Scope of Project:

Our team has designed a buying and selling platform which has many functions of creating a users account and then the user can list items to sell on our website and also have function to raise a request to buy products listed by other users over the site and the users can also add items to their wishlist and can delete them also from website and all these information is stored in database so this is the scope in which our database works

Stakeholders:

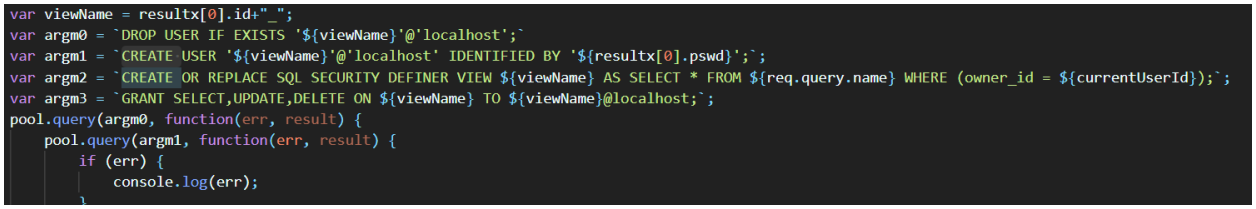
Some stakeholders of this project are:
1)Users(sellers) who can list their products on the website for selling

2)Users(buyers) who can raise a request to the other user for buying products .

ER-DIAGRAM & SCHEMA-:

https://lucid.app/lucidchart/a7cb4b1b-d276-421b-84eb-1c54d7f99edf/edit?invitationId=inv_763997a8-57c8-41e1-b73c-babdc7223cdf&page=0_0#

Views and grants:



```
var viewName = resultx[0].id+"_";
var argm0 = `DROP USER IF EXISTS '${viewName}'@'localhost';`
var argm1 = `CREATE USER '${viewName}'@'localhost' IDENTIFIED BY '${resultx[0].pswd}';`;
var argm2 = `CREATE OR REPLACE SQL SECURITY DEFINER VIEW ${viewName} AS SELECT * FROM ${req.query.name} WHERE (owner_id = ${currentUserId});`;
var argm3 = `GRANT SELECT,UPDATE,DELETE ON ${viewName} TO ${viewName}@localhost;`;
pool.query(argm0, function(err, result) {
    pool.query(argm1, function(err, result) {
        if (err) {
            console.log(err);
```

```
CREATE VIEW USERS_VIEW AS SELECT name,email FROM users;
CREATE VIEW REQUEST_VIEW AS SELECT customer_id,productName FROM requests;
CREATE VIEW car_VIEW AS
SELECT name,price
FROM car;
CREATE VIEW phone_VIEW AS
SELECT name,price
FROM phone;
CREATE VIEW other_VIEW AS
SELECT name,price
FROM other;
CREATE VIEW home_VIEW AS
SELECT adTitle,price
FROM home;
CREATE VIEW bike_VIEW AS
SELECT image,price
FROM bike;
CREATE VIEW electronicapp_VIEW AS
SELECT name,price FROM electronicapp;
CREATE VIEW furniture_VIEW AS SELECT name,price FROM furniture;
CREATE VIEW wishlist_VIEW AS SELECT owner_id,customer_id FROM wishlist;
```

We have created these views as shown above:

1) For the users table we have created a view for name and email.
2) For the requests table we have created a view for customer_id and productName.
3) For the car, phone, electronicapp, other and furniture table we have created a view for name and price.
4) For the bike table we have created a view for image and price.
5) For the home table we have created a view for adTitle and price.
6) For the wishlist table we have created a view for owner_id and customer_id.

Sql queries:

1. INSERT INTO users (name,email,phone,pswd) VALUES ("Ksh","aa@a","+919205564288","a");

2. select email,count(id) as numOfRows,id,pswd from users where email = "aa@a" ;

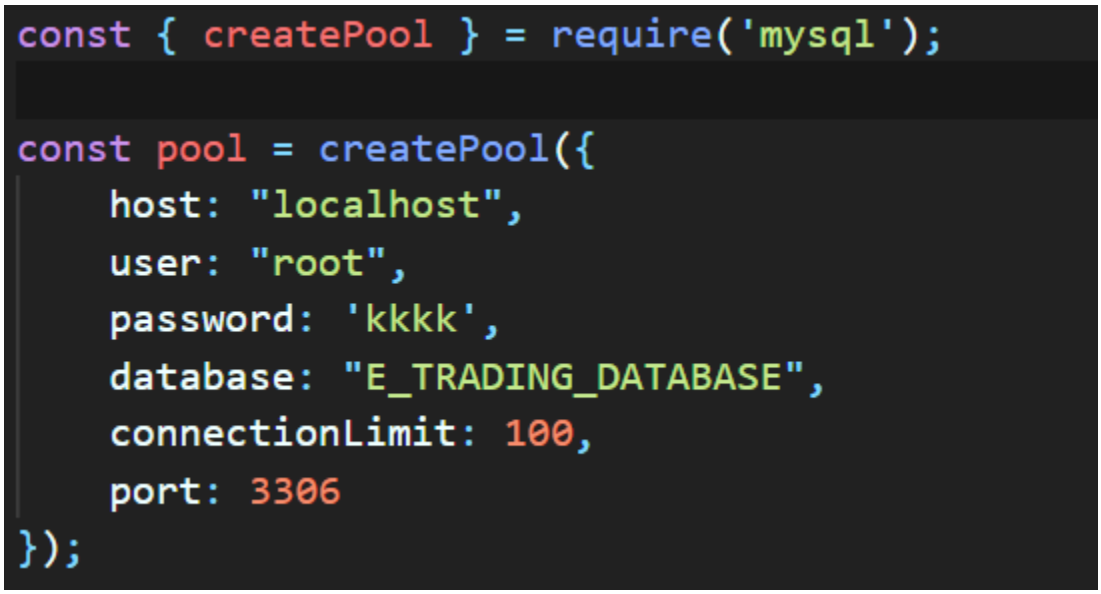
3. select 'car' as table_name, car.entryTime,car.owner_id,car.name,car.price,car.image,car.location from car union all select 'pet'
as table_name, pet.entryTime,pet.owner_id,pet.name,pet.price,pet.image,pet.location from pet union all select 'phone' as

```
table_name, phone.entryTime,phone.owner_id,phone.name,phone.price,phone.image,phone.location from
phone union all select 'other' as
table_name, other.entryTime,other.owner_id,other.name,other.price,other.image,other.location from other
union all select 'furniture'
as table_name,
furniture.entryTime,furniture.owner_id,furniture.name,furniture.price,furniture.image,furniture.location from
furniture
union all select 'bike' as table_name,
bike.entryTime,bike.owner_id,bike.name,bike.price,bike.image,bike.location from bike union all
select 'electronicapp' as table_name,
electronicapp.entryTime,electronicapp.owner_id,electronicapp.name,electronicapp.price,electronicapp.imag
e,electronicapp.location
from electronicapp union all select 'home' as table_name,
home.entryTime,home.owner_id,home.adTitle,home.price,home.image,home.location from home;

4. select AVG(price) from furniture;

5. CREATE OR REPLACE SQL SECURITY DEFINER VIEW 1_ AS SELECT * FROM phone WHERE (owner_id
=3);

6. delete from requests where owner_id = 10 and entryTime = '2022-4-27 15:0:42' and status = 'pending';


7. select (select name from users where id = r.owner_id) as o_name ,u.name as c_name,u.email as
c_email,r.productName,r.entryTime,r.purchaseTime,r.status,r.owner_id,r.customer_id
        from users as u INNER JOIN requests as r on r.customer_id = u.id;

8. select car.entryTime,car.owner_id,car.name,car.price,car.image,car.location from car where car.price >= 2
and car.price <= 5 order by car.price asc;


9. INSERT  wishlist (customer_id,owner_id,tableName,entryTime) SELECT  10,1,'car', '1998-1-23 12:59:56'
WHERE NOT EXISTS
  (  SELECT  1
     FROM    wishlist
     WHERE   customer_id = 10
     AND     owner_id = 1
     AND     tableName = 'car'
     AND     entryTime = '1998-1-23 12:59:56'
  );

10. INSERT requests(customer_id,owner_id,productName,entryTime,purchaseTime,status)
          SELECT 9,10,'Kshitij kumar','2022-4-27 15:0:42',null,'pending' WHERE NOT EXISTS
            (  SELECT  1
               FROM    requests
               WHERE   customer_id = 9
               AND     owner_id = 10
               AND     productName = 'Kshitij kumar'
               AND     entryTime = '2022-4-27 15:0:42'
               AND           purchaseTime is null
               AND           status = 'pending'
            );


11. UPDATE requests
  SET purchaseTime = CASE WHEN status = 'pending'
                  THEN '2022-4-27 17:20:9'
```

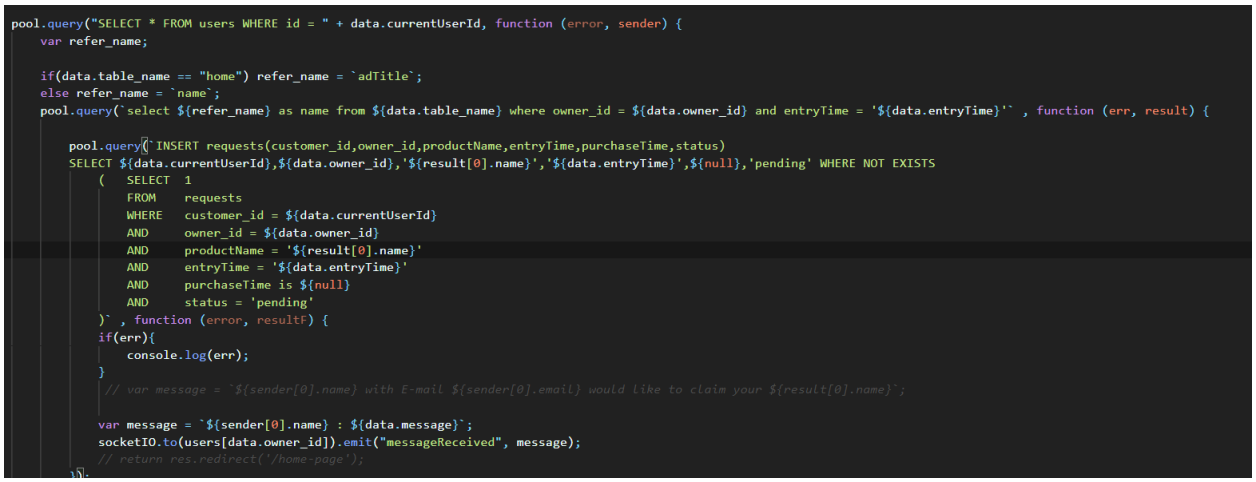
```
                ELSE purchaseTime
            END,
    status = CASE WHEN status = 'pending'
                THEN 'approved'
                ELSE status
            END
WHERE owner_id= 10 and entryTime = '2022-4-27 15:0:42';
```

Embedded SQL queries:

At various places in the backend code we have used embedded SQL queries to fetch the data from the database we have created.



```
const { createPool } = require('mysql');

const pool = createPool({
    host: "localhost",
    user: "root",
    password: 'kkkk',
    database: "E_TRADING_DATABASE",
    connectionLimit: 100,
    port: 3306
});
```



```
pool.query("SELECT * FROM users WHERE id = " + data.currentUserId, function (error, sender) {
    var refer_name;

    if(data.table_name == "home") refer_name = `adTitle`;
    else refer_name = `name`;
    pool.query(`select ${refer_name} as name from ${data.table_name} where owner_id = ${data.owner_id} and entryTime = '${data.entryTime}'` , function (err, result) {

        pool.query(`INSERT requests(customer_id,owner_id,productName,entryTime,purchaseTime,status)
        SELECT ${data.currentUserId},${data.owner_id},'${result[0].name}','${data.entryTime}',${null},'pending' WHERE NOT EXISTS
            (   SELECT  1
                FROM    requests
                WHERE   customer_id = ${data.currentUserId}
                AND     owner_id = ${data.owner_id}
                AND     productName = '${result[0].name}'
                AND     entryTime = '${data.entryTime}'
                AND     purchaseTime is ${null}
                AND     status = 'pending'
            )` , function (error, resultF) {
            if(err){
                console.log(err);
            }
            // var message = `${sender[0].name} with E-mail ${sender[0].email} would like to claim your ${result[0].name}`;

            var message = `${sender[0].name} : ${data.message}`;
            socketIO.to(users[data.owner_id]).emit("messageReceived", message);
            // return res.redirect('/home-page');
        });
```

```
pool.query(`select count(id) as numOfRows from users where email = "${req.body.email}" `, function(err, result) {
            if (err) {
                return console.log(err);
            }
            var num = result[0].numOfRows;
            if(num == 0){

                pool.query(`INSERT INTO users (name,email,phone,pswd) VALUES ("${req.body.name}","${req.body.email}","${req.body.phone}","${req.body.password}")`, function(err, result) {
                    if (err) {
                        return console.log(err);
                    }
                });
                return res.render("signIn");
            }else{
                return res.redirect('back');
            }
        });
```

```
pool.query(`select * from wishlist where customer_id = ${currentUserId}`,  function(err, wishlist) {
   if(wishlist.length == 0){
    pool.query(`select (select name from users where id = r.owner_id) as o_name ,u.name as c_name,u.email as c_email,r.productName,r.entryTime,r.purchaseTime,r.status,r.owner_id,r.customer_id
    from users as u INNER JOIN requests as r on r.customer_id = u.id`, function(err, request) {
        if(err)console.log(err);
        // console.log(request);
        for(let k  = 0;k<request.length;k++){
            request[k].entryTime = giveDate(request[k].entryTime);
            if(request[k].purchaseTime != null){
                request[k].purchaseTime = giveDate(request[k].purchaseTime);
            }
        }
        return res.render('home',{finalRes:finalRes,user_id:user_id,wishlistItems:wishlistItems,currentUserId:currentUserId,request,request});
    });
   }
```

```
app.get('/wishlistA', function(req, res){
    if(!req.isAuthenticated()){
        //try to redirect
        return res.render('signIn');
    }

    pool.query(`INSERT  wishlist (customer_id,owner_id,tableName,entryTime) SELECT  ${currentUserId},${req.query.owner_id},'${req.query.name}', '${req.query.entryTime}' WHERE NOT EXISTS
    (   SELECT  1
        FROM    wishlist
        WHERE   customer_id = ${currentUserId}
        AND     owner_id = ${req.query.owner_id}
        AND     tableName = '${req.query.name}'
        AND     entryTime = '${req.query.entryTime}'
    );`, function(err, result) {
        if(err)console.log(err);
        req.flash('success',"Added to Wishlist");
        return res.redirect('/home-page');
    });
```

Query optimization:

Will be shown during the demo.

Indexing:

We have identified 10 attributes to create index tables for our sql queries:

create index cartime on car(price,entryTime);
create index biketime on bike(price,entryTime);
create index pettime on pet(price,entryTime);
create index phonetime on phone(price,entryTime);
create index othertime on other(price,entryTime);
create index hometime on home(price,entryTime);
create index electronicapptime on electronicapp(price,entryTime);
create index furnituretime on furniture(price,entryTime);
create index wishlisttime on wishlist(entryTime);
create index requesttime on request(entryTime);

Triggers:

1) Created car_w, pet_w, phone_w, other_w, furniture_w, bike_w, electronicapp_w, home_w triggers for deleting the product from each users wishlist(if selected) if someone deletes his product from site.

```
DELIMITER  $$
CREATE TRIGGER car_W
BEFORE DELETE ON car FOR EACH ROW
BEGIN
delete from wishlist where owner_id = old.owner_id and tableName ='car' and entryTime = old.entryTime;
END $$
DELIMITER ;

DELIMITER  $$
CREATE TRIGGER phone_W
BEFORE DELETE ON phone FOR EACH ROW
BEGIN
delete from wishlist where owner_id = old.owner_id and tableName ='phone' and entryTime = old.entryTime;
END $$
DELIMITER ;

DELIMITER  $$
CREATE TRIGGER pet_W
BEFORE DELETE ON pet FOR EACH ROW
BEGIN
delete from wishlist where owner_id = old.owner_id and tableName ='pet' and entryTime = old.entryTime;
END $$
DELIMITER ;
```

```
DELIMITER  $$
CREATE TRIGGER other_W
BEFORE DELETE ON other FOR EACH ROW
BEGIN
delete from wishlist where owner_id = old.owner_id and tableName ='other' and entryTime = old.entryTime;
END $$
DELIMITER ;

DELIMITER  $$
CREATE TRIGGER furniture_W
BEFORE DELETE ON furniture FOR EACH ROW
BEGIN
delete from wishlist where owner_id = old.owner_id and tableName ='furniture' and entryTime = old.entryTime;
END $$
DELIMITER ;

DELIMITER  $$
CREATE TRIGGER bike_W
BEFORE DELETE ON bike FOR EACH ROW
BEGIN
delete from wishlist where owner_id = old.owner_id and tableName ='bike' and entryTime = old.entryTime;
END $$
DELIMITER ;
```

```
DELIMITER  $$
CREATE TRIGGER electronicapp_W
BEFORE DELETE ON electronicapp FOR EACH ROW
BEGIN
delete from wishlist where owner_id = old.owner_id and tableName ='electronicapp' and entryTime = old.entryTime;
END $$
DELIMITER ;

DELIMITER  $$
CREATE TRIGGER home_W
BEFORE DELETE ON home FOR EACH ROW
BEGIN
delete from wishlist where owner_id = old.owner_id and tableName ='home' and entryTime = old.entryTime;
END $$
DELIMITER ;
```