DBMS REPORT

Team Members:

Pushparaj Shetty K S

PES1UG21CS460

PREETHAM C

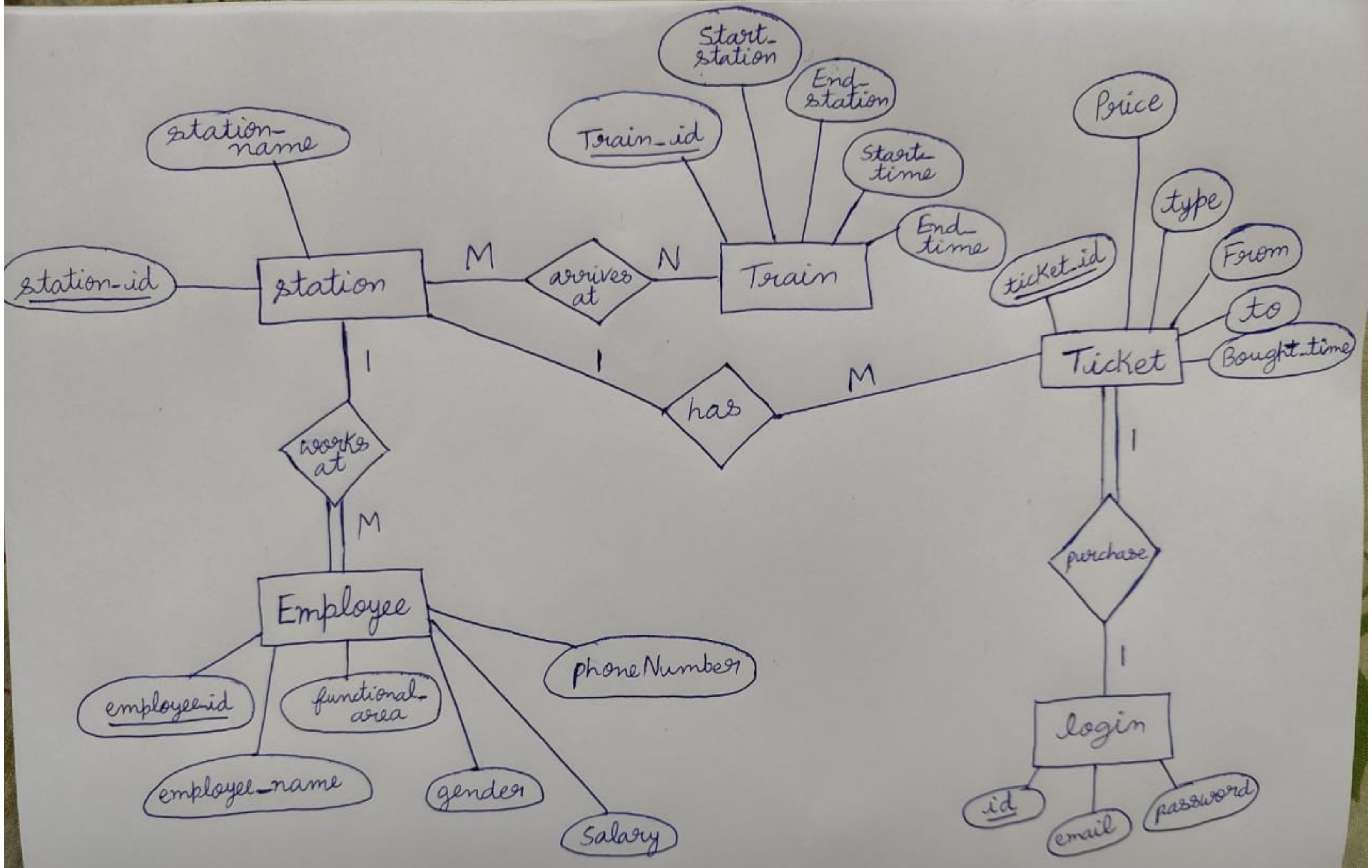
PES1UG21CS447

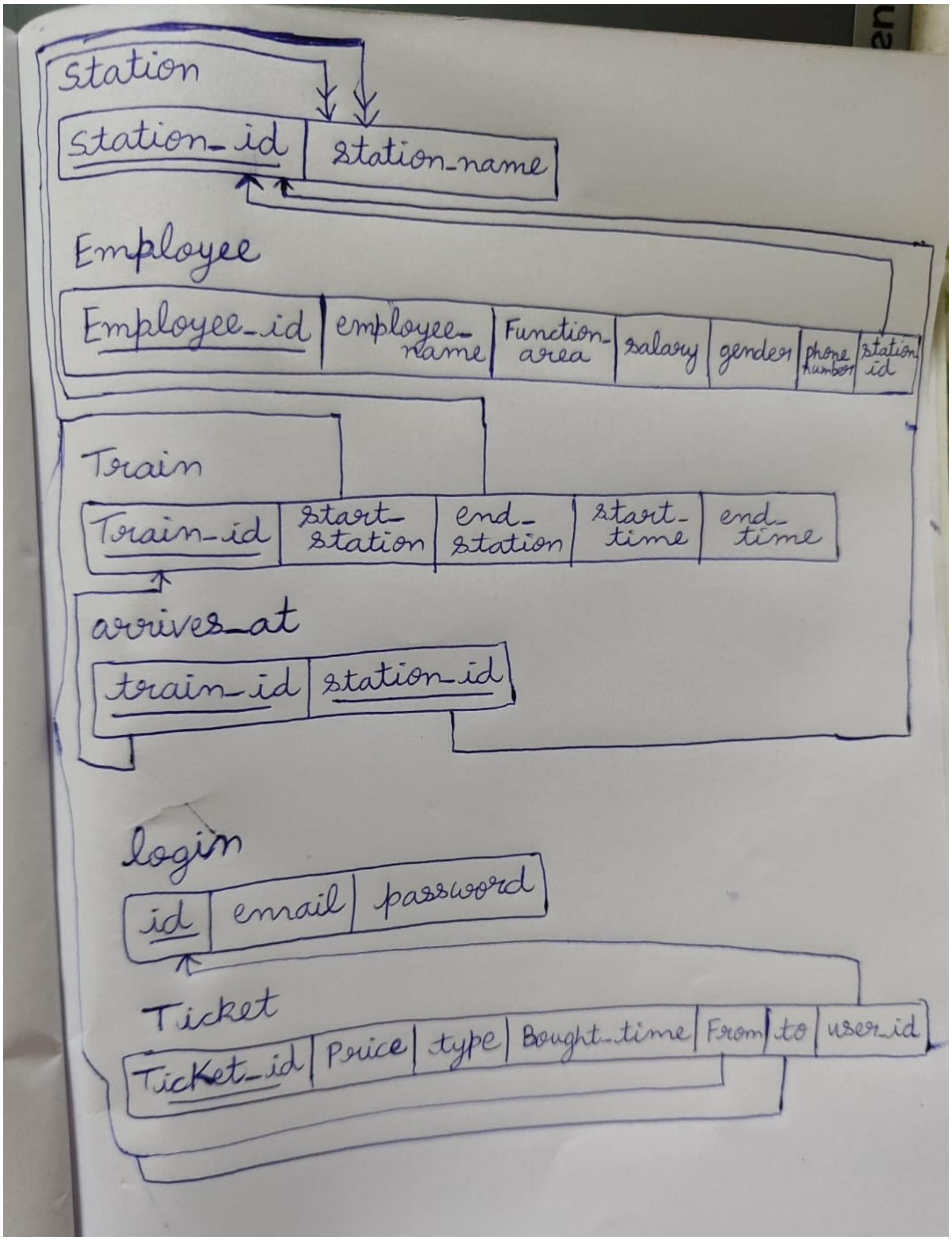
PROJECT TITLE: METRO MANAGEMENT SYSTEM

GITHUB LINK: https://github.com/PushparajShetty/dbms

Abstract:

The Metro Management System, built using React for the frontend, Node.js for the backend, and MySQL for the database, offers an integrated solution with distinct Admin and User modules. Administrators benefit from tools to monitor schedules, and manage station,train,emplpyees and user accounts. Users enjoy a user-friendly interface for route planning, ticket purchase, and real-time updates. The system aims to optimize metro operations, ensuring efficiency and a seamless experience for both administrators and commuters.


station_id
station_name
station
M
arrives at
N
Train
Train_id
Start_station
End_station
Start_time
End_time
1
1
has
M
works at
M
Employee
employee_id
employee_name
functional_area
gender
Salary
phoneNumber
Ticket
ticket_id
Price
type
From
to
Bought_time
1
purchase
1
login
id
email
password



Station

Station_id	station_name

Employee

Employee_id	employee-name	Function-area	salary	gender	phone number	station id

Train

Train_id	start-station	end-station	start-time	end-time

arrives_at

train_id	station_id

login

id	email	password

Ticket

Ticket_id	Price	type	Bought_time	From	to	user_id

DDL SQL COMMANDS:

```
create table station(station_id int primary key auto_increment,station_name varchar(40) unique);
```

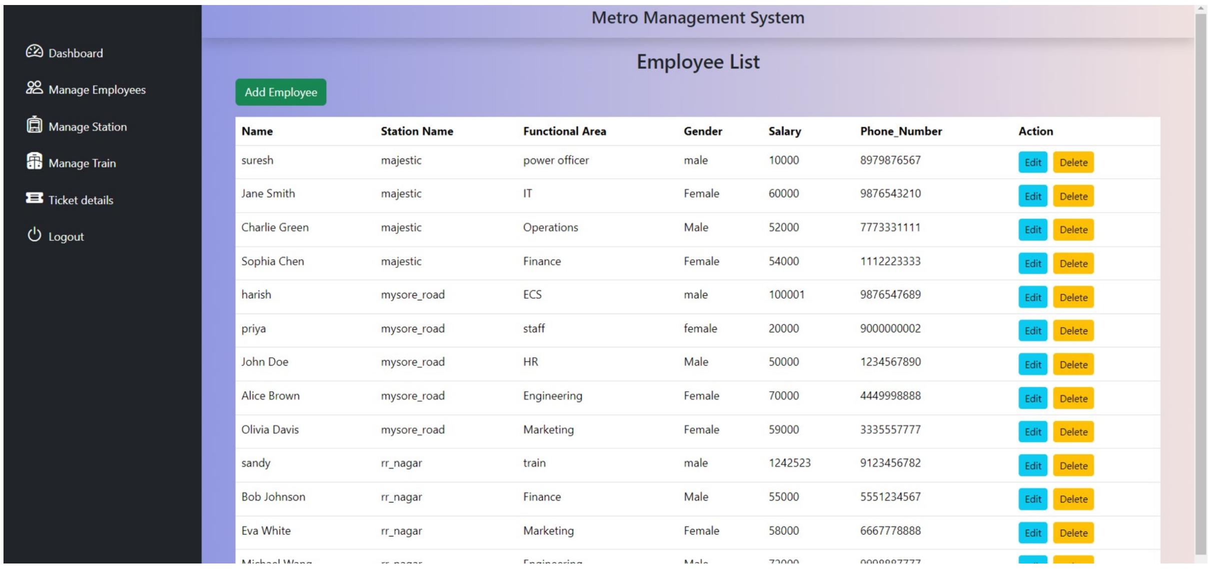
```
create table employee(employee_id int primary key auto_increment,employee_name varchar(40),functional_area varchar(40),gender varchar(10),salary int,station_id int,phoneNumber int,foreign key (station_id) references station(station_id) on delete cascade);
```

```
create table train(train_id int primary key auto_increment,start_station varchar(40),end_station varchar(40),start_time time,end_time time,foreign key (start_station) references station(station_name),foreign key (end_station) references station(station_name));
```

```
create table arrives_at(train_id int,station_id int,primary key(train_id,station_id),foreign key (station_id) references station(station_id),foreign key (train_id) references train(train_id));
```

```
create table login(id int primary key auto_increment,email varchar(40),password varchar(40));
```

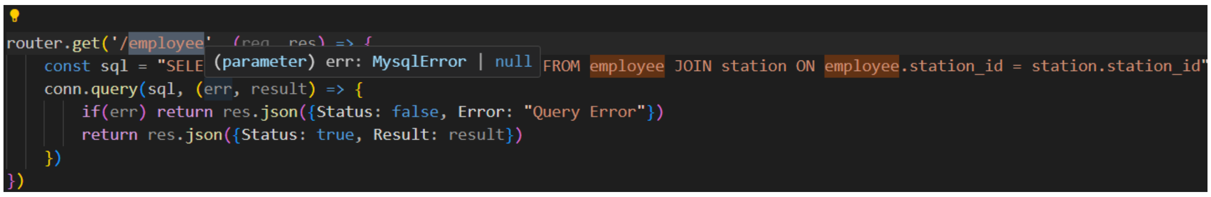
```
create table ticket(ticket_id int primary key auto_increment,price int,type varchar(20),bought_time time,from varchar(40),to varchar(40),user_id int,foreign key (user_id) references login(id),foreign key (from) references station(station_name),foreign key (to) references station(station_name));
```



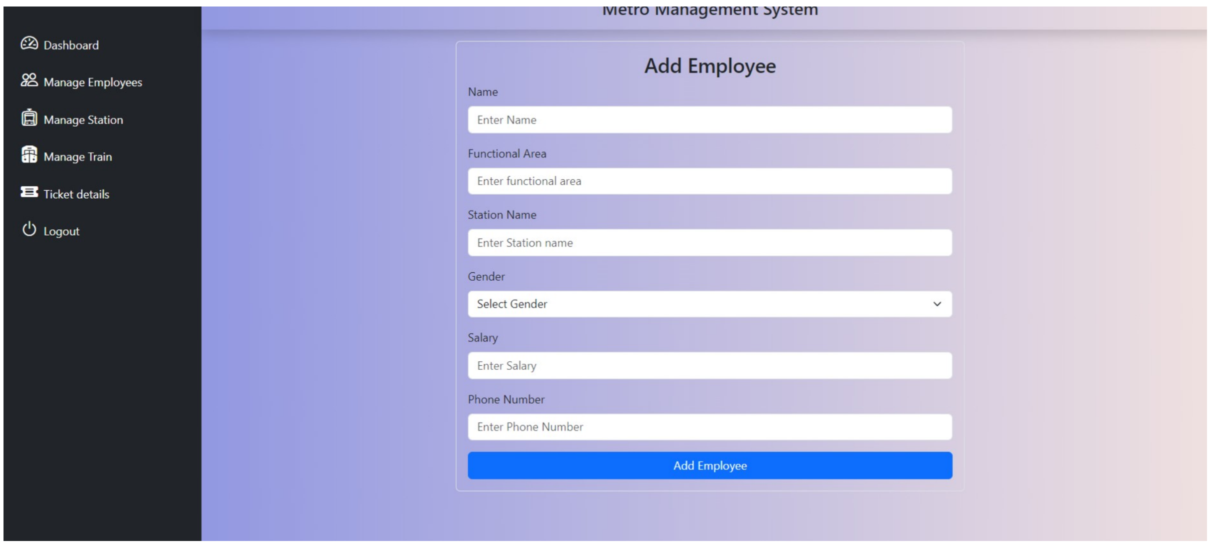
Metro Management System

Employee List

Name	Station Name	Functional Area	Gender	Salary	Phone_Number	Action
suresh	majestic	power officer	male	10000	8979876567	Edit Delete
Jane Smith	majestic	IT	Female	60000	9876543210	Edit Delete
Charlie Green	majestic	Operations	Male	52000	7773331111	Edit Delete
Sophia Chen	majestic	Finance	Female	54000	1112223333	Edit Delete
harish	mysore_road	ECS	male	100001	9876547689	Edit Delete
priya	mysore_road	staff	female	20000	9000000002	Edit Delete
John Doe	mysore_road	HR	Male	50000	1234567890	Edit Delete
Alice Brown	mysore_road	Engineering	Female	70000	4449998888	Edit Delete
Olivia Davis	mysore_road	Marketing	Female	59000	3335557777	Edit Delete
sandy	rr_nagar	train	male	1242523	9123456782	Edit Delete
Bob Johnson	rr_nagar	Finance	Male	55000	5551234567	Edit Delete
Eva White	rr_nagar	Marketing	Female	58000	6667778888	Edit Delete



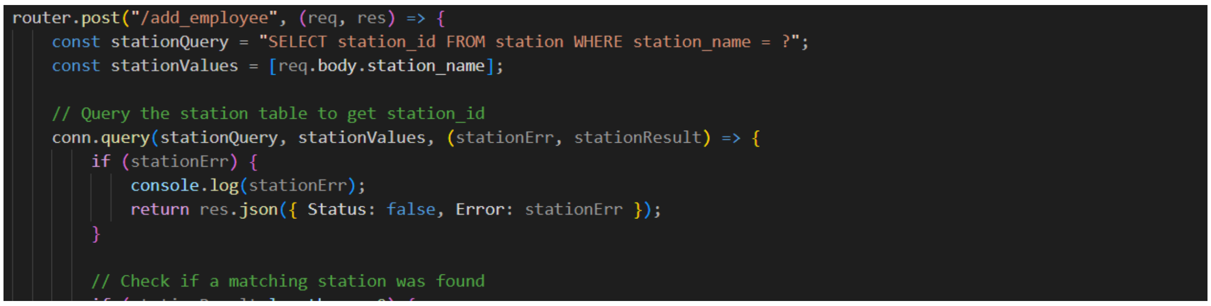
```
router.get('/employee', (req, res) => {
    const sql = "SELE (parameter) err: MysqlError | null FROM employee JOIN station ON employee.station_id = station.station_id"
    conn.query(sql, (err, result) => {
        if(err) return res.json({Status: false, Error: "Query Error"})
        return res.json({Status: true, Result: result})
    })
})
```



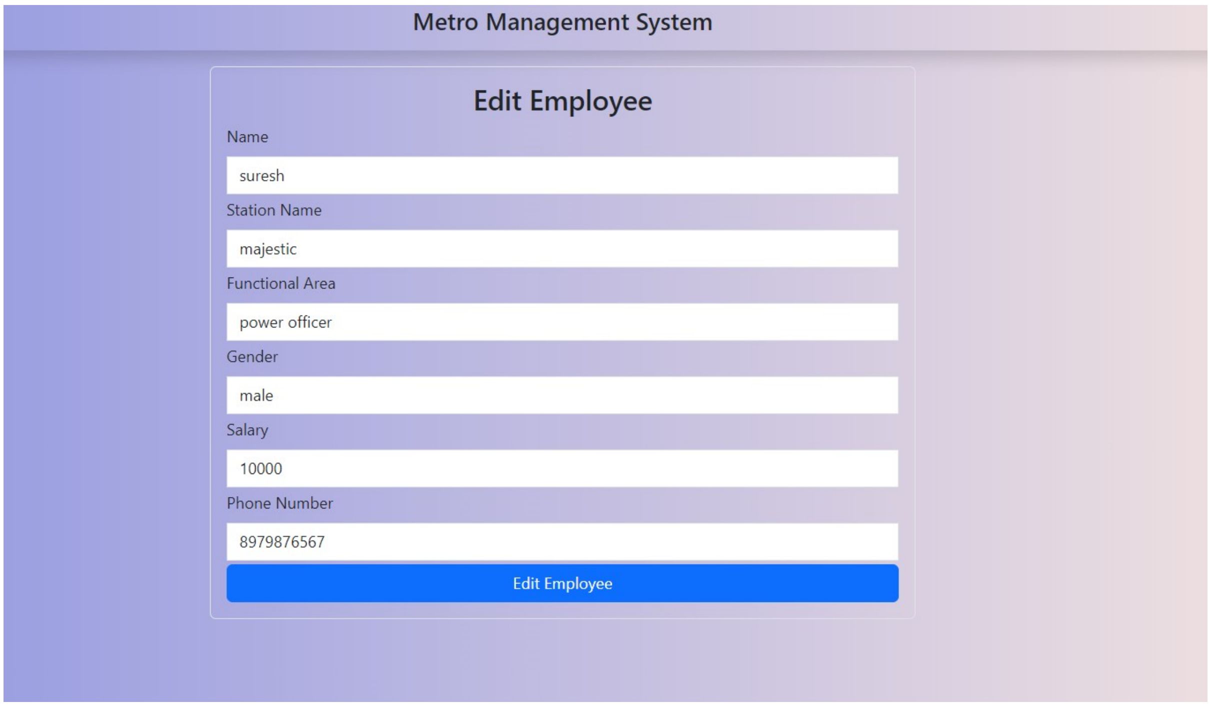
Metro Management System

Add Employee

Name	Enter Name
Functional Area	Enter functional area
Station Name	Enter Station name
Gender	Select Gender
Salary	Enter Salary
Phone Number	Enter Phone Number

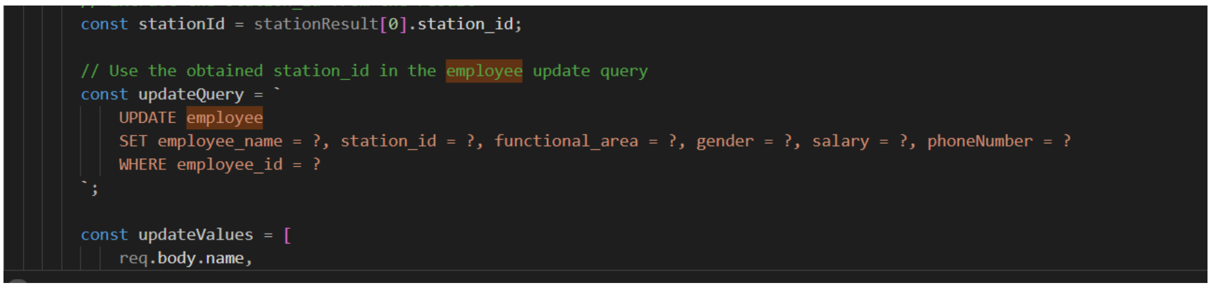


```
router.post("/add_employee", (req, res) => {
    const stationQuery = "SELECT station_id FROM station WHERE station_name = ?";
    const stationValues = [req.body.station_name];

    // Query the station table to get station_id
    conn.query(stationQuery, stationValues, (stationErr, stationResult) => {
        if (stationErr) {
            console.log(stationErr);
            return res.json({ Status: false, Error: stationErr });
        }

        // Check if a matching station was found
```

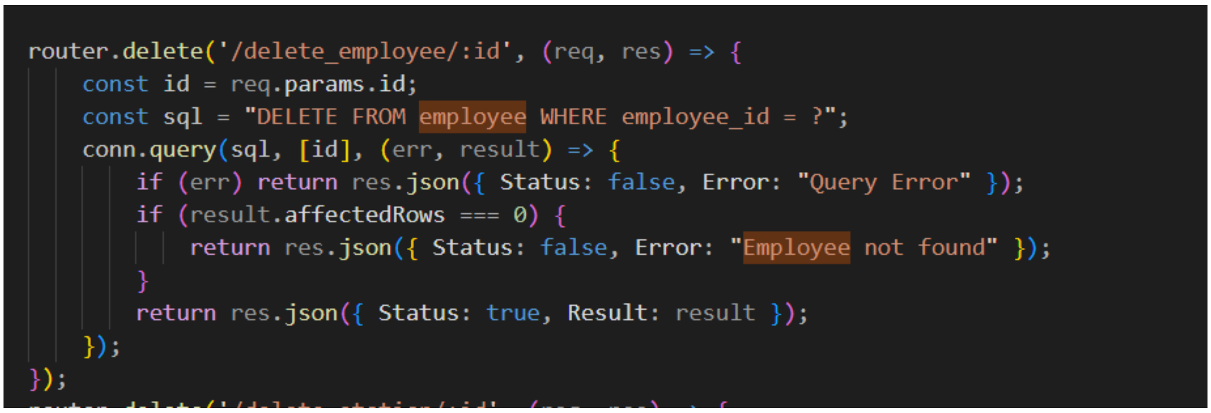


Edit Employee

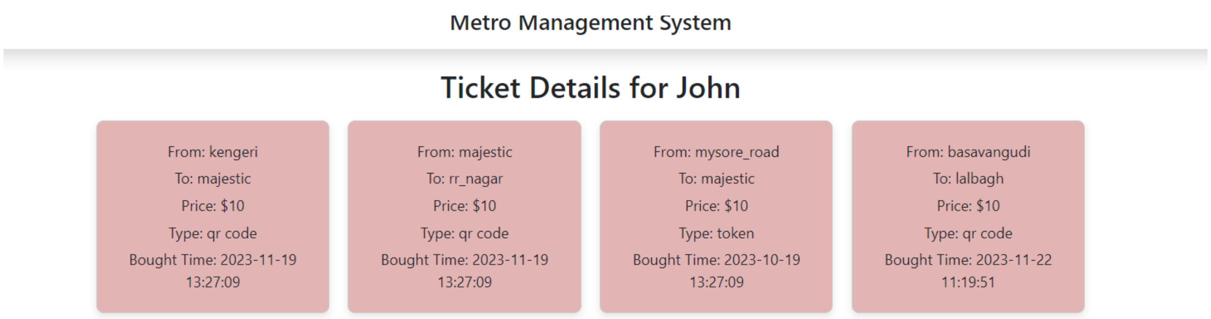
Name	suresh
Station Name	majestic
Functional Area	power officer
Gender	male
Salary	10000
Phone Number	8979876567



```
const stationId = stationResult[0].station_id;

// Use the obtained station_id in the employee update query
const updateQuery = `
    UPDATE employee
    SET employee_name = ?, station_id = ?, functional_area = ?, gender = ?, salary = ?, phoneNumber = ?
    WHERE employee_id = ?
`;

const updateValues = [
    req.body.name,
```



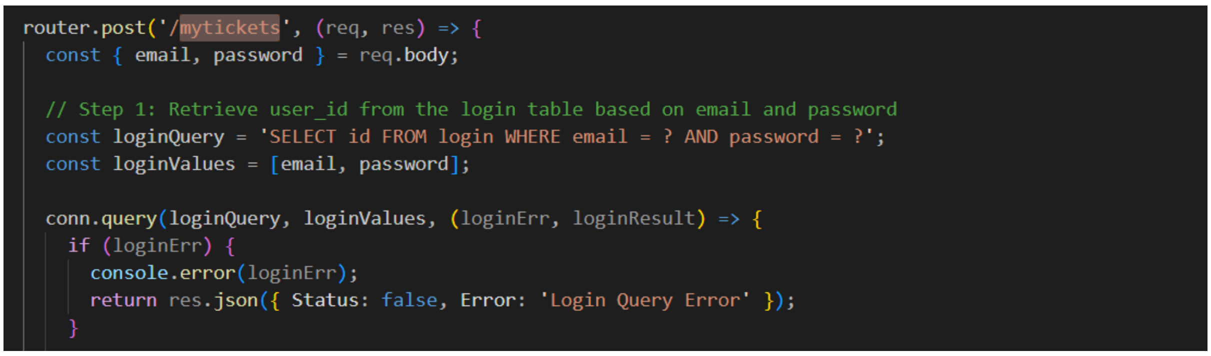
```
router.delete('/delete_employee/:id', (req, res) => {
    const id = req.params.id;
    const sql = "DELETE FROM employee WHERE employee_id = ?";
    conn.query(sql, [id], (err, result) => {
        if (err) return res.json({ Status: false, Error: "Query Error" });
        if (result.affectedRows === 0) {
            return res.json({ Status: false, Error: "Employee not found" });
        }
        return res.json({ Status: true, Result: result });
    });
});
router.delete('/delete_station/:id', (req, res) => {
```



Metro Management System

Ticket Details for John

From	To	Price	Type	Bought Time
kengeri	majestic	$10	qr code	2023-11-19 13:27:09
majestic	rr_nagar	$10	qr code	2023-11-19 13:27:09
mysore_road	majestic	$10	token	2023-10-19 13:27:09
basavangudi	lalbagh	$10	qr code	2023-11-22 11:19:51



```
router.post('/mytickets', (req, res) => {
  const { email, password } = req.body;

  // Step 1: Retrieve user_id from the login table based on email and password
  const loginQuery = 'SELECT id FROM login WHERE email = ? AND password = ?';
  const loginValues = [email, password];

  conn.query(loginQuery, loginValues, (loginErr, loginResult) => {
    if (loginErr) {
      console.error(loginErr);
      return res.json({ Status: false, Error: 'Login Query Error' });
    }
```

1)delimiter $$

create function revenue_from_user_specific_ticket_type(userid int,type_tic varchar(20))

returns int deterministic

begin

declare revenue int;

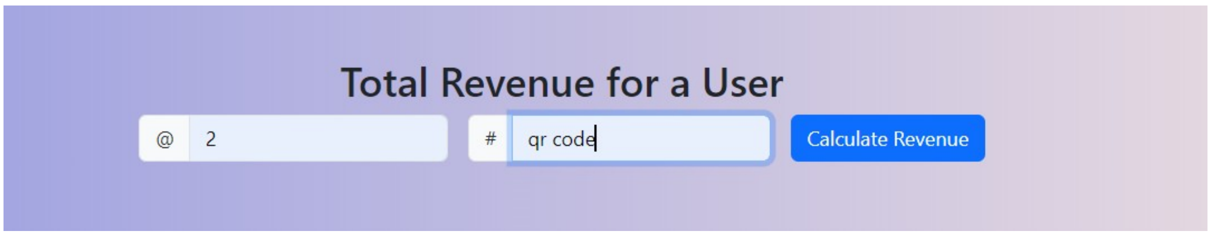
select sum(price) into revenue from (select user_id,price from ticket where type=type_tic) as o where user_id=userid;

return revenue;

end$$

delimiter ;

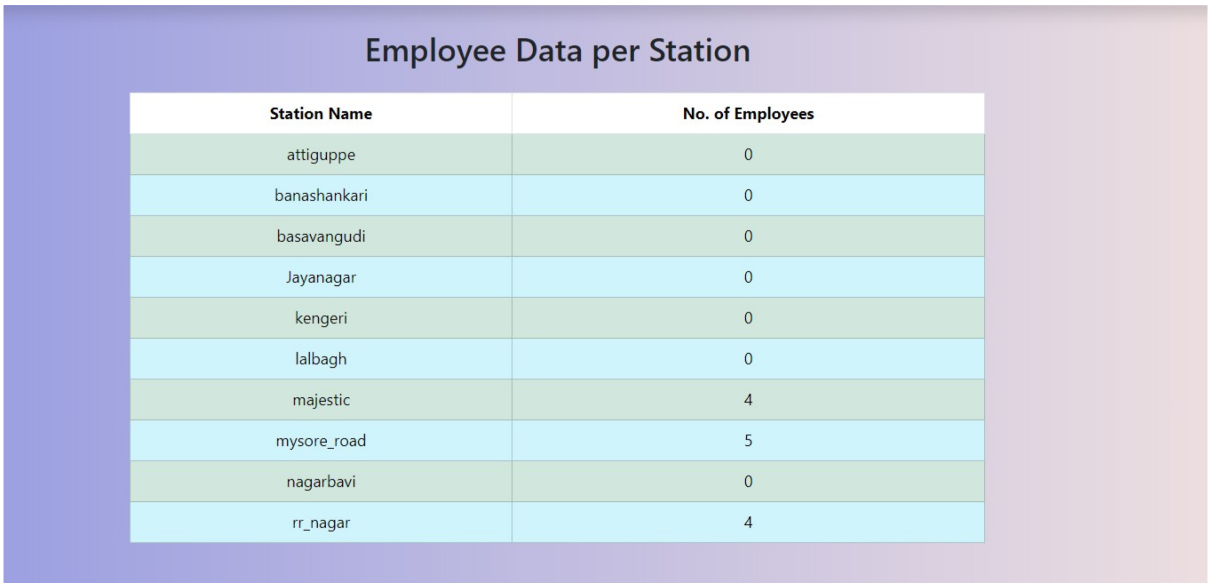
select revenue_from_user_specific_ticket_type(,'');



Total Revenue for a User

@	2
#	qr code

```
2)delimiter //

create procedure no_of_employee_per_station()

begin

select station.station_name,count(distinct employee_id) as no_of_employee from station left join
employee on station.station_id=employee.station_id group by station_name;

end//

delimiter ;

call no_of_employee_per_station();
```



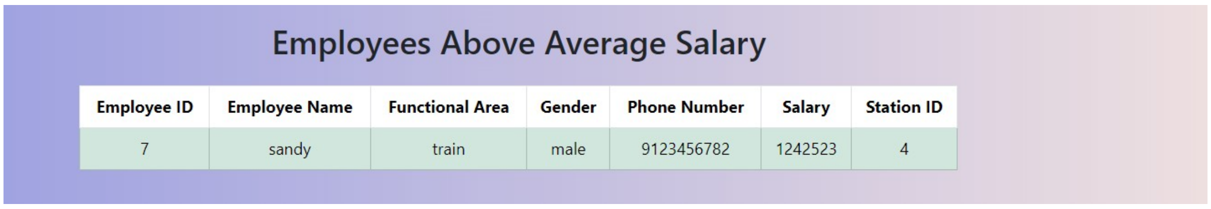
Employee Data per Station

Station Name	No. of Employees
attiguppe	0
banashankari	0
basavangudi	0
Jayanagar	0
kengeri	0
lalbagh	0
majestic	4
mysore_road	5
nagarbavi	0
rr_nagar	4

3)

```
delimiter //

create procedure employee_above_avg_salary()

begin

select * from employee where salary > (select avg(salary) from employee where
employee_id=employee.employee_id);

end//

delimiter ;
```

call employee_above_avg_salary();



Employees Above Average Salary

Employee ID	Employee Name	Functional Area	Gender	Phone Number	Salary	Station ID
7	sandy	train	male	9123456782	1242523	4

4)

delimiter $$

create function no_of_tickets_bought_at_a_station(stationname varchar(20))

returns int deterministic

begin

declare no int;

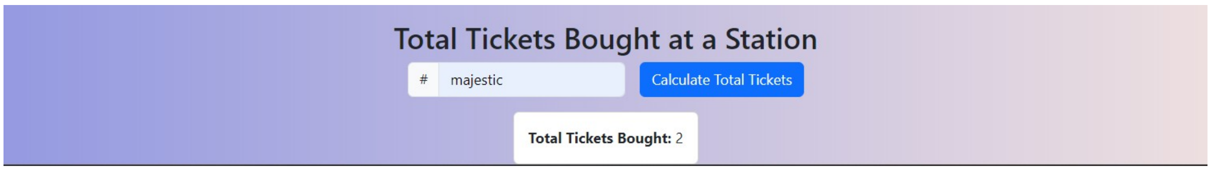
select count(ticket_id) into no from ticket t where exists (select 1 from station where t.station_name=station.station_name and t.station_name=stationname);

return no;

end$$

delimiter ;

select no_of_tickets_bought_at_a_station('');



Total Tickets Bought at a Station

Total Tickets Bought: 2

5)

delimiter //

create trigger arrival_station

```
after insert
on train for each row
begin
declare a int;
declare b int;
select station_id into a from station where station_name=new.start_station;
select station_id into b from station where station_name=new.end_station;
insert into arrives_at values (new.train_id,a);
insert into arrives_at values (new.train_id,b);
end;
//
delimiter ;
```

6)

```
delimiter //
create trigger train_delete
before delete
on train for each row
begin
delete from arrives_at where train_id=old.train_id;
end;
//
delimiter ;
```

7)

```
delimiter //
create trigger station_delete
before delete
on station for each row
```

```
begin

delete from train where start_station=old.station_name or end_station=old.station_name

end;

//

delimiter ;
```