Så här skickar du JSON data.

connection.setRequestProperty("Content-Type"

, "application/json; charset=UTF-8");

JSONObject json = new JSONObject();

json.put("Erik", "Ollon");

json.put("Yllka", "Rama");

json.put("Fuck", "Knäpp");

writer.write(json.toString());

**//skicka data via websocket.**

**socket.emit("acceptDeal",json);**

**//Lyssnar så här kan du uppdatera för alla**

**MainFragment mf = new MainFragment();**

**socket.on("newMessage", mf.onNewMessage ); //lyssnar**

**socket.off("newMessage"); //Ta bort för något du lyssnar på**

**//Main fragment klassen**

**public class MainFragment extends Fragment {**

**public Emitter.Listener onNewMessage = new Emitter.Listener() {**

**@Override**

**public void call(final Object... args) { //args är inkommande data**

**Log.*e*("tag", args[0].toString());**

**JSONObject data = (JSONObject) args[0];**

**String username;**

**String message;**

**try {**

**username = data.getString("username");**

**message = data.getString("message");**

**} catch (JSONException e) {**

**return;**

**}**

**// add the message to view**

**}**

**};**

**}**

**Register:**

Client skickar:

* Användarnamn
* Lösenord (hashat)
* type (“cleaner”/”company”)

Server skickar

* True eller false om användaren finns.

**Login:**

* Användnamn - String
* Lösenord - String
* type (t.ex företagare, städare, osv) - String

Client skickar:

* userName (Variabelnamn)
* password (Variabelnamn)
* Förväntar dig: Typ av anställning för användaren.
* Skickar som JSON.

Server skickar:

* typ av anställning.

**Om användare är städare (Cleaner):**

**Hitta lediga agreements**

Client skickar:

* GET-request (ingen data behöver skickas med)

Server skickar:

* TABLE NAME: Agreement
* name *varchar(64),*
* price *float*
* city *varchar(64)*
* hours *int*
* accepted *boolean*
* done *boolean*

NOTERA: om en annan städare accepterar ett uppdrag så ska detta skickas via Websockets, även om ett nytt uppdrag tillkommer så skall denna uppdateras via Websockets.

**Om “Städare” klickar på “My Assigned Task”**

Client skickar:

* POST-request (userName)

Server skickar:

* TABLE NAME: ongoingAssignments
* name *varchar(64),*
* price *float*
* city *varchar(64)*
* hours *int*
* accepted *boolean*
* done *boolean*
* Where activeAssignment = true AND username = “username” (till den här städaren).
* Kan modifiera tasks så att (done = true) task är färdig.

NOTERA: Om de kommer ett nytt uppdrag så ska denna lista

uppdateras via websockets.

**Om användare är Företag (Company):**

1. **Server skickar** - sql: Hämta alla uppdrag som tillhör den här beställaren (mha usernamn) Lista så att alla ofärdiga högst upp.
2. **Client skickar -** alternativ: Om beställare vill skapa fler uppdrag så kan beställaren trycka på en knapp som heter “Create New Task”. Fyll in formulär(name, price, city, hours, dt, accepted, done) och “insert” data till sql.

Client skickar:

* Skickar POST-request
  + userName

Server skickar:

* TABLE NAME: Agreement
* name *varchar(64),*
* price *float*
* city *varchar(64)*
* hours *int*
* accepted *boolean*
* done *boolean*

NOTERA: Här behövs INGEN uppdatering via websockets.

Login

typeOfEmployment