

DATABASE MANAGEMENT SYSTEM
UE19CS301
INTELLIGENT METRO MANAGEMENT SYSTEM

Team-members:

Venuthurla Venkata Pradeep Reddy

PES1UG19CS564

Y Nikhil Bharadwaj PES1UG19CS586

Yamajala Siddhardha PES1UG19CS588

Languages used for frontend :

1. HTML
2. CSS

Languages used for Database connection :

1. Python.

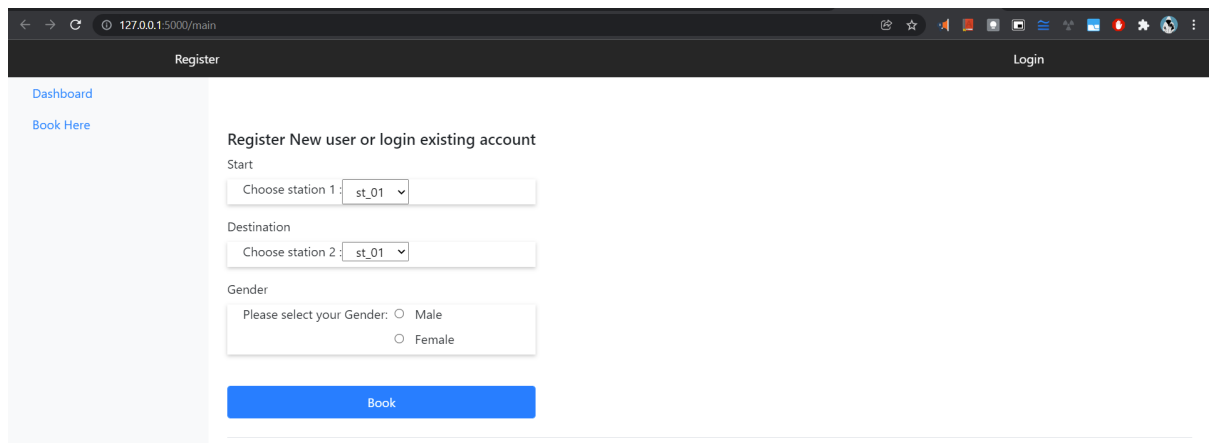
Dependencies installed :

1. Flask: Flask is a web framework, it's a Python module that lets you develop web applications easily. It has a small and easy-to-extend core: it's a microframework that doesn't include an ORM (Object Relational Manager) or such features. It does have many cool features like url routing, template engine. It is a WSGI web app framework.
2. Psycopg2: Psycopg is the most popular PostgreSQL database adapter for the Python programming language. Its main features are the complete implementation of the Python DB API 2.0 specification and the thread safety (several

threads can share the same connection). It was designed for heavily multi-threaded applications that create and destroy lots of cursors and make a large number of concurrent INSERTs or UPDATEs.

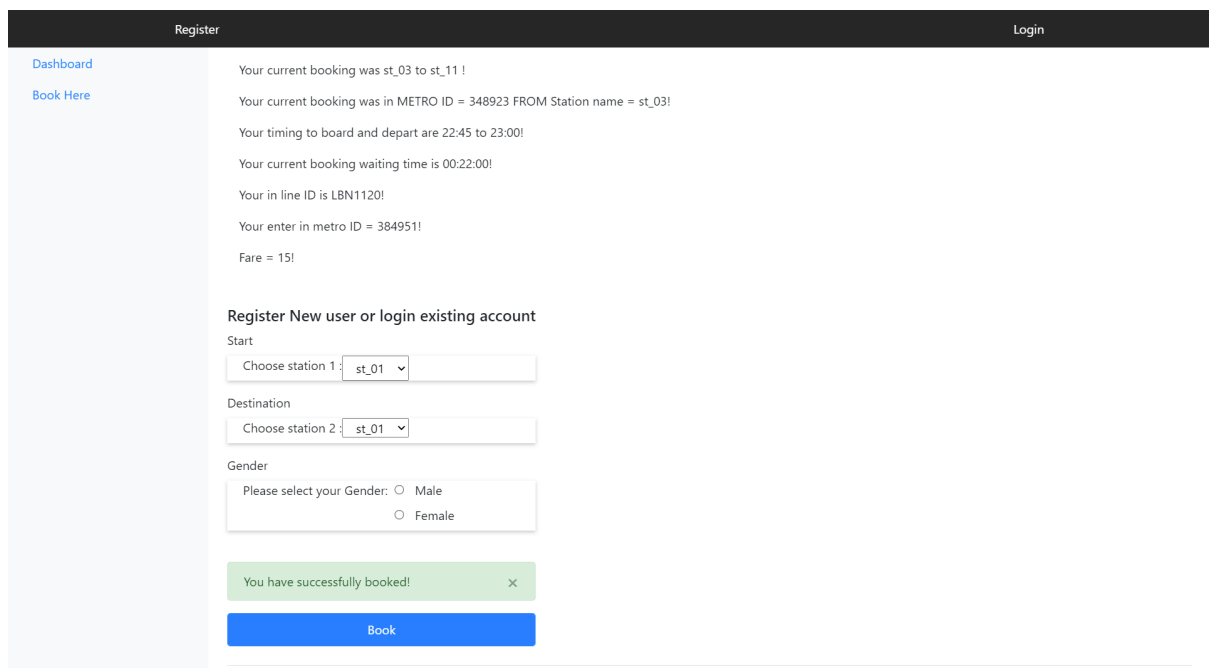
Queries implemented in frontend:

Main page:



The screenshot shows a web browser window with the URL 127.0.0.1:5000/main. The page has a dark header with "Register" on the left and "Login" on the right. A sidebar on the left contains links for "Dashboard" and "Book Here". The main content area is titled "Register New user or login existing account". It includes a "Start" section with a dropdown menu for "Choose station 1" set to "st_01". Below this is a "Destination" section with a dropdown menu for "Choose station 2" also set to "st_01". A "Gender" section asks the user to select their gender with radio buttons for "Male" and "Female". At the bottom of the form is a blue "Book" button.

This page is used by the unregistered users to book tickets for their metro travel.



This screenshot shows the same web page after a successful booking. The main content area now displays several lines of confirmation text: "Your current booking was st_03 to st_11 !", "Your current booking was in METRO ID = 348923 FROM Station name = st_03!", "Your timing to board and depart are 22:45 to 23:00!", "Your current booking waiting time is 00:22:00!", "Your in line ID is LBN1120!", "Your enter in metro ID = 384951!", and "Fare = 15!". Below this text is the same "Register New user or login existing account" form as seen in the previous screenshot. At the bottom of the form, there is a green notification box that says "You have successfully booked!" with a close button (X). The blue "Book" button remains at the bottom of the form.

Platform table:

Query Editor

Query History

1

SELECT * FROM public.platform

2

Data Output

Explain

Messages

Notifications

| | platform_no integer | arrival_time character varying (30) | departure_time character varying (30) | waiting_time character varying (10) | station_id integer | user_id integer |
|----|------------------------|--|--|--|-----------------------|--------------------|
| 94 | 1 | 22:55 | 23:20 | 00:27:00 | 348921 | 10097 |
| 95 | 1 | 22:45 | 23:00 | 00:22:00 | 348923 | 10098 |
| 96 | 1 | 22:50 | 23:05 | 00:23:00 | 348922 | 10099 |

Ticket Table:

1SELECT * FROM public.ticket

2ORDER BY ticket_id ASC

Data Output

Explain

Messages

Notifications

| | ticket_id [PK] integer | date_cur character varying (15) | start_time character varying (30) | end_time character varying (30) | fare integer | start character varying (30) | destination character varying (30) | user_id integer |
|----|---------------------------|------------------------------------|--------------------------------------|------------------------------------|-----------------|---------------------------------|---------------------------------------|--------------------|
| 93 | 349483 | 07:12:2021 | 22:25:20 | 23:20 | 25 | st_01 | st_11 | 10096 |
| 94 | 349484 | 07:12:2021 | 22:28:40 | 23:20 | 25 | st_01 | st_11 | 10097 |
| 95 | 349485 | 08:12:2021 | 22:23:26 | 23:00 | 15 | st_03 | st_11 | 10098 |

Boards Table:

Query Editor

Query History

1

2

SELECT

*

FROM

public.boards

Data Output

Explain

Messages

Notifications

metro_id

integer

user_id

integer

station_id

integer

93

384951

10096

348921

94

384951

10097

348921

95

384951

10098

348923

Users Table:

Query Editor

Query History

1

SELECT * FROM public.users

2

ORDER BY user_id ASC

Data Output

Explain

Messages

Notifications

user_id

[PK] integer

gender

character varying (10)

station_id

integer

96

10096

M

348921

97

10097

M

348921

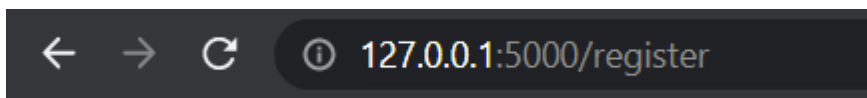
98

10098

M

348923

"User Registration" is for those who are looking for a metrocard for frequent usage of metro services.



Register New user or login existing account

Your Name

Enter your Name

Your Email

Enter your Email

Phone Number

Enter your Phone Number

Phone Number 2

Enter your second Phone Number

Gender

gender

Password

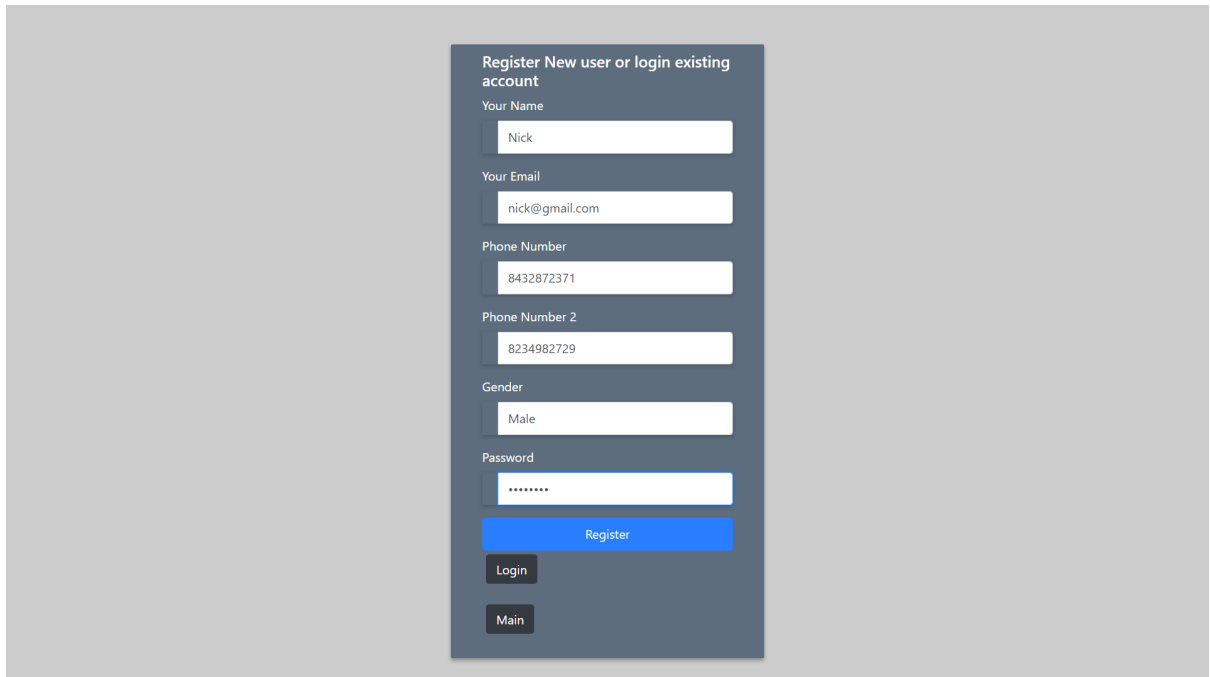
Password

Register

Login

Main

After filling out the fields:



Register New user or login existing account

Your Name

Your Email

Phone Number

Phone Number 2

Gender

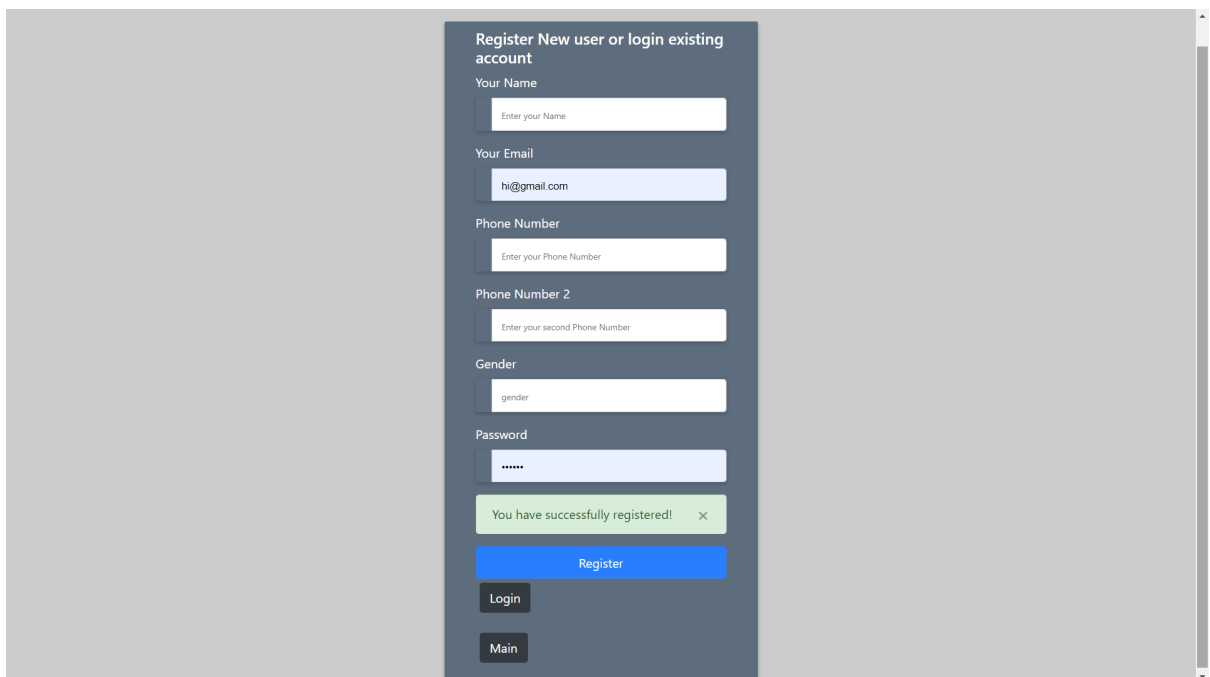
Password

[Register](#)

[Login](#)

[Main](#)

Successfully registered:



Register New user or login existing account

Your Name

Your Email

Phone Number

Phone Number 2

Gender

Password

[You have successfully registered! x](#)

[Register](#)

[Login](#)

[Main](#)

Members Table:

```
1 SELECT * FROM public.members
2 ORDER BY email_id ASC
```

Data Output Explain Messages Notifications

| | email_id [PK] character varying (30) | name character varying (20) | gender character varying (10) | password character varying (30) |
|----|---|--------------------------------|----------------------------------|------------------------------------|
| 10 | hi@gmail.com | hi | M | qweqwe |
| 11 | kamarthikaran@gmail.com | karan | M | 123456 |
| 12 | nick@gmail.com | Nick | Male | nick@123 |


Metrocard Table:

```
1 SELECT * FROM public.metro_card
2 ORDER BY card_id ASC
```

Data Output Explain Messages Notifications

| | card_id [PK] integer | balance integer | start character varying (30) | destination character varying (30) | s_time character varying (30) | d_time character varying (30) | email_id character varying (30) |
|----|-------------------------|--------------------|---------------------------------|---------------------------------------|----------------------------------|----------------------------------|------------------------------------|
| 15 | 98738405 | 150 | st_01 | st_01 | 17:04:53 | 17:04:53 | vishnu@gmail.com |
| 16 | 98738406 | 200125 | st_01 | st_11 | 22:55 | 23:20 | sasankcharishma143@gmail.com |
| 17 | 98738407 | 635 | st_02 | st_10 | 22:50 | 23:05 | nick@gmail.com |

Payment Table:

 metro11/postgres@PostgreSQL 13

Query Editor

Query History




```
1 SELECT * FROM public.payment
2 ORDER BY payment_id ASC
```

Data Output

Explain

Messages

Notifications

| | payment_id  [PK] integer | status  character varying (20) | card_id  integer |
|----|--|--|--|
| 15 | 1817206 | paid | 98738404 |
| 16 | 1817207 | paid | 98738405 |
| 17 | 1817208 | paid | 98738406 |
| 18 | 1817209 | paid | 98738407 |

Phone No of Members Table:

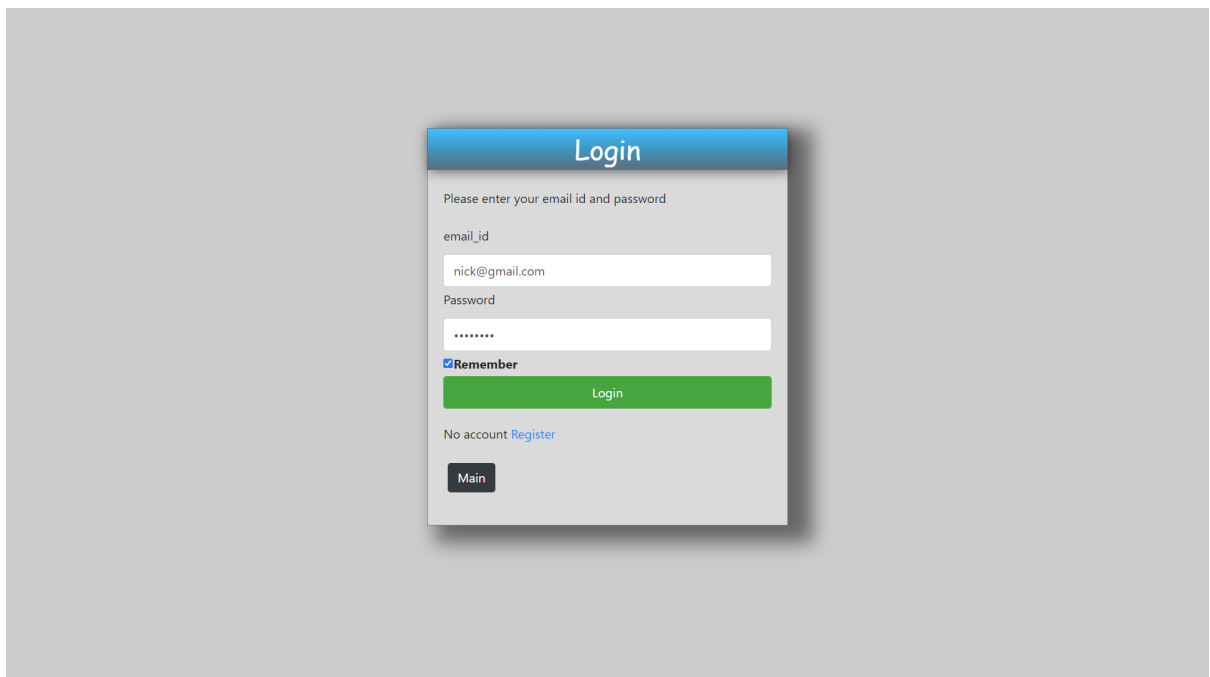
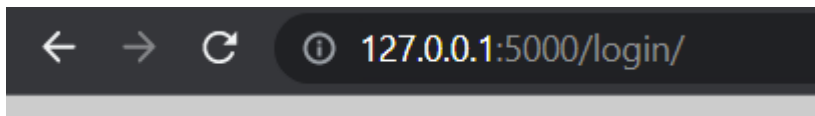
```

1 SELECT * FROM public.phoneno
2

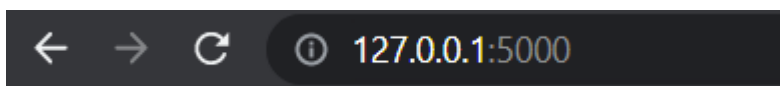
```

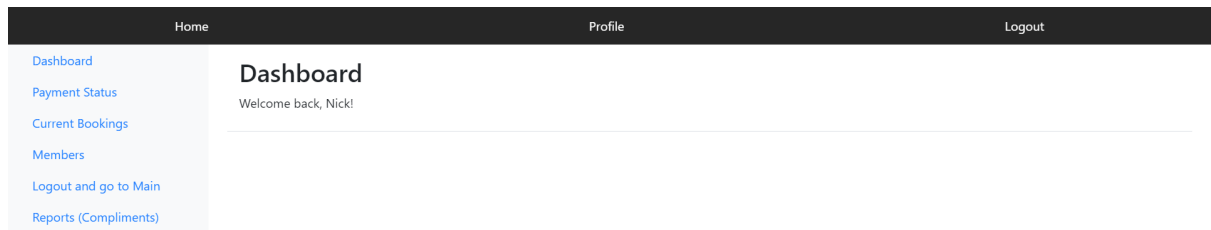
| Data Output | Explain | Messages | Notifications |
|-------------|--------------------------------|---------------------------------|------------------------------------|
| | p_no character varying (15) | p_no2 character varying (15) | email_id character varying (30) |
| 15 | 9390137310 | 9000915863 | vishnu@gmail.com |
| 16 | 9411431432 | 1431431431 | sasankcharishma143@gmail.com |
| 17 | 8432872371 | 8234982729 | nick@gmail.com |

Login Page:

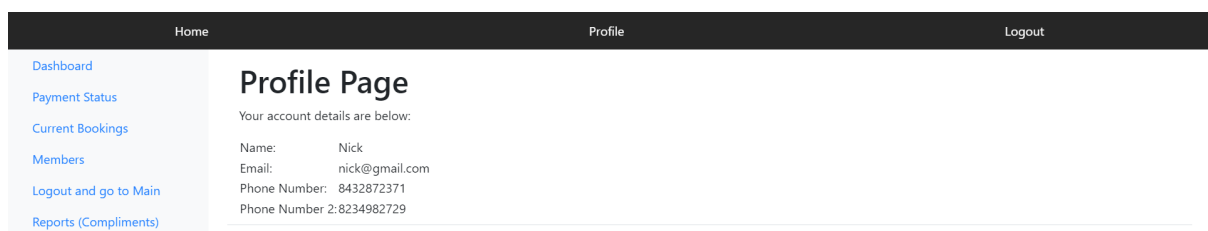
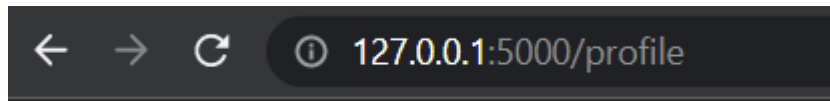


After Login getting inside dashboard:



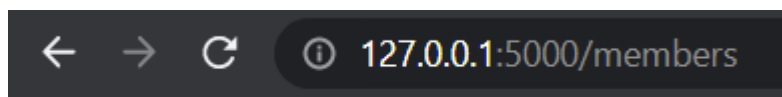


“Profile Page” of Members:



“Members” where member can check balance add balance:

Checking Balance:



The screenshot shows a web application interface with a dark header containing 'Home', 'Profile', and 'Logout' links. A left sidebar lists navigation options: 'Dashboard', 'Payment Status', 'Current Bookings', 'Members' (highlighted), 'Logout and go to Main', and 'Reports (Compliments)'. The main content area is titled 'Members Metro Card Info' and displays account details for a user named Nick. Below the details is a form to 'Register New user or login existing account' with a text input field for 'amount to add' and a blue 'ADD Amount' button. The current balance is shown as 150.

| Home | | Profile | Logout |
|-----------|--|---------|--------|
| Dashboard | <h2>Members Metro Card Info</h2> <p>Your account details are below:</p> <p>Name: Nick Email: nick@gmail.com Phone Number: 8432872371 Card ID: 98738407 Balance: 150</p> <p>Register New user or login existing account</p> <p>amount to add</p> <input type="text" value="Enter the amount to add"/> <input type="button" value="ADD Amount"/> <p>Card ID: 98738407 Balance: 150</p> | | |

Adding Balance: “ Updating Balance” for the member who logged in.

This screenshot is identical to the previous one, but the 'amount to add' input field now contains the value '500'. Consequently, the 'Balance' displayed at the bottom of the account details has been updated from 150 to 135.

| Home | | Profile | Logout |
|-----------|--|---------|--------|
| Dashboard | <h2>Members Metro Card Info</h2> <p>Your account details are below:</p> <p>Name: Nick Email: nick@gmail.com Phone Number: 8432872371 Card ID: 98738407 Balance: 135</p> <p>Register New user or login existing account</p> <p>amount to add</p> <input type="text" value="500"/> <input type="button" value="ADD Amount"/> <p>Card ID: 98738407 Balance: 135</p> | | |

The balance is updated accordingly in the below image and also in database:

[Home](#)[Profile](#)[Logout](#)

[Dashboard](#)[Payment Status](#)[Current Bookings](#)[Members](#)[Logout and go to Main](#)[Reports \(Compliments\)](#)

Members Metro Card Info

Your account details are below:

Name: Nick
Email: nick@gmail.com
Phone Number: 8432872371
Card ID: 98738407
Balance: 635

Register New user or login existing account

amount to add

ADD Amount

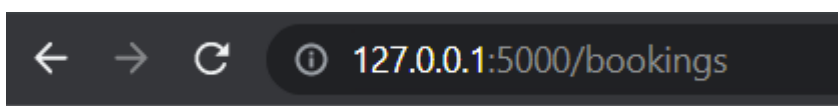
Card ID: 98738407
Balance: 635

“Bookings” of Members:

Here one can see there previous bookings or current booking if they did now and will be updated with the new booking status:

Here Member can see which metro one has to enter, waiting time for the metro to arrive at the station the member is boarding, start time, end time of the metro journey.

The fare for the current booking will be automatically deducted from the members metrocard balance.



Home
Profile
Logout

Dashboard
Payment Status
Current Bookings
Members
Logout and go to Main
Reports (Compliments)

Your current booking was, st_02 to st_10 !
Your current booking was from, FROM STATION NAME = st_02!
Your current booking times was 22:50 to 23:05!
Your current booking waiting time = 0:23:00!
Your in line ID LBN1120!
Your enter in metro ID = 384951!
Your current metro card balance = 135!

Register New user or login existing account
Start
Choose station 1 : st_01
Destination
Choose station 2 : st_01
Gender
Please select your Gender: ☐ Male
☐ Female

You have successfully booked!
Book

Updated balance Metrocard Table:

| Query Editor | Query History | Scratch Pad |
|---|------------------------|------------------------|
| <pre> 1 SELECT * FROM public.metro_card 2 ORDER BY card_id ASC </pre> | | |
| Data Output | Explain | Messages |
| Notifications | | |
| card_id | balance | start |
| [PK] integer | integer | character varying (30) |
| destination | s_time | d_time |
| character varying (30) | character varying (30) | character varying (30) |
| email_id | | |
| character varying (30) | | |
| 17 | 98738407 | 635 |
| st_02 | st_10 | 22:50 |
| 23:05 | nick@gmail.com | |

“Current payment status”:

Here the Member can see their payment status:

Home
Profile
Logout

Dashboard
Payment Status
Current Bookings
Members
Logout and go to Main
Reports (Compliments)

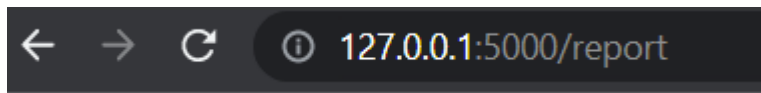
Dashboard

Welcome back, Nick!

Your Payment for current booking was, ['paid']!

“Members can Report a problem” using Report Complaints page:

By writing Subject and Content of the report:

A screenshot of a web application's 'Report Page'. The page has a dark header with 'Home', 'Profile', and 'Logout' links. A left sidebar contains a list of links: 'Dashboard', 'Payment Status', 'Current Bookings', 'Members', 'Logout and go to Main', and 'Reports (Compliments)'. The main content area is titled 'Report Page' and shows account details for 'Nick' (Male, nick@gmail.com). A 'report here' modal is open, containing two text input fields labeled 'Subject of Report' and 'Report Content', and a blue 'report' button at the bottom.

HomeProfileLogout

DashboardPayment StatusCurrent BookingsMembersLogout and go to MainReports (Compliments)

Report Page

Your account details are below:

Name: Nick
Gender: Male
Email: nick@gmail.com

report here

Subject of Report

Hey write your subject here

Report Content

This is what I want to say

report

HomeProfileLogout

DashboardPayment StatusCurrent BookingsMembersLogout and go to MainReports (Compliments)

Report Page

Your account details are below:

Name: Nick
Gender: Male
Email: nick@gmail.com

report here

Subject of Report

Enter your report subject

Report Content

Enter your Report message

You have successfully reported!

report

Reports Table:

12

SELECT * FROM public.report

| | Data Output | Explain | Messages | Notifications |
|----|---|---------|--|---|
| | <div>report_subject</div> <div>character varying (50)</div> | | <div>report</div> <div>character varying (300)</div> | <div>email_id</div> <div>character varying (30)</div> |
| 10 | Hey write your subject here | | This is what I want to say | nick@gmail.com |

> Business expansion can lead to lots of schema and constraint changes in the Database.

For example if we want to give offers for certain users then we have to create a separate table and record the validity of the offer for the specific use. Assuming we need to restrict the number of customers who are taking the rooms to the customers who are having age more prominent than 18 then we want to add the requirement in the data set so customers with age under 18 can't save the rooms.

Database migration :

If the current database should be migrated to an alternative one, we choose mongoDB. A modern, document-based database such as MongoDB can be a great choice over an RDBMS like PostgreSQL.

Because,

It considers the schema changes that would be best for your data, while keeping in mind MongoDB schema best practices and avoiding anti-patterns.

Querying data and their associated data objects is often faster in MongoDB than in SQL, which uses expensive JOIN statements.

MongoDB distributed, scale-out architecture allows your database to grow as your application grows. This is in contrast to PostgreSQL, which is not natively distributed.

MongoDB's flexible schema enables changes to an application's data model or schema to be deployed quickly and flexibly, without the need for migrations or to update query statements in legacy code.

Contributions:

All members have contributed equally in all aspects , but we individually worked more on some aspects parallely

Yamajala Siddhardha - Creation of database , insertion of values , Queries , linking the database with front - end ,helped in creation of front - end

Y Nikhil Bharadwaj - E-R diagram , Granting permissions to different users in our database , Queries , Creation of some HTML pages,helped in creation of front-end

V V Pradeep Reddy - Creation of database , insertion of values , linking the database with the front end , Creation of HTML pages + CSS , creation of front - end

Time spent on each aspect :

- 1) E-R diagram + conversion to relational schema (5-6 hours)
- 2) Creation of database + insertion of values (4-5 hours)
- 3) Granting permission and Queries (4-5 hours)
- 4) front - end (took about a week)