***Programme name = Btech (ICT)***

Semester: 4nd semester

Course Name: Database and management Systems

Project Title: Hotel Management

Group Member 1 – **1741022**

**----Mudra Suthar**

Group Member 2 – 1741092

----Hardi Trivedi

2)

Description of Project::  
Hotel Project ::

1. As the name recommends, the Online Hotel Management System permits the clients globally to reserve the rooms in the hotel at any time. It will provide a unique search and book functionally for the customers anywhere in the world. This project can be used by the clients and the travel agencies who will help them to reserve the hotel of their choice. Sometimes people face problem in reserving the hotels in the places where they want to visit. In that situation, this project can be of great relief to them. The user interface must be simple and easy to understand even by the common man.

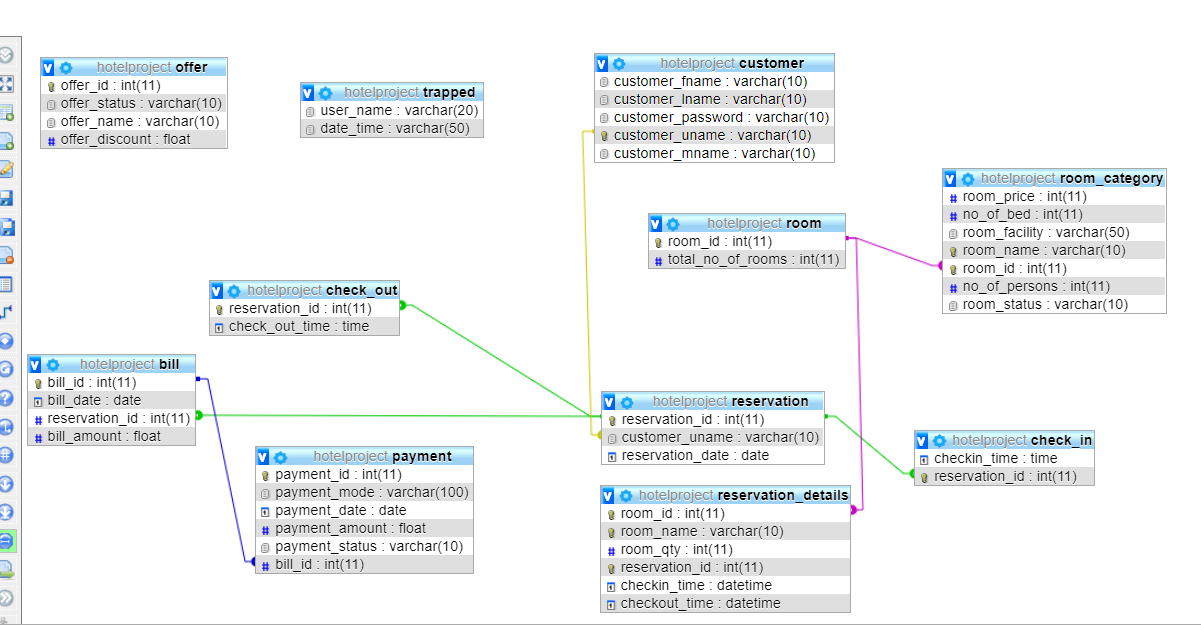
The admin has the right to update or add the hotel or the room information. The clients can cancel the already booked room in the particular hotel on a particular day. This will be one of the exciting projects that one can implement in real time world. The highlights that can be added to the online hotel management system are given below:

* **Access anywhere**: The online hotel reservation project can be reached at any time and anywhere from the world with great efficiency.
* **Easy booking**: The rooms in the particular hotel can be reserved easily through this site with great expertise.
* **Hotel description**: This project will contain the hotel information of the hotels that are present in the particular area.
* **Room availability**: The rooms that are open in the particular hotel must be refreshed in real time.
* **Category**: The hotels can be classified based on the appropriate region of your preference.
* **Price**: The price or rent of the appropriate rooms in the appropriate hotel in the chosen section will also be updated.
* **Payment options**: The clients will be having the option of paying the rent of the hotels through the online mode without any difficulty.

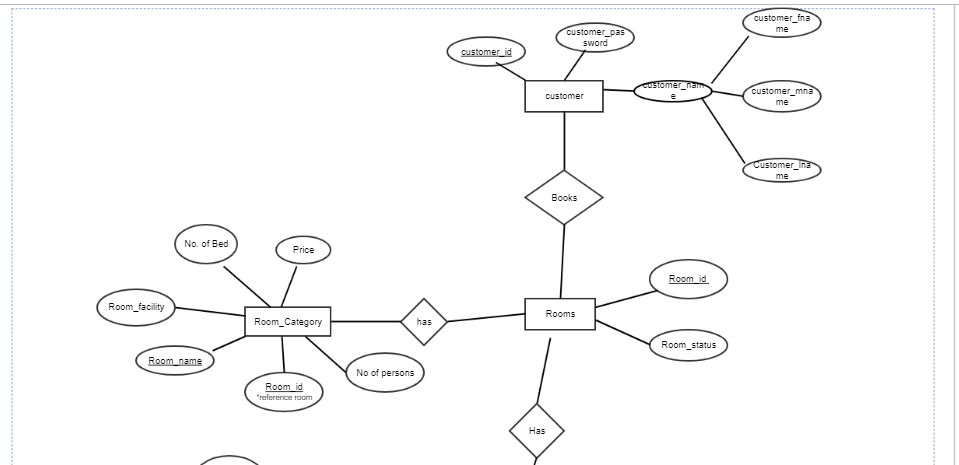
System Requirement specifications :: Xamp server (Localhost phpmyadmin) for accessing mysql at backend. (64 bit xampp application)

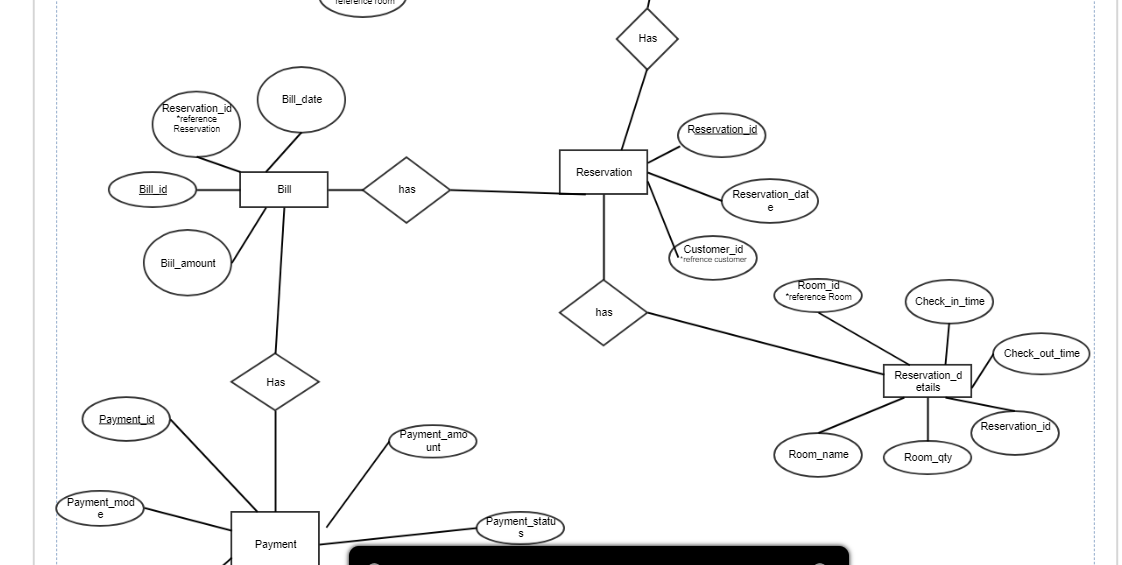
Front end :: php ,html ,css text editor = sublime

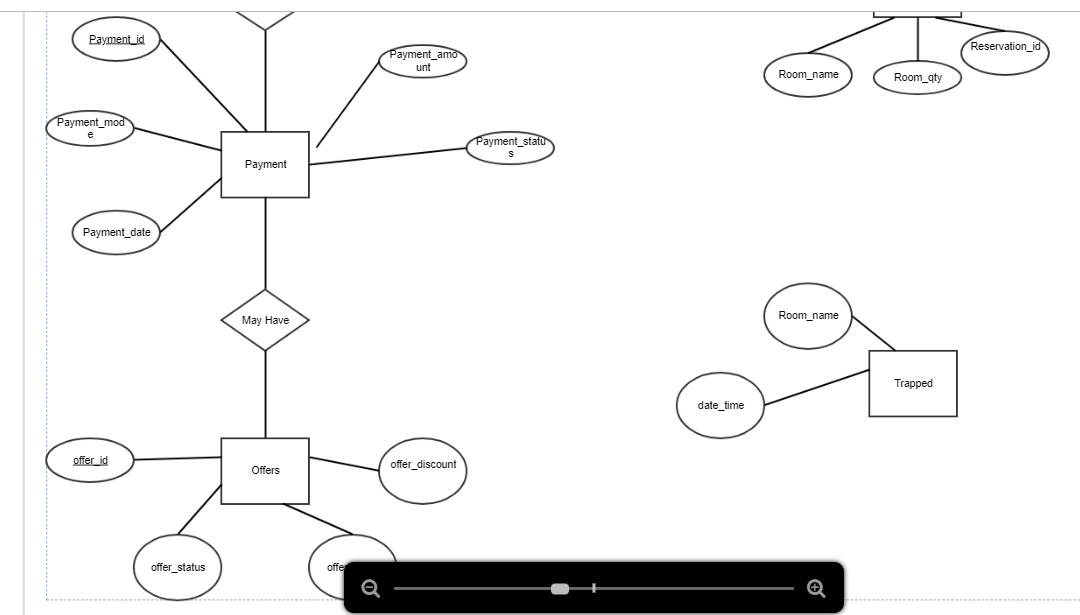
b)Entity Relationship Diagram(Relational schema)



ER diagram below is shown in three parts::







**DATA dictionary**

**bill**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Links to** | **Comments** | **MIME** |
| bill\_id *(Primary)* | int(11) | No |  |  |  |  |
| bill\_date | Date | No |  |  |  |  |
| reservation\_id | int(11) | No |  | reservation -> reservation\_id |  |  |
| bill\_amount | Float | No |  |  |  |  |

**Indexes**

| **Keyname** | **Type** | **Unique** | **Packed** | **Column** | **Cardinality** | **Collation** | **Null** | **Comment** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PRIMARY | BTREE | Yes | No | bill\_id | 4 | A | No |  |
| const\_res | BTREE | No | No | reservation\_id | 4 | A | No |  |

**check\_in**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Links to** | **Comments** | **MIME** |
| checkin\_time | time | No |  |  |  |  |
| reservation\_id | int(11) | No |  | reservation -> reservation\_id |  |  |

**Indexes**

| **Keyname** | **Type** | **Unique** | **Packed** | **Column** | **Cardinality** | **Collation** | **Null** | **Comment** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| rese | BTREE | Yes | No | reservation\_id | 1 | A | No |  |

**check\_out**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Links to** | **Comments** | **MIME** |
| reservation\_id | int(11) | No |  | reservation -> reservation\_id |  |  |
| check\_out\_time | time | No |  |  |  |  |

**Indexes**

| **Keyname** | **Type** | **Unique** | **Packed** | **Column** | **Cardinality** | **Collation** | **Null** | **Comment** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ressss | BTREE | Yes | No | reservation\_id | 1 | A | No |  |

**customer**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Links to** | **Comments** | **MIME** |
| customer\_fname | varchar(10) | No |  |  |  |  |
| customer\_lname | varchar(10) | No |  |  |  |  |
| customer\_password | varchar(10) | No |  |  |  |  |
| customer\_uname *(Primary)* | varchar(10) | No |  |  |  |  |
| customer\_mname | varchar(10) | No |  |  |  |  |

**Indexes**

| **Keyname** | **Type** | **Unique** | **Packed** | **Column** | **Cardinality** | **Collation** | **Null** | **Comment** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PRIMARY | BTREE | Yes | No | customer\_uname | 4 | A | No |  |

**offer**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Links to** | **Comments** | **MIME** |
| offer\_id *(Primary)* | int(11) | No |  |  |  |  |
| offer\_status | varchar(10) | No |  |  |  |  |
| offer\_name | varchar(10) | No |  |  |  |  |
| offer\_discount | float | No |  |  |  |  |

**Indexes**

| **Keyname** | **Type** | **Unique** | **Packed** | **Column** | **Cardinality** | **Collation** | **Null** | **Comment** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PRIMARY | BTREE | Yes | No | offer\_id | 4 | A | No |  |

**payment**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Links to** | **Comments** | **MIME** |
| payment\_id *(Primary)* | int(11) | No |  |  |  |  |
| payment\_mode | varchar(100) | No |  |  |  |  |
| payment\_date | date | No |  |  |  |  |
| payment\_amount | float | No |  |  |  |  |
| payment\_status | varchar(10) | No |  |  |  |  |
| bill\_id | int(11) | No |  | bill -> bill\_id |  |  |

**Indexes**

| **Keyname** | **Type** | **Unique** | **Packed** | **Column** | **Cardinality** | **Collation** | **Null** | **Comment** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PRIMARY | BTREE | Yes | No | payment\_id | 4 | A | No |  |
| bill\_index | BTREE | No | No | bill\_id | 4 | A | No |  |

**reservation**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Links to** | **Comments** | **MIME** |
| reservation\_id *(Primary)* | int(11) | No |  |  |  |  |
| customer\_uname | varchar(10) | No |  | customer -> customer\_uname |  |  |
| reservation\_date | date | No |  |  |  |  |

**Indexes**

| **Keyname** | **Type** | **Unique** | **Packed** | **Column** | **Cardinality** | **Collation** | **Null** | **Comment** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PRIMARY | BTREE | Yes | No | reservation\_id | 6 | A | No |  |
| cons\_uname | BTREE | No | No | customer\_uname | 6 | A | No |  |

**reservation\_details**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Links to** | **Comments** | **MIME** |
| room\_id *(Primary)* | int(11) | No |  | room -> room\_id |  |  |
| room\_name *(Primary)* | varchar(10) | No |  |  |  |  |
| room\_qty | int(11) | No |  |  |  |  |
| reservation\_id *(Primary)* | int(11) | No |  |  |  |  |
| checkin\_time | datetime | No |  |  |  |  |
| checkout\_time | datetime | No |  |  |  |  |

**Indexes**

| **Keyname** | **Type** | **Unique** | **Packed** | **Column** | **Cardinality** | **Collation** | **Null** | **Comment** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PRIMARY | BTREE | Yes | No | room\_id | 2 | A | No |  |
| room\_name | 4 | A | No |
| reservation\_id | 4 | A | No |
| const\_room\_name | BTREE | No | No | room\_name | 4 | A | No |  |
| const\_reservation\_id | BTREE | No | No | reservation\_id | 4 | A | No |  |

**room**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Links to** | **Comments** | **MIME** |
| room\_id *(Primary)* | int(11) | No |  |  |  |  |
| total\_no\_of\_rooms | int(11) | No |  |  |  |  |

**Indexes**

| **Keyname** | **Type** | **Unique** | **Packed** | **Column** | **Cardinality** | **Collation** | **Null** | **Comment** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PRIMARY | BTREE | Yes | No | room\_id | 3 | A | No |  |

**room\_category**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Links to** | **Comments** | **MIME** |
| room\_price | int(11) | No |  |  |  |  |
| no\_of\_bed | int(11) | No |  |  |  |  |
| room\_facility | varchar(50) | No |  |  |  |  |
| room\_name *(Primary)* | varchar(10) | No |  |  |  |  |
| room\_id *(Primary)* | int(11) | No |  | room -> room\_id |  |  |
| no\_of\_persons | int(11) | No |  |  |  |  |
| room\_status | varchar(10) | No |  |  |  |  |

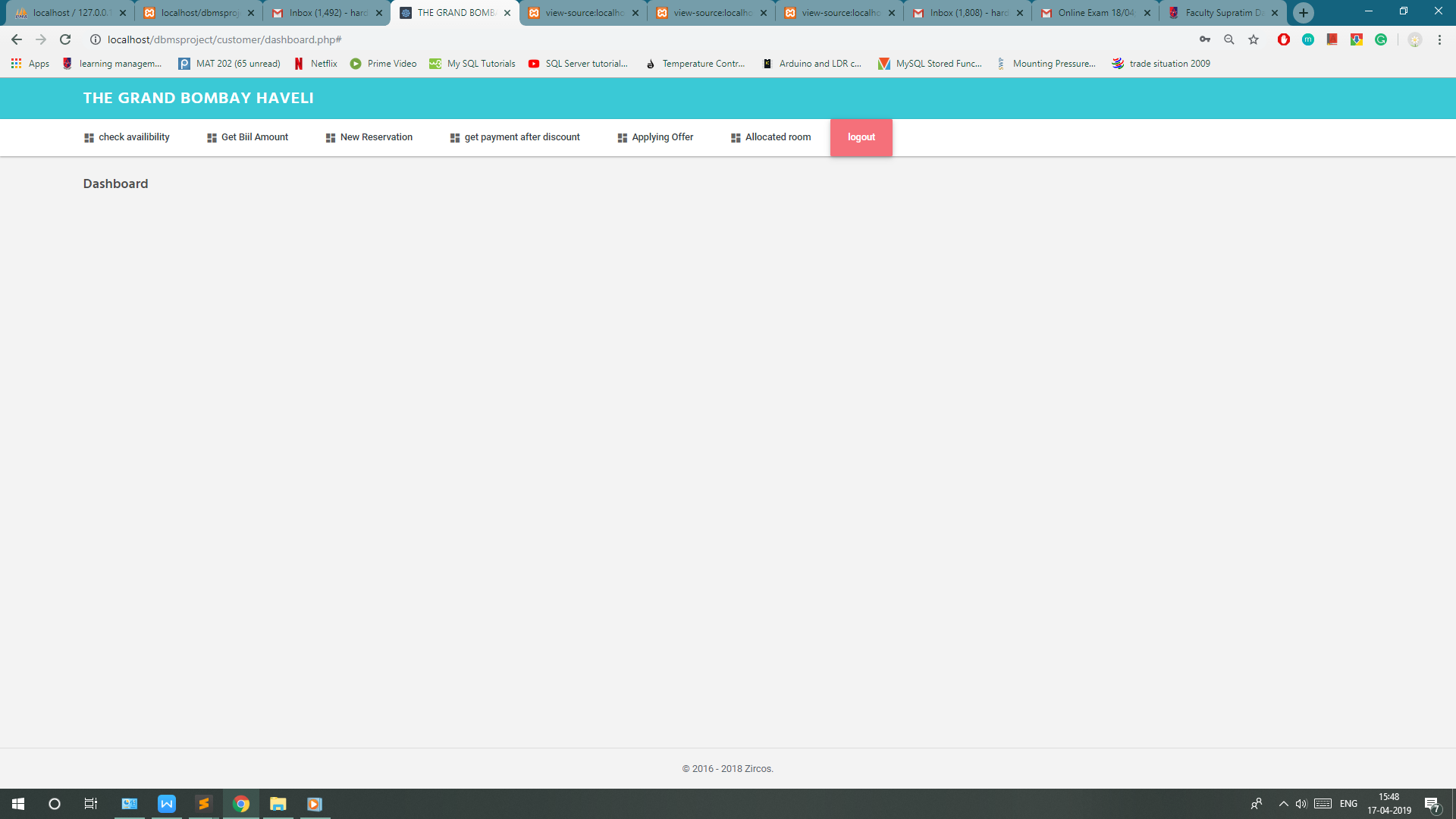
**Indexes**

| **Keyname** | **Type** | **Unique** | **Packed** | **Column** | **Cardinality** | **Collation** | **Null** | **Comment** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PRIMARY | BTREE | Yes | No | room\_name | 3 | A | No |  |
| room\_id | 3 | A | No |
| room\_id | BTREE | No | No | room\_id | 3 | A | No |  |

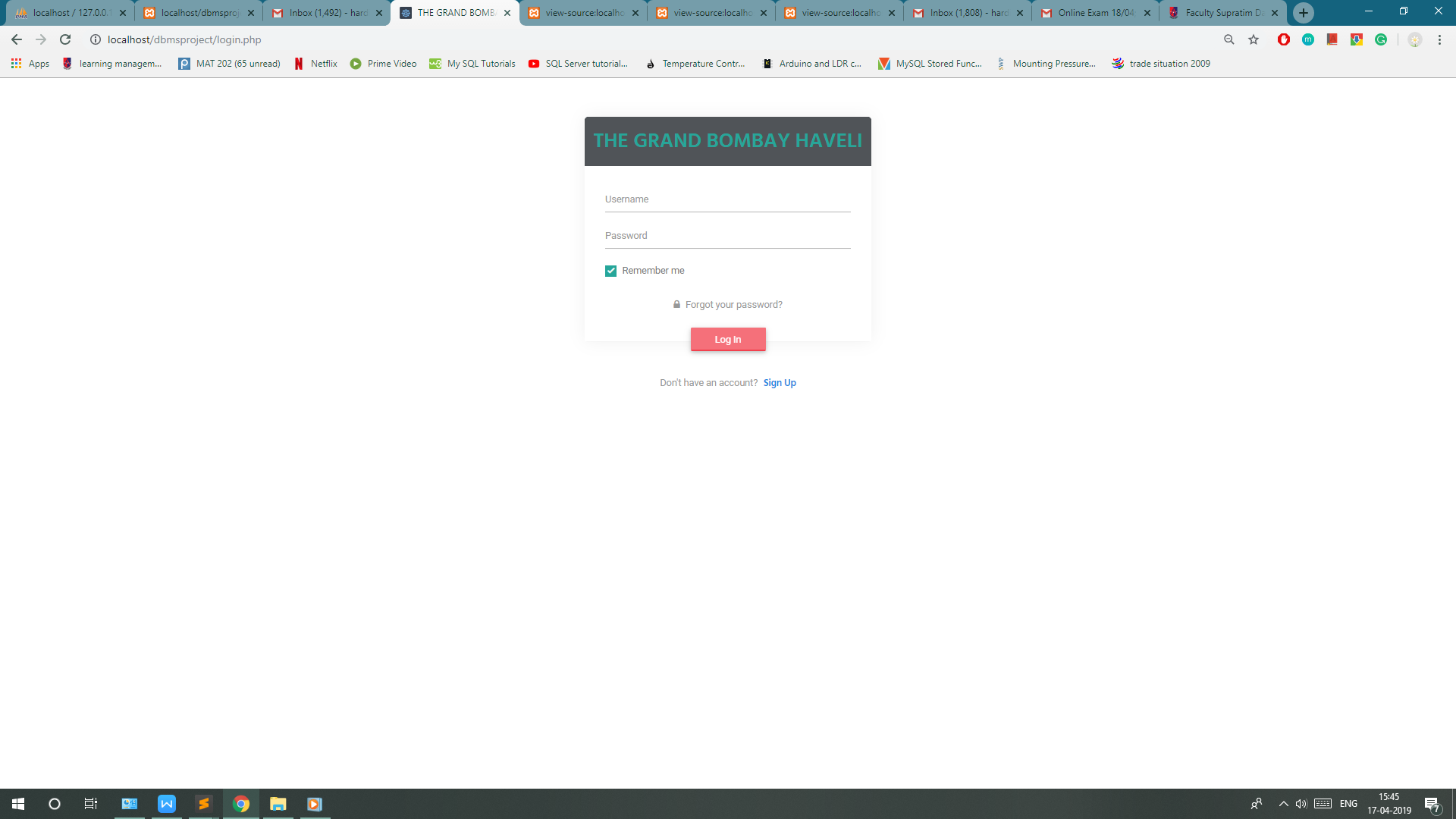
**trapped**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** | **Links to** | **Comments** | **MIME** |
| user\_name | varchar(20) | No |  |  |  |  |
| date\_time | varchar(50) | No |  |  |  |  |

**[Open new phpMyAdmin window](http://localhost/phpmyadmin/db_datadict.php?db=hotelproject&table=trapped&server=1&target=)DASHBOARD**



**Login page**



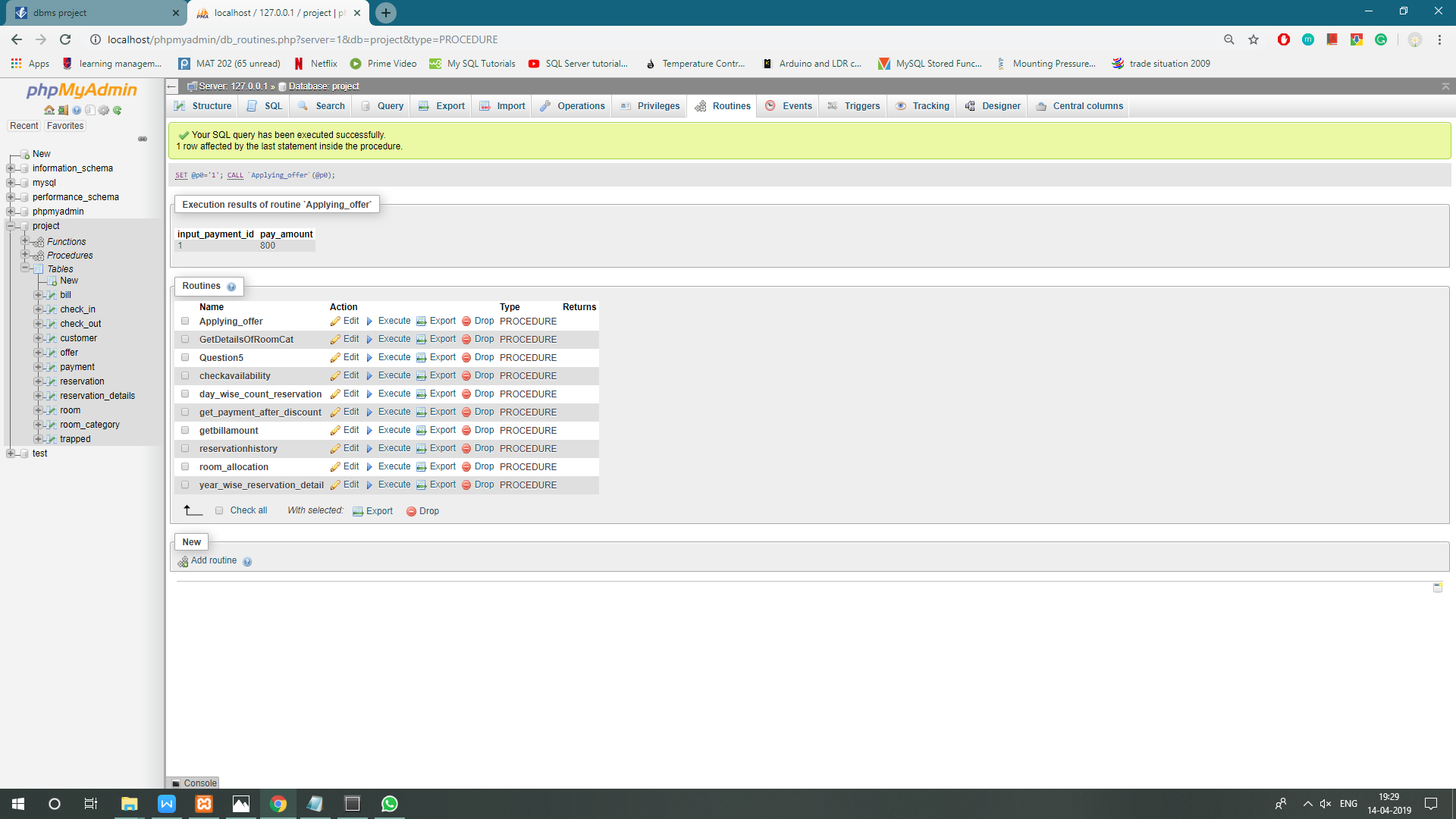
**Stored procedures and triggers (code)**

1)**Applying\_offer** = Calculates payment amount after checking the offer status of any of the offers.If offer status is true and with the help of that offer id accordingly offer discount is calculated and deducted from payment amount.For this it also checks the payment status of payment amount if payment status is unpaid then and then only it will calculate payment amount applying offer.

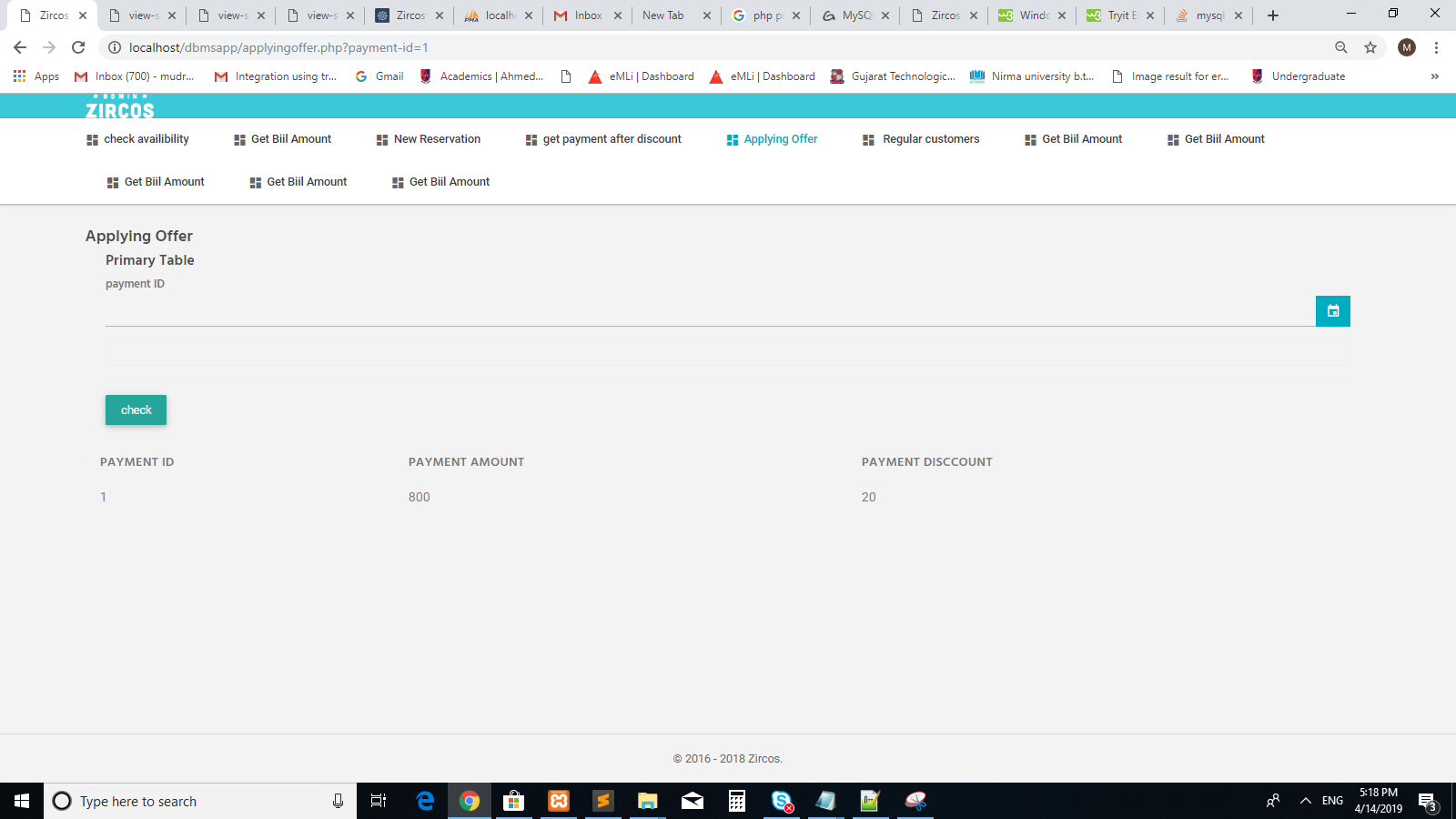
Output :: payment amount(calculated after applying specific discount according to offer)

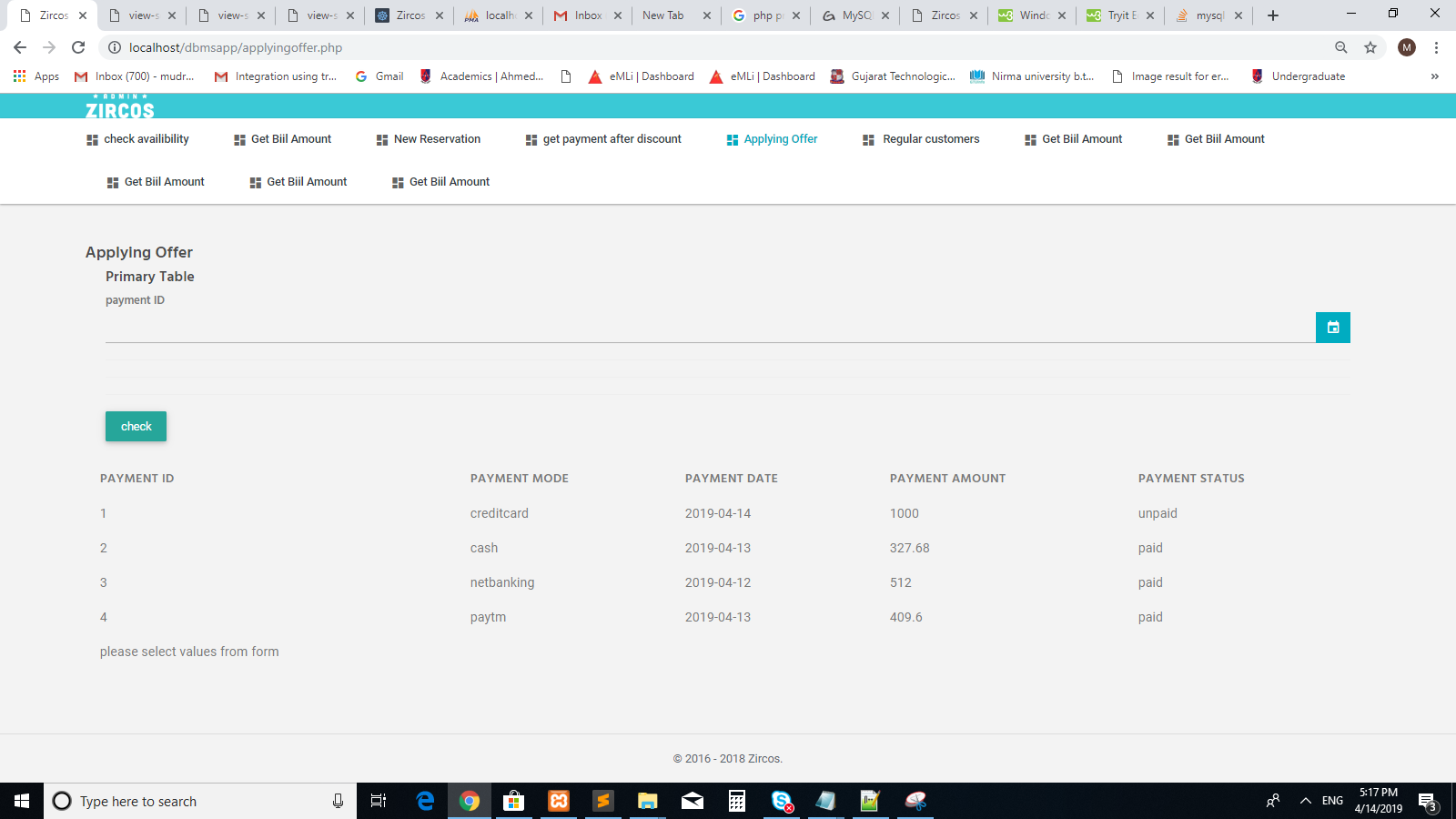
Payment status of payment amount is changed to paid from unpaid.

Screenshots(at backend)



Screenshot(at frontend)





-->Type procedure

Parameters :: in input\_payment int

begin

declare pay\_amount , dis float;

declare pay\_status varchar(10);

declare k ,pay\_id int;

declare cur1 cursor for select offer.offer\_discount , payment.payment\_amount from offer , payment where offer.offer\_status = 'true' and (input\_payment\_id = payment.payment\_id) and payment.payment\_id in (select payment\_id from payment where payment\_status = 'unpaid') ;

declare continue handler for not found set k = 1;

open cur1;

set k = 0;

fetch cur1 into dis, pay\_amount;

while k = 0 do

set pay\_amount = (pay\_amount - ((dis\*pay\_amount)/100)) ;

select input\_payment\_id,pay\_amount;

fetch cur1 into dis, pay\_amount;

end while;

close cur1;

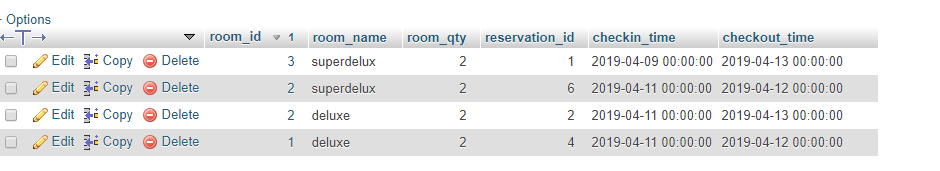
update payment set payment\_amount = pay\_amount where input\_payment\_id = payment\_id ;

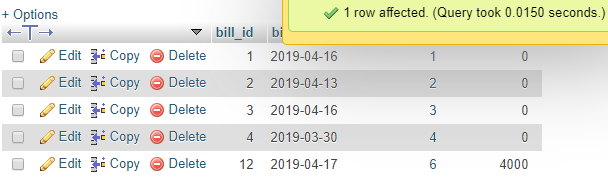
update payment set payment\_status='paid' where input\_payment\_id = payment\_id ;

end

2) **trigger**

**to get a bill directly with specified discount after making a reservation for a customer** :: will calculate bill amount according to the number of reservations made.For example if a customer has made total 5 number of reservations then at the time of 6th reservation he/she will get 5 percentage of discount and that value will be directly calculated in that(6th ) reservation and bill amount will be calculated with that particular discount.

Here we are inserting reservation id 6 of customer\_uname hardi ,so we can see in the table of bill final price is calculated.



As shown in the above table 4000 amount is calculated directly .

3**)regular\_customers** – prints names of those customer who have highest no of bills in a given year.

Type :: Procedure

Parameter :: year\_input int

begin

declare c\_uname varchar(10) ;

declare display\_count int;

declare k int;

declare b\_dt date ;

declare cur1 cursor for select COUNT( reservation.customer\_uname ), bill.bill\_date,reservation.customer\_uname from reservation , bill where year\_input = extract(YEAR from bill.bill\_date)

and reservation.reservation\_id = bill.reservation\_id

group by reservation.customer\_uname having COUNT(reservation.customer\_uname) = ( select max(mycount) from ( select reservation.customer\_uname, COUNT(reservation.customer\_uname) as mycount from bill , reservation where reservation.reservation\_id = bill.reservation\_id group by reservation.customer\_uname) as T ) ;

declare continue handler for not found set k = 1;

open cur1;

set k = 0;

fetch cur1 into display\_count, b\_dt , c\_uname ;

while k = 0 do

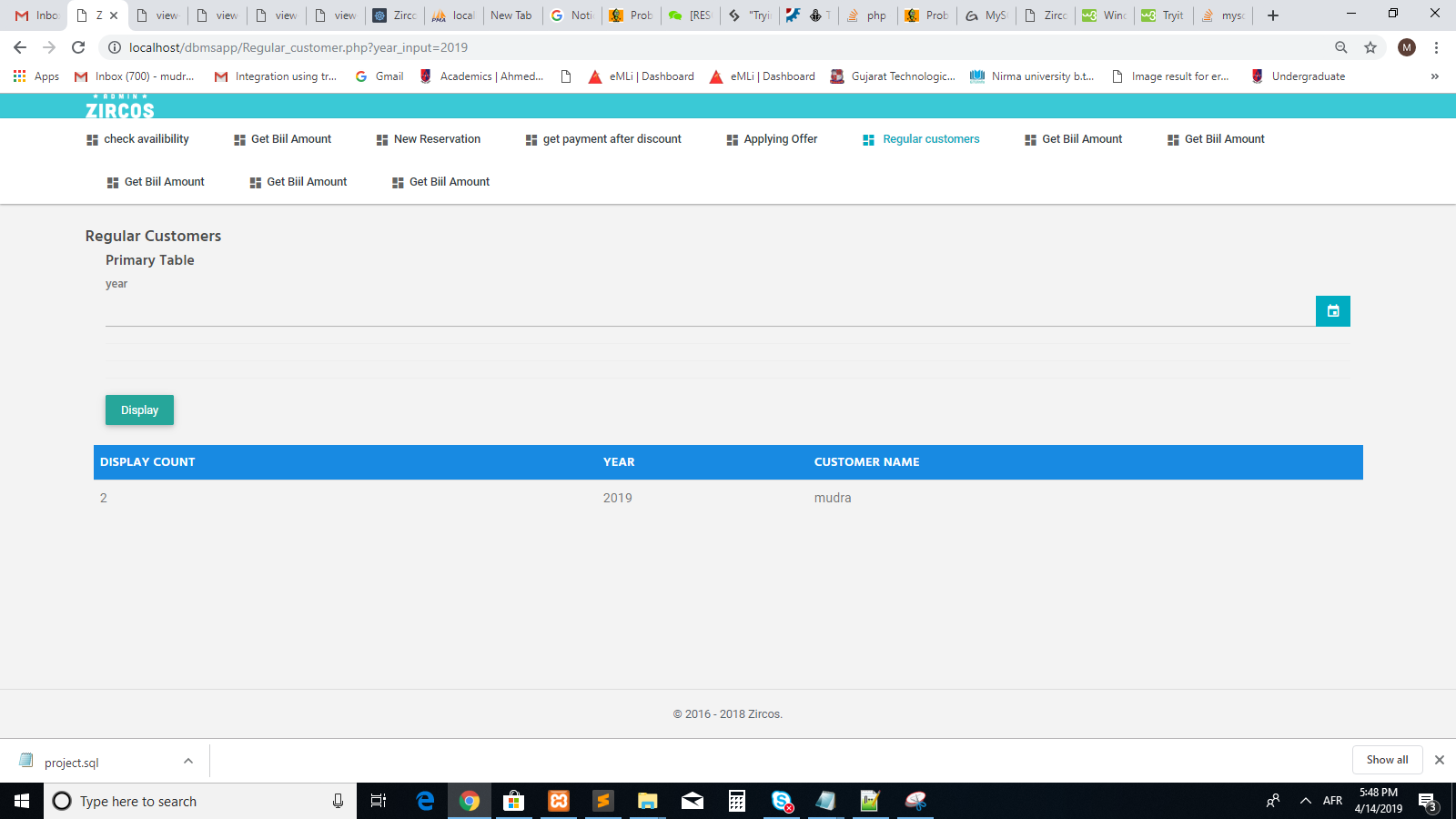
select display\_count, b\_dt, c\_uname;

fetch cur1 into display\_count, b\_dt , c\_uname ;

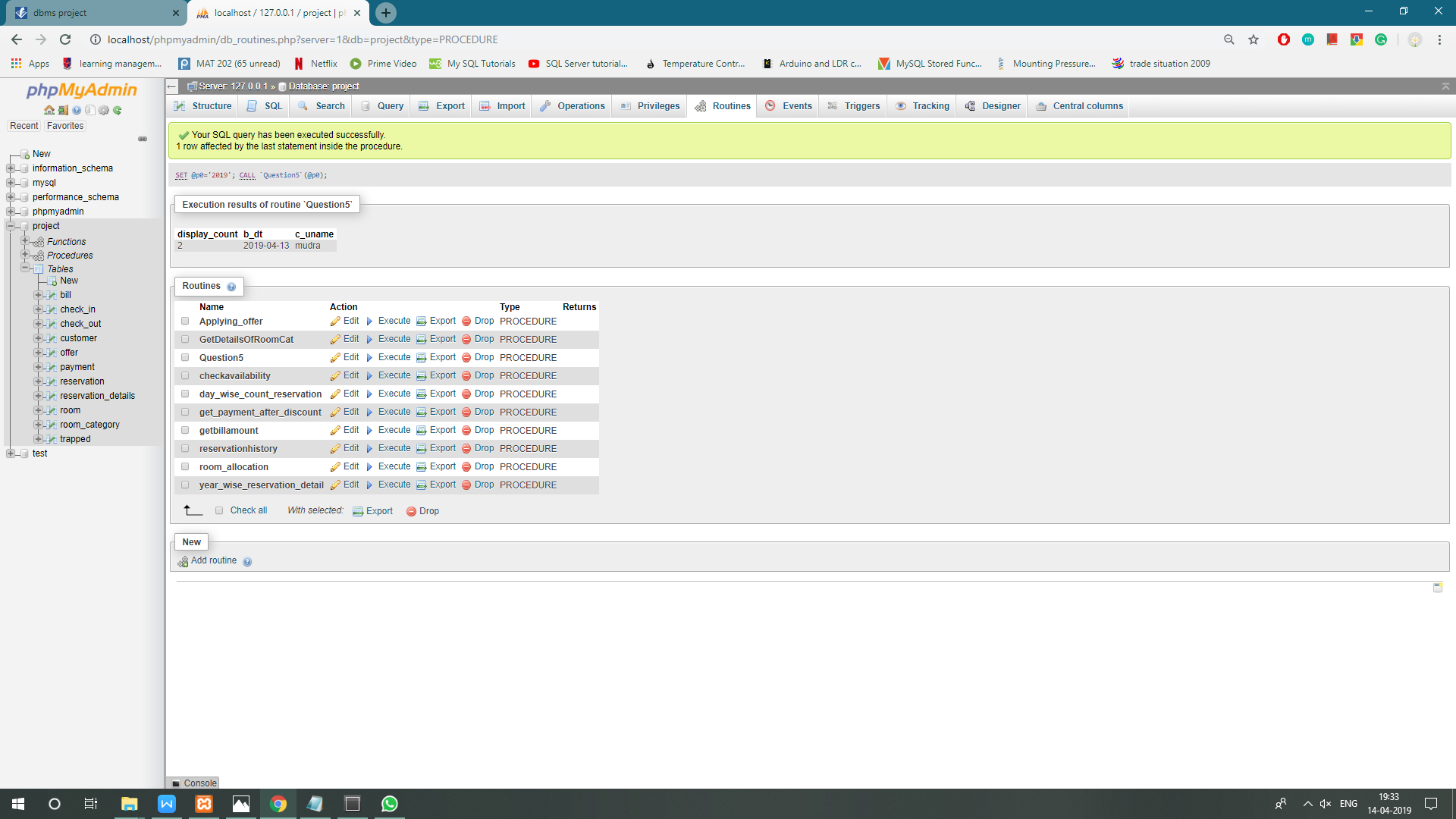
end while;

end

screenshot(at front end)



Screenshot(at backend)



**4)get\_payment\_after\_discount :** after knowing which payment mode it is and whether payment status is paid or unpaid , will give discount to payment amount accordingly .If payment status is paid then nothing will be executed further else,after getting appropriate discount payment amount will be calculated and payment status of payment table will be set to paid from unpaid.

Type :: Procedure

Parameters :: input :: input\_payment\_id

input :: input\_payment\_mode

begin

declare pay\_amount , dis float;

declare pay\_mode ,pay\_status varchar(10);

declare k ,pay\_id int;

declare cur1 cursor for select payment\_id , payment\_mode , payment\_amount , payment\_status from payment where input\_payment\_mode = payment\_mode and input\_payment\_id = payment\_id;

declare continue handler for not found set k = 1;

open cur1;

set k = 0;

fetch cur1 into pay\_id, pay\_mode, pay\_amount , pay\_status;

while k = 0 do

if pay\_mode = 'creditcard' && pay\_status='unpaid' then

set dis = 0.5 ;

elseif pay\_mode = 'cash' && pay\_status='unpaid' then

set dis = 0.4 ;

elseif pay\_mode = 'netbanking' && pay\_status='unpaid' then

set dis = 0.2 ;

elseif pay\_mode = 'paytm' && pay\_status='unpaid' then

set dis = 0.3 ;

end if;

set pay\_amount = pay\_amount - (dis\*pay\_amount) ;

select pay\_id,pay\_mode,pay\_amount;

fetch cur1 into pay\_id, pay\_mode, pay\_amount,pay\_status;

end while;

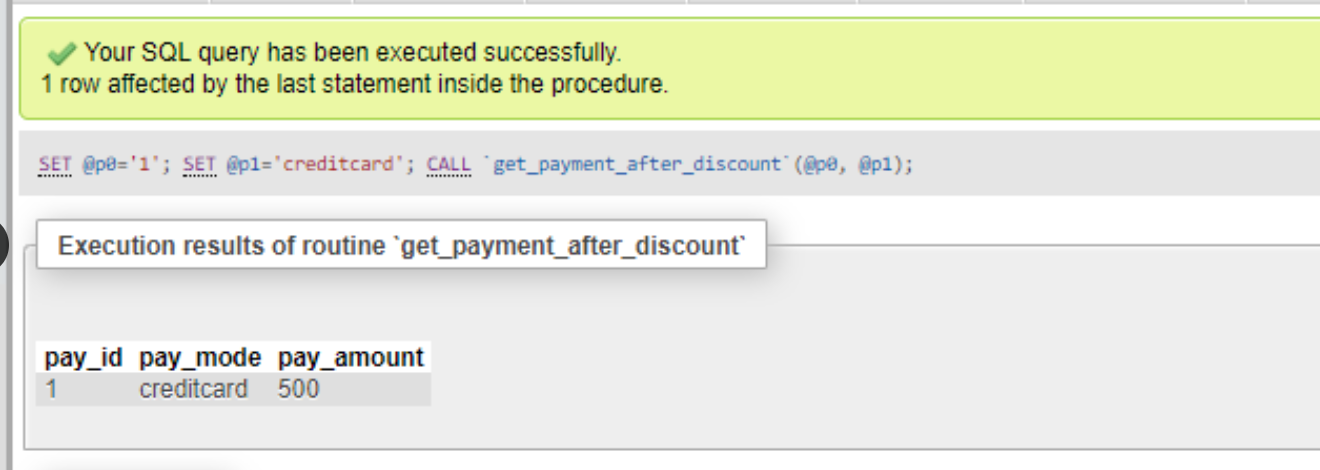
close cur1;

update payment set payment\_amount = pay\_amount where input\_payment\_mode = payment\_mode and input\_payment\_id = payment\_id and payment\_status= 'unpaid';

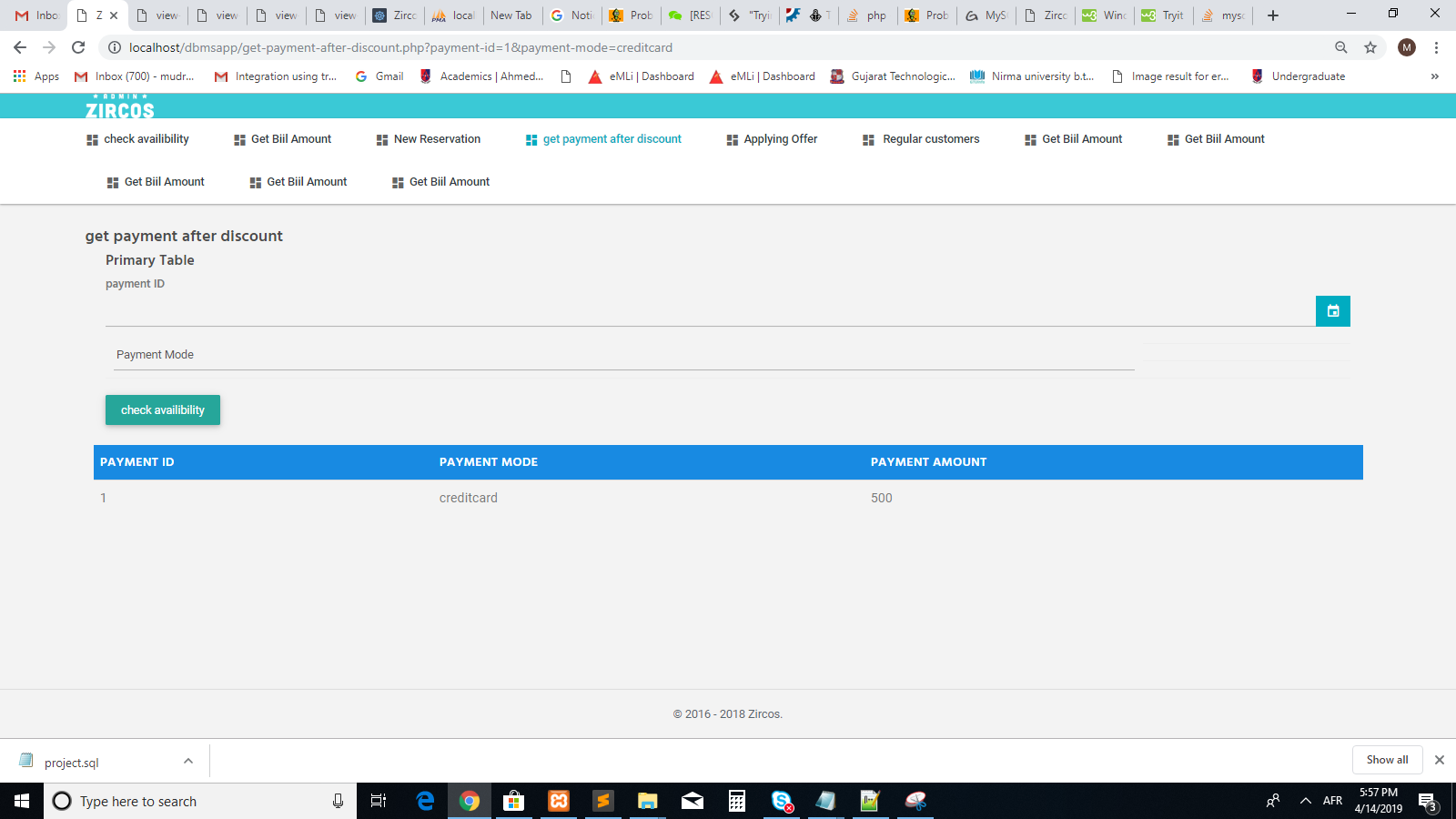
update payment set payment\_status='paid' where input\_payment\_mode = payment\_mode and input\_payment\_id = payment\_id ;

end

screenshot(at backend)



Screenshot(at frontend)



**5)check\_availaibility** :: checks categories of rooms which are available between check\_in date and check\_out date as given by user.

Parameters :: input :: input\_check\_in datetime

input :: input\_check\_out datetime

begin

select r.room\_id , r.room\_name from room\_category as r inner join reservation\_details as re on r.room\_id = re.room\_id WHERE

(input\_checkin\_date < re.checkin\_time )

OR

(input\_checkout\_date > re.checkout\_time)

AND

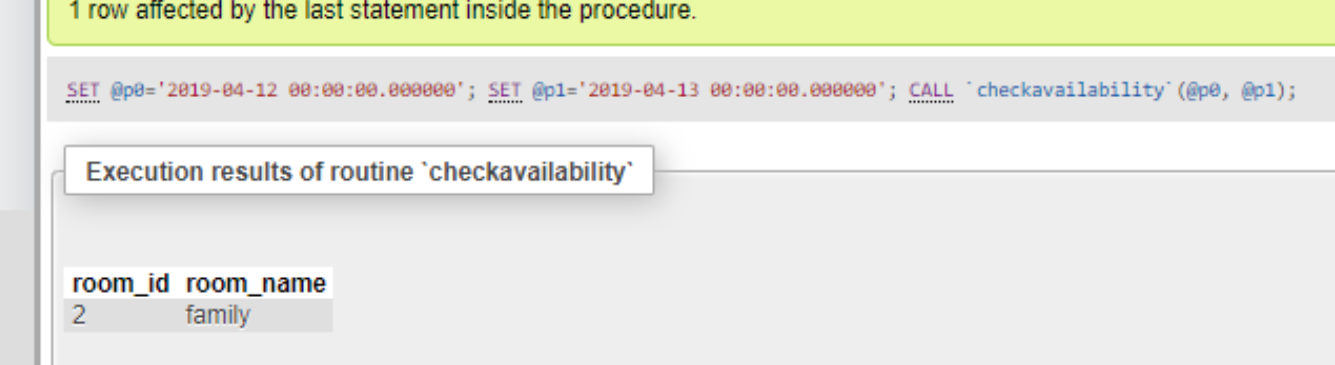
(input\_checkout\_date < re.checkin\_time)

OR

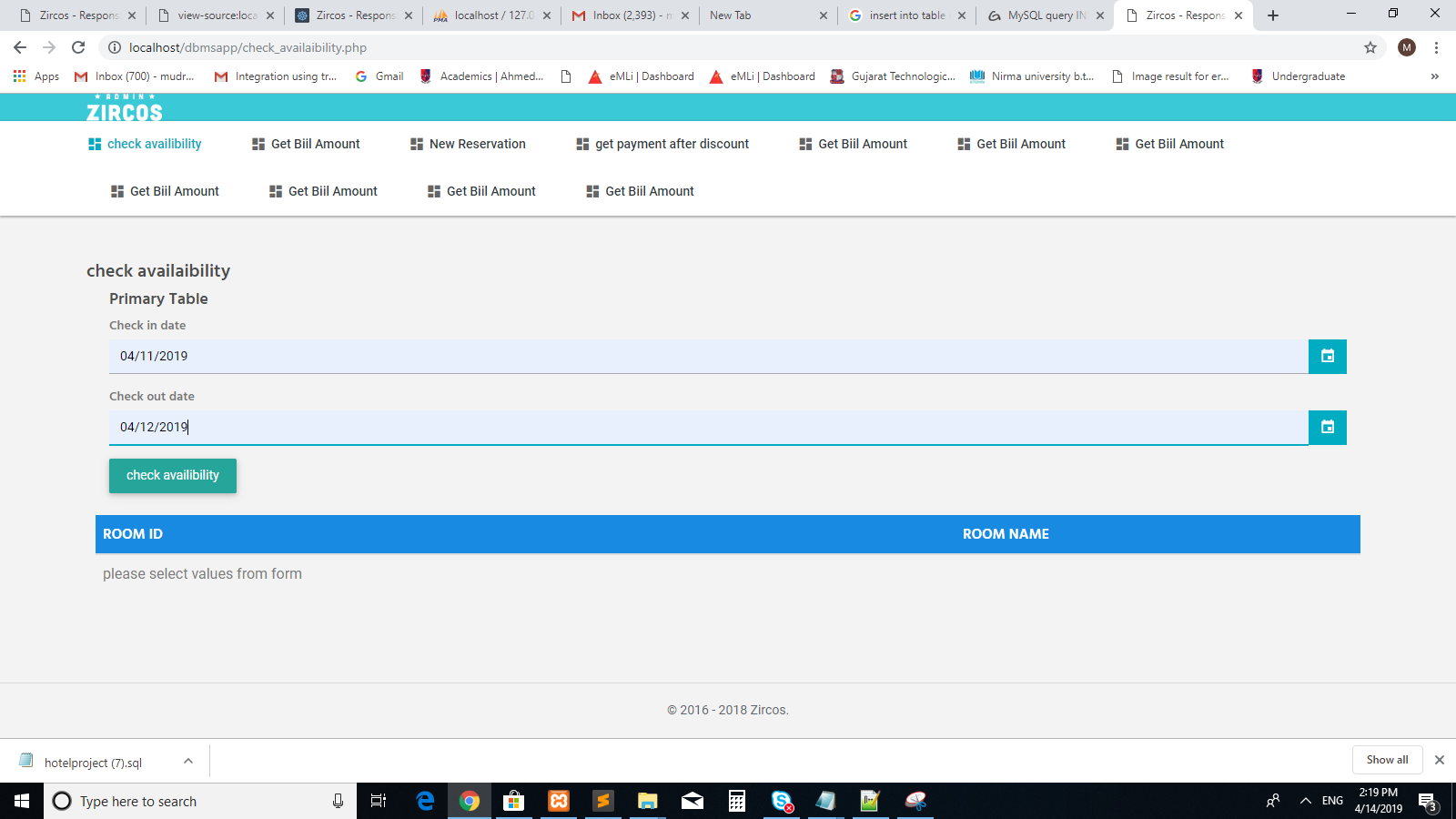
(input\_checkout\_date > re.checkout\_time);

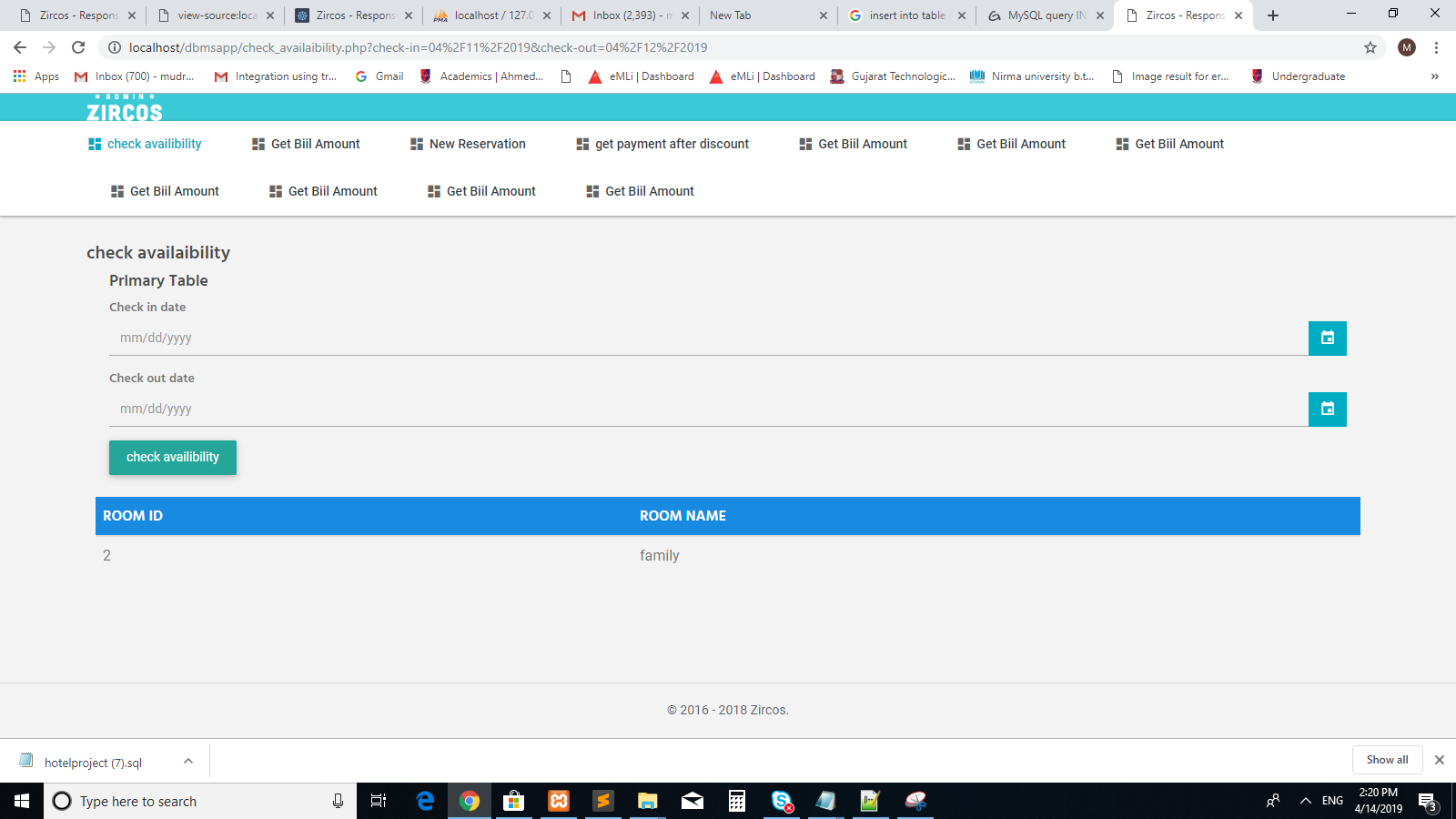
end

screenshot(at backend)



Screenshot(at frontend)





**6) Type :: Trigger NOT ON WEEKENDS**

Trigger\_name = If admin tries to update any room category on weekends then it will show a msg error that it cannot be updated into weekends and will store that data into a table named trapped.

Table :: room\_category

Time :: Before

Event :: update

begin

declare msg varchar(128);

if(dayname(curdate()) = 'saturday' || dayname(curdate())= 'sunday') then insert into trapped values (user(),curdate());

set msg = 'ERROR : You cannot change it on weekends ....';

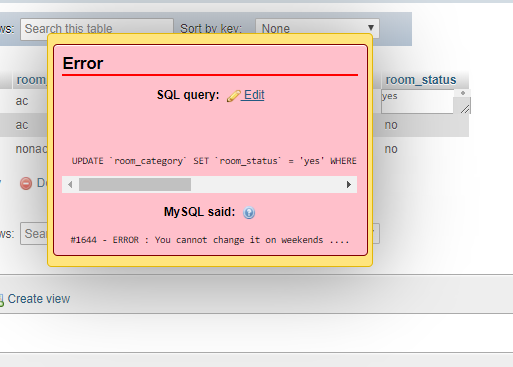
signal sqlstate '45001' set message\_text = msg ;

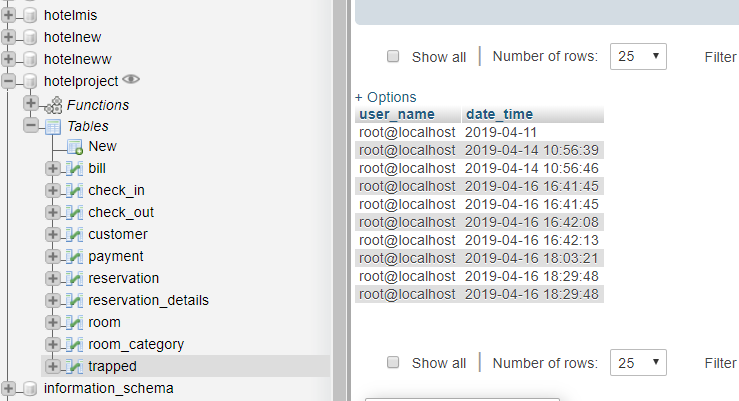
elseif (hour(curtime()) >=10 && hour(curdate()) <= 18) then insert into trapped values (user(),current\_timestamp ());

end if;

end

screenshot(at backend)





**7)get level of customer** :: calculates level of customer that is platinium ,silver etc on the basis of no of reservations made by the customer in a given period of time.

delimiter $

drop procedure getlevel$

delimiter $

create procedure getlevel()

begin

declare level varchar(20);

declare no\_reservations, r\_id , k int;

declare r\_customer\_uname varchar(10) ;

declare cur1 cursor for select reservation.customer\_uname , reservation.reservation\_id ,count(reservation\_details.reservation\_id) as no\_reservations FROM reservation\_details , reservation where reservation.reservation\_id = reservation\_details.reservation\_id group by reservation\_id ORDER BY no\_reservations DESC ;

declare continue handler for not found set k=1;

open cur1;

set k=0;

fetch cur1 into r\_customer\_uname, r\_id , no\_reservations;

while k=0 do

if no\_reservations > 20 then

set level = 'PLATINUM';

elseif no\_reservations <= 20 and no\_reservations >= 10 then

set level = 'GOLD' ;

elseif no\_reservations < 10 then

set level = 'SILVER';

end if;

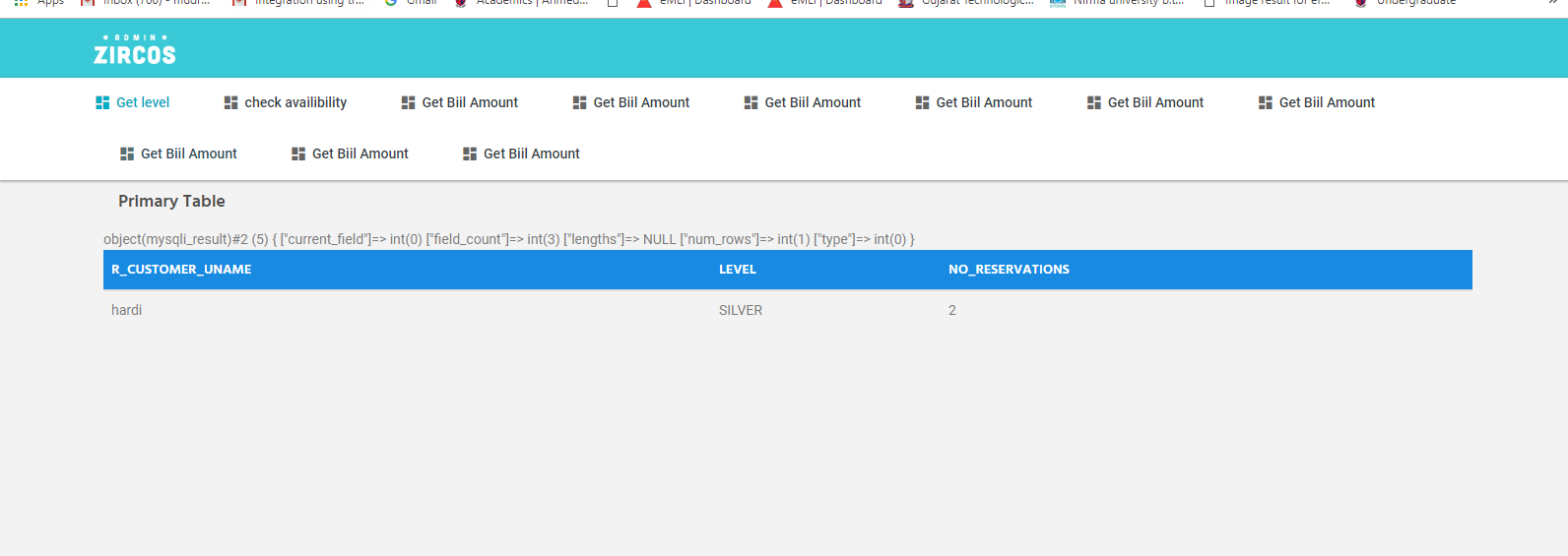
select r\_customer\_uname,level,no\_reservations;

fetch cur1 into r\_customer\_uname, r\_id , no\_reservations;

end while;

close cur1;

end $



**8)day\_wise\_count\_reservation ::** displays count of total no of reservations done on a particular day.

Type :: procedure

Parameters :: input :: input\_date DATE

c\_type varchar(10)

begin

declare reservation\_cnt int;

if c\_type = 'admin' then

select count(distinct reservation.reservation\_id) into reservation\_cnt from reservation inner join reservation\_details on reservation.reservation\_id = reservation\_details.reservation\_id and input\_date = reservation.reservation\_date group by reservation\_date;

select reservation\_cnt;

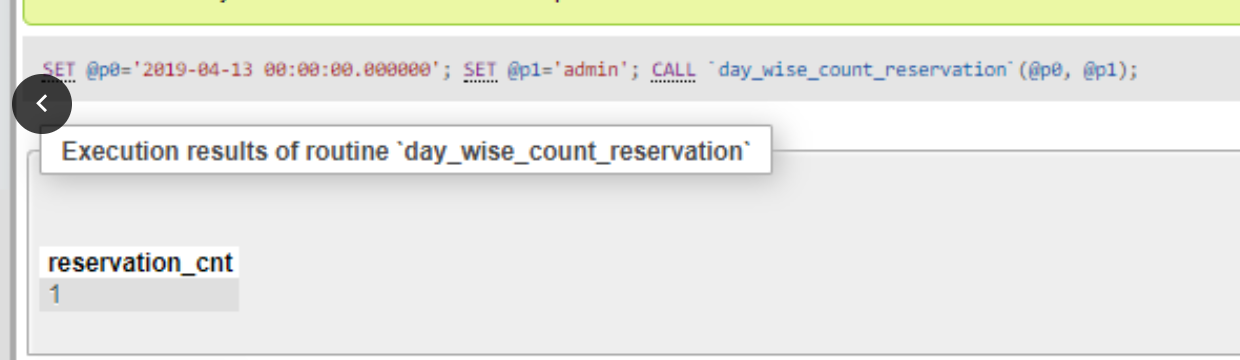
else

select 'this information is only visible to admin';

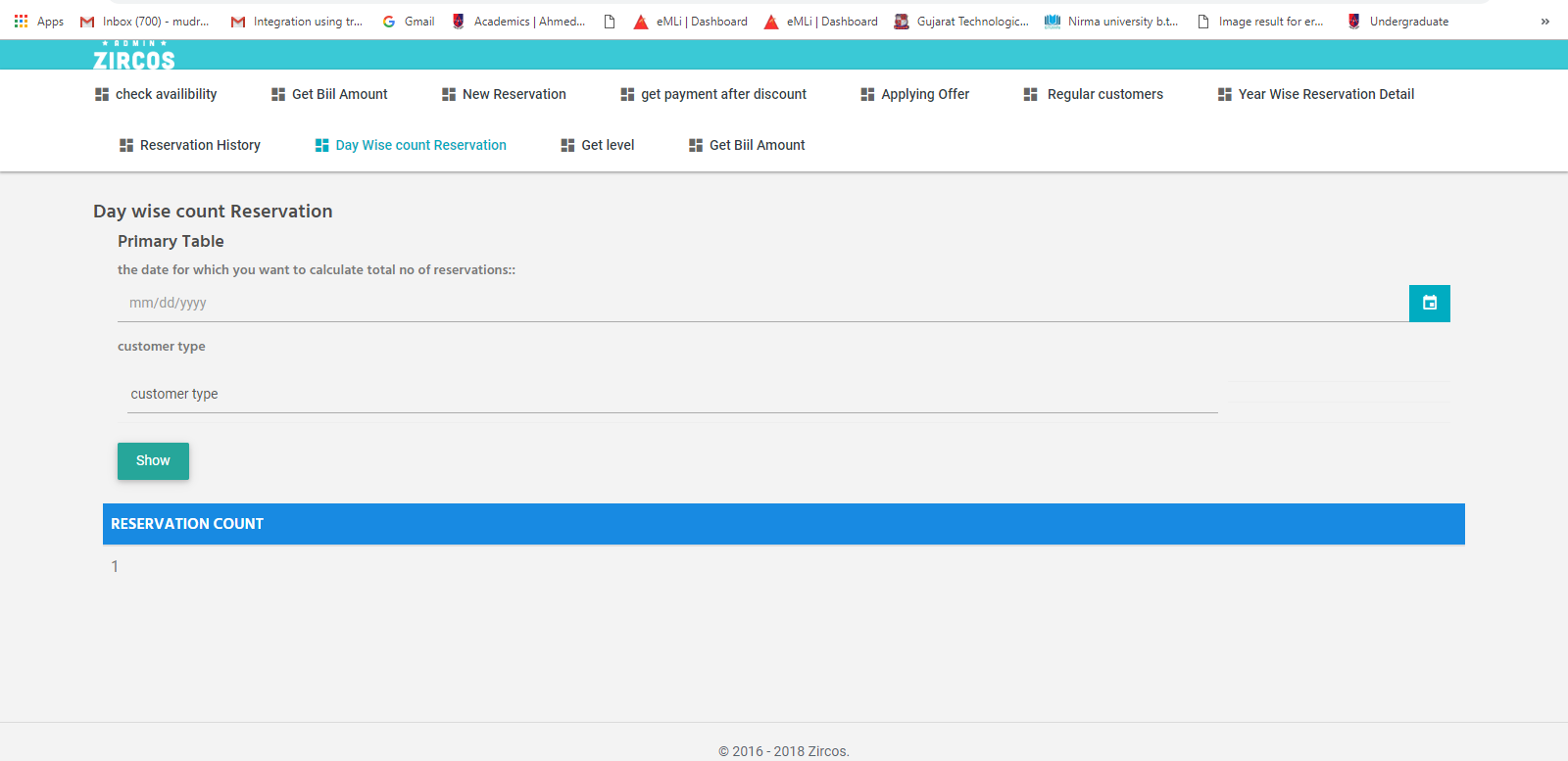
end if;

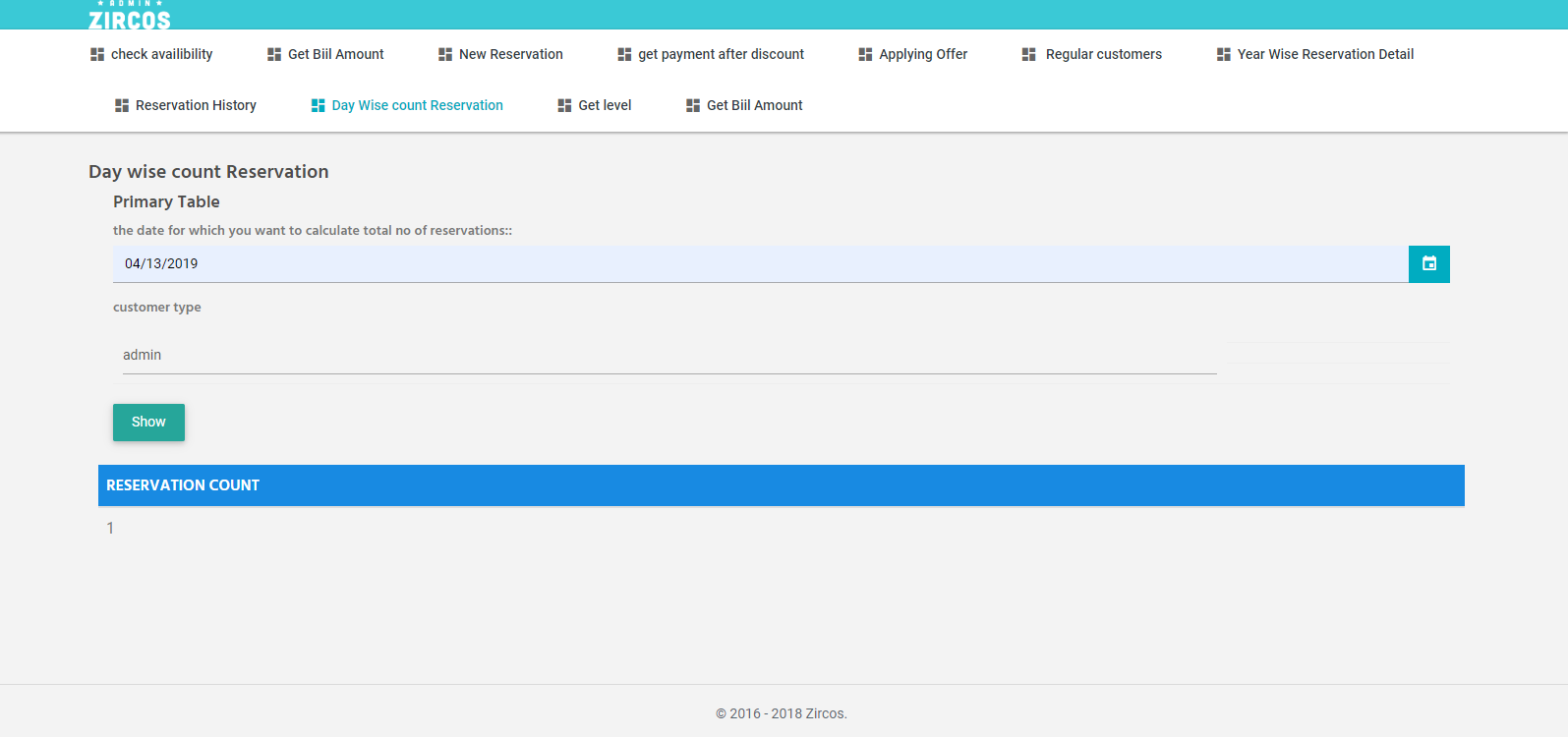
end

screenshot(at backend)



Screenshot(at frontend)





**9) getbillamount** == calculates total bill amount from reservation id only.(Note a reservation can have more than one room category and any no of quantity of any given specific room category .)

Type :: procedure

parameters :: input re\_id int

begin

declare billamount , k int;

declare cur1 cursor for select reservation\_id ,sum(room\_category.room\_price\*reservation\_details.room\_qty) as billamount from room\_category, reservation\_details where reservation\_id=re\_id and room\_category.room\_name = reservation\_details.room\_name GROUP BY reservation\_id;

declare continue handler for not found set k=1;

open cur1;

set k=0;

fetch cur1 into re\_id , billamount;

while k=0 do

select re\_id , billamount;

fetch cur1 into re\_id , billamount;

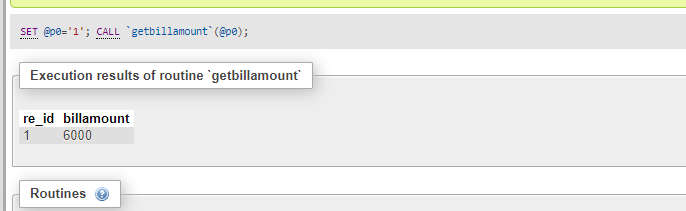
end while;

insert into bill values(re\_id,CURRENT\_DATE, re\_id, billamount );

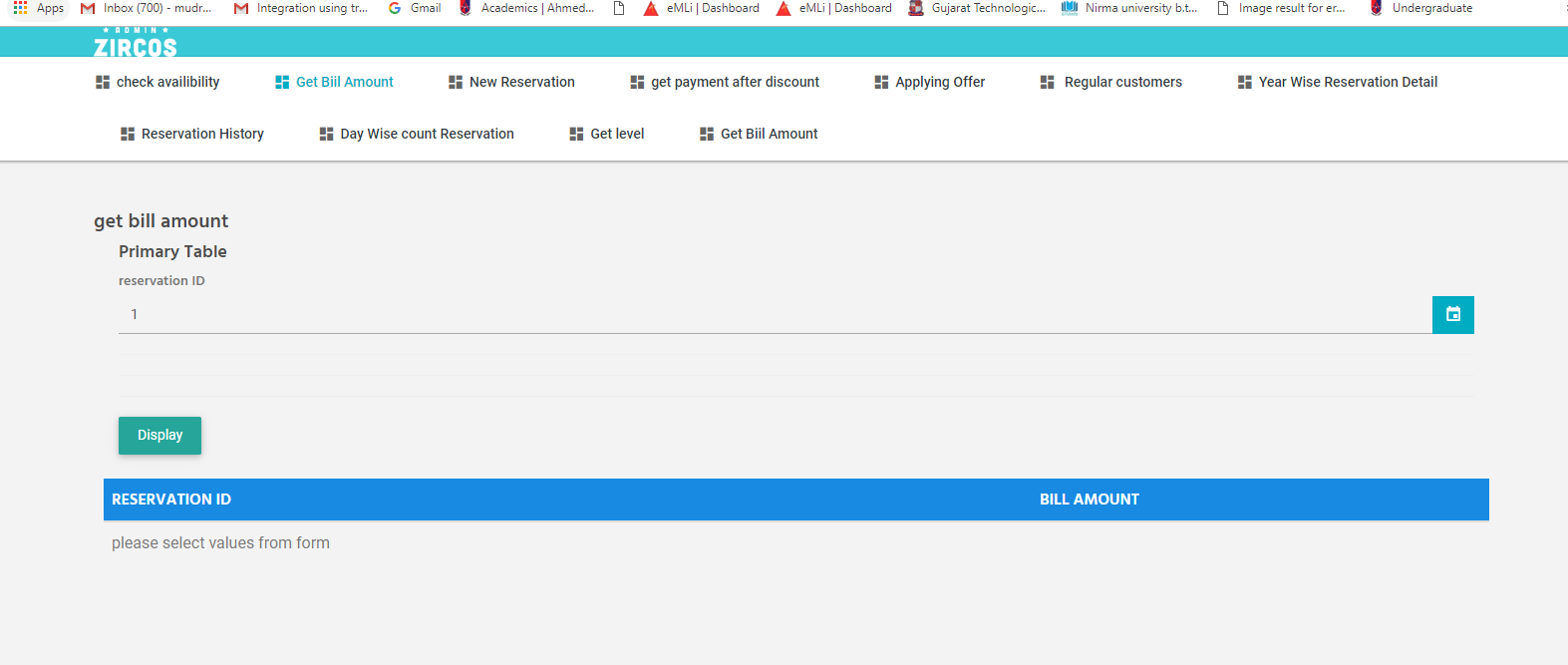
close cur1;

end

screenshot(at backend)



Screenshot(at frontend)



**10)reservation\_history** :: will get all the reservation details of rooms booked till date as per year and also in which month.

Type : Procedure

BEGIN

declare b int ;

declare rdate date;

declare cur1 cursor for select reservation.reservation\_date from reservation inner join reservation\_details on reservation.reservation\_id = reservation\_details.reservation\_id group by reservation\_details.reservation\_id order by reservation.reservation\_date;

declare continue handler for not found set b=1;

open cur1;

set b=0;

fetch cur1 into rdate;

while b = 0 do

select extract(year from rdate);

select extract(month from rdate);

select \* from reservation inner join reservation\_details on reservation.reservation\_id = reservation\_details.reservation\_id where reservation.reservation\_date = rdate;

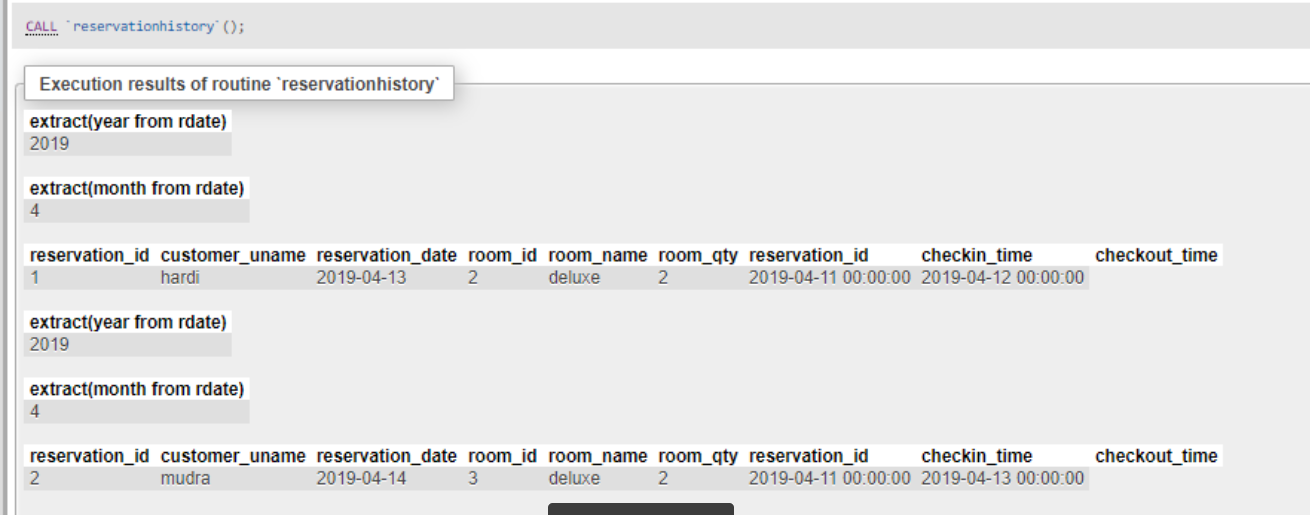
fetch cur1 into rdate;

end while;

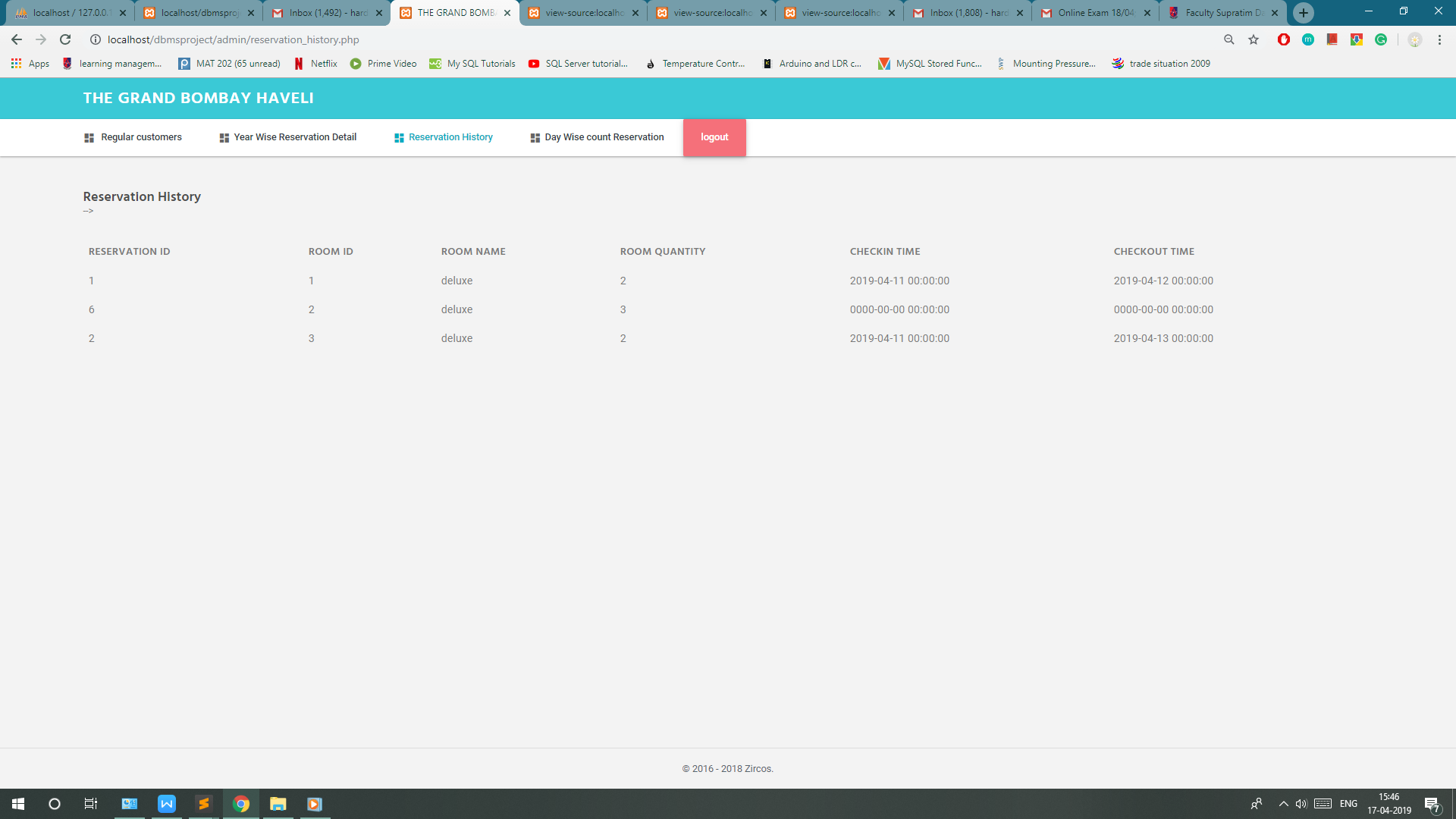
close cur1;

end

screenshot(at backend)



Screenshot(at frontend)



Type : Procedure

**11) room\_allocation :=** On the basis of no of members ,reservation id and room type(name) allocates room to the customers

As soon as the room is allocated it inherently sets room status corresponding to that room id to be true ,that means that room is allocated and cannot be further allocated to other customer till that customer check outs from the room.

Type :: Procedure

Parameters :: input members int , input reserve\_id int , input r\_name varchar(10);

begin

declare k int;

declare a\_occupied varchar(3);

declare r\_nm varchar(10);

declare c, room\_id\_ int;

declare temp\_roomid int;

declare cur1 cursor for select room.room\_id, room\_category.no\_of\_persons, room\_category.room\_status , room\_category.room\_name from room, room\_category where room.room\_id=room\_category.room\_id ;

declare continue handler for not found set k=1;

open cur1;

set k=0;

fetch cur1 into room\_id\_ ,c ,a\_occupied ,r\_nm;

while k=0 do

if ( c = members && c <= 4 && a\_occupied ='no' && r\_nm = r\_name) then

set temp\_roomid= room\_id\_;

end if;

fetch cur1 into room\_id\_, c ,a\_occupied,r\_nm ;

end while;

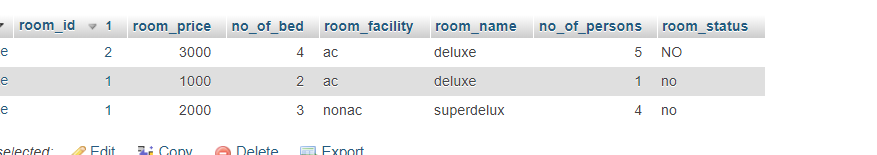
close cur1;

update room\_category set room\_status='yes' where room\_id= temp\_roomid;

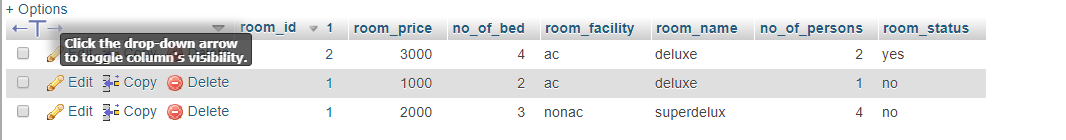
update reservation\_details set room\_id =temp\_roomid where reservation\_id =reserve\_id;

end

table room category before executing above procedure::



After inserting values 2,2,deluxe note here first one is reservation id and not room id and 2nd one is room capacity so the room status corresponding to all the rooms of that reservation id will get room status to be true.even if there were 2 different rooms of different room category that would have set to true.



**12)insert\_reservation\_detail(booking and calculating total bill directly)** ::

Type :: procedure (In roomid int , in roomname varchar, in roomqty int , in reservationid int)

Code::

begin

declare reservation\_total\_price ,room\_p int;

declare k int;

declare cur1 cursor for select room\_category.room\_id, room\_category.room\_price, room\_category.room\_name , reservation\_details.room\_qty ,reservation\_details.reservation\_id from room\_category , reservation\_details where reservation\_details.room\_id = room\_category.room\_id ;

declare continue handler for not found set k = 1;

set reservation\_total\_price = 0;

open cur1;

set k = 0;

fetch cur1 into roomid , room\_p , roomname , roomqty ,reservationid;

while k = 0 do

set reservation\_total\_price = reservation\_total\_price + room\_p\*roomqty;

insert into reservation\_details(room\_id , room\_name , room\_qty , reservation\_id,checkin\_time,checkout\_time)

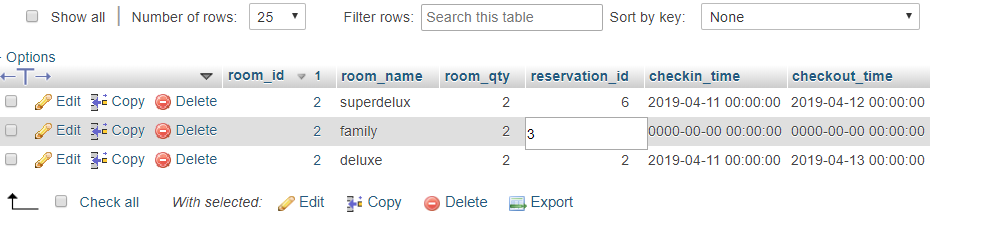
values(roomid, roomname, roomqty, reservationid, 2019-07-04, 2019-07-05);

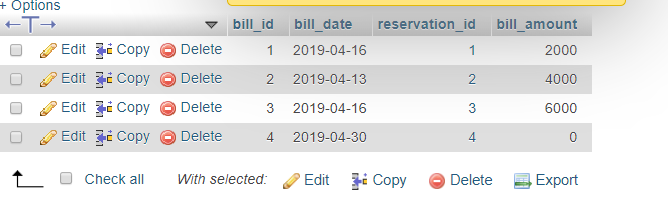
fetch cur1 into roomid , room\_p , roomname , roomqty ,reservationid;

end while;

close cur1;

end





**12)year\_wise\_reservation\_detail ::** after giving any year as input it will give complete set of records of reservation details categorized by that particular year.This procedure can only be accessed by admin so the user type is also taken as input , if user type is admin then and only then it will display year\_wise\_reservation\_detail.

Type :: procedure

input :: input\_year year ;

input\_year c\_type varchar(10);

begin

select input\_year,c\_type;

if c\_type = 'admin' then

select \* from reservation where input\_year = extract(year from reservation.reservation\_date);

select r.reservation\_id , r.reservation\_date , r.customer\_uname ,rd.room\_id,rd.room\_qty from reservation r inner join reservation\_details rd on r.reservation\_id = rd.reservation\_id and input\_year = extract(year from r.reservation\_date);

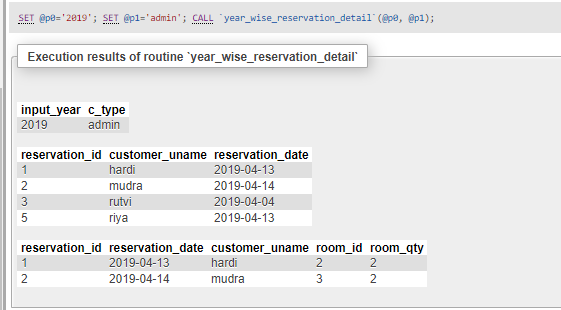
ELSE

select 'these details are available only for admin' ;

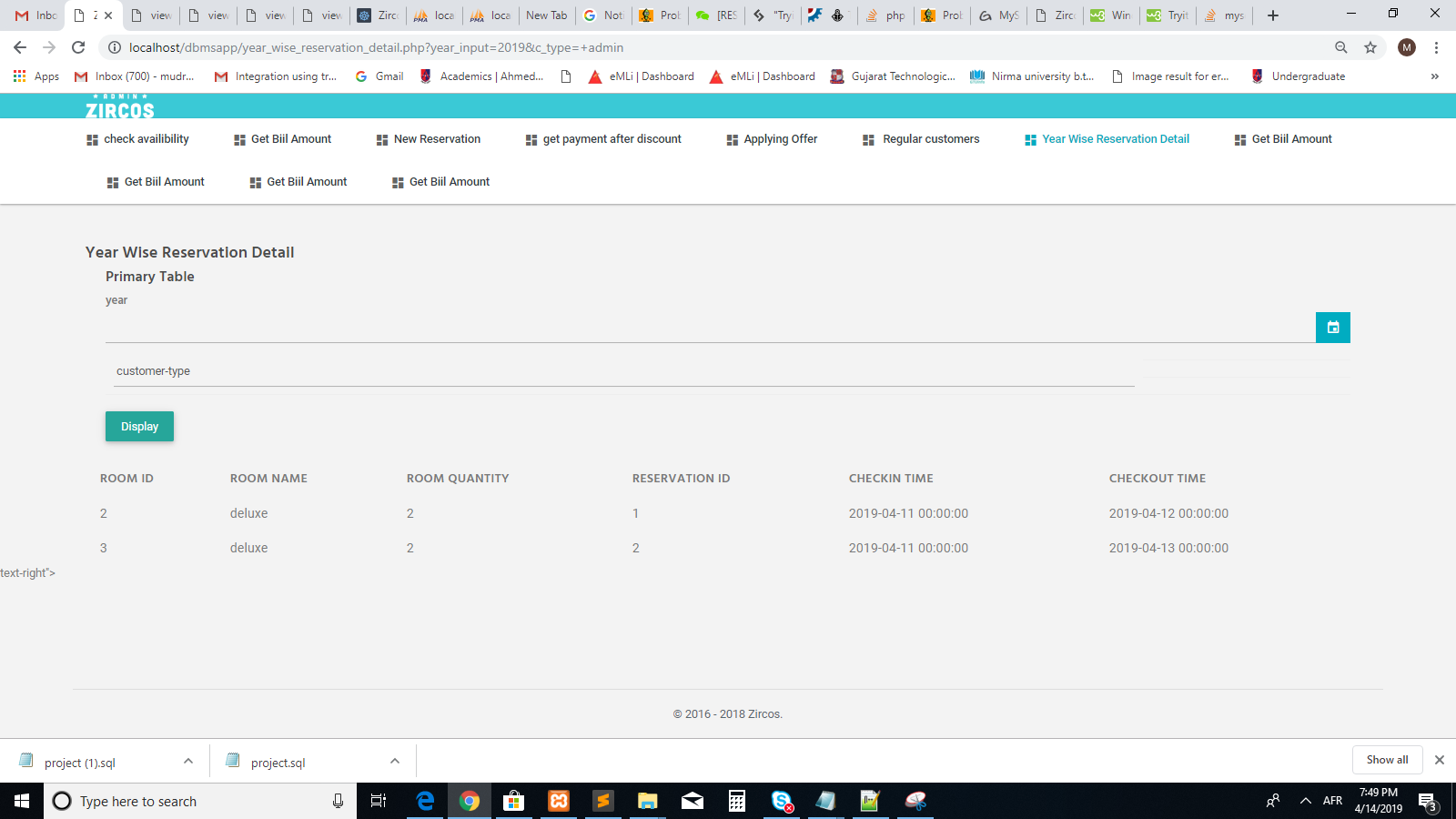
end if;

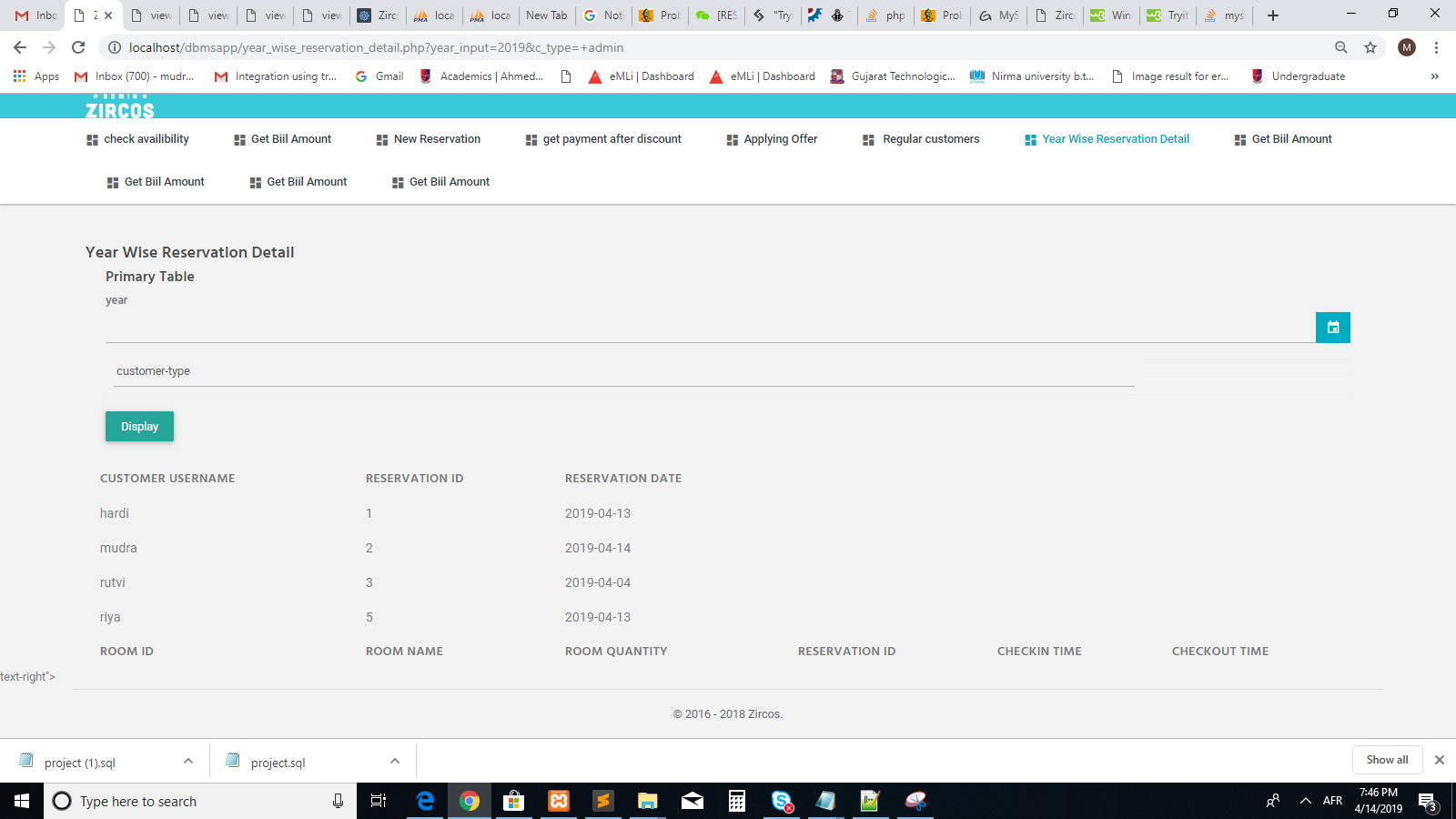
end

screenshot(at backend)



Screenshot(frontend)





**13)Type Trigger**

Trigger\_name :: checkin\_enforcement :: Before inserting into reservation\_details table it will check whether checkin date is smaller than check\_out date, if it is not it will return an error.

Table :: reservation\_details

Time : Before

Event :INSERT

begin

declare msg varchar(128);

if new.checkin\_time > new.checkout\_time then

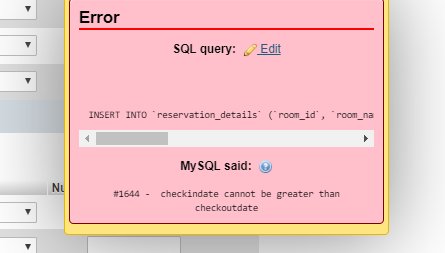
set msg = ' checkindate cannot be greater than checkoutdate';

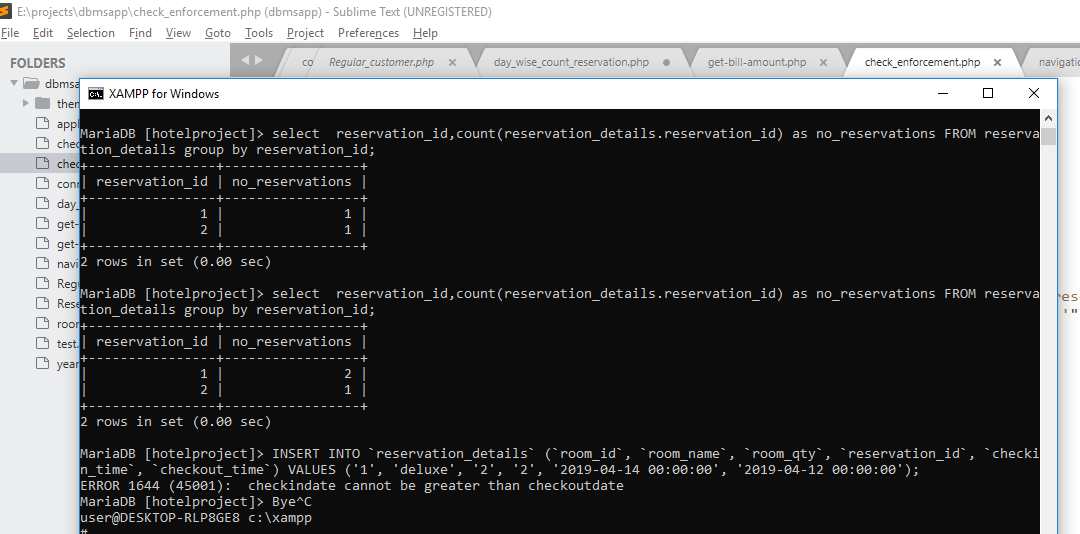
signal sqlstate '45001' set message\_text = msg;

end if;

end

screenshot(backend)





**Screenshot (at frontend)**

