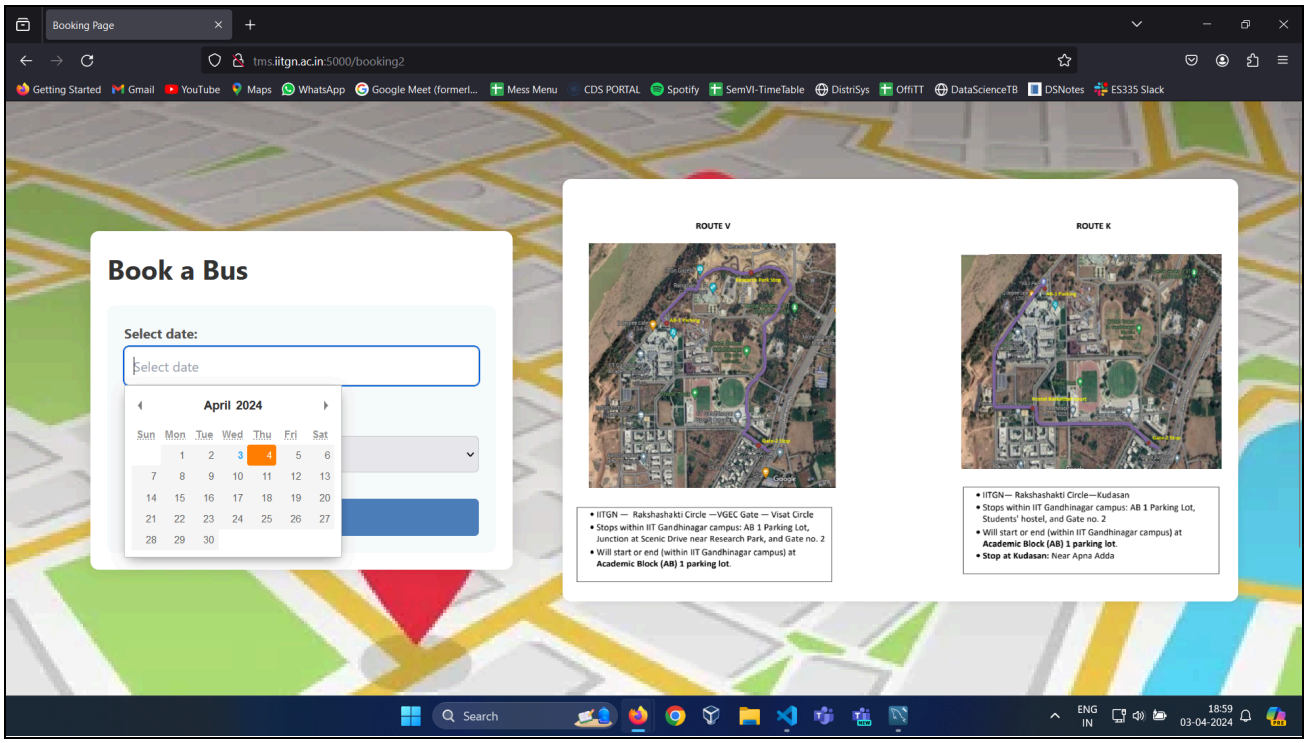
Logging in to the application after signing up:



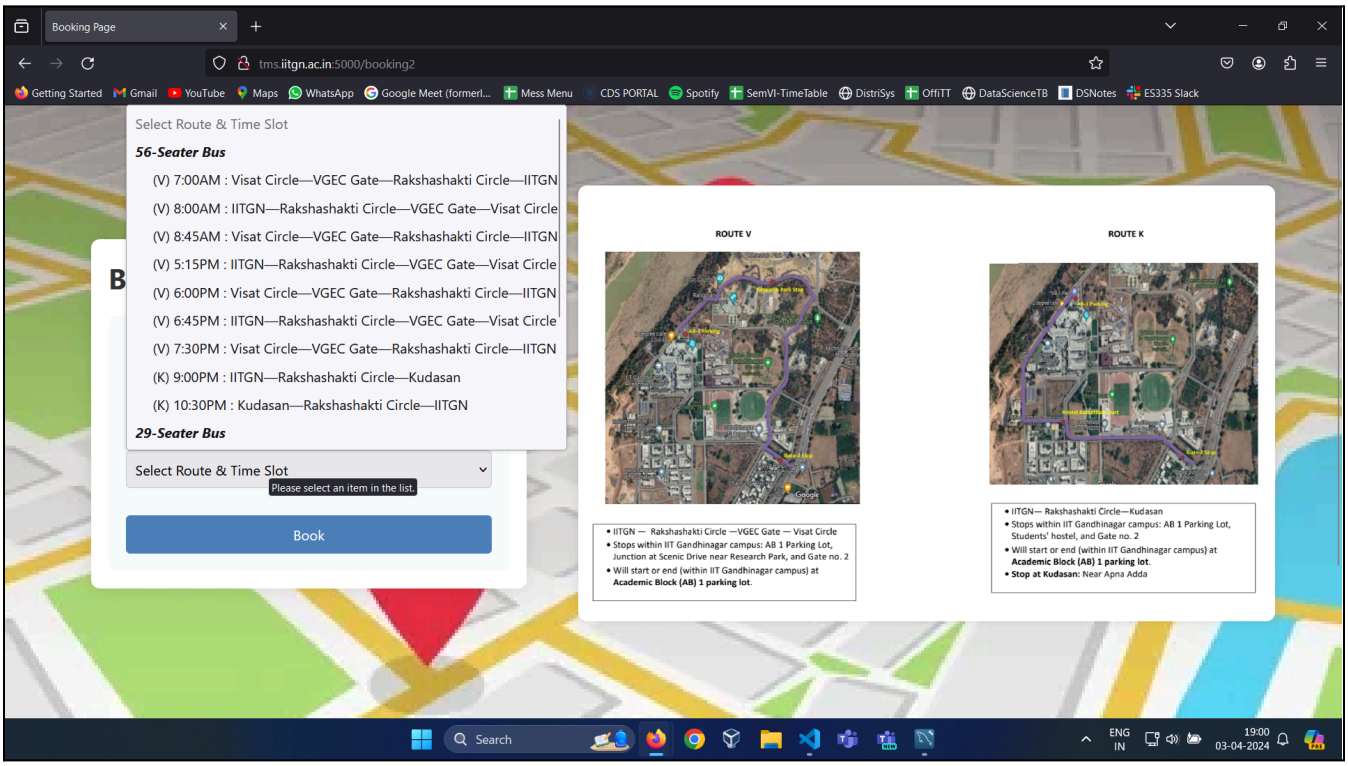
After Logging in the booking page will appear where the user can book seat in the bus:



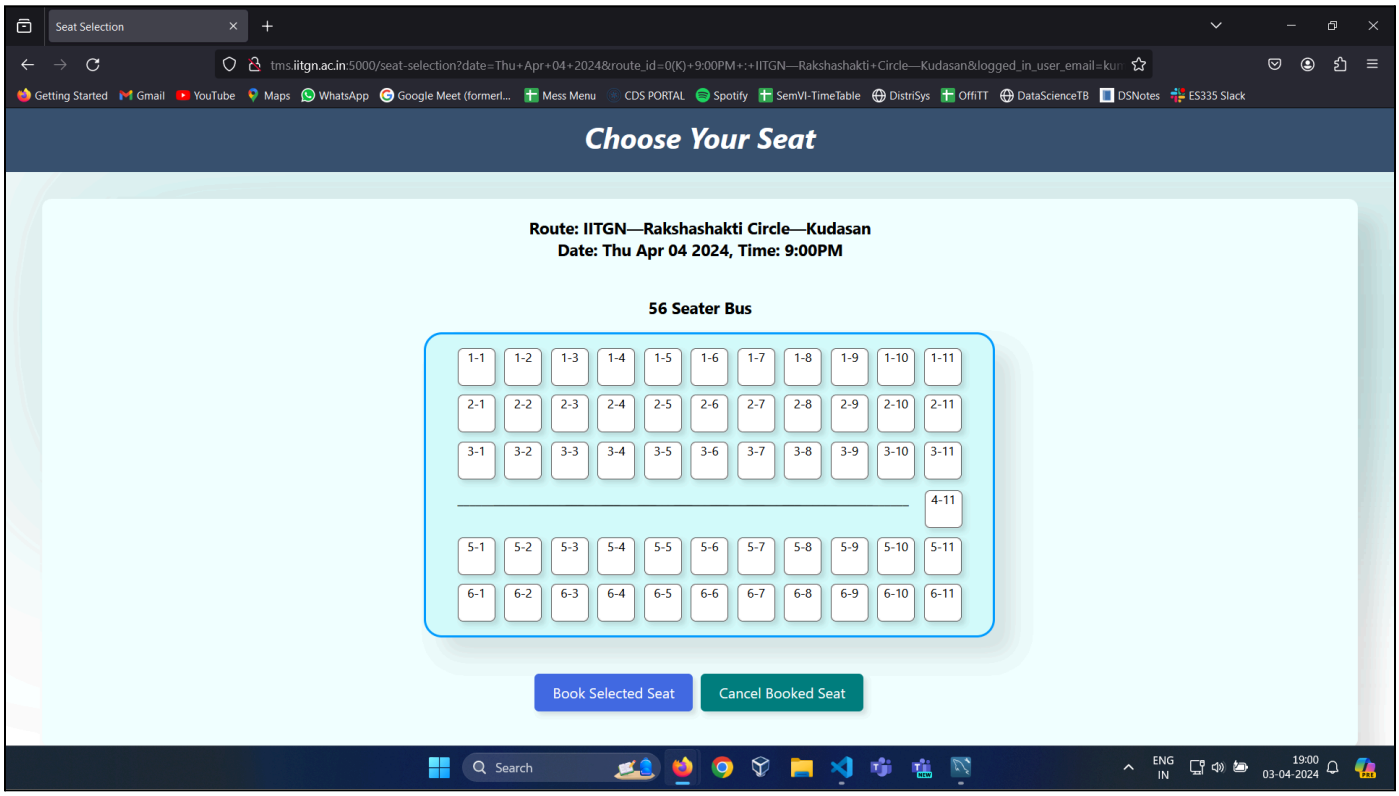
Choosing the date of travel:

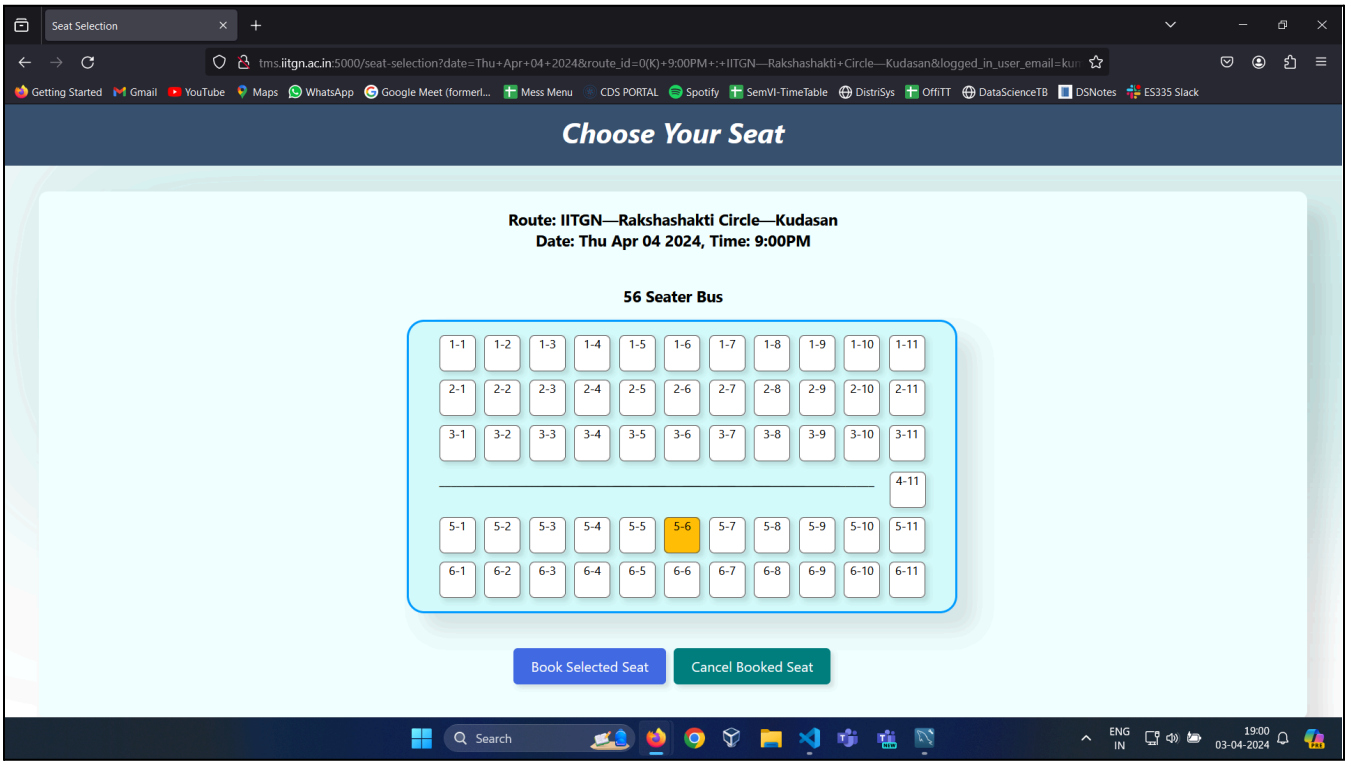


Selecting the route and the time slot:

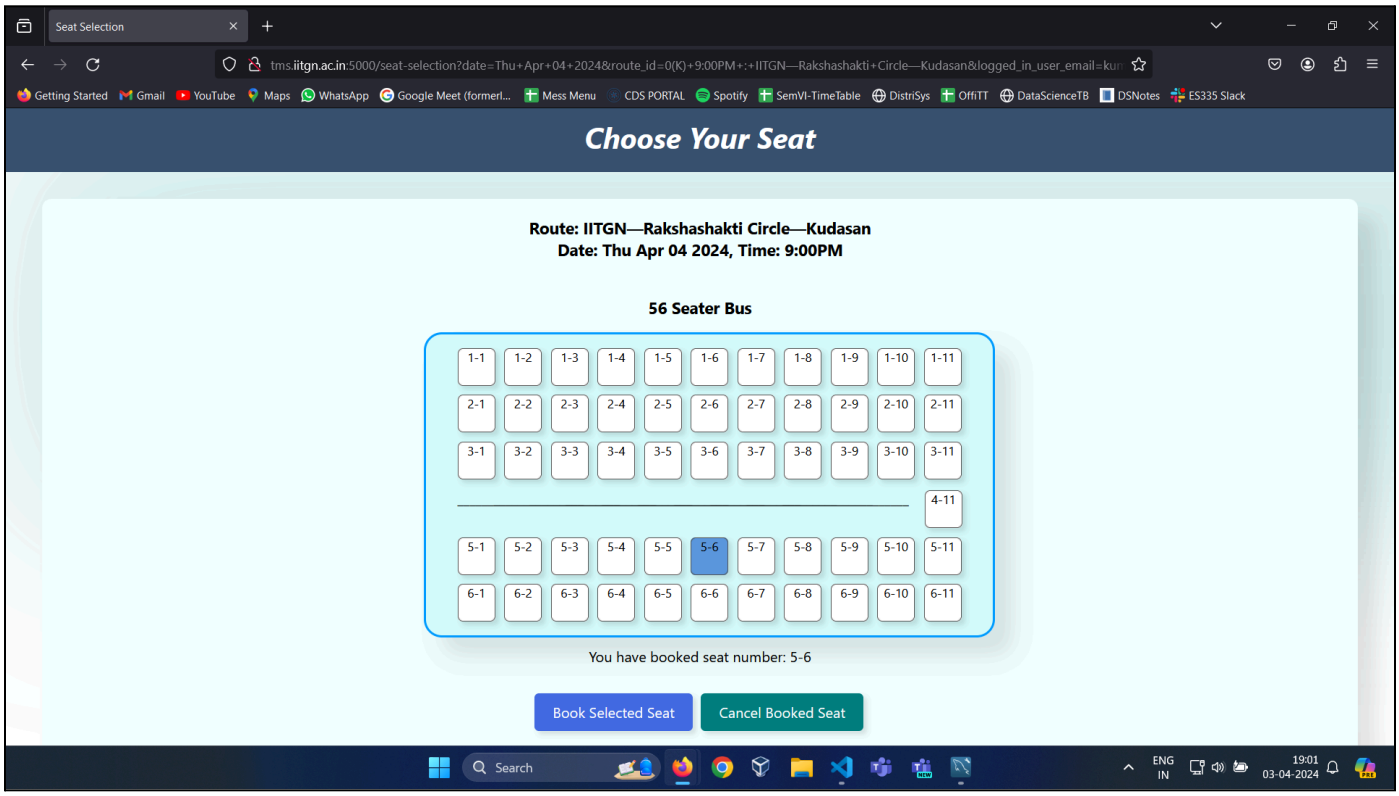


Selecting the seat in the bus:

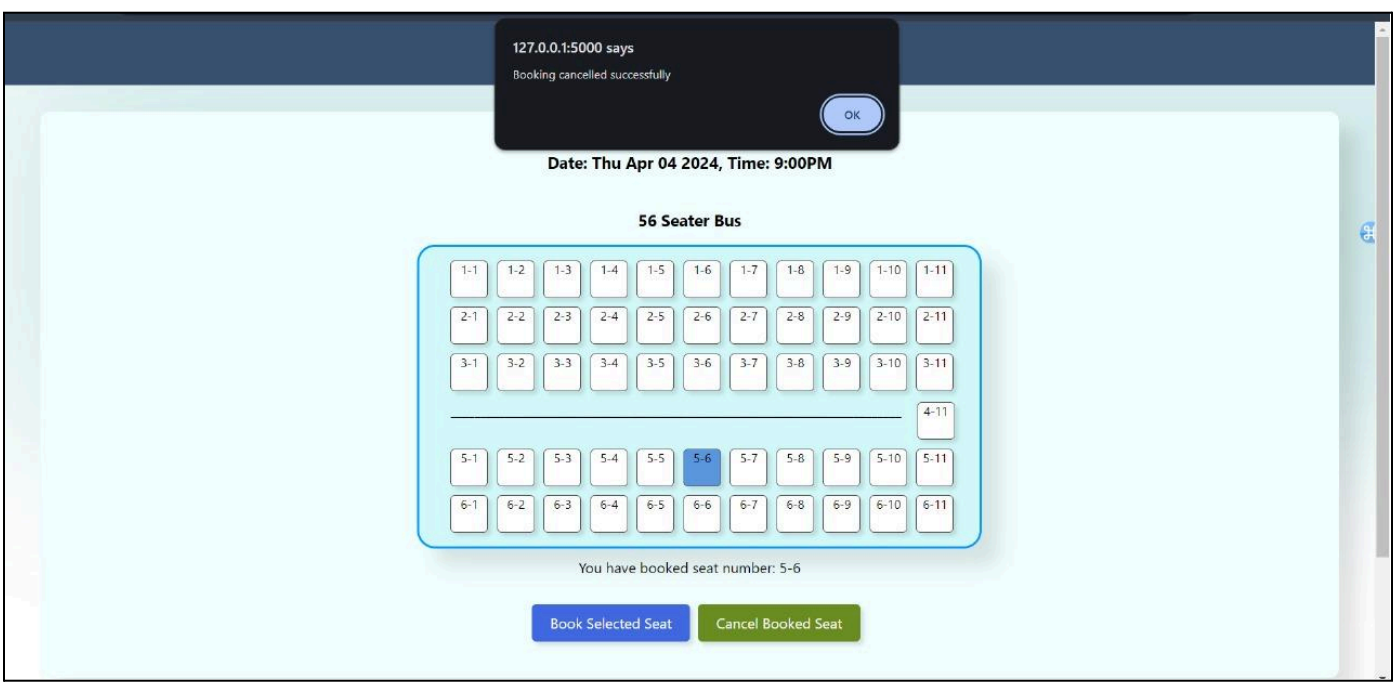


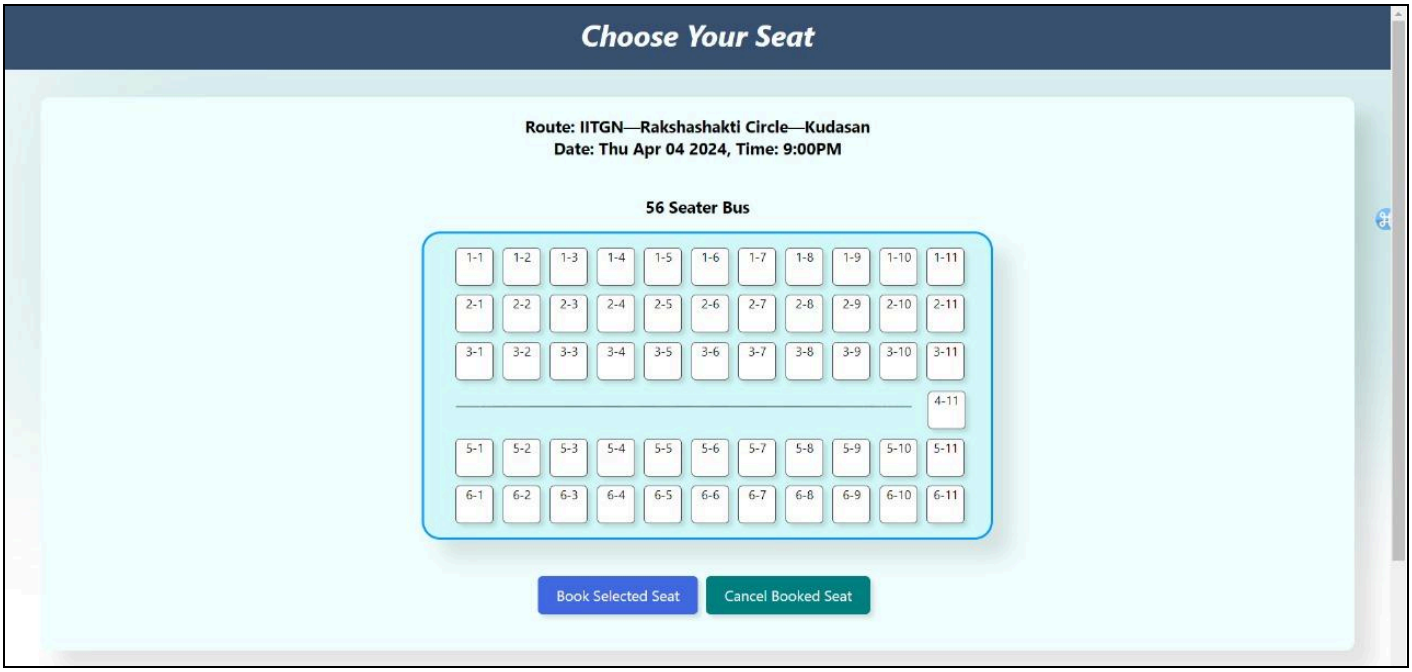


Booking Confirmed:

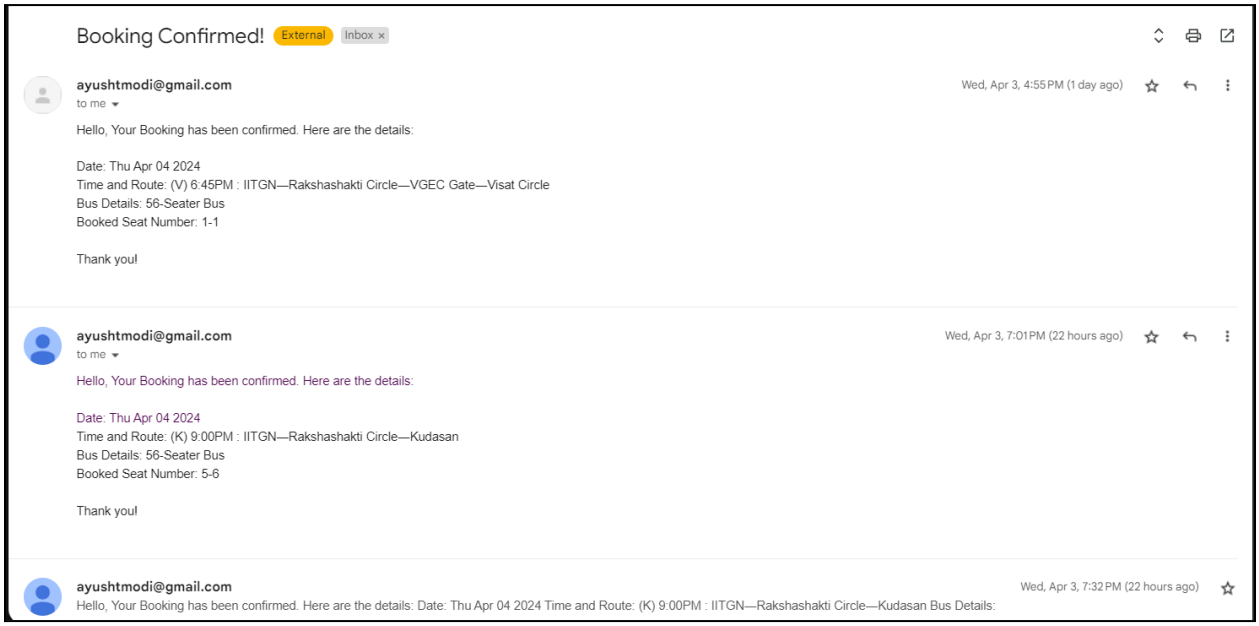


Booking Cancelled:

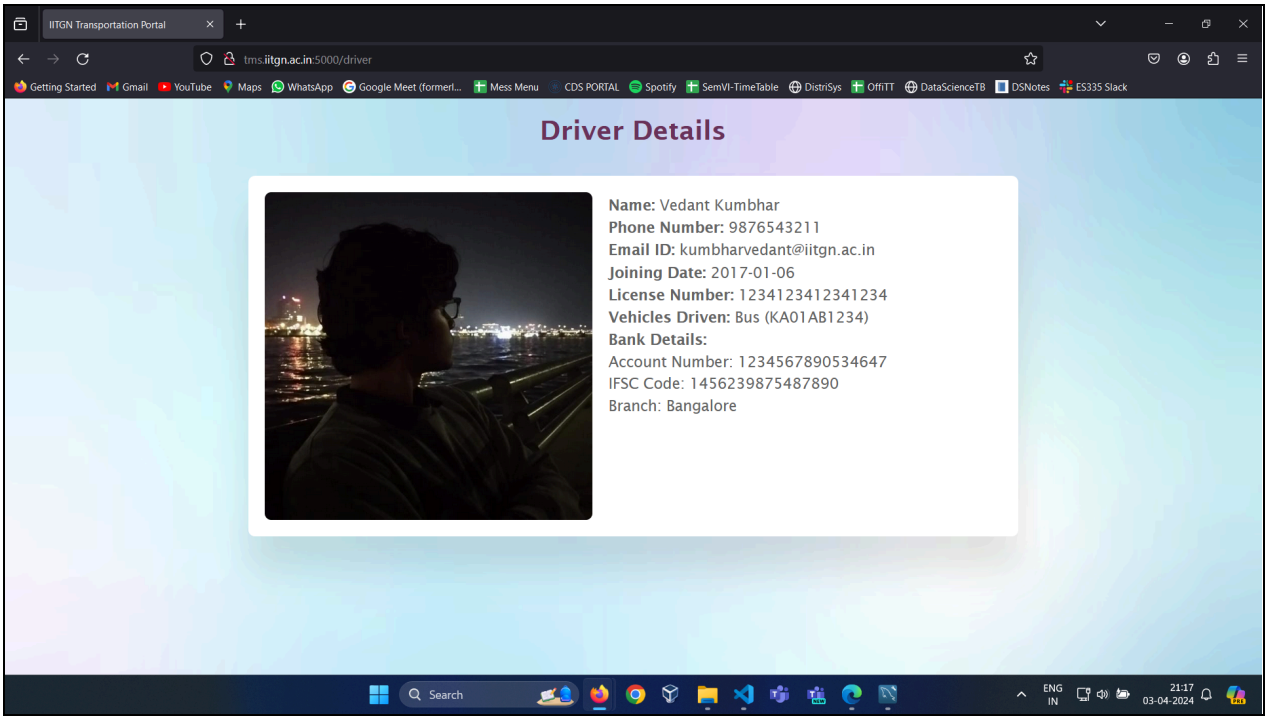




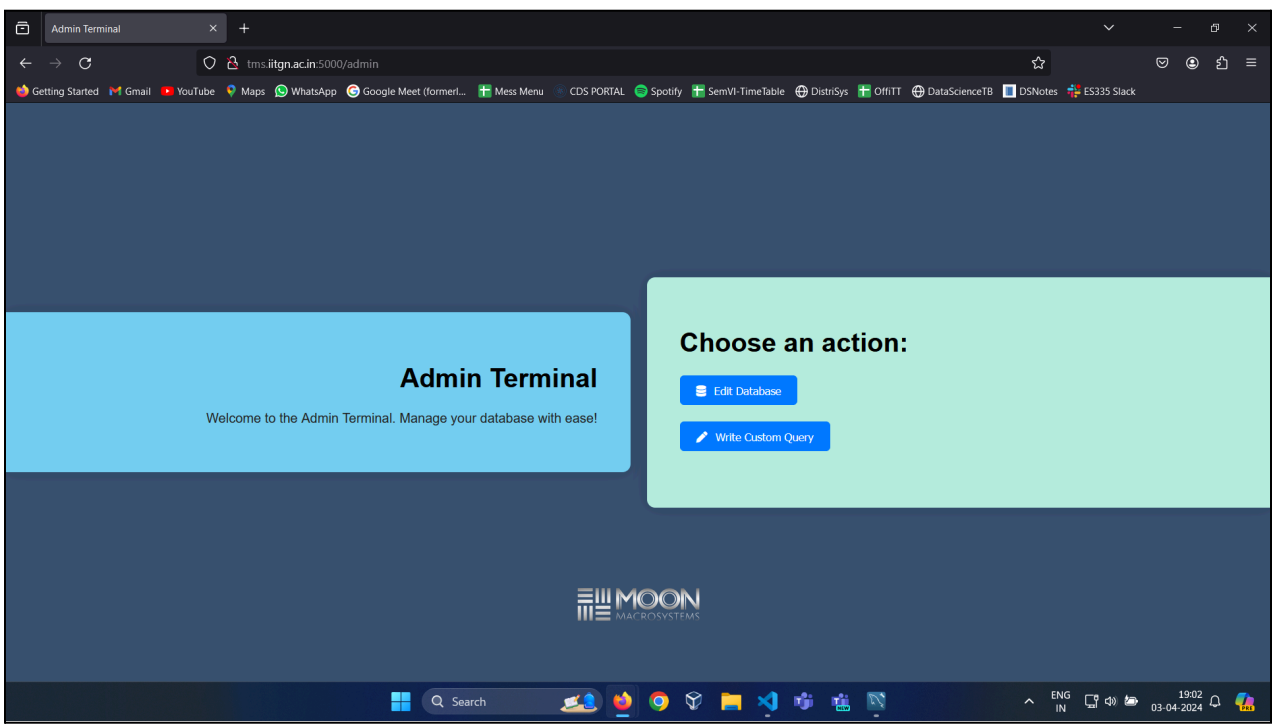
We also receive an email for the confirmation of the bookings.



If the user is a driver then he/she would be directed to the driver's page:

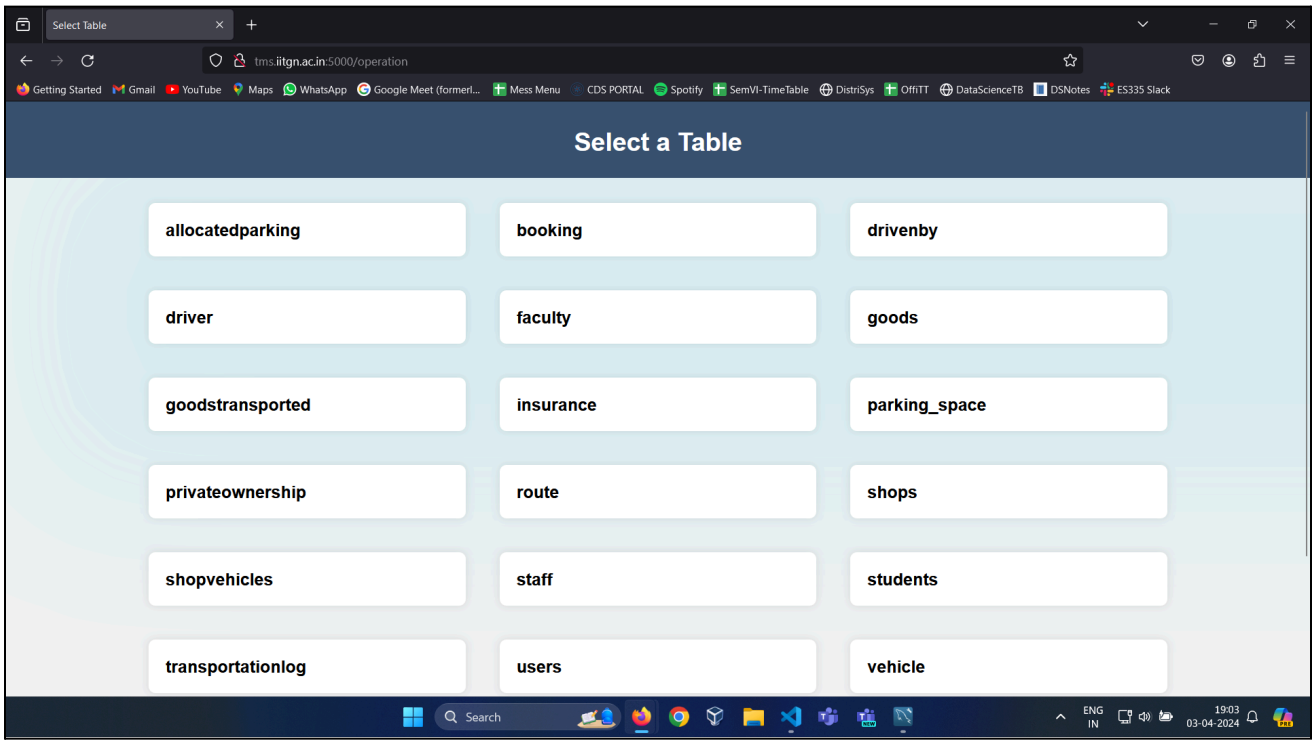


Screenshots of the Admin Terminal:



Choosing the button to Edit Database:

In the admin terminal tables can be selected and some operations can be performed on them:



3. Contribution

G1:

Ayush Modi
Shreesh Agrawal
Anushk Bhana

G2:

Mithil Pechimuthu
Vedant Kumbhar

(We had initially divided in this manner, but we realized later that it was getting extremely difficult to divide the frontend and backend as they were hard to decouple. Also, the frontend team required the specific things to display from the backend which the backend team updated from time to time. So instead we divided the specific web pages and some other designing and debugging tasks among us.)

Ayush Modi:

1) Worked on the Booking Part, building the Frontend and Backend for booking seats from the seat-selection UI.
2) Created the Edit Database part of the admin terminal. Built the apis for the various operations and the frontend aspect of displaying the tables before and after execution of the operation.
3) Beautified the web-app by adding CSS stylings.
4) Helped in the documentation.

Mithil Pechimuthu:

1) Built the backend and framework for the admin terminal of the webpage.
2) Designed and structured the website workflow.
3) Wrote the README.md and the final writeup.pdf.
4) Implemented all the stitching code.

Vedant Kumbhar:

1) Developed the user profile page inclusive of the backend and the frontend.
2) Worked on the HTMLand CSS styling of some of the pages.
3) Helped identify the bugs in the overall functionality of the website workflow.
4) Worked on the backend and frontend of the Signup page.

Anushk Bhana:

1) Developed the home page layout and design.
2) Contributed login and signup functionality.
3) Contributed to the development of the booking feature.
4) Enhanced user experience through styling and UI improvements.

Shreesh Agrawal:

1) Worked on the Drivers Page.
2) Wrote the code for the frontend using HTML and did styling using CSS.
3) Built the backend for the drivers page using Flask.
4) Connected the database with the page/app using SQLAlchemy (a Python SQL toolkit).
5) Updated the database SQL code and dummy data according to the desired needs.
6) Contributed to the ideation of the structure of the final web application.
7) Did the final documentation.

References

1) https://docs.python.org/3/library/difflib.html
2) https://www.sqlalchemy.org/
3) https://www.youtube.com/watch?v=3aZa-3TYzZo
4) https://pypi.org/project/diff-match-patch/
5) https://github.com/RahulVC02/Alumni-Database-Webapp
6) https://github.com/frank-chris/HospitalDatabaseSystem/blob/main/README.pdf