

CS355 Mini Project Assignment

Autumn 2020

Team-14

Name: Unique Kaushik Roll no: 1901CS66

Name: Vishesh Jain Roll No: 1901CS71

Name: Ranjeet Khichar Roll No:1901CS45

Guest House:

1901CS66, Unique Kaushik

Market Shop:

1901CS71, Vishesh Jain

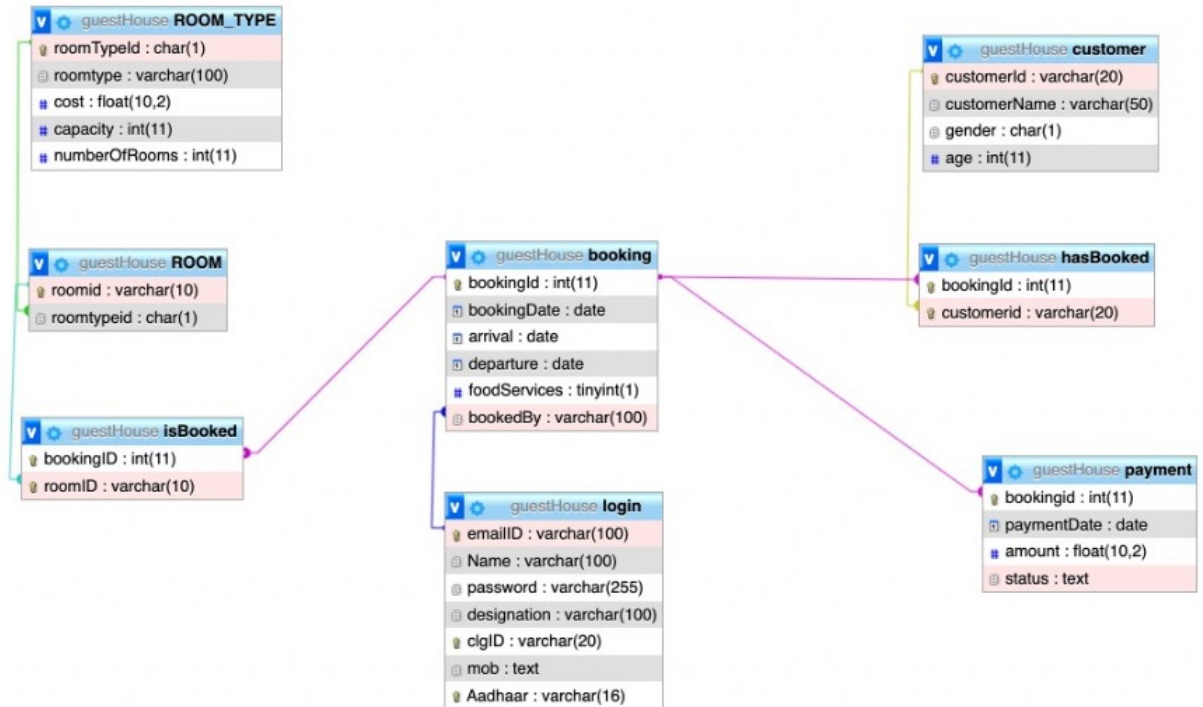
Landscape:

1901CS45, Ranjeet Khichar

GuestHouse :

Database :

ER-diagram :



Tables used in Guest House:

1. **Login Table** : This is used to store information about members of IIT Patna.

PRIMARY KEY: emailID



Unique: clgID, Aadhaar

#	Name	Type	Collation	Attributes	Null	Default
1	emailID	varchar(100)	utf8mb4_general_ci		No	None
2	Name	varchar(100)	utf8mb4_general_ci		No	None
3	password	varchar(255)	utf8mb4_general_ci		No	None
4	designation	varchar(100)	utf8mb4_general_ci		No	None
5	clgID	varchar(20)	utf8mb4_general_ci		No	None
6	mob	text	utf8mb4_general_ci		No	None
7	Aadhaar	varchar(16)	utf8mb4_general_ci		No	None

2. Booking Table : This table is used to store information about the bookings.


Primary Key : bookingId

Foreign Key: bookedBy

#	Name	Type	Collation	Attributes	Null	Default
1	bookingId 	int(11)			No	None
2	bookingDate	date			No	current_timestamp()
3	arrival	date			No	None
4	departure	date			No	None
5	foodServices	tinyint(1)			No	None
6	bookedBy 	varchar(100)	utf8mb4_general_ci		No	None

3. Customer Table : Information about guests for whom the room is booked.




Primary Key : customerId

#	Name	Type	Collation	Attributes	Null	Default
1	customerId 	varchar(20)	utf8mb4_general_ci		No	None
2	customerName	varchar(50)	utf8mb4_general_ci		No	None
3	gender	char(1)	utf8mb4_general_ci		No	None
4	age	int(11)			No	None

4. HasBooked : This table stores customerId corresponding to bookingId.

Primary Key: (bookingId, customerId)


Foreign Key : bookingId,customerId

#	Name	Type	Collation	Attributes	Null	Default
1	bookingId 	int(11)			No	None
2	customerId  	varchar(20)	utf8mb4_general_ci		No	None

5. isBooked : This table stores roomID corresponding to bookingID.

Primary Key : (bookingId,roomId)


Foreign Key : bookingId,roomId

#	Name	Type	Collation	Attributes	Null	Default
1	bookingID 	int(11)			No	None
2	roomId  	varchar(10)	utf8mb4_general_ci		No	None

6. **Payment** : This table stores payment details corresponding to bookingID.


Primary Key : bookingId.

Foreign Key : bookingId.

#	Name	Type	Collation	Attributes	Null	Default
1	bookingId 	int(11)			No	None
2	paymentDate	date			Yes	NULL
3	amount	float(10,2)			No	None
4	status	text	utf8mb4_general_ci		No	'PENDING'

7. **Room_Type** : This table contains details of different room types.



Primary Key : roomTypeID

#	Name	Type	Collation	Attributes	Null	Default
1	roomTypeID 	char(1)	utf8mb4_general_ci		No	None
2	roomtype	varchar(100)	utf8mb4_general_ci		No	None
3	cost	float(10,2)			No	None
4	capacity	int(11)			No	None
5	numberOfRooms	int(11)			No	None

8. **Room** : This table stores roomID corresponding to roomtypeID.

Primary Key : roomID.

Foreign Key : roomtypeID.

#	Name	Type	Collation	Attributes	Null	Default
1	roomId 	varchar(10)	utf8mb4_general_ci		No	None
2	roomtypeId 	char(1)	utf8mb4_general_ci		Yes	NULL

WebPages :

Index Webpage : Here, a brief description of campus and its guesthouse is provided. There are details about different types of rooms here. One can visit the 'register' web page if not registered and the 'login' page from here directly.

Register WebPage :Here, information from the user like iitp webmail, Name, Password, designation, collegeID/RollNo, Mobile No and Aadhar is taken. Every IITP member should remember their email and passwords for logging in.

[Note: Designations : Student, Staff and IITP

One can input designations Student/Staff while registering depending on his status in college but the designation(IITP) is only provided to ADean/Admin/Clubs etc.]

SQL queries:

1. //checking if email is already registered
SELECT * from login WHERE emailID = '____@iitp.ac.in';
2. // inserting values in login table if email is not registered
INSERT INTO login VALUES ('email','name','password','des',clgid,mob,aadhaar);

Login WebPage: People can login from here using their email and passwords. From here one can jump to different web pages:

1. For Receptionist: If a person logs in with receptionist's email and password.
2. For Booking: All other IITP members than receptionist

SQL query:

```
// Query to find user in login table and match the password if found.  
SELECT * from login WHERE emailID = 'email';
```

Options Available for IITP Members:

Welcome WebPage : From here a person can do multiple tasks . He can do following things from here :-

- He can see the details about different types of rooms.
- There are some important points also.
- He can see the details of previous and current booking as well.
- He can see guest details as well corresponding to those bookings.

SQL queries :

```
// to show details of user's previous booking  
SELECT * FROM booking NATURAL JOIN isBooked NATURAL JOIN room  
NATURAL JOIN payment WHERE bookedBy = 'username' AND departure <  
curdate();  
// to show details of user's current booking  
SELECT * FROM booking NATURAL JOIN isBooked NATURAL JOIN room  
NATURAL JOIN payment WHERE bookedBy = 'username' AND departure >=  
curdate();  
// details of guests for current bookings  
SELECT bookingId, customer.* FROM hasBooked NATURAL JOIN customer  
NATURAL JOIN booking WHERE bookedBy = 'username' AND departure >  
curdate();
```

Book Room WebPage :

Process of Booking a Room in GuestHouse:

1. First the user needs to select checkin and checkout dates and click on the “Check Availability” button. While selecting checkin and checkout dates he need to take care of some points like :
 - Checkout should be greater than Check In.
 - Checkin and checkout are not in the past.In these cases a warning comes on screen showing “Invalid Inputs”.
2. After selecting valid checkin and checkout dates. He needs to select which type of room he/she wants to book.

SQL Query :

```
SELECT roomtypeid, COUNT(roomtypeid) as cnt from ROOM_TYPE NATURAL  
JOIN ROOM WHERE roomid IN (SELECT roomid from ROOM WHERE roomid  
NOT IN (SELECT roomid from isBooked NATURAL JOIN booking WHERE  
'checkin' < departure AND arrival < '$checkout')) GROUP BY roomtypeid;
```

3. After selecting the room type, he needs to fill in how many people would be staying there. It Should be less than/ equal to that room’s max capacity otherwise it shows “Invalid Inputs”.

SQL Query :

```
SELECT capacity FROM ROOM_TYPE WHERE roomTypeID = 'roomtypeid;
```

4. After he needs to input whether the guests would be consuming food while staying.
5. Click on the Submit button.

SQL Query :

```
INSERT INTO booking (bookingId,arrival,departure,foodservices,bookedBy)  
VALUES (bookingid,'checkin','checkout',foodservices,'bookedby');
```

6. After that he comes to a new web page, here, he can see the total cost for the booking as well as he is asked to fill information about the Guests.

SQL Queries :

1. // inserting guest information in table
INSERT INTO customer VALUES
(‘customerid’,‘customername’,‘customergender’,customerage);
2. // finding empty room in given time interval and of given type
SELECT roomid from ROOM_TYPE NATURAL JOIN ROOM WHERE
roomtypeid = 'roomtypeid' AND roomid IN (SELECT roomid from ROOM WHERE
roomid NOT IN (SELECT roomid from isBooked NATURAL JOIN booking
WHERE '\$checkin' < departure AND arrival < '\$checkout')) LIMIT 1;

3. INSERT INTO isBooked VALUES (bookingid,'roomid');
4. INSERT INTO payment (bookingId,amount) VALUES (bookingid,amount);

Cancel Booking Web Page :

- On this web Page, the user sees his current bookings.
“Bookings can only be cancelled before 7 days of Check IN date!!”.
- If the user selects a valid booking ID, then corresponding booking will be cancelled.
[Note : Once payment is done it is non-refundable.]

SQL Queries :

1. //checking if bookingid given is valid
 SELECT * FROM booking WHERE bookingId = 'bookingid' AND bookedBy = 'bookedby';
2. // cancelling booking
 DELETE FROM hasBooked WHERE bookingId = 'bookingid';
 DELETE FROM isBooked WHERE bookingId = 'bookingid';
 DELETE FROM payment WHERE bookingId = 'bookingid';
 DELETE FROM booking WHERE bookingId = \$bookingid';
3. // displaying current booking of user
 SELECT * FROM booking NATURAL JOIN isBooked NATURAL JOIN room
 NATURAL JOIN payment WHERE bookedBy = 'username" AND departure > curdate()

Logout WebPage: Once your booking is done, the user can logout by clicking the logout button placed at top of 'Welcome page'.

Options Available for Receptionist:

Home Page: At this page, various relevant information is provided to the receptionist like:

1. All current booking details or bookings with PENDING payment
2. details about different types of rooms

SQL Queries :

1. // to show current booking details or bookings with PENDING payment
 SELECT * FROM booking NATURAL JOIN payment WHERE departure >= curdate() OR status = 'PENDING'

Booking Details: On this webpage a receptionist can see booking details of particular BookingID. He/She can see Booking details, Corresponding customers details, room details, payment details.
 He can also set the payment status of a booking equal to DONE.

SQL Queries :

1. // to check booking id is valid
SELECT * FROM booking WHERE bookingId = \$bookingid
2. // to display customers details, room details, payment details.
SELECT * FROM payment WHERE bookingid = \$bookingid
SELECT customer.* FROM hasBooked NATURAL JOIN customer WHERE
bookingid = \$bookingid
SELECT room.* FROM isBooked NATURAL JOIN room WHERE bookingid =
\$bookingid
3. // updating payment status if marked as DONE
UPDATE payment SET payment.status = 'DONE' WHERE bookingid =
\$bookingid
UPDATE payment SET paymentDate = curdate() WHERE bookingid =
\$bookingid

Check Availability: Here they can see how many rooms are unoccupied corresponding to Room_Type between two dates.

Note : CheckIn and CheckOut dates should be in order.

SQL Queries :

1. // query to find how many rooms are unoccupied corresponding to Room_Type between two dates.

SELECT roomtypeid, COUNT(roomtypeid) as cnt from ROOM_TYPE NATURAL
JOIN ROOM WHERE roomid IN (SELECT roomid from ROOM WHERE roomid
NOT IN (SELECT roomid from isBooked NATURAL JOIN booking WHERE
'\$checkin' < departure AND arrival < '\$checkout')) GROUP BY roomtypeid

Monthly Expenditure : Here, the receptionist needs to provide a month and year for which he wants to know about expenditures. After that he gets the information about expenditures because of different rooms booked and food services provided during that “Month-Year” on the same webpage.

SQL Queries :

1. // finding bookings in which customer stayed in given month
SELECT bookingID, foodServices FROM booking WHERE arrival <= '\$end' AND
departure > '\$start'
2. // finding room details for booking to know the cost of room
SELECT * FROM room NATURAL JOIN isBooked NATURAL JOIN
ROOM_TYPE WHERE bookingid = '\$id'
3. // finding number of guest stayed
SELECT count(*) as cnt FROM hasBooked WHERE bookingID = 'id';

Food Billings : Here also, the receptionist needs to provide a month and year for which he wants to know about expenditures. After that he gets the information about expenditures because of food services provided during that “Month-Year” on the same webpage.

SQL Queries :

1. // finding bookings in which customer stayed in given month and availed foodservices
SELECT bookingID FROM booking WHERE arrival <= 'end' AND departure > 'start' AND foodservices = 1;
2. SELECT roomid FROM room NATURAL JOIN isBooked WHERE bookingid = 'id';
3. // finding number of guest stayed
SELECT count(*) as cnt FROM hasBooked WHERE bookingID = 'id';
4. SELECT arrival,departure FROM booking WHERE bookingid = 'id';

Monthly Bookings : Here also, the receptionist needs to provide a month and year for which he wants to know about expenditures. After that he gets the information about bookings done during that “Month-Year” on the same webpage.

SQL Queries :

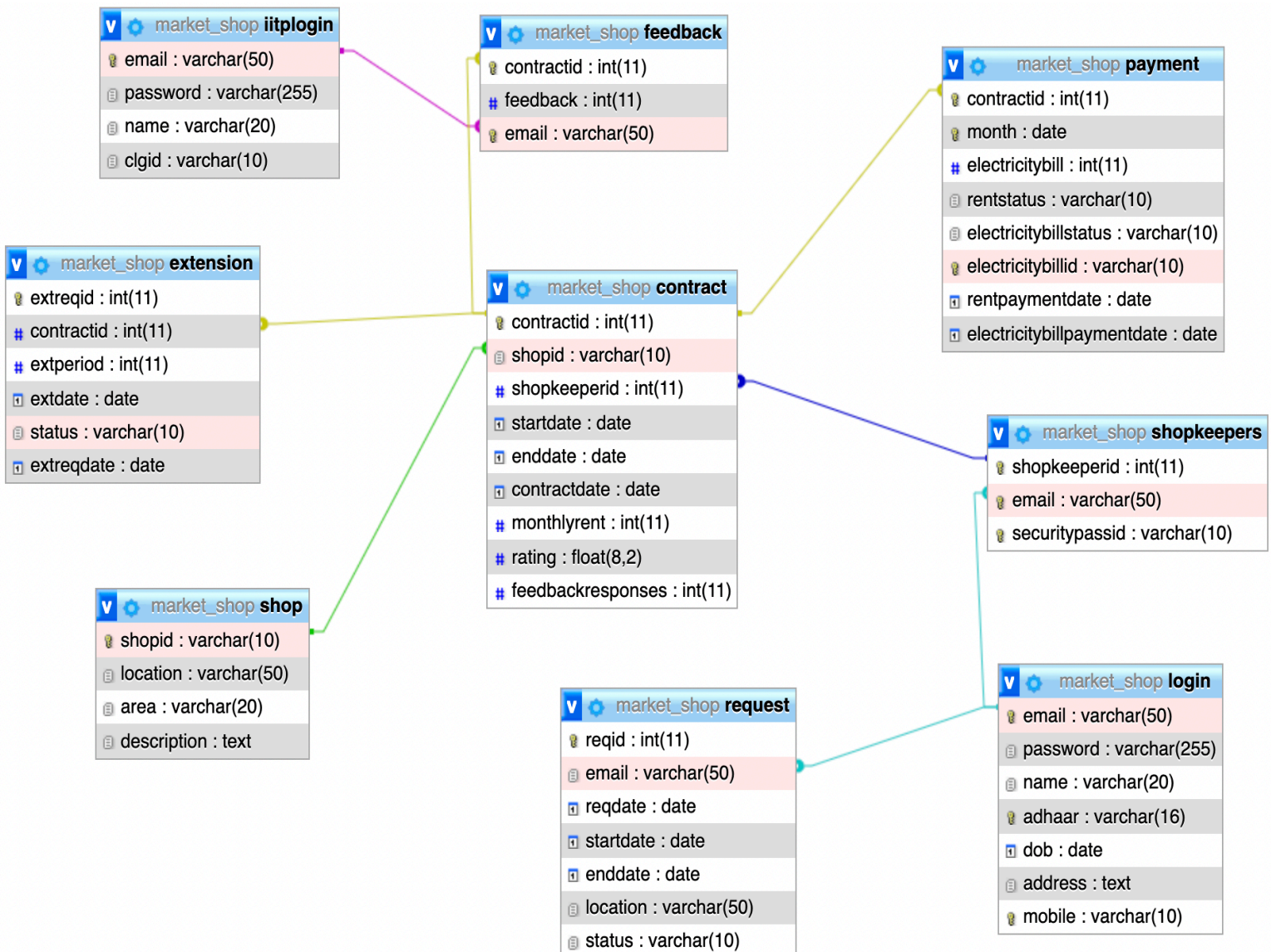
```
// to show bookings whose booking date is in given month
SELECT * FROM booking NATURAL JOIN isBooked NATURAL JOIN room
NATURAL JOIN payment WHERE bookingdate <= 'end' AND bookingdate >= 'start';
```

Logout : Receptionist can log out by clicking the logout button on the navigation bar.

Market Shop

Database:

ER-Diagram:






Tables:

Following tables are used in Market Shop:

1. **Login Table:** This is used to store information about admin and shopkeepers.

Primary Key: email


Unique: Aadhaar, mobile

#	Name	Type	Collation	Attributes	Null	Default
1	email 	varchar(50)	utf8mb4_general_ci		No	None
2	password	varchar(255)	utf8mb4_general_ci		No	None
3	name	varchar(20)	utf8mb4_general_ci		No	None
4	adhaar 	varchar(16)	utf8mb4_general_ci		No	None
5	dob	date			No	None
6	address	text	utf8mb4_general_ci		No	None
7	mobile 	varchar(10)	utf8mb4_general_ci		No	None

2. **IITP Login Table:** This is used to store information about the members of IIT Patna (to collect their feedback).


Primary Key: email

Unique: clgid

#	Name	Type	Collation	Attributes	Null	Default
1	email 	varchar(50)	utf8mb4_general_ci		No	None
2	password	varchar(255)	utf8mb4_general_ci		No	None
3	name	varchar(20)	utf8mb4_general_ci		No	None
4	clgid	varchar(10)	utf8mb4_general_ci		No	None

3. **Shop Table:** This table stores information about shops.

Primary Key: shopid.




#	Name	Type	Collation	Attributes	Null	Default
1	shopid 	varchar(10)	utf8mb4_general_ci		No	<i>None</i>
2	location	varchar(50)	utf8mb4_general_ci		No	<i>None</i>
3	area	varchar(20)	utf8mb4_general_ci		No	<i>None</i>
4	description	text	utf8mb4_general_ci		Yes	<i>NULL</i>

4. **Shopkeepers Table:** This table stores information about shopkeepers who have at least once run a shop in IITP.

Primark Key: shopkeeperid

Foreign Key: email (from login table)




Unique: securitypassid

#	Name	Type	Collation	Attributes	Null	Default
1	shopkeeperid 	int(11)			No	<i>None</i>
2	email 	varchar(50)	utf8mb4_general_ci		Yes	<i>NULL</i>
3	securitypassid 	varchar(10)	utf8mb4_general_ci		No	<i>None</i>

5. **Contract Table:** This table stores information about contracts allotted to shopkeepers by the admin for the market shop.

Primary Key: contractid



Foreign Key: contractid, shopid, shopkeeperid

#	Name	Type	Collation	Attributes	Null	Default
1	contractid 	int(11)			No	None
2	shopid 	varchar(10)	utf8mb4_general_ci		No	None
3	shopkeeperid 	int(11)			No	None
4	startdate	date			No	None
5	enddate	date			No	None
6	contractdate	date			No	current_timestamp()
7	monthlyrent	int(11)			No	None
8	rating	float(8,2)			No	0.00
9	feedbackresponses	int(11)			No	0

6. **Extension Table:** This table stores information about the extension request made by shopkeepers for a shop currently owned by him.

Primary Key: extreqid



Foreign Key: contractid

#	Name	Type	Collation	Attributes	Null	Default
1	extreqid 	int(11)			No	None
2	contractid 	int(11)			No	None
3	extperiod	int(11)			No	None
4	extdate	date			No	None
5	status	varchar(10)	utf8mb4_general_ci		No	PENDING
6	extreqdate	date			No	current_timestamp()

7. **Request Table:** This table stores information about the requests made by any person for getting a shop in IITP.

Primary Key: reqid




Foreign Key: email (from login table)

#	Name	Type	Collation	Attributes	Null	Default
1	reqid 	int(11)			No	None
2	email 	varchar(50)	utf8mb4_general_ci		No	None
3	reqdate	date			No	current_timestamp()
4	startdate	date			No	None
5	enddate	date			No	None
6	location	varchar(50)	utf8mb4_general_ci		No	None
7	status	varchar(10)	utf8mb4_general_ci		No	PENDING

8. **Payment Table:** This table stores information about monthly bills for a shop.

Primary Key: (contractid,month).



Unique: electricitybillid

#	Name	Type	Collation	Attributes	Null	Default
1	contractid 	int(11)			No	None
2	month 	date			No	None
3	electricitybill	int(11)			Yes	NULL
4	rentstatus	varchar(10)	utf8mb4_general_ci		No	PENDING
5	electricitybillstatus	varchar(10)	utf8mb4_general_ci		No	PENDING
6	electricitybillid 	varchar(10)	utf8mb4_general_ci		Yes	NULL
7	rentpaymentdate	date			Yes	NULL
8	electricitybillpaymentdate	date			Yes	NULL

9. **Feedback Table:** This table stores information about feedback of shops given by IITP members.

Primary Key: (contractId,email)

Foreign Key: email (from iitplogin table)

#	Name	Type	Collation	Attributes	Null	Default
1	contractid 	int(11)			No	None
2	feedback	int(11)			No	None
3	email 	varchar(50)	utf8mb4_general_ci		No	None

Web Pages: (Works done here automatically reflect in the database)

1) Index Web Page: Initially, we land on the index page. Information about the IITP campus and shops is provided here. From here a person can go to the *Register* web page, if he/she has not registered yet and he/she wants to get a shop in IITP. People who have already registered can go to the *Login* web page directly from here. IITP members can go to the *Feedback* web page to give their feedback.

2) Register Web Page: Here, we take several information from the user like Email ID, Name, Password, Address, Date of Birth, Mobile No. and Aadhaar Card No. Every user should remember their email and passwords for logging in. If already registered, the user can visit the *Login* web page.

3) Login Web Page: There are two type of pages where one can land after logging in:

1. If someone logs in through the shopkeeper email and password, he goes on the shopkeeper welcome web page.
2. If *Admin* logs in using 'msadmin@iitp.ac.in', he goes to another welcome page.

SQL Query: SELECT * from login WHERE email = 'email';

// Query to find user in login table and match the password if found.

Options Available (If user is not 'admin'):

Welcome web page:

- User can see details about different types of shops in IITP as well as his own details.
- He can review the status of his previously made requests (if any) to get a shop in IITP. He can cancel this request if it is still 'PENDING'.

- Reminders are shown if less than 30 days are remaining before license period is expired.
- He can also see details of his/her ongoing contracts of shops (if any).
- From here he can directly go to multiple webpages like *Shop Request* web page, *Extension Request* web page, *Cancel Extension Request* web page, *Payment Details* web page, and finally logout.

SQL Queries:

1. Queries to cancel 'PENDING'
 // find PENDING request which user wants to cancel
 SELECT * FROM request WHERE email = 'email' AND request.status = 'PENDING';

 // to delete request from table if found in above query
 DELETE FROM request WHERE reqid = reqid;
2. // to show reminders
 SELECT * FROM market_shop.contract NATURAL JOIN shopkeepers NATURAL JOIN market_shop.login WHERE email = 'email' AND enddate >= curdate();
3. // to show all DENIED or PENDING requests that user have made
 SELECT * FROM request WHERE email = username AND request.status <> 'ACCEPTED'
4. //query to show all the ongoing contracts of the user
 SELECT * FROM market_shop.contract NATURAL JOIN shopkeepers NATURAL JOIN shop WHERE email = username AND enddate >= curdate();
5. // to show user details
 SELECT * FROM market_shop.login WHERE login.email = '\$email'
 SELECT * FROM shopkeepers WHERE email = '\$email'
 SELECT max(enddate) as validity FROM contract WHERE shopkeeperid = \$shopkeeperid AND enddate >= curdate() AND startdate <= curdate()

Shop Request Web Page:

(Assumption: A person can have only one 'PENDING' shop request at a time.)

- Here a person can see his/her previously made 'shop request' details.
- For a new shop request, he needs to provide the time period for which he wants to get a shop (start month and end month) in corresponding blanks.

[Note: He can't fill past dates, if he does so, an error comes on screen, showing "Invalid Details"].

- After providing relevant time period, he needs to select the location at which he desires to get a shop they want.

[Note: If no shop is free in the corresponding time period, an error comes on screen, showing "No Available Shops"].

SQL Queries:

1. // to check if user already have a PENDING shop request or not
SELECT * FROM request WHERE email = 'username' AND request.status = 'PENDING';
2. // finding shops available in given time interval and location
SELECT shopid FROM shop WHERE shop.location = 'location' AND shopid NOT IN (SELECT shopid FROM market_shop.contract NATURAL JOIN shop WHERE shop.location = 'location' AND startdate < enddate AND enddate > startdate);
3. // finding max reqid from request table
SELECT max(reqid) FROM request;
4. // inserting values in request table if user has no PENDING request
INSERT INTO request (reqid,email,startdate,enddate,request.location) VALUES (reqid,'username','startdate','enddate','location');
5. // displaying user's PENDING/DENIED requests
SELECT * FROM request WHERE email = username AND request.status <> 'ACCEPTED';

Extension Request Web Page:

(Assumption: A user can have only one extension request per ongoing contractid having status 'ACCEPTED' or 'PENDING'.)

- At the top of this webpage, the user can see previous extension request details and ongoing contract details.
- After that he needs to fill Contract ID and by how many months he wants to get that contract extended.

[Note: If the user does not provide valid details, screen shows "Invalid Details".]

SQL Queries:

1. Queries for extension request :
// checking if contract id provided by user is valid or not
SELECT * FROM market_shop.contract NATURAL JOIN shopkeepers WHERE email = 'email' AND contractid = contractid;

// checking if user already have a ACCEPTED/PENDING extension request
SELECT * FROM extension NATURAL JOIN market_shop.contract NATURAL JOIN shopkeepers WHERE email = 'email' AND contractid = contractid AND extension.status <> 'DENIED';

// finding max extreqid from extension table to calculate new extreqid
SELECT max(extreqid) as id FROM extension;

```
// to get contract details
SELECT * FROM market_shop.contract WHERE contractid = contractid;
```

```
// inserting values in extension request table if all condition are met
INSERT INTO extension (extreqid,contractid,extperiod,extdate) VALUES
(id,contractid,extperiod,'extdate');
```

2. Query to display extension requests made by user:
SELECT * FROM extension NATURAL JOIN market_shop.contract NATURAL JOIN shopkeepers WHERE email = username;
3. //query to show all the ongoing contracts of the user
SELECT * FROM market_shop.contract NATURAL JOIN shopkeepers NATURAL JOIN shop WHERE email = username AND enddate >= curdate();

Cancel Extension Request Web Page:

- At the top of the webpage shopkeeper can see previous extension request details.
- After that he/she needs to write a valid Extension Request ID which they want to cancel.
[Note: In case of invalid Extension Request ID, screen shows “Invalid Details”.]

SQL Queries:

1. Queries to cancel request:

```
// checking if valid extreqid is provided
SELECT * FROM extension NATURAL JOIN market_shop.contract NATURAL JOIN
shopkeepers WHERE email = 'email' AND extreqid = id AND extension.status =
'PENDING';
```

```
// canceling request if valid
DELETE FROM extension WHERE extreqid = id;
```

2. Query to display extension requests made by user:
SELECT * FROM extension NATURAL JOIN market_shop.contract NATURAL JOIN shopkeepers WHERE email = username;

Payment Details Web Page:

- Here the shopkeeper can see two types of bill details, one for the rent of the shop and one for the electricity bill.
- In Shop bill details, he/she can see bill details till the ‘end month’ of the present contract.
- In Electricity bills he/she can see only bills till present month.

SQL Queries:

1. // displaying rent payment details
SELECT * FROM payment NATURAL JOIN market_shop.contract NATURAL JOIN shopkeepers NATURAL JOIN shop WHERE email = 'email';
2. // displaying electricity bill payment details
SELECT * FROM payment NATURAL JOIN market_shop.contract NATURAL JOIN shopkeepers NATURAL JOIN shop WHERE email = 'email' AND electricitybill = 'NULL';

Logout Web Page: Finally, the user can logout by clicking on Logout button on navigation bar.

Options Available for Admin:

Admin Welcome Web Page:

Admin can see many details on this webpage like:

1. Details of Ongoing Contracts.
2. Details of all the shopkeepers till this date.
3. Details of shops.

Request Details Web Page:

- At the top, the admin can see the details of 'PENDING' shop requests.
- Admin can accept or deny any request on this web page. For that he/she needs to write *Request ID* and select *Accept / Deny* option accordingly.
[Note: Admin can only write Request ID from above table otherwise screen will show "Invalid Details"].
- Now, if admin approves a request here, then he/she lands on a new web page where he/she needs to fill in Shop ID and Monthly rent of the shop which will be provided to the shopkeeper.
[Note: As soon as admin approves shop to shopkeeper, 'payment' table will automatically be updated.]

SQL Queries:

1. // checking if reqid is valid
SELECT * FROM request WHERE reqid = reqid AND request.status = 'PENDING';
2. // Query If request id DENIED
UPDATE request SET request.status = 'DENIED' WHERE reqid = reqid;

// queries if request is ACCEPTED :

// get details of request
SELECT * FROM request WHERE reqid = \$reqid

```
// insert a new contract in table
INSERT INTO market_shop.contract
(contractid,shopid,shopkeeperid,startdate,enddate,monthlyrent) VALUES
($contractid,$shopid,$shopkeeperid,$startdate,$enddate,$monthlyrent')

// delete the accepted request from table
DELETE FROM request WHERE reqid = $reqid
3. // to display details of 'PENDING' shop requests
SELECT * FROM request WHERE request.status = 'PENDING';
```

Add bills Web page:

- Here, admin adds the monthly electricity bill of a shop.
 - To add electricity bill admin needs to fill Contract ID, Month, Electricity Bill ID and Electricity Bill here.
- [Note:** If admin provides invalid Contract ID, screen shows “Invalid Contract ID”]

SQL Queries:

1. // updating electricitybillid
UPDATE payment SET electricitybillid = 'electricitybillid' WHERE contractid = contractid AND payment.month = 'month';
2. // updating electricity bill amount
UPDATE payment SET electricitybill = electricitybill WHERE contractid = contractid AND payment.month = 'month';

Update Bills Web Page:

- On this web page, admin updates the status of bills paid by a shopkeeper.
 - He provides a Contract ID and a Month. After that he/she selects 'Pending/Paid/NA' for both the Electricity bill and Rent bill section.
- NA represents that the payment status of that section need not be updated.

SQL Queries:

1. // updating payment status for electricity bill
UPDATE payment SET electricitybillstatus = 'ebillstatus' WHERE contractid = contractid AND payment.month = 'month';
2. // updating payment status for rent
UPDATE payment SET rentstatus = 'rentstatus' WHERE contractid = contractid AND payment.month = 'month';

Extension Request Web Page:

- Here admin can see the 'PENDING' extension requests made by shopkeepers at top of the page.
 - Admin needs to provide a valid Extension Request ID and then choose accept/deny accordingly.
- [Note:** If admin chooses to approve the extension request, then 'payment' table will be updated automatically.]

SQL Queries:

1. Queries to accept/deny extension requests:

```
// update only status of request if DENIED
UPDATE extension SET extension.status = 'DENIED' WHERE extreqid = extreqid;
```

```
// checking if extreqid is valid
SELECT * FROM extension WHERE extreqid = extreqid AND extension.status =
'PENDING';
```

```
// getting contract details of the contract for which extension is requested
SELECT * FROM extension NATURAL JOIN market_shop.contract WHERE extreqid
= extreqid;
```

```
// checking if shop is already booked or not in the extension period provided
SELECT * FROM contract WHERE shopid = 'shopid' AND startdate <= 'en' AND
startdate >= 'st';
```

```
// update status of request to ACCEPTED
UPDATE extension SET extension.status = 'accept_deny' WHERE extreqid =
extreqid;
```

```
// inserting payment details for extended months
INSERT INTO payment (contractid,payment.month) VALUES (contractid,'x');
```

```
// updating end date of contract to extension date
UPDATE contract SET enddate = 'en' WHERE contractid = contractid;
```

2. // displaying PENDING extension requests:
SELECT * FROM extension WHERE extension.status = 'PENDING';

Logout Web Page: At the end, admin can log out by clicking Logout button on navigation bar.

- 4) **Feedback Home Web Page:** On this web page, the average rating and the number of responses provided for all shops are shown here. For giving feedback, the IITP member first has to login by clicking on login button on navigation bar. After that, he/she lands to a new web page where he/she can give feedback to any shop only if he/she has not given it earlier. After giving feedback he/she can logout from here.

SQL Queries used for logging in:

1. // Query to find user in iitplogin table and match the password if found.
SELECT * from iitplogin WHERE email = 'email';

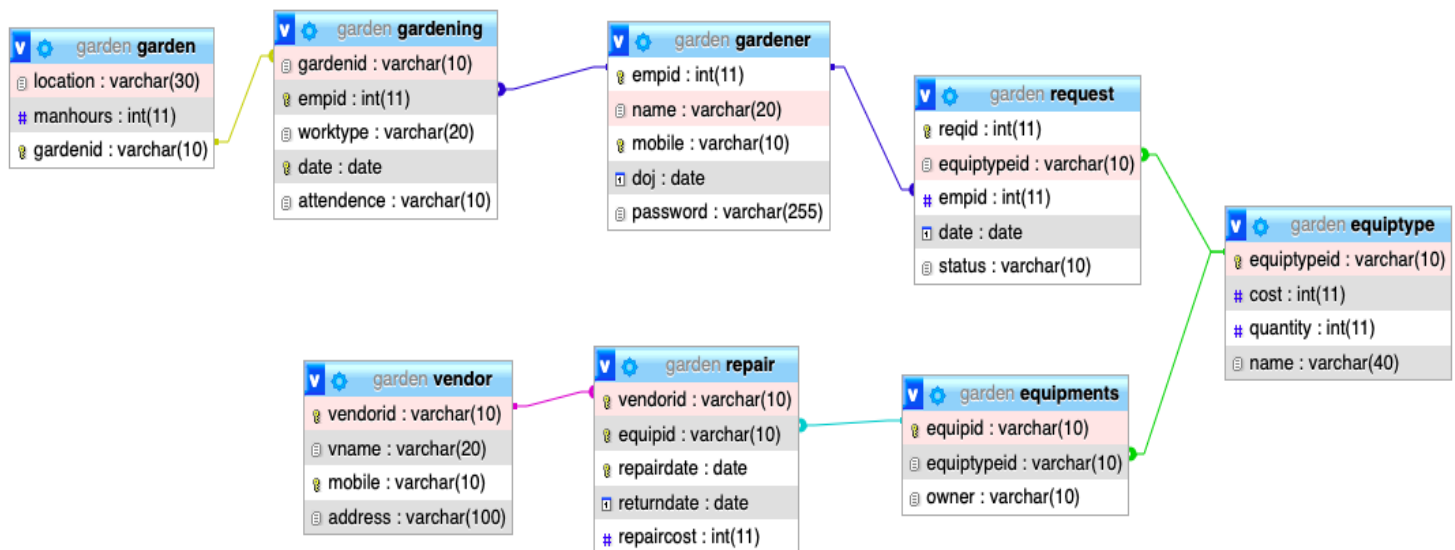
SQL Queries used for providing feedback:

1. // checking if user has already provided feedback for that shop
SELECT * FROM feedback NATURAL JOIN contract WHERE email = 'email' AND shopid = 'shopid';
2. // inserting feedback given by user
INSERT INTO feedback VALUES (contractid,rating,'email');
3. // updating rating of shop
UPDATE contract SET rating = newrating WHERE contractid = contractid;
4. // updating count of feedback responses
UPDATE contract SET feedbackresponses = newresponse WHERE contractid = contractid;

Landscaping

Database :


ER-Diagram:



Tables used in Landscaping:



1. **Garden Table** : This table used to store information about gardens and man hours per day.

Primary Key : GardenId

#	Name	Type	Collation	Attributes	Null	Default
1	location	varchar(30)	utf8mb4_general_ci		No	None
2	manhours	int(11)			No	None
3	gardenid 	varchar(10)	utf8mb4_general_ci		No	None

2. **Gardener Table** : This table used to store information about gardeners .




Primary Key : empid.

#	Name	Type	Collation	Attributes	Null	Default
1	empid 	int(11)			No	None
2	name	varchar(20)	utf8mb4_general_ci		No	None
3	mobile 	varchar(10)	utf8mb4_general_ci		No	None
4	doj	date			No	current_timestamp()
5	password	varchar(255)	utf8mb4_general_ci		No	None

3. **Gardening Table** : This table used to store information about a gardener's work.

Primary Key : (gardenid, date).




Foreign Key : empid,gardenid

#	Name	Type	Collation	Attributes	Null	Default	Comments
1	gardenid 	varchar(10)	utf8mb4_general_ci		No	None	
2	empid 	int(11)			No	None	
3	worktype	varchar(20)	utf8mb4_general_ci		No	None	
4	date 	date			No	None	
5	attendance	varchar(10)	utf8mb4_general_ci		Yes	NULL	

4. **Request Table** : This table used to store information about requests by gardeners to take equipment.


Primary Key : reqid

Foreign Key : equiptypeid, empid

#	Name	Type	Collation	Attributes	Null	Default
1	reqid 	int(11)			No	None
2	equiptypeid 	varchar(10)	utf8mb4_general_ci		No	None
3	empid 	int(11)			No	None
4	date	date			No	current_timestamp()
5	status	varchar(10)	utf8mb4_general_ci		No	PENDING

5. **EQUIPTYPE Table** : This table used to store information about equipments .



Primary Key : equiptypeid.

#	Name	Type	Collation	Attributes	Null	Default	Comments
1	equiptypeid 	varchar(10)	utf8mb4_general_ci		No	None	
2	cost	int(11)			No	None	
3	quantity	int(11)			No	None	
4	name	varchar(40)	utf8mb4_general_ci		No	None	

6. **Equipments Table** : This table used to store information about different equipment types.

Primary Key : (equipid,equiptypeid)

Foreign Key : equipid,equiptypeid.

#	Name	Type	Collation	Attributes	Null	Default	Comments
1	equipid 	varchar(10)	utf8mb4_general_ci		No	None	
2	equiptypeid 	varchar(10)	utf8mb4_general_ci		No	None	
3	owner	varchar(10)	utf8mb4_general_ci		No	IITP	

7. **Vendor Table** : This table is used to store information about vendors .
Primary Key : vendorid

#	Name	Type	Collation	Attributes	Null	Default	Comments
1	vendorid 🔑	varchar(10)	utf8mb4_general_ci		No	None	
2	vname	varchar(20)	utf8mb4_general_ci		No	None	
3	mobile 🔑	varchar(10)	utf8mb4_general_ci		No	None	
4	address	varchar(100)	utf8mb4_general_ci		No	None	

8. **Repair Table** : This table used to store information about repairs done.
Primary Key : (vendorid,equipid,date)
Foreign Key: vendorid,equipid

#	Name	Type	Collation	Attributes	Null	Default
1	vendorid 🔑	varchar(10)	utf8mb4_general_ci		No	None
2	equipid 🔑 🔑	varchar(10)	utf8mb4_general_ci		No	None
3	repairdate 🔑	date			No	current_timestamp()
4	returndate	date			Yes	NULL
5	repaircost	int(11)			Yes	NULL

Web Page: (Works done here automatically reflect on the database.)

Register Web Page:

Here the information from the user like mobile no, Name, Password is taken. Every user should remember their mobile no and passwords for logging in. From here, the user can go to the login web page through the link provided at the end.

Login Web Page:

There are two type of login pages:

1. For Gardeners.
2. For Admin/Supervisor. (mobile no.: admin and password = admin)

// Query to find user in login table and match the password if found.

SQL Query: SELECT * from gardener WHERE mobile = 'mobile';

Options Available for Gardener:

Gardener Welcome web page :(Assumption: username is mobile no of gardener)

Here a gardener can see various details:

1. Monthly Work.
2. Equipment Request Status.
3. Borrowed Equipments

SQL Queries:

1. // to get empid of gardener
SELECT * FROM gardener WHERE mobile = 'username';
2. // to show whole month's work assigned
SELECT * FROM gardener NATURAL JOIN gardening NATURAL JOIN garden
WHERE empid = empid AND DATE_FORMAT(date, '%Y-%m') =
DATE_FORMAT(curdate(), '%Y-%m');"
3. // to show PENDING/DENIED equipment requests
SELECT * FROM request WHERE empid = empid AND request.status <>
'ACCEPTED';
4. // to show equipments borrowed from IITP
SELECT * FROM equipments NATURAL JOIN equiptype WHERE owner = 'empid';

Equipment Request Web Page :

- Here a gardener can see his/her previous equipment request details.
- For a new equipment request, he needs to submit equipment type from given equipment type options.

SQL Queries:

1. // to get empid of gardener
SELECT * FROM gardener WHERE mobile = 'username';
2. // finding max reqid to calculate new id
SELECT max(reqid) as id FROM request;
3. // inserting new request made by gardener
INSERT INTO request(reqid, equiptypeid, empid) VALUES(id, 'equiptype', empid);
4. // to show PENDING/DENIED equipment requests
SELECT * FROM request NATURAL JOIN equiptype WHERE empid = empid AND
request.status <> 'ACCEPTED';
5. // to show types of equipments gardener can borrow
SELECT * FROM equiptype;

Cancel Equipment Request Web Page :

- Here, the gardener can see his/her 'PENDING' equipment request details.
- After that he/she needs to provide a valid requestID of the request, he/she wants to cancel.
[Note: In case of invalid requestID a error shows on screen with "Invalid Details".]

SQL Queries :

1. // to get empid of gardener
SELECT * FROM gardener WHERE mobile = 'username';
2. // checking if reqid is valid
SELECT * FROM request WHERE reqid = reqid AND empid = empid AND status = 'PENDING';
3. // cancelling request
DELETE FROM request WHERE reqid = reqid;
4. // to show PENDING equipment requests
SELECT * FROM request NATURAL JOIN equiptype WHERE empid = empid AND request.status = 'PENDING';

Logout Link : Finally there is a logout button which takes the user back to login page.

Options Available for Supervisor:

Supervisor Welcome Web Page :

- Supervisor can see many details on this webpage like :
 1. Work Assignments Today Onwards of this month.
 2. Garden Details
 3. Gardener Details
 4. Equipments in IITP stock
 5. Equipments borrowed by gardeners
 6. Equipments sent for repair
- From here a supervisor can go to multiple web pages like Mark attendance web page, Update work web page, request details web page, repair web page, repair return web page, equipment return web page.

SQL Queries :

- // Queries to display above mentioned
1. SELECT * FROM gardener NATURAL JOIN gardening NATURAL JOIN garden WHERE date >= curdate();
 2. SELECT * FROM garden;
 3. SELECT * FROM gardener WHERE empid <> 0;
 4. SELECT * FROM equipments NATURAL JOIN equiptype WHERE owner = 'IITP';
 5. SELECT * FROM equipments NATURAL JOIN equiptype WHERE owner NOT LIKE 'V-%' AND owner <> 'IITP';
 6. SELECT * FROM gardener WHERE empid = owner;
 7. SELECT * FROM equipments NATURAL JOIN equiptype NATURAL JOIN repair NATURAL JOIN vendor WHERE owner LIKE 'V-%';

Mark Attendance Web Page :

- Here a supervisor can see details about gardeners who have duties today.
- He can update the attendance of gardeners.

SQL Queries :

1. // mark attendance as provided by supervisor
UPDATE gardening SET attendance = 'attendance' WHERE empid = empid AND date = curdate();
2. // get all gardener details
SELECT * FROM gardener WHERE empid <> 0;
3. // counting attendance of gardener
SELECT count(*) as cnt FROM gardening WHERE empid = empid AND attendance = 'PRESENT' AND date >= 'start' and date <= curdate();
4. // get today's attendance
SELECT * FROM gardening WHERE empid = empid AND date = curdate();
5. SELECT * FROM gardening NATURAL JOIN gardener WHERE date = curdate();

Update Work WebPage :

- Here, the supervisor can see details of Gardeners and Gardens.
- Supervisor can update work for any gardener for upcoming dates.

SQL Queries :

1. // updating work for gardener
UPDATE gardening SET gardenid = 'gardenid' WHERE empid = empid AND date = 'date';
UPDATE gardening SET worktype = 'worktype' WHERE empid = \$empid AND date = 'date';
2. // display all gardener details
SELECT * FROM gardener WHERE empid <> 0;
3. // display all garden details
SELECT * FROM garden;

Request Details Web Page :

- Here the supervisor can see details of 'PENDING' equipment requests made by gardeners.
- Supervisor needs to fill requestID and select accept/deny accordingly.
[Note : The request corresponding to the input Request ID must be 'PENDING'.]

SQL Queries :

1. // checking if reqid is valid
SELECT * FROM request WHERE reqid = reqid AND status = 'PENDING';
2. // if request is DENIED set status = 'DENIED'
UPDATE request SET request.status = 'DENIED' WHERE reqid = reqid;

3. // If ACCEPTED

// checking if equipment is available

```
SELECT * FROM equipments WHERE owner = 'IITP' AND equiptypeid =  
'equiptypeid' LIMIT 1;
```

//set status = 'DENIED' if not available

```
UPDATE request SET request.status = 'DENIED' WHERE reqid = reqid;
```

// set status = 'ACCEPTED' and owner = 'empid' if available

```
UPDATE request SET request.status = 'ACCEPTED' WHERE reqid = reqid;  
UPDATE equipments SET owner = 'empid' WHERE equipid = '$equipid';
```

4. // Displaying PENDING requests

```
SELECT *, gardener.name as name1 FROM request JOIN gardener JOIN equiptype  
WHERE request.empid = gardener.empid AND request.equiptypeid =  
equiptype.equiptypeid AND request.status = 'PENDING';
```

Repair Web page :

- Supervisor can see Equipment in IITP stock and Vendor details.
- For a repair of equipment, the supervisor needs to fill in valid equipmentID and vendorID.

SQL Queries :

1. // checking if equipid is valid and is in IITP stock
SELECT * FROM equipments WHERE equipid = 'equipid' AND owner = 'IITP';
2. // send in repair by setting owner to vendorid
UPDATE equipments SET owner = 'vendorid' WHERE equipid = 'equipid';
3. // insert repair details in repair table
INSERT INTO repair(vendorid, equipid) VALUES ('vendorid', 'equipid');
4. // displaying vendor details and equipment in stock
SELECT * FROM equipments NATURAL JOIN equiptype WHERE owner =
'IITP';SELECT * FROM vendor;

Repair return Web Page :

- At the top of this web page the supervisor can see details of equipment which is going for repair. Details like equipmentID, equipmenttype, equipmenttypeid, date of sent in repair, vendorID, vendor name and their mobile no.
- On this page supervisors need to fill equipmentID which must be present above and cost of that repair .
- At last, clicking on "Return" shows "Successful".

SQL Queries :

1. // checking if equipid is valid and is with a vendor
SELECT * FROM equipments WHERE equipid = 'equipid' AND owner LIKE 'V-%';

2. // if valid return to IITP stock and update cost , date in repair table
 UPDATE equipments SET owner = 'IITP' WHERE equipid = 'equipid';
 UPDATE repair SET repaircost = cost WHERE equipid = 'equipid' and vendorid = 'owner' and returndate = '2001-01-01';
 UPDATE repair SET returndate = 'today' WHERE equipid = 'equipid' and vendorid = 'owner' and returndate = '2001-01-01';
3. // displaying equipments in repair
 SELECT * FROM equipments NATURAL JOIN equiptype NATURAL JOIN repair NATURAL JOIN vendor WHERE owner LIKE 'V-%';

Equipment return web page :

- On this web page the supervisor can see details of equipment which are sent for repair.
- Here the supervisor needs to fill equipmentID which has been returned.

SQL Queries :

1. // checking if equipid is valid and is with a gardener
 SELECT * FROM equipments WHERE equipid = 'equipid' AND owner <> 'IITP' AND owner <> 'V-%';
2. // if valid return to IITP stock
 UPDATE equipments SET owner = 'IITP' WHERE equipid = 'equipid';
3. // displaying equipments borrowed by gardeners
 SELECT * FROM equipments NATURAL JOIN equiptype WHERE owner NOT LIKE 'V-%' AND owner = 'IITP';
 SELECT * FROM gardener WHERE empid = owner;

Logout Web page : At the end admin can log out by clicking the logout button on navigation bar.