DOKUMENTASI PROYEK “WAREGARAGE”
OPREC NETLAB 2023

Muhammad Rizky Utomo, 2106731320

Universitas Indonesia

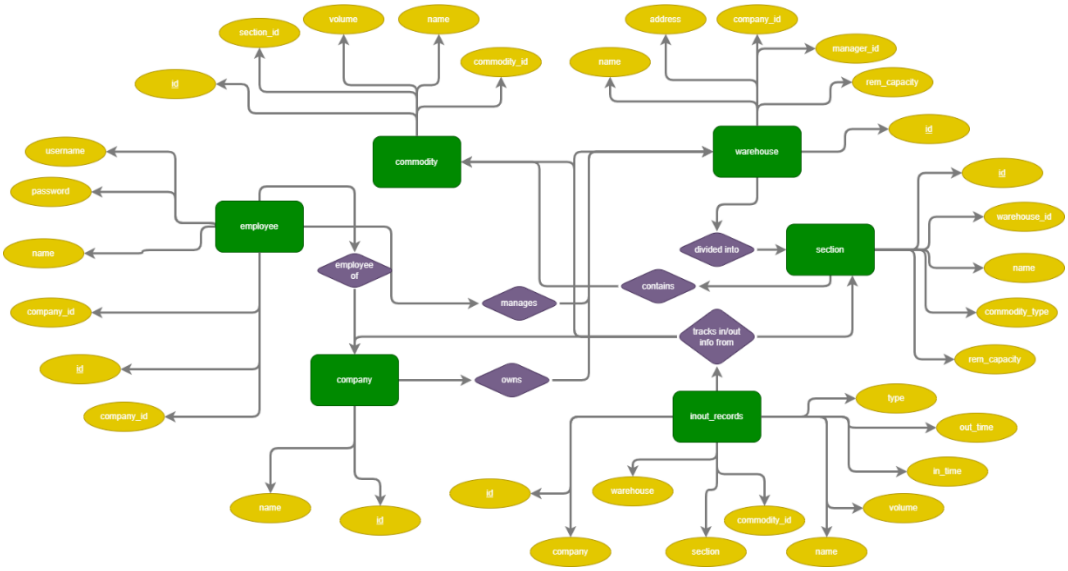
I. Topologi

Ini adalah repositori dari aplikasi WareGarage, sebuah aplikasi yang memudahkan Anda dalam melakukan manajemen terhadap gudang penyimpanan.

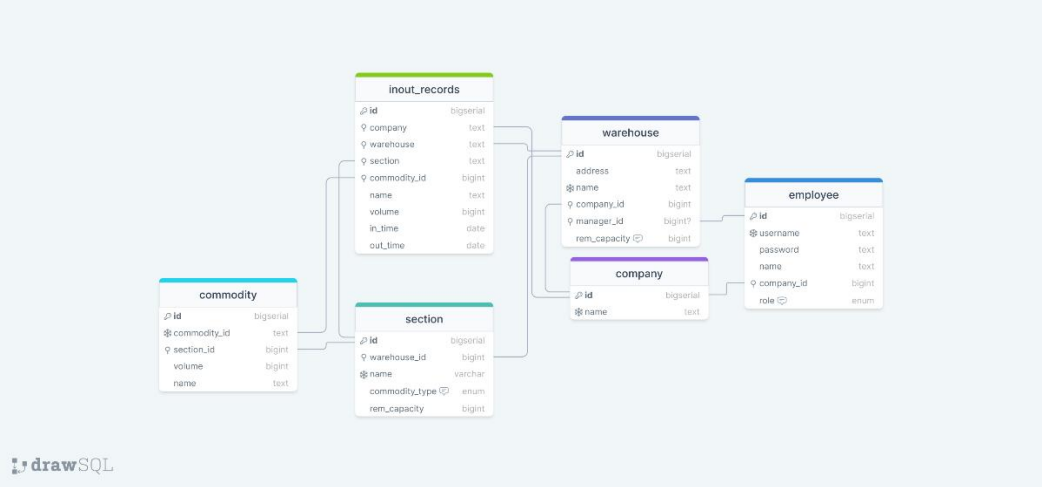
Sebuah perusahaan dapat membuat beberapa tempat penyimpanan. Tempat penyimpanan tersebut memiliki kapasitas tertentu dan dapat dibagi menjadi beberapa section yang tiap sectionnya menyimpan barang tertentu. Jika barang tidak terlalu banyak dan masih ada ruang di dalam gudang, alokasi untuk sebuah jenis barang bisa ditambah dan dikurang. Barang dapat disimpan dan ditarik di mana tiap barang memiliki ID yang menyesuaikan jenisnya dan tanggal masuk. Tiap gudang memiliki sebuah akun pengurus dengan sebuah akun utama yang bisa membuat perusahaan dan mengakses semua gudang (OWNER dan MANAGER). Perusahaan juga memiliki data keluar masuk barang yang ditandai dengan tanggal masuk, tanggal keluar, dan kapasitas yang keluar.

Untuk menjalankan database gudang dan konten-kontennya, pengembang menggunakan PostgreSQL yang berjalan di server cloud milik Neon. Database ini terhubung ke backend berupa Node.js yang terhubung ke server PostgreSQL menggunakan pg. Backend ini menjadi server untuk frontend berupa aplikasi Android yang dikembangkan dengan Android Studio.

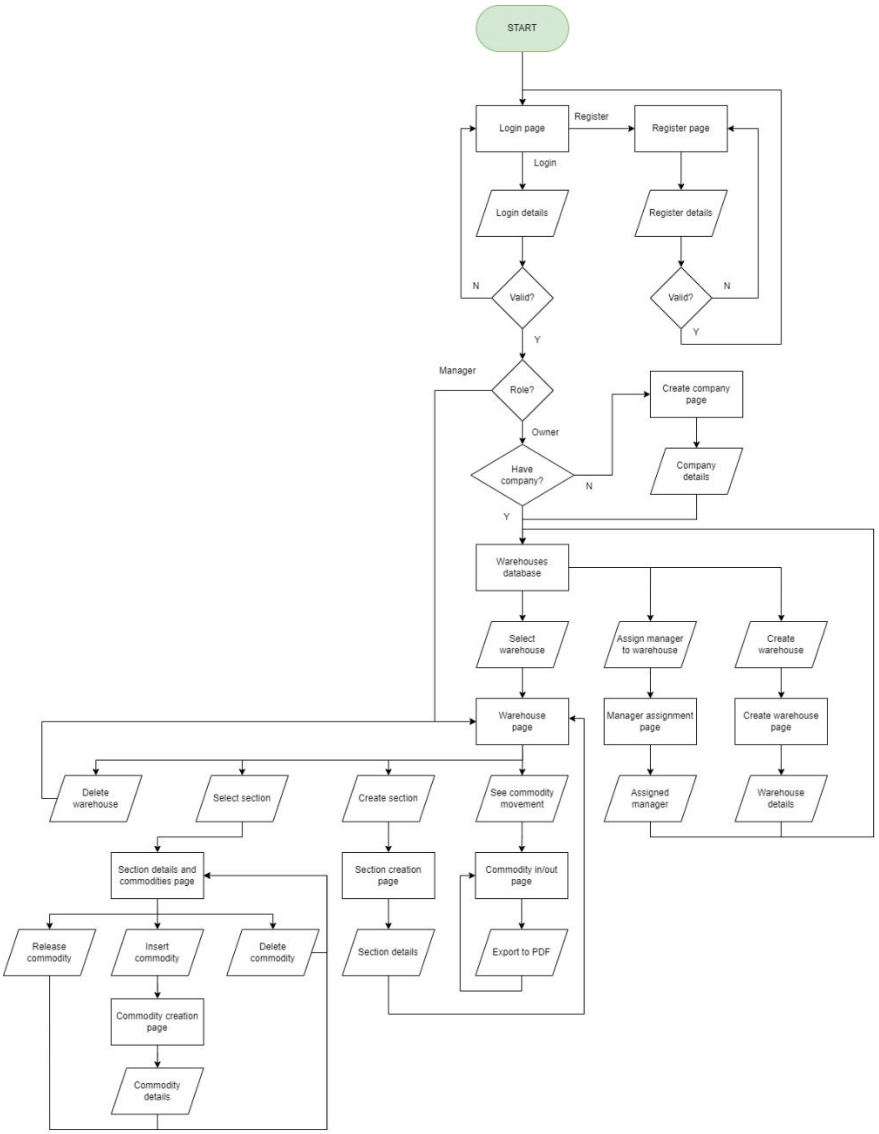
ERD



UML



Flowchart



II. Tabel SQL

Dengan kriteria seperti di bagian I, ini adalah tabel SQL-nya.

Employee

name	type	constraint
id	bigserial	primary key
username	text	unique not null
password	text	not null
name	text	not null
company_id	bigint	not null
role	employee_type	not null

Company

name	type	constraint
id	bigserial	primary key
name	text	unique not null

Warehouse

NAME	TYPE	CONSTRAINT
id	bigserial	primary key
address	text	not null
name	text	not null
company_id	bigint	not null
manager_id	bigint	N/A
rem_capacity	bigint	not null

Inout_records

name	type	constraint
id	bigserial	primary key
company_id	bigint	not null
warehouse	text	not null
section	text	not null
commodity_id	text	not null
name	text	not null
type	commodities	not null
volume	bigint	not null
in_time	date	N/A
out_time	date	N/A

Section

NAME	TYPE	CONSTRAINT
id	bigserial	primary key
warehouse_id	bigint	not null
name	text	not null
commodity_type	commodities	not null
rem_capacity	bigint	not null

Commodity

name	type	constraint
id	bigserial	primary key
commodity_id	text	unique not null
section_id	bigint	not null
volume	bigint	not null
name	text	not null

III. Konfigurasi SQL, Android, dan Node.js

SQL

```
create type employee_type as enum ('MANAGER', 'OWNER');

create table employee (
id bigserial primary key,
username text unique not null,
password text not null,
name text not null,
company_id bigint not null,
```

```
role employee_type not null
);

create table company (
id bigserial primary key,
name text unique not null
);

create table warehouse (
id bigserial primary key,
address text not null,
name text not null,
company_id bigint not null,
manager_id bigint,
rem_capacity bigint not null
);

create table inout_records (
id bigserial primary key,
company_id bigint not null,
warehouse text not null,
section text not null,
commodity_id text not null,
name text not null,
type commodities not null,
volume bigint not null,
in_time date,
out_time date
);

create type commodities as enum ('LIVE', 'FOOD', 'EARTH', 'ELECTRONIC',
'PHARMA', 'FURNITURE', 'TRANSPORT');

create table section (
```

```
id bigserial primary key,
warehouse_id bigint not null,
name text not null,
commodity_type commodities not null,
rem_capacity bigint not null
);

create table commodity (
id bigserial primary key,
commodity_id text unique,
section_id bigint not null,
volume bigint not null,
name text not null
);
```

Node.js (Index.js)

```
const express = require('express')
const app = express()
const  { Client }  = require('pg')
const bcrypt = require('bcrypt');
const bp = require('body-parser')
app.use(express.json())

const db = new Client({
   connectionString: 'postgres://muhammad.rizky18:HEGdpPmn8S9B@ep-
hidden-mode-314042.ap-southeast-1.aws.neon.tech/proyek_oop',
   sslmode: "require",
   ssl: true
})

db.connect((err)=>{
   if(err){
      console.log(err)
```

```
        return
    }
    console.log('Database berhasil terkoneksi')
})

app.post('/register-owner',(req,res)=>{
    const username = req.body.username,
    password = req.body.password,
    name = req.body.name,
    company_name = req.body.company_name

    bcrypt.hash(password, 8, (err, hashedPassword) => {
        if (err) {
            console.log(err)
            res.status(400).send
            return
        }

        const query1 = `insert into company(name) values ('${company_name}')`;
        db.query(query1, (err, results) => {
            if(err){
                console.log(err)
                res.status(400).send
                return
            }
            const query = `insert into employee(username, password, name, role,
company_id) values ('${username}', '${hashedPassword}', '${name}', 'OWNER',
(select id from company where name = '${company_name}'))`;
            db.query(query, (err, results) => {
                if(err){
                    console.log(err)
                    res.status(400).send
                    return
                }
```

```
            });

        });
    });
    res.status(200).send
})

app.post('/register-manager',(req,res)=>{
    const username = req.body.username,
    password = req.body.password,
    name = req.body.name,
    company_id = req.body.company_id

    bcrypt.hash(password, 8, (err, hashedPassword) => {
        if (err) {
            console.log(err)
            res.status(400).send
            return
        }

        const query = `insert into employee(username, password, name, role,
company_id) values ('${username}', '${hashedPassword}', '${name}', '${role}',
cast('${company_id}' as bigint)`;
        db.query(query, (err, results) => {
            if(err){
                console.log(err)
                res.status(400).send
                return
            }

            res.status(200).send
        });
```

```
    });

})

/*
app.post('/change-employee-detail',(req,res)=>{
    const {name, username, password, employee_id} = req.body

    bcrypt.hash(password, 8, (err, hashedPassword) => {
        if (err) {
            console.log(err)
            res.status(400).send
            return
        }

        const query = `update employee set name = '${name}', username =
'${username}', password = '${hashedPassword}' where id = cast('${employee_id}'
as bigint)`;
        db.query(query, (err, results) => {
            if(err){
                console.log(err)
                res.status(400).send
                return
            }
        });
    });

    res.status(200).send
})*/


app.post('/login',(req,res)=>{
    const username = req.body.username
    const password = req.body.password
```

```
    bcrypt.hash(password, 8, (err, hashedPassword) => {
        const query = `SELECT * from employee WHERE username='${username}'
AND password='${hashedPassword}'`; //query ambil data user untuk login
        bcrypt.compare(password, hashedPassword, (err, isMatch) => {
            if( err ) {
                res.status(404).send
                return err;
            }
            // If password matches then display true
            console.log(isMatch);
            db.query(query, (err, results) => {
                if (err) {
                    console.log(err)
                    res.status(404).send
                    return
                }

                res.status(200).send(JSON.stringify(results))
             });

        });
    });
})

app.post('/get-company-name',(req,res)=>{
    const company_id = req.body.company_id
    db.query(`SELECT * FROM company WHERE id = ${company_id}`
,(err,results)=>{
        if(err){
            console.log(err)
            res.status(400).send
            return
        }
```

```
        res.status(200).send(JSON.stringify(results))
    })
})

app.post('/company',(req,res)=>{
    const company_id = req.body.company_id
    db.query(`SELECT * FROM warehouse WHERE company_id =
${company_id}` ,(err,results)=>{
        if(err){
            console.log(err)
            res.status(400).send
            return
        }
        res.status(200).send(JSON.stringify(results))
    })
})

app.post('/create-warehouse',(req,res)=>{
    const address = req.body.address,
    name = req.body.name,
    company_id = req.body.company_id,
    rem_capacity = req.body.rem_capacity

    const query = `insert into warehouse(address, name, company_id,
rem_capacity) values ('${address}', '${name}', cast('${company_id}' as bigint),
cast('${rem_capacity}' as bigint))`;
    db.query(query, (err, results) => {
        if(err){
            console.log(err)
            res.status(400).send
            return
        }

        res.status(200).send
```

```
    });
})

app.post('/delete-warehouse',(req,res)=>{
    const warehouse_id = req.body.warehouse_id
    const query01 = `select * from sections where warehouse_id =
${warehouse_id}`;
    db.query(query01, (err, results) => {
        if(err){
            console.log(err)
            res.status(400).send
            return
        }

        if(results != NULL){
            res.status(500).send
            return
        }

        const query02 = `delete from warehouse where id = ${warehouse_id}`;
        db.query(query02, (err, results) => {
            if(err){
                console.log(err)
                res.status(400).send
                return
            }

            res.status(200).send
        });
    });
})

app.post('/get-manager',(req,res)=>{
    const company_id = req.body.company_id
```

```
    db.query(`SELECT * FROM employee WHERE company_id =
${company_id} and role = 'MANAGER` ,(err,results)=>{
        if(err){
            console.log(err)
            res.status(400).send
            return
        }
        res.status(200).send(JSON.stringify(results))
    })
})

app.post('/get-manager-name',(req,res)=>{
    const manager_id = req.body.manager_id
    db.query(`SELECT * FROM employee WHERE id = ${manager_id}`
,(err,results)=>{
        if(err){
            console.log(err)
            res.status(400).send
            return
        }
        res.status(200).send(JSON.stringify(results))
    })
})

app.post('/assign-manager',(req,res)=>{
    const manager_id = req.body.manager_id,
    warehouse_id = req.body.warehouse_id

    const query = `update warehouse set manager_id = ${manager_id} where id =
${warehouse_id}`;
    db.query(query, (err, results) => {
        if(err){
            console.log(err)
            res.status(400).send
```

```
            return
        }

        res.status(200).send
    });
})

app.post('/unassign-manager',(req,res)=>{
    const warehouse_id = req.body.warehouse_id

    const query = `update warehouse set manager_id = NULL where id =
${warehouse_id}`;
    db.query(query, (err, results) => {
        if(err){
            console.log(err)
            res.status(400).send
            return
        }

        res.status(200).send
    });
})

app.post('/warehouse',(req,res)=>{
    const warehouse_id = req.body.warehouse_id

    db.query(`SELECT * FROM section WHERE warehouse_id =
${warehouse_id}` ,(err,results)=>{
        if(err){
            console.log(err)
            res.status(400).send
            return
        }
        res.status(200).send(JSON.stringify(results))
```

```
    })
})

app.post('/create-section',(req,res)=>{
    const warehouse_id = req.body.warehouse_id,
    name = req.body.name,
    commodity_type = req.body.commodity_type,
    rem_capacity = req.body.rem_capacity
    const query1 = `insert into section(warehouse_id, name, commodity_type,
rem_capacity) values (cast('${warehouse_id}' as bigint), '${name}',
'${commodity_type}', cast('${rem_capacity}' as bigint)`;
    db.query(query1, (err, results) => {
        if(err){
            console.log(err)
            res.status(400).send
            return
        }
        res.status(200).send
    });

    const query2 = `update warehouse set rem_capacity = rem_capacity -
${rem_capacity} where id = ${warehouse_id}`;
    db.query(query2, (err, results) => {
        if(err){
            console.log(err)
            res.status(400).send
            return
        }

        res.status(200).send
    });
})

app.post('/delete-section',(req,res)=>{
```

```
    const {section_id, rem_capacity, warehouse_id} = req.body
    const query01 = `select * from commodity where section_id = ${section_id}`;
    db.query(query01, (err, results) => {
        if(err){
            console.log(err)
            res.status(400).send
            return
        }

        if(results != NULL){
            res.status(500).send
            return
        }

        const query02 = `delete from section where id = ${section_id}`;
        db.query(query02, (err, results) => {
            if(err){
                console.log(err)
                res.status(400).send
                return
            }
        });
    });

    const query2 = `update warehouse set rem_capacity = rem_capacity +
${rem_capacity} where id = ${warehouse_id}`;
    db.query(query2, (err, results) => {
        if(err){
            console.log(err)
            res.status(400).send
            return
        }
    });
```

```
    res.status(200).send
})

app.post('/section',(req,res)=>{
    const section_id = req.body.warehouse_id

    db.query(`SELECT * FROM commodity WHERE section_id = ${section_id}`
,(err,results)=>{
        if(err){
            console.log(err)
            res.status(400).send
            return
        }
        res.status(200).send(JSON.stringify(results))
    })
})

app.post('/commodity-in', (req, res) =>{
    const {company_id, warehouse_id, section_id, name, type, volume, in_time} =
req.body

    var commodity_id = `C` + `${company_id}` + `W` + `${warehouse_id}` `S` +
`${section_id}` + `TI` + `${in_time}`

    var query = `insert into commodity(section_id, volume, name) values
(cast('${section_id}' as bigint), ${volume}, ${name})`
    db.query(query, (err, results) => {
        if(err){
            console.log(err)
            res.status(400).send
            return
        }
```

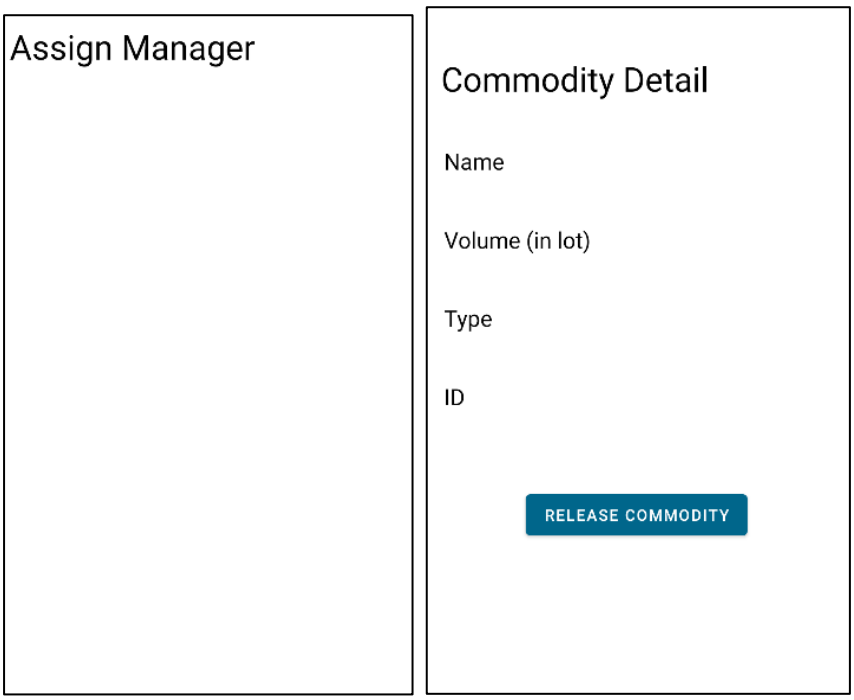
```
        query = `update commodity set commodity_id =
concat('${commodity_id}','IN', cast(max(id) as text)) where id = max(id)`
        db.query(query, (err, results) => {
            if(err){
                console.log(err)
                res.status(400).send
                return
            }
        });
    });

    query = `update section set rem_capacity = rem_capacity - ${volume} where id
= ${section_id}`
    db.query(query, (err, results) => {
        if(err){
            console.log(err)
            res.status(400).send
            return
        }
    });

    query = `insert into inout_records values   (   company_id, warehouse_id,
section_id, commodity_id, name, type, volume, in_time) values
                                    (   cast('${company_id}' as bigint),
                                        (select name from warehouse where id =
cast('${warehouse_id}' as bigint)),
                                        (select name from section where id =
cast('${section_id}' as bigint)),
                                        '${commodity_id}', '${name}', '${type}',
                                        cast('${volume}' as bigint),
                                        to_date('${in_time}', 'yyyy-mm-dd'))`;
    db.query(query, (err, results) => {
        if(err){
            console.log(err)
```

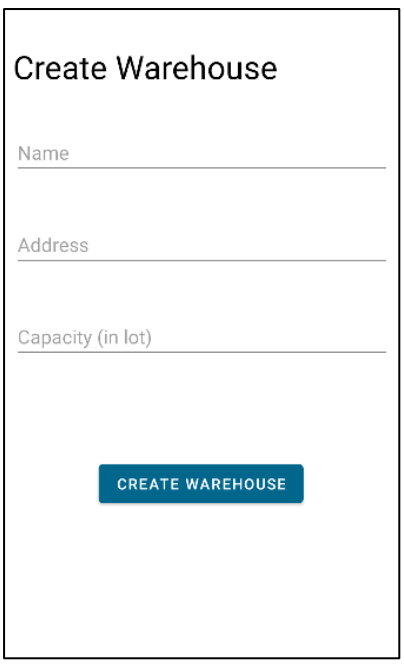
```
            res.status(400).send
            return
        }
    });
    res.status(200).send
})

app.post('/commodity-out', (req, res) =>{
    const {commodity_id, section_id, volume, out_date} = req.body

    var query = `delete from commodity where commodity_id =
'${commodity_id}'`
    db.query(query, (err, results) => {
        if(err){
            console.log(err)
            res.status(400).send
            return
        }
    });

    query = `update section set rem_capacity = rem_capacity + cast('${volume}' as
bigint) where id = cast('${section_id}' as bigint)`
    db.query(query, (err, results) => {
        if(err){
            console.log(err)
            res.status(400).send
            return
        }
    });

    query = `update inout_records set out_time = cast('${out_date}' as date) where
commodity_id = '${commodity_id}'`;
    db.query(query, (err, results) => {
        if(err){
```

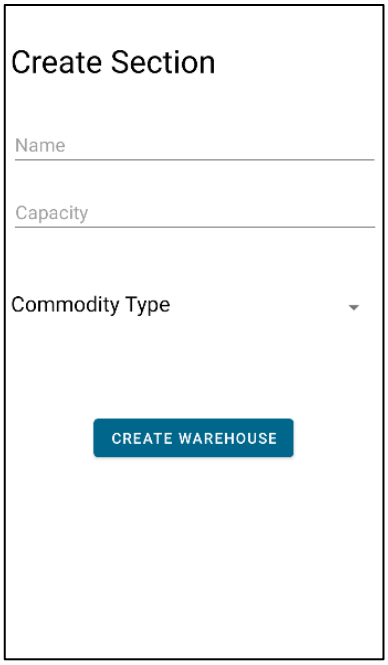
```
            console.log(err)
            res.status(400).send
            return
        }
    });

    res.status(200).send
})

app.post('/inout_records',(req,res)=>{
    const {
        company_id,
        in_time_start,
        in_time_end,
        out_time_start,
        out_time_end} = req.body

    var query = `SELECT * FROM inout_records WHERE company_id =
cast('${company_id}' as bigint) AND in_time BETWEEN
to_date('${in_time_start}', 'yyyy-mm-dd') and to_date('${in_time_end}', 'yyyy-
mm-dd') AND out_time BETWEEN to_date('${out_time_start}', 'yyyy-mm-dd')
and to_date('${out_time_end}', 'yyyy-mm-dd')`;

    if(out_time_start == null || out_time_end == null) {
        query = query.replace(` AND out_time BETWEEN
to_date('${out_time_start}', 'yyyy-mm-dd') and to_date('${out_time_end}', 'yyyy-
mm-dd')`, ``);
    }

    if(in_time_start == null || in_time_end == null) {
        query = query.replace(` AND in_time BETWEEN to_date('${in_time_start}',
'yyyy-mm-dd') and to_date('${in_time_end}', 'yyyy-mm-dd')`, ``);
    }
```

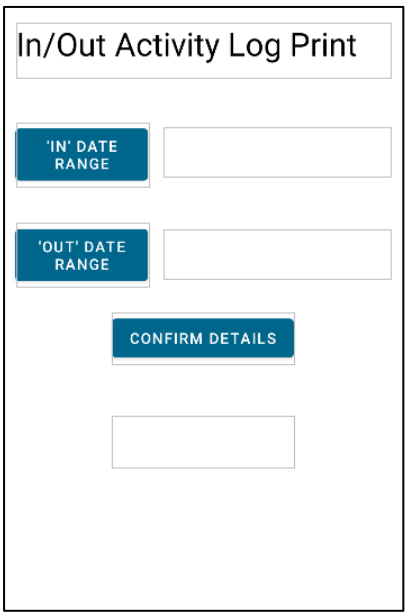
```
    db.query(query ,(err,results)=>{
        if(err){
            console.log(err)
            res.status(400).send
            return
        }
        res.status(200).send(JSON.stringify(results))
    })
})

app.listen(1325 /*angka terakhir harusnya 0*/, ()=>{
    console.log('Port 1325 tersambung')
})
```

Android (Activity)

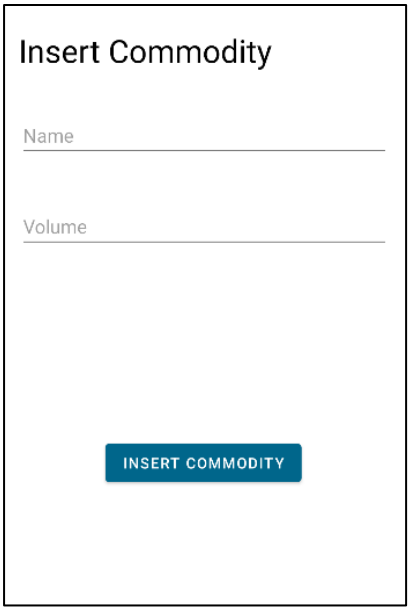


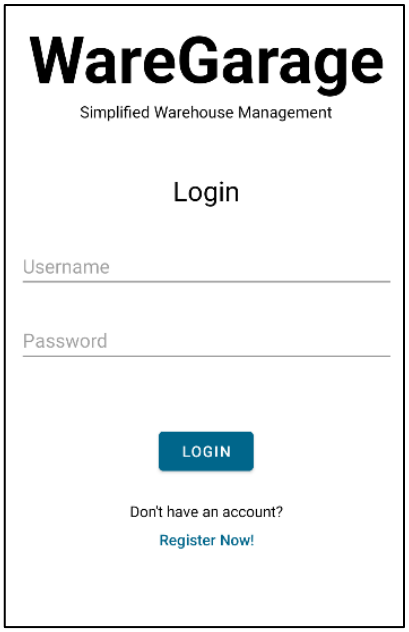

Create Warehouse
Name
Address
Capacity (in lot)
CREATE WAREHOUSE


Create Section
Name
Capacity
Commodity Type
CREATE WAREHOUSE

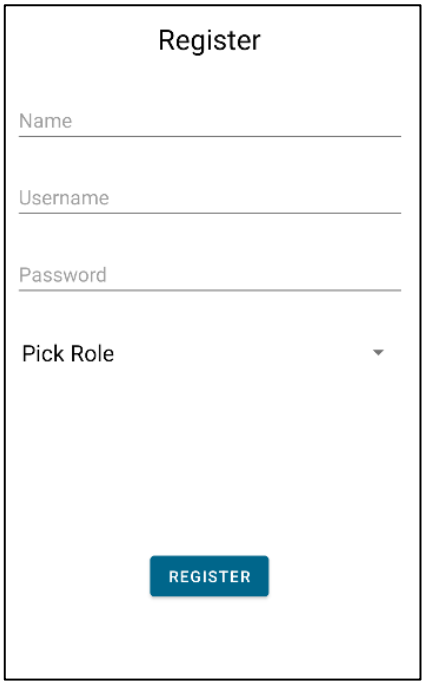

In/Out Activity Log Print
'IN' DATE RANGE
'OUT' DATE RANGE
CONFIRM DETAILS

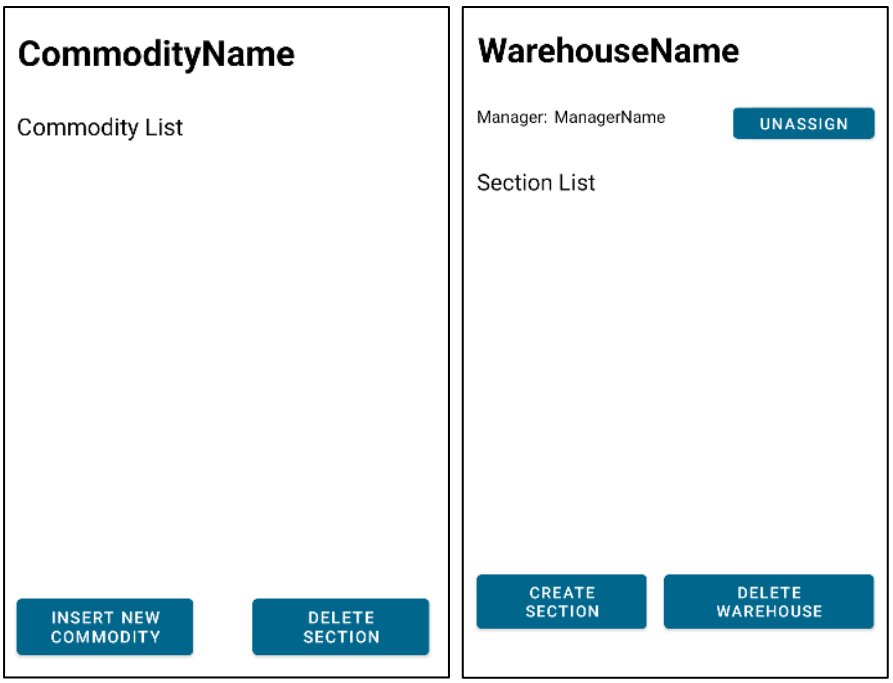

Your Profile
Name
Username
Role
Company


Insert Commodity
Name
Volume
INSERT COMMODITY


WareGarage
Simplified Warehouse Management
Login
Username
Password
LOGIN
Don't have an account?
Register Now!


CompanyName
ID:
Warehouse List
CREATE WAREHOUSE


Register
Name
Username
Password
Pick Role
REGISTER


CommodityName
Commodity List
INSERT NEW COMMODITY
DELETE SECTION
WarehouseName
Manager: ManagerName
UNASSIGN
Section List
CREATE SECTION
DELETE WAREHOUSE