TALLINNA TEHNIKAÜLIKOOL

Infotehnoloogia teaduskond

Paul Ööbik 163920IACB

ENERGIATARBE MÕÕTMISE JA VISUALISEERIMISE LAHENDUS

Bakalaureusetöö

Juhendaja: Karl Laanemets

MsC

Mairo Leier

PhD

Tallinn 2021

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Paul Ööbik

08.01.2021

Annotatsioon

Käesoleva bakalaureusetöö eesmärk on pakkuda välja energiamõõtmise lahendus, mida saaks kasutada iseseisvalt, vajamata tööks internetiühendust. Lahenduses on olemas kõik vajalik selle normaalseks toimimiseks, peale esialgset seadistust saab süsteemi kasutada ilma kõrvaliste seadmeteta.

Töö peamine probleem seisneb selles, et olemasolevate energiatarbe mõõtmise seadmetel läheb oma toimimiseks vaja internetiühendust ning need on ka suhteliselt kallid. Probleemi lahendamise käigus seati nõuded süsteemile, ja nendest lähtuvalt valiti komponendid, mille põhjal arendada energia mõõtmise lahendust. Tulemusena valmis lahendus energiamõõtja, serveri ja graafilise kasutajaliidesega. Korraldatud on nende omavaheline suhtlus üle Wi-Fi ja Etherneti. Mõõtetulemused salvestatakse ja kasutajaliidese kaudu on kättesaadav nii üldine vaade kui ka ajalugu.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 35 leheküljel, 5 peatükki, 21 joonist, 5 tabelit.

Abstract

Solution to measure and visualize energy consumption

The purpose of this thesis is to offer a possible solution to measure energy consumption, that could be used as standalone and offline. The solution contains everything for its normal operation, and after an initial setup the system can be used with no extra devices needed.

The main problem of the thesis resides in the available energy measurement and monitoring solutions needing a connection to the Internet and being rather expensive. In the current thesis, certain requirements were set, according to which the components were chosen, and the solution was developed. The system was divided into three devices: a device for measuring power, a central server and a graphical user interface.

The developed measurement device consists of a current sensor attached to a microcontroller to measure current, calculate power and send the data. User interface was realized on a developer board with an integrated 4.3'' touch screen and a microcontroller. For communication between devices and data management, a mini pc was utilized to run a server with an interface confirming to the set requirements.

HTTP requests are utilized as the data channel for their ease of realization and fault-proofing. The physical layer consists of Wi-Fi on the measurement device and Ethernet on the interface device. Both channels are provided by the server acting as an access point.

All in all, the solution conforms to the set requirements and is able to measure power effectively and send the data wirelessly to the server. On the graphical user interface, saved readings and information about the sensors can be shown and visualized as graphs.

The thesis is in Estonian and contains 35 pages of text, 5 chapters, 21 figures, 5 tables.

Lühendite ja mõistete sõnastik

AC	Alternating Current ehk vahelduvvool
ADC	Analog – digital converter ehk analoog-digitaalmuundur
AP	Access Point ehk pääsupunkt
API	Application Programming Interface ehk rakendusliides
Back-end	Tagarakendus/põhiprogramm
Endpoint	Ligipääs API-le kindla toimingu jaoks
Flash (memory)	Välkmälu
Front-end	Eessüsteem/kasutajaliides
GPIO	General Purpose Input-Output ehk üldotstarbeline sisend-väljund
HTTP	Hypertext Transfer Protocol eht hüperteksti edastusprotokoll
IDE	Integrated Development Environment ehk integreeritud arenduskeskkond
IEEE	Elektri- ja Elektroonikainseneride Instituut
LAN	Local Area Network ehk Kohtvõrk
REST	Representational State Transfer
RTOS	Real-Time Operating System ehk reaalaja operatsioonisüsteem
SD-kaart	Secure Digital Card, SD standardil mälukaart
SQL	Structured Query Language ehk struktuurpäringukeel
SSID	Service Set IDentifier ehk võrgunimi ehk mestiident
URI	Uniform Resource Identifier ehk ühtne ressursiidentifikaator
USB	Universal Serial Bus ehk universaalne järjestiksiin
UI	User Interface ehk kasutajaliides

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Elekter on üks suurimaid kommunaalkulusid ja Eestis selle tootmine tänu põlevkivile ka suur koormus keskkonnale. Energiatarbimise optimeerimine aitaks kaasa mõistlikumale ressursside kasutamisele, mis omakorda vähendaks reostamist, looduse kurnamist ning lõpptarbija kulusid.

Efektiivse tarbimise üks eeldusi on asjakohase informatsiooni olemasolu, olles teadlik oma tarbimisharjumustest, saab teha neis vajalikke parandusi, et hoida kokku elektrit, ja sellele kuluvat raha. Tarbimise jälgimine ja selle ajaloo salvestamine ning selle käigus kogutud andmed võimaldavad teha teadlikumaid otsuseid, näiteks koju päikesepaneelide või tuulegeneraatori paigaldamisel.

Andmete kogumine tavalise elektriarvestiga on võrdlemisi keeruline, kuna tarve summeeritakse ja ühe arvu põhjal on detailsemaid järeldusi raske teha. Lahendusi energia mõõtmiseks ja analüüsimiseks on turul juba olemas, aga need on kallid ja vajavad oma tööks internetiühendust [1], [2], [3].

Käesoleva lõputöö raames pakutakse välja võimalik lahendus ülaltoodud probleemile: eraldiseisev, internetiühenduse olemasolu mittevajav mõistlikku hinnaklassi jääv energiatarbe monitoorimise süsteem.

2 Eesmärgid

Töö eesmärkideks oli luua 2 seadet, millest ühega oleks võimalik mõõta nii ühe- kui kolmefaasilisel voolul töötavate seadmete energiatarvet ning teisega võimaldada kasutajal mõõtetulemusi kuvada.

2.1 Turul olevad lahendused

Lahendusi energia mõõtmiseks ja analüüsimiseks on turul juba olemas, näiteks *Sense Energy Monitor*, *Efergy Engage* ja *Emporia Vue Smart Home Energy Monitor*, mille baaslahenduse hinnad on vastavalt $299 [2] $99,99 [1] ja $59,99 [3].

Sense on mõeldud ühefaasilise süsteemi energiatarbe mõõtmiseks, ilma päikesepaneelide tootluse mõõteseadmeta on selle hind $299. Lahenduses on juhtme ümber käivad sensorid ja seade, millega mõõdetakse tarvet ning saadetakse mõõtetulemused Wi-Fi kaudu edasi pilve. Selle üks suurimaid plusse on andmete töötlus, ühilduvus teiste IoT seadmetega ja kasutajaliides, millele saab ligi mobiilirakendusest või internetist [2].

Joonis 1: Efergy Engage [4]

Joonis 1 kujutatud *Efergy* lahendus on mõeldud kas ühe- või kolmefaasilise voolu energiatarbe mõõtmiseks kodudesse, baaslahenduse hind on $99,99 ning 3 faasi mõõtmiseks $129,99. See koosneb mõõteseadmest ja juhtme ümber käivatest sensoritest

ning keskseadmest. Mõõteseadmest tulnud andmed saadetakse juhtmevabalt keskseadmele, mis, olles ühendatud kaabliga internetiga, saadab andmed edasi *Efergy* serveritesse. Sealt saab neile ligi nutiseadme või arvutiga [1].

Emporia on väljatoodud seadmetest odavaim, hinnad on $59,99 baaslahenduse eest ja $69,99, kui soovitakse mõõta kolme faasi. Seade koosneb sensoritest ning kontrollerist, mis mõõdab voolu, teeb arvutused ja saadab informatsiooni üle Wi-Fi pilve. Andmetele ligipääsuks on vajalik nutiseadme või arvuti olemasolu. Välja on toodud ka mõõteseadmete täpseim töövahemik 2 A – 250 A [3].

Ülaltoodud seadmed on kallid ja mis põhiline – kõik vajavad oma normaalseks toimimiseks internetiühenduse olemasolu ja *Sense*'i ning *Emporia* lahenduses lisaks ka Wi-Fi levikut. See piirab nende kasutusvõimalusi näiteks suvilates, kuurides, garaažides ja mujal, kus tarbitakse elektrit, kuid ei pruugi olemas olla internetiühendust. Lisaks puudub eelmainitud süsteemidel füüsiline sisend-väljund seade kasutajale andmete kuvamiseks.

2.2 Nõuded

Turul olevatest lahendustest on suur enamik kallid ja nõuavad internetiühendust. Käesoleva projekti raames luuakse võimalikult odavalt valmislahendustega samaväärset süsteemi, mis toimib ka ilma internetiühenduseta, et muuta seda paindlikumaks. Keskenduda otsustati andmete kogumise paindlikkusele ja täpsusele ning toote hinnale, mitte niivõrd andmete tarkvaralisele analüüsimisele, milles konkureerivatel toodetel on rohkemal või vähemal määral eelis. Sellest tulenevalt jäi lõpplahenduse kasutajaliidesesse vähem funktsionaalsust ning see on lihtsakoelisem kui konkureerivatel toodetel. Samas on informatsioon nii ühe- kui ka kolmefaasilise voolu energiatarbe kohta kättesaadav, seda ka ilma internetiühenduse või nutiseadme olemasoluta.

Lahendus peab võimaldama järgmist:

- Võimalikult lihtne ja mugav energiatarbe mõõtmine
- Nii ühe kui ka kolmefaasiliste vooluvõrkude mõõtmine
- Info kuvamine reaalaja energiatarbe ning tarbimisajaloo kohta

- Mitme mõõte- ja väljundseadme kasutusvõimalus
- Toimimine ilma internetiühenduseta
- Suhtlus mõõteseadmetega juhtmevabalt
- Füüsiline kasutajaliides, millelt saab informatsiooni reaalaja energiatarbe ning tarbimisajaloo kohta
- Mõistlik hind, et konkureerida turul olevate toodetega

Et mõõtmine võimalikult mugav oleks ning ei peaks sensori juures tarbimist vaatamas käima, otsustati eraldada kasutajaliides mõõtmisi tegevast seadmest. See võimaldab mõõteseadme jätta näiteks elektrikappi ja läbi ekraaniga kasutajaliidese jälgida oma energiatarvet, ilma sensori kõrval olemata. Lahendus peab võimaldama ka tarbimisajaloo jälgimist, mis eeldab, et tarbimisandmed salvestatakse ja neile on võimalik ligi pääseda. Mitme mõõteseadme kasutamiseks oli vaja keskset andmebaasi, kuhu infot salvestada ja kust seda pärida, seega otsustati kasutada andmehalduseks lisaseadet. Andmeedastus protokollina otsustati HTTP päringute kasuks, kuna see võimaldab mõlemapidist suhtlust, aga erinevalt Bluetoothist, on andmete ülekanne oluliselt töökindlam.

Lahendus otsustati jagada kolme osasse: mõõteseade, kasutajaliides ja keskseade.

2.2.1 Sensor

Selleks, et konkureerida eelmainitud valmis lahendustega on mõõteseadmele järgnevad nõuded:

- Seade peab olema võimeline mõõtma ühe-, kolme- kui ka segafaasilist voolu.
- Pärast esmakordset töökorda seadmist peab see töötama iseseisvalt, ka pärast toitevoolu ajutist kadumist.
- Mõõteseadme parameetreid peab olema võimalik dünaamiliselt muuta, st arvutist või nutitelefonist saab ligi liidesele, mille kaudu muuta seadeid.
- Andmeedastus juhtmevabalt

Selleks, et pakkuda võimalikult universaalset lahendust, peab mõõdetav olema nii kodumasinate ja -elektroonika energiatarve ühe voolufaasi peal, kui ka tööstusseadmete energiatarve kolmel voolufaasil.

Lisaks peab süsteem võimaldama mõõta neid kahte ka korraga, kuna tihti on seadmeid mõlemat tüüpi, näiteks korterites, kus elektripliit on kolmefaasilisele voolule mõeldud, aga ülejäänud ühefaasilisele.

2.2.2 Kasutajaliides

Lahenduses välditakse pilvelahendusi ja sellega muudetakse projekt sõltumatuks internetiühenduse olemasolust, andes sellega juurde paindlikkust võimalike kasutuskohtade suhtes. Selleks peab süsteemis olema füüsiline väljund andmete kuvamiseks.

Kasutajaliides peab võimaldama järgmisi kasutusjuhte:

- Kõigi mõõteseadmete kohta hetketulemuste kuvamine.
- Ühe mõõteseadme tulemuste ajaloo ja selle graafiku kuvamine.

2.2.3 Keskseade

Seadmetevaheline infovahetus planeeriti töötama HTTP päringutel, selleks peab andmehalduse seadmel töötama HTTP server ning samas paiknema ka andmebaas.

Seade peab võimaldama nii mõõteseadmete kui kasutajaliidese tööd, selleks on vajalik:

- HTTP päringute vastu võtmine ning neile vastamine
- Eelmisest punktist tulenevalt ühenduvus eeltoodud seadmetega
- Andmete salvestamise võimalus

3 Riistvara

Lahendus on riistvaraliselt jagatud kaheks põhiliseks seadmeks, mille andmevahetus käib läbi keskseadme, kokku kolm seadet:

- Mõõteseade koos sensoriga, kust tuleksid mõõtetulemused
- Andmeid kuvav ekraaniga kasutajaliides, millel kuvada mõõtetulemusi
- Keskseade, mis tegeleb andmehaldusega ja on vahelüliks eelmainitute vahel

Lahenduses samade või analoogsete, funktsionaalsuse poolest võrdväärsete seadmete korral tuleb hinnaks $123,5 kolme sensoriga ja $115,41 ühe sensoriga lahenduse puhul. Tabel 1 on välja toodud kirjutamise hetkel leitud hindadega komponendid.

Tabel 1: Lahenduses kasutatud komponentide hinnad

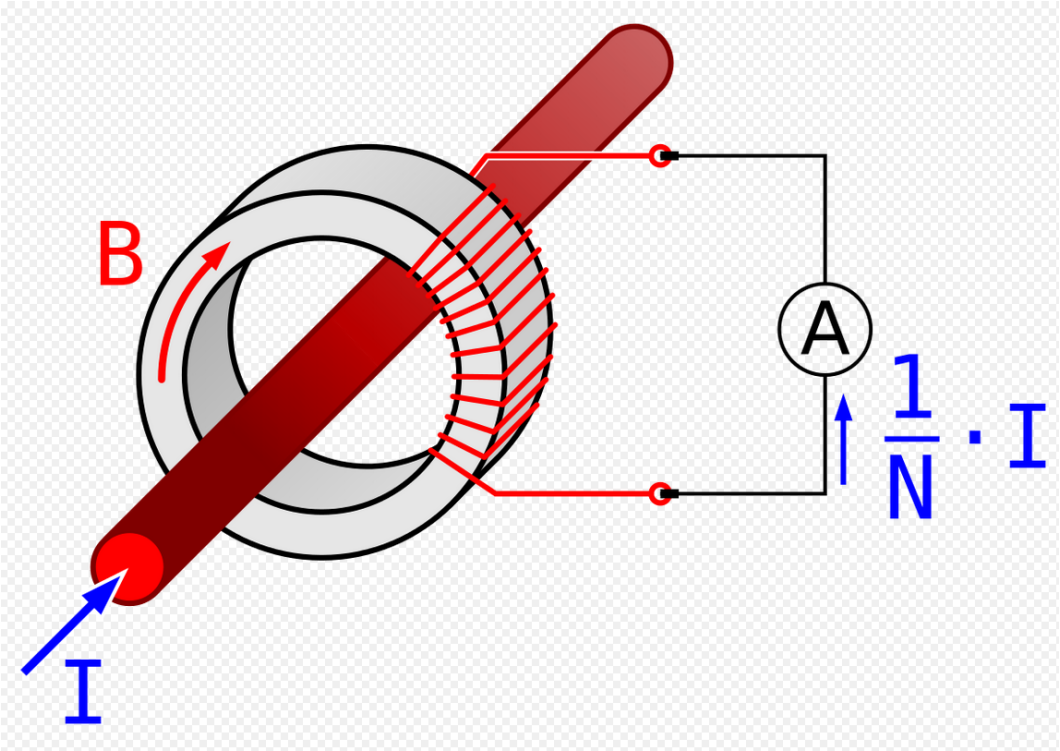
Komponent	Hind
Vool-pinge muundur SCT-013	$4 [5]
Mikrokontroller ESP32	$2,45 [6]
Arendusplaat STM32F746G-Discovery	$54,00 [7]
Raspberry Pi 4 B	$43 [8]
8 GB mälukaart	$7 [9]
Mälupulk > 1 GB	$3 [10]
Kokku ühe sensoriga/kolme sensoriga	$115,5 / $123,5

Mõõteseade kujutab endast voolusensoriga ühendatud mikrokontrollerit, kasutajaliides koosneb ekraanist ja mikrokontrollerist, keskseadmeks kasutati miniarvutit. Suhtluseks HTTP päringute kaudu kasutatakse nende peamisi füüsilisi kanaleid: Wi-Fi suhtluseks mõõteseadmega ja Ethernet kasutajaliidesega.

3.1 Mõõteseade

Ampermeetriga voolu mõõtmiseks on vaja juhe lahti ühendada ja sinna vahele ampermeeter ühendada. Elektrikilbis või ka mujal juhtmetes liikuva voolu mõõtmiseks ei ole tihti mugav juhtmeid lõhkuda ja pärast parandada. Selle vältimiseks on kasutusel ümber juhtme käivad magnetilise südamikuga vool-vool muundurid, mis vastavalt

keerdude arvudele muundavad ühes juhtmes liikuva voolu magnetvälja toimel teises juhtmes liikuvaks vooluks. Muunduri tööpõhimõte on toodud Joonis 2.



Joonis 2: Vool-vool muunduri tööpõhimõte [11]

Peajuhe, mille voolu I mõõdetakse – joonisel punane juhe – läheb läbi magnetsüdamiku, millele on mähitud N keeruga sekundaarne juhe. Südamikus tekkiv magnetväli B tekitab mähise juhtmes voolu, mis on A = 1/N*I.

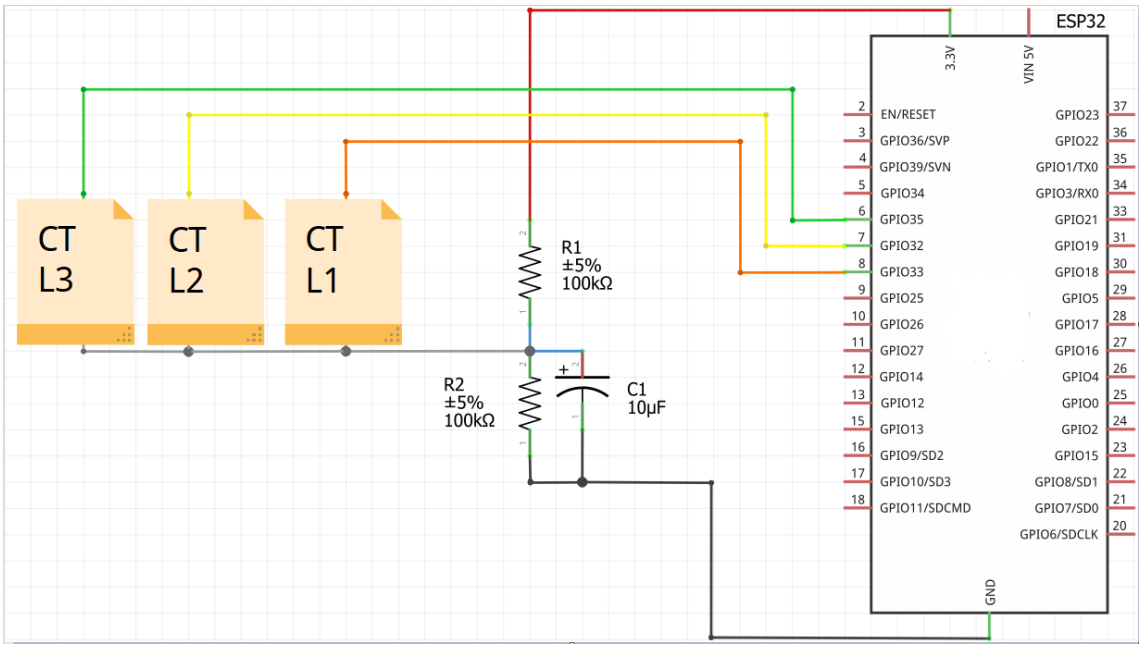
Sellise seadmega voolu mõõtmine on siiski tülikas, sest juhe on vaja lahti ühendada, see südamikust läbi juhtida ja tagasi ühendada, mõõtmine ära teha ja siis uuesti lahti ühendada. Probleemi lahendamiseks on võimalik südamik poolitada ja mõõtmisi endiselt täpselt samal põhimõttel läbi viia.

Vältimaks riske, mis võivad tekkida elektrikilbis juhtmete ühendamisel, ning kasutusmugavuse suurendamiseks on sensoritena kasutul samal põhimõttel töötava, lahtikäiva südamikuga vool-pinge muundurid YHDC SCT013-030 30A/1V. Need on kahest ferriidist U-südamikust ja mähisest koosnevad AC-AC muundurid. Seade toimib eelmainitud põhimõttel, ent voolust pinge tekitamiseks on vooluahelasse juurde lisatud madala takistusega (<10Ω) koormustakisti, kuhu mähisesse tekkinud voolu korral jääb pinge. Kasutatud sensor on näidatud Joonis 3.

Joonis 3: YHDC SCT013 sensor [12].

Muunduri väljund on tavaline 3,5mm stereo audio pistik (ülevalt alla *sleeve*, *right*, *left*), kus alumisse klemmi (*left*) ühendatakse etalonpinge V_{ref}, mis on pinge nullpunktiks-maanduseks, mille suhtes väljundpinge väärtus tekib, ja ülemise (*sleeve*) ning alumise klemmi vahelt mõõdetakse väljundpinge V_{out}. Keskmine klemm (*right*) jääb kasutamata [13].

Voolu arvutamiseks ja selle edastamiseks ühendati sensor mikrokontrolleriga SparkFun ESP32 Thing. Antud mikrokontrolleri eelisteks on odav hind ja võimekus ühenduda WiFi-ga, lisaks mitu ADC(analoog-digitaal muundur)-võimekusega GPIO (üldotstarbeline sisend-väljund) ühendust, mis võimaldab ühe kontrolleriga siduda mitu sensorit. Kontrollerile saab 5V toite sisendi anda *Micro*-USB(universaalne järjestiksiin) kaudu [14].



Joonis 4: Sensori liidestamine kontrolleriga

Joonis 4 on näidatud sensori liidestamine kontrolleriga. Antud skeem käib segafaasilise voolutarbe mõõtmise kohta, see tähendab, et on nii 3- kui 1-faasilisi tarbijaid. Näiteks korterites, kus on 3-faasiline elektripliit, aga ülejäänud süsteem on ühe faasi peal. 1 faasi korral ei ole vaja 3 sensorit, vaid piisab ainult ühest.

Mikrokontrolleri ADC sisend on võimeline hindama vaid positiivse pingega väärtusi, aga vahelduvvoolu korral vool kõigub positiivse ja negatiivse vahel, seega ka voolumõõtja väljundisse tekib negatiivset pinget. Seetõttu tõsteti V_{ref} poolde ADC maksimaalse mõõdetava pinge ulatusse pingejaguriga, mis on skeemil välja toodud R1 ja R2 ning R2 toetava kondensaatoriga C1. Sensori V_{ref} antud juhul jääb ideaalis 1,65V juurde ning V_{out} jääb pingevahemikku 0,65V – 2,65V. Seda on mikrokontroller võimeline mõõtma.

Sellega on teada juhtmes liikuv vool, millest saab arvutada võimsuse. Ühe faasi voolu võimsus on U*I ja „mistahes ühendusel ja mistahes koormusel on kolmefaasilise voolu võimsus võrdne kolme faasi võimsuste summaga“ [15]. Võimus kokku avaldub seega valemis $P = P_1 + P_2 + P_3$, kus $P_i = U * I_i$.

3.2 Kestkeade

Andmevahetuseks ja salvestamiseks vajalik API server asub miniarvutil Raspberry PI 4B, mis valiti tänu oma heale ühenduvusele ja võimekale, 64-bitisele, 1,5 GHz, neljatuumalisele Broadcom BCM2711 protsessorile, mis muudavad serveri käitamise

võimalikuks ja seadete ühendamise lihtsaks ja mugavaks. Seadmel on olemas nii 2,4 GHz kui ka 5 GHz Wi-Fi võimekus ja lisaks ka Gigabitine Etherneti ühenduse võimalus [16]. Tänu sellele saab tekitada võrgu, kuhu sensori mikrokontroller saab ühenduda Wi-Fi kaudu ja ekraani kontroller läbi Etherneti.

Raspberry PI on ka üldlevinud, mis omakorda tähendab, et sellel on olemas hea toega ja optimeeritud operatsioonisüsteem, nimelt Debianil põhinev Linuxi distributsioon Raspbian [17]. Vältimaks mälukaardi kulumist, hoitakse andmebaasi mälupulgal, SD-kaardile (mälukaart) jäeti ainult lugemisõigused, seal olevaid andmeid seade muuta ei saa.

3.3 Kasutajaliides

Mõõtetulemuste kuvamiseks ja visualiseerimiseks on kasutatud ST Microelectronicsi poolt loodud 32F746GDISCOVERY arendusplaati, millel on integreeritud 4,3'' 480x272px puutetundlik värviekraan, Arm Cortex M7-l põhinev protsessor ja RJ45 Etherneti pesa. Plaadi programmeerimine on tehtud võrdlemisi lihtsaks tänu integreeritud programmeerija-silujale. Seadmele saab 5V toite ühendada spetsiaalselt selleks mõeldud pordi ning 7-12V DC (alalisvool) toite ka GPIO klemmide kaudu [18].

ST Microelectronicsi kodulehelt tellides oli kirjutamise hetkel arendusplaadi hind $54,00 [7], mis sisaldas endas ekraani, koos mikrokontrolleriga. Samas näiteks Raspberry Pi-le samaväärsed ekraanid maksid ca $38,00-64,00 [19], mis koos arvuti endaga läheb juba üle $100 maksma. Arendusplaadil olevad ekraan ja mikrokontroller on juba ka ühendatud ning tänu sellele ei ole enam vaja lisajuhtmete ja nende haldamisega tegeleda. Sarnased ekraani ja mikroprotsessori komplektid olid veel kordades kallimad. Arendusplaadist on pilt Joonis 5

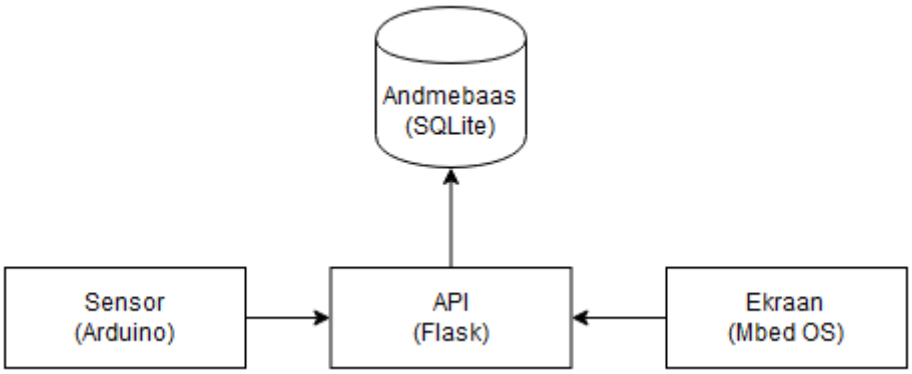
Joonis 5: STM32F746GDISCOVERY [20]

4 Tarkvara

Nõuetest tulenevalt suhtlevad seadmed omavahel üle HTTP (hüperteksti edastusprotokoll) päringute ning peamised osad on:

- mõõteseade, mis teeb mõõtmisi ja saadab neid edasi
- kasutajaliides, kus kuvada mõõtetulemusi
- server, mis on võimeline mõõtetulemusi vastu võtma, neid salvestama ja vastama kasutajaliidese tehtud andmepäringutele

Komponentide omavahelist seotust ning kasutatud raamistikke illustreerib Joonis 6.



Joonis 6: Realiseeritud süsteemi osad

Komponentidevaheline suhtlus käib läbi Flaskil põhineva API (rakendusliidese), HTTP päringute kaudu. Arduino-raamistiku põhjal programmeeritud mõõteseade saadab andmed tulemustega serverisse, mis salvestab need SQLite-il põhinevasse andmebaasi. Mbed-OS-i peal töötav kasutajaliides pärib andmed API-st ja kuvab need.

Kõik raamistikud ja kasutatud tehnoloogiad on välja toodud järgnevas peatükis.

4.1 Kasutatud tehnoloogiad

Alljärgnevalt on välja toodud peamised projektis kasutatud teegid, programmid ja raamistikud ning neid tutvustatud.

4.1.1 Raspbian Buster Lite

Raspbian on Debianil põhinev, Raspberry Pi Foundationi poolt ametlikult toetatud operatsioonisüsteem, mis on mõeldud Raspberry Pi-le [17]. See on mõeldud jooksma väga minimaalse riistvara peal ning optimeeritud olemaks võimalikult hea jõudlusega just Raspberry PI-l [21] Kuigi sama väljalaset on mitmeid erinevaid versioone, siis PI kodulehel on välja toodud 3 põhilist: Raspbian Buster Lite ilma graafilise kasutajaliidese ja lisatarkvarata, Raspbian Buster graafilise töölauaga ning Raspbian Buster graafilise töölaua ning soovitusliku lisatarkvaraga [21].

Raspbiani kasutati operatsioonisüsteemina, millel töötas API server, kuna on hästi optimeeritud ja juba mitmenda iteratsiooni projekt. Käesolevas lahenduses kasutati Lite versiooni, kuna vajadus graafilise liidese järgi puudus ning see oleks olnud seadmele lisakoormuseks.

4.1.2 Apache HTTP Server ja mod_wsgi

Apache HTTP Server [22] on enim kasutatud veebiserver, selle installeerimine, käitamine ja seadistamine toimub lihtsate käskude ja konfiguratsioonifailide kaudu. Mod_wsgi [23] on Apache moodul, mille kaudu saab läbi Apache jooksutada Pythoni veebirakendusi. See võimaldab Apache serveril vastu võtta päringuid ja neid vastavasse Pythoni skripti edasi anda ning skriptist tulnud päringuid edasi saata. Serveri valikul lähtuti ühilduvusest võimalike raamistikega ja töökorda seadmise lihtsusest.

4.1.3 Flask

Flask on programmeerimiskeele Python üks populaarsemaid raamistik veebirakenduste loomiseks, sellel põhinevad näiteks Pinterest ja LinkedIn [24]. Flaski loojad asetavad selle „mikroraamistiku“ kategooriasse: raamistiku põhiosa on hoitud lihtne ja elementaarne ning ilma millegi üleliigseta. Samas on olnud suunitlus ka maksimaalse modulaarsuse poole, jättes arendajale võimalikult vabad käed erinevate tööriistade ja tehnoloogiate valikul, näiteks andmebaasihaldus, veebilehe elementide ülesehitus, autentimine jne, mida saab erinevate moodulite kaudu tuumale lisada [25].

Flaski kasutati raamistikuna, mille põhjal oli kirjutatud API. Kuna antud projekti API kasutusjuhud olid lihtsamakoelised ja funktsionaalsus üsna piiratud, siis Flask võimaldas valida täpselt need komponendid, mida vaja ja jätta süsteem võimalikult „kergeks“. Flaski

kasutamist soodustas ka asjaolu, et autor oli juba varasemalt loonud selle raamistikuga rakendusi ning põhimõtetega tuttav. Käesoleva lõputöö näitelahenduses on kasutatud Flaski laiendeid Flask-Restful – *RESTful* ehk REST põhimõtteid arvestava API programmeerimiseks – ning Flask-SQLAlchemy – C keeles kirjutatud SQLite teegi ning Flaski liidestamiseks ning SQL (struktuurpäringukeel) päringute abstraheerimiseks mõeldud moodul – andmete haldamiseks.

4.1.4 SQLite

SQLite on üks enimkasutatumaid C teeke – seda kasutatakse peaaegu kõigis nutiseadmetes, veebilehitsejates, arvutites ja modernsemate autode multimeediasüsteemides ning paljuski mujal. Teek on mõeldud andmehalduseks ja andmebaasifailide loomiseks ning nendega liidestamiseks. SQLite'il loetakse aktiivsete baaside arvuks umbkaudu triljonit [26].

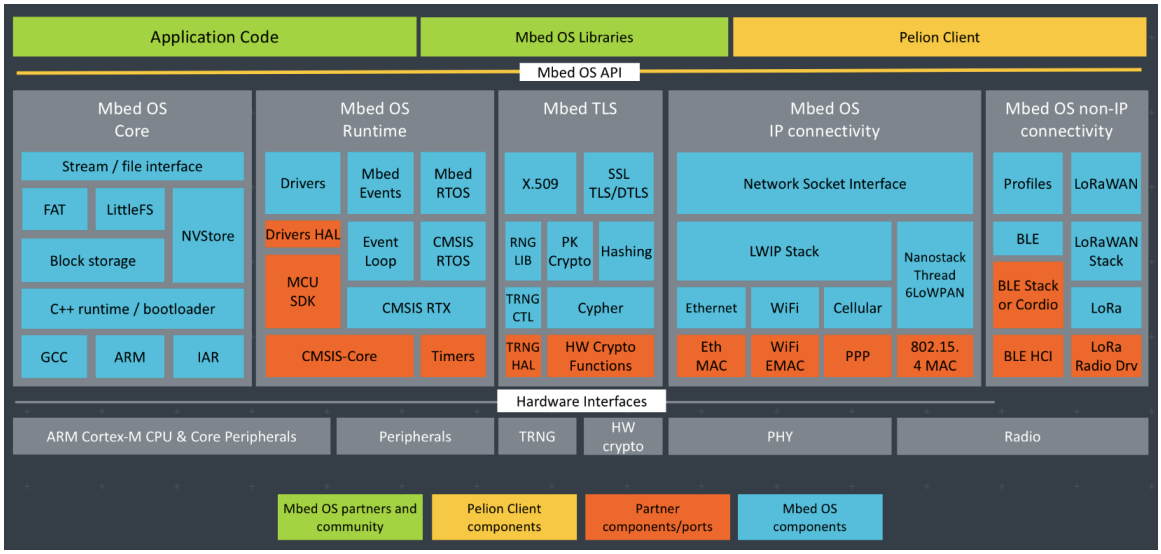
Projekti API implementatsioonis kasutati SQLite'i andmete salvestamiseks ja haldamiseks just selle lihtsuse põhimõtte tõttu: failisüsteemis on andmebaasifail, mille kasutamine teegi kaudu on vähese ressursi- ning lisakuludega. Erinevalt näiteks MySQL'ist ei nõua SQLite'i kasutamine lisaprotsesse ega eraldi serveri käitamist.

4.1.5 Mbed OS

Arm Mbed OS on madalama jõudlusega seadmetele, näiteks mikrokontrolleritele mõeldud avatud lähtekoodiga operatsioonisüsteem. Selles on kasutusel HAL (riistvara abstraktsioonikiht) enamlevinud komponentidele ja seadmetele, millega suhtlemine käib läbi OS-i API. Läbi API riistvaraga suhtlemine muudab koodi kirjutamise oluliselt efektiivsemaks, kuna puudub vajadus käsitsi registrite muutmiseks ja bittide seadmiseks, riistvara käitumise suunamiseks. Üks Mbed OS-i põhikomponente on RTOS, mille kaudu on võimalik kasutada mitut lõime – eriti oluline graafiliste programmide loomisel. Lisaks on ära abstraheeritud ühenduvusepoolne riistvara, mis tähendab, et näiteks Etherneti ja Wi-Fi ühendust saab rakendada läbi operatsioonisüsteemi pakutava liidese [27].

HAL-i ja konkreetse OS-i kasutamise eelis on lisaks koodikirjutamise lihtsustamisele ka koodi taaskasutus. Mbed OS 5.15 toetab hetkel 111 erinevat arendusplaati [28] ning samaväärse riistvaraga kontrollerite vahetamine ei nõua koodis suuri muutusi, sest

rakendused on kirjutatud just pigem operatsioonisüsteemi kui kindla seadme jaoks. Mbed OS-ist arhitektuurilise ülevaate saamiseks on see kujutatud Joonis 7.



Joonis 7: Mbed OS diagramm [27]

Operatsioonisüsteemi kasutati lahenduses ekraani mikrokontrolleri programmi alusena, sest kõik, mida ülalmainitud operatsioonisüsteem pakub, lihtsustab tarkvara sidumist riistvaraga läbi HAL-i ja draiverite ning puudub vajadus koodi nullist riistvarale kirjutada. Peaaegu kõik enamlevinud riistvara funktsioonid on kapseldatud lihtsasti kasutatavatesse teekidesse, millele programmeerijal on ligipääs. Samas ei ole välistatud ka otse riistvara poole pöördumine ja millegi väga spetsiifilise tegemine ei jää vajaduse korral operatsioonisüsteemi taha kinni. Käesoleva dokumendi kirjutamise hetkel oli viimane versioon Mbed OS 5.15.3 [29]

4.1.6 LittlevGL

LittlevGL on tasuta ja vabavaraline C keeles kirjutatud teek, mis on suunatud graafilise kasutajaliidese arendamiseks just pigem sardsüsteemidele, tänu vähesele ressursikasutusele, võimaldades seejuures modernse välimuse, efektide ja viibete kasutamist, mida tänapäeval puutetundlikelt liidestelt normiks peetakse. LvGL vajab tööks 2 funktsiooni perioodilist väljakutsumist, massiivi ekraani puhvrile ning sisendi lugemiseks ja pildi kuvamiseks 2 n-ö draiveri funktsiooni. Teegi käitamiseks on vajalik 80 kB *flash*-mälu (välkmälu) ja 12 kB RAM-i (suvapöördusmälu) [30].

Projektis kasutati ekraani graafilise liidese tegemiseks LittlevGL-i, sest Mbed OS-il sisseehitatud lahendust graafika tegemiseks ei olnud ja LittlevGL ühildub pea kõigega, võimaldades samas ilusa ja modernse välimusega liidest. Teegi kasutamine on lihtne ja loogiline, ainsa miinusena võib välja tuua, et erinevalt näiteks JavaFX-ist või Windowsi .NET raamistikest, ei ole sellele kvaliteetset graafilist tööriista, millega kasutajaliidest kokku panna.

4.1.7 Arenduskeskkonnad

Arenduskeskkonnana sensori ja API tarkvara arendusel kasutati Visual Studio Code'i (edaspidi ka VS Code) [31], millega on tänu laiendustele ja pistikprogrammidele arendamine pea kõigile enamlevinud keeltele ja platvormidele võrdlemisi mugavaks tehtud. ESP32 kasutatud teekide halduseks ja programmeerimiseks kasutati laiendust nime PlatformIO (edaspidi ka PIO) [32], mis on mõeldud sardsüsteemide arenduse ja projektihaldamise lihtsustamiseks ning väga käepäraseks lähtekoodi teekidega liidestamiseks.

STM32 ja graafika arendusel kasutati Mbed OS-i IDE-t (integreeritud arenduskeskkond) Mbed Studio, mille eelisteks on jällegi mugav teekide integreerimine projektidesse ning lisaks hea optimeeritus koodi kompileerimisel ning seadme programmeerimisel [33]. Autor proovis ka selle arendamisel kasutada PIO keskkonda, kuid ilmnes mitmeid ebameeldivusi just STM32 plaadi ning Mbed OS-iga. Välja võib tuua, et sama koodi, teekide ja operatsioonisüsteemiga pärast esmakordset kompileerimist võtsid järgnevad kompileerimised PIO-s üle 220 sekundi, Mbed Studios aga 10-20 s.

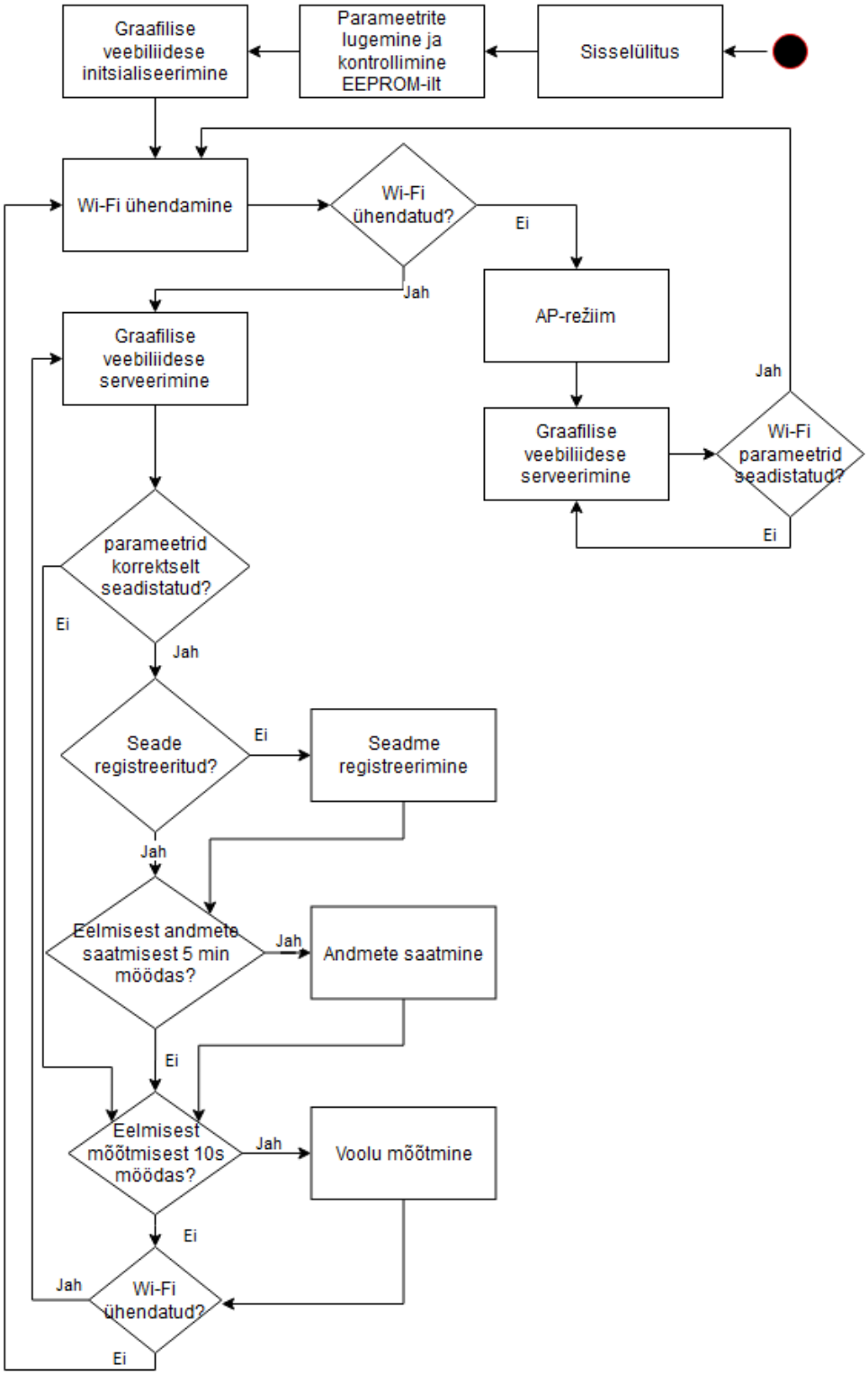
4.2 Mõõteseade

ESP32 arendus oli tehtud Arduino raamistiku põhjal, kasutades eelmainitud PlatformIO ja VS Code keskkonda. Sensorile olid järgnevad nõuded:

- Seade peab olema võimeline mõõtma ühe-, kolme- kui ka segafaasilist voolu.
- Pärast esmakordset töökorda seadmist peab see töötama iseseisvalt, ka pärast toitevoolu ajutist kadumist.

- Seadeid peab olema võimalik dünaamiliselt muuta, st arvutist või nutitelefonist saab ligi liidesele, mille kaudu muuta seadme parameetreid.

Seadme töötsükkel on kirjeldatud Joonis 8.



Joonis 8: Sensori töötsükkel

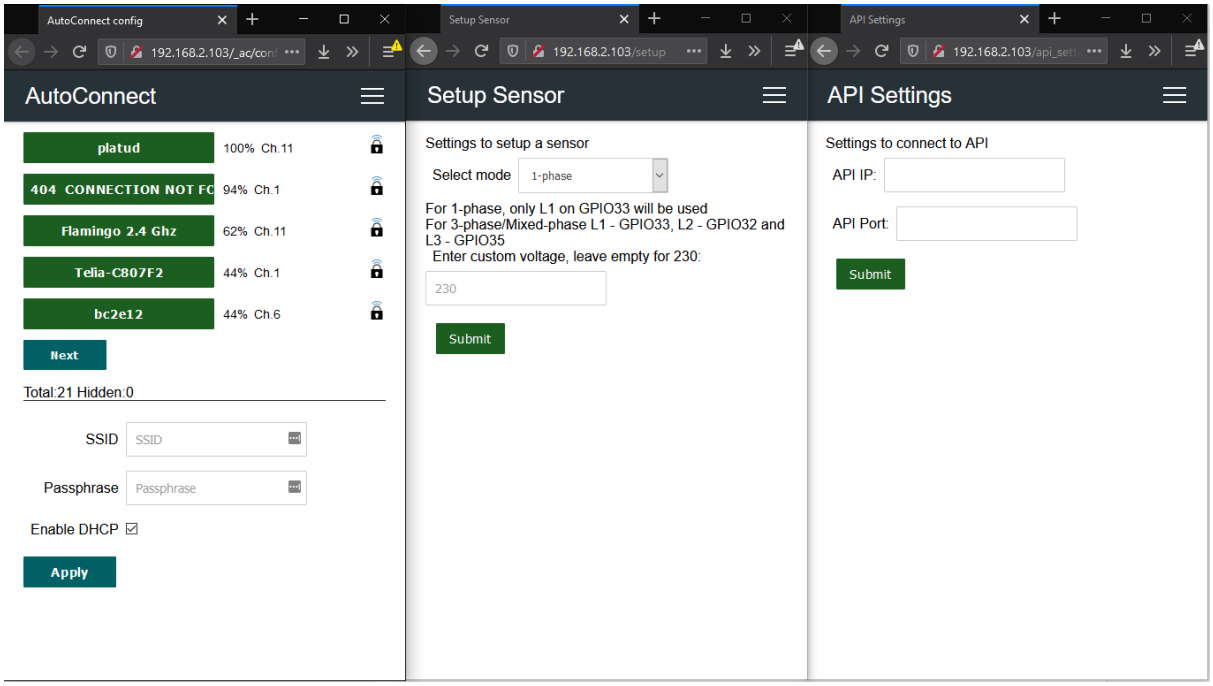
Seadme sisselülitamisel loetakse ja kontrollitakse tööks vajalikud parameetrid EEPROM-ilt, seejärel initsialiseeritakse veebiliidese tööks vajalik osa ja proovitakse leida salvestatud Wi-Fi võrku ja sellega ühenduda. Kui see ebaõnnestub, siis läheb seade AP (pääsupunkt) režiimi ja serveerib graafilist veebiliidest, millele saab ligi seadme AP-võrku ühendades. Kui Wi-Fi parameetrid on seadistatud, siis proovitakse uuesti ühenduda, see jätkub, kuni ühendus saavutatakse. Seejärel toimub tsüklis graafilise liidese serveerimine, kust saab seadistada tööks vajalikke parameetreid. Kui need on korrektsed, siis kontrollitakse, kas seade on registreeritud, kui ei, siis seade registreeritakse. Seejärel kontrollitakse, kas on piisav aeg – 5 min – möödas viimasest andmete saatmisest, kui on, siis saadetakse andmed API-sse, vastasel juhul ei saadeta. Pärast seda, kui on möödunud 10 sekundit viimasest mõõtmisest, siis mõõdetakse voolu ja tehakse vajalikud arvutused energiatarbe arvutamiseks. Realiseeritud töötsükli lähtekoodi asukoht on leitav Lisa 4, lk 53.

4.2.1 Seadistamine

Selleks, et sensor oleks võimeline andmeid saatma, vajab see võrguühendust, API aadressi, API-lt saadud id-d ning seadistatud töörežiimi: 1-, 3- või segafaasiline. Võrguga ühendumiseks on mikrokontrolleril Wi-Fi liides, sellega ühendumine nõuab SSID-d (mestiident) ning vajadusel salasõna, mille järgi tuvastada õige võrk ja sellega autentida. Selleks, et ühendamise ja seadete parameetrite muutmine oleks lõppkasutajale võimalikult lihtne, otsustati graafilise kasutajaliidese kasuks. Et andmevahetuseks vajalik Wi-Fi kiip on mikrokontrolleris juba olemas, realiseerida kasutajaliides veebilehena, kuhu oleks ligipääs arvuti või nutiseadmega.

Tekitamaks keskkonda, kuhu telefoni või arvutiga ligi pääseda ning seadeid muuta, kasutati Arduino teeki AutoConnect, mis võimaldab vähese vaevaga käitada ESP32-l ka veebiserverit, millelt serveerida veebisaiti, kus mainitud keskkond asuks [34]. AutoConnecti korrektsel kasutamisel ning mitte ühegi võrguga ühenduses olles tekitab mikrokontroller Wi-Fi AP ning sellega ühendumisel on võimalik valida Wi-Fi võrk, millesse seadmega liituda soovitakse ning seada salasõna. Pärast ühendumist salvestatakse andmed EEPROMile ning ka pärast seadme taaskäivitamist ühendub see automaatselt. Seadmel olev keskkond asub URI-l (ühtne ressursiidentifikaator) „/_ac“, kuhu mugavamaks pääsemiseks suunati juur-URI-lt sinna edasi.

Lisaks sisseehitatud Wi-Fi seadete haldusvahendile saab eelmainitud teegiga samasse võrgukeskkonda (vt ekraanipilt Lisa 3, lk 52) ka lisaks mugavalt lihtsamaid lehti tekitada, seega ka ülejäänud kasutajaliidese realiseerimiseks kasutati sama keskkonda. Võimalik on kasutada samu põhilisi elemente, mida ka tavaliselt HTML lehelt, näiteks tekstisisestuslahtreid, nuppe, HTML-vorme jne.



Joonis 9: Mõõteseadme seadete vaated

Lahenduses realiseeritud keskkonnas saab lisaks võrguühendusele, Joonis 9 vasakpoolne pilt, seadistada mõõteseadme töörežiimi ning muuta vajaduse korral elektrivõrgu pinget lehel „*Setup Sensor*“, Joonis 9 keskmine pilt, ja lehel „*API Settings*“ seada API-le ligipääsuks vajalikke IP aadressi ja porti, Joonis 9 parempoolne pilt.

Back-endina (tagasüsteem/põhiprogramm) kasutati Arduino raamistikku sisseehitatud HTTPClienti, mille instantsi kaudu käis suhtlus nii *front-endi*(kasutajaliides) kui serveriga.

4.2.2 Mõõtmised

Kuna kasutatud sensoritega vooluvõrgu pinget ei olnud võimalik mõõta, siis arvutused tehti eeldusel, et pinge on 220V. Riistvara on kokku ühendatud, nagu on näidatud peatükis 3.1. Sensorist saadud pinge tõlgendamiseks juhtmes olevaks vooluks kasutati funktsiooni *calcIrms*, mis põhineb suurelt Githubi kasutaja capella-ben projektis emon_esp32 [35]

paikneval funktsioonil, millega mõõdetakse juhtmes liikuvat voolu. Funktsioonis mõõdetakse etteantud arv kordi etteantud klemmilt pinget, võetakse mõõdetud pingete ruutkeskmine ja arvutatakse selle põhjal mõõteperioodi keskmine vool.

1-faasilise voolu jaoks sellest piisab ning energiatarbe saab arvutada, korrutades tulemuse pinge ehk 220V-ga. 3- ja segafaasilise voolu korral leiti sama funktsiooniga kõigis kolmes juhtmes liikuva voolu keskmine ja arvutati saadud voolude põhjal energiatarve.

Eeldati, et lahenduse kasutajail ei ole kodus tööstuslikel otstarvetel masinaid, vaid ainult soojusmasinad, näiteks pliit. Seetõttu ei kasutatud arvutustes võimsustegurit. Kuid see on lähtekoodi vajaduse korral lihtsasti lisatav.

4.2.3 Kalibreerimine

Riistvarast tulenevate iseärasuste tõttu võivad mikrokontrolleriga mõõdetud tulemused erineda reaalsest energiatarbest. Seadme kalibreerimiseks kasutati Tesatek pistikusse käivat energiamõõtjat ning eraldatud juhtmetega pikendusjuhet. Testseadmetena kasutati ca 2200 W võimsusega veekeedukannu, 700 W võimsusega võileivagrilli, 60 W elektripirni, 55 W televiisorit ja 25 W jootekolbi. Sensor ühendati pikendusjuhtme ühe juhi ümber, pikendusjuhe ühendati Tesateki energiamõõtja külge ning see omakorda pistikusse. Enne katsetusi mõõdeti vooluvõrgu pingeks 234 V, seda kasutati ka mikrokontrolleri seadetes. Mõõtmised toimusid pistikus oleva vattmeetri ja mikrokontrolleri abil samaaegselt.

Tabel 2 kujutab keskmiseid mõõtetulemused enne paranduste tegemist.

Tabel 2: Mõõtmistulemused enne kalibreerimist

Seade	Mikrokontroller (W)	Tesatek (W)	Erinevus
Tühi	96	0	-
Jootekolb	115	27	326%
Televiisor	127	55	131%
Lambipirn	138	64	116%
Võileivagrill	720	772	-7%
Veekeedukann	2050	2150	-5%
Kann ja grill	2700	2850	-5%

Katsetustest järeldati, et mikrokontrolleri ADC mõõdetud vool saab väärtusteks ca 0,4 A rohkem kui tegelikkuses. Sellest tulenevalt parandati voolumõõtmise funktsiooni, et saadud voolust lahutataks 0,4 A. Tehtud parandustega jäi pikendusjuhtmes mõõdetud

energiatarve koormuseta kõikuma -3 W ja 8 W piirkonda. Katsetati ka muid konstante, aga tulemust täpsemaks ei saadud ning võttes arvesse, et tegu on odava mikrokontrolleri ADC-ga, loeti saadud tulemus piisavaks ning tehti järgmine katsetus, mille mõõtetulemused on välja toodud Tabel 3.

Tabel 3: Mõõtetulemused pärast nullkoha parandamist

Seade	Mikrokontroller (W)	Tesatek (W)	Erinevus
Tühi	4	0	-
Jootekolb	21	27	-22%
Televiisor	36	55	-35%
Lambipirn	43	64	-32%
Võileivagrill	670	772	-13%
Veekeedukann	1950	2150	-9%
Kann ja grill	2600	2850	-9%

Mikrokontrolleriga tehtud mõõtetulemused erinesid tegelikkusest keskmiselt endiselt üle 10%. Arvestades suuremate koormustega saadud tulemusi, lisati mikrokontrolleris voolutugevusele 10% ning tehti uued mõõtmised, mille tulemusi kirjeldab Tabel 4.

Tabel 4: Parandustega mõõtetulemused

Seade	Mikrokontroller (W)	Tesatek (W)	Erinevus
Tühi	3	0	-
Jootekolb	25	27	-7%
Televiisor	46	55	-16%
Lambipirn	53	64	-16%
Võileivagrill	720	772	-7%
Veekeedukann	2100	2150	-2%
Kann ja grill	2790	2850	-2%

Koos nullpunkti paranduse ja vooluarvutustesse lisatud kordajaga jääb mõõtetulemuste erinevus rahuldavasse täpsuspiirkonda, eriti kõrgematel koormustel.

4.3 API

Et andmete lugemine, salvestamine ja näitamine toimusid kolmes erinevas seadmes, oli vaja need omavahel siduda. API ülesanne on olla vahelüli erinevaid protsesse läbi viiva koodi ja välismaailma vahel. Nagu klaviatuuri klahvile vajutamine tekitab ekraanile tähe, siis see, kuidas täpselt signaal jõuab arvutisse, sealt operatsioonisüsteemi, edasi konkreetsesse programmi jne kuni piksliteni ekraanil, ei ole oluline dokumenti kirjutades. Samal põhimõttel töötavad ka API-d, mille klientide jaoks on see must kast. API pakub

vastavad *endpointid*, mille kaudu on võimalik panna süsteem läbi viima soovitud toimingut, samamoodi, nagu klahvid klaviatuuril võimaldab kirjutada erinevaid tähti.

Endpointide kasutamine realiseeriti üle HTTP päringute, kuna see annab süsteemile väga suure paindlikkuse. Soovi korral saab seadmed panna tööle teisele poole maakera, rääkimata mõne(kümne)st meetrist. Projekt ei olnud kuigi keerukas ning selleks, et lahendus hoida võimalikult väike, põhines andmevahetuseks vajalik API päringute vastuvõtmiseks eelmainitud Flask Pythoni raamistikul.

Kõik kasutusjuhud, mida nõuetest tulenevalt süsteemi täielikuks tööks vaja läheb, on:

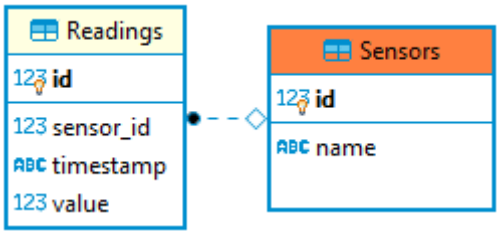
- Sensori lisamine
- Mõõtmistulemuse salvestamine andmebaasi
- Kõikide sensorite ja nende viimaste tulemuste pärimine
- Kindla sensori ja selle mõõtmistulemuste ajaloo pärimine

Seadme lähtekoodiga repositooriumi asukoht on välja toodud Lisa 4, lk 53.

4.3.1 Andmebaas

Kõik kasutusjuhud eeldavad andmebaasi olemasolu, seega esimene etapp on luua andmestruktuur.

Vastavalt kasutusjuhtudele on andmed jagatud kaheks: andmed sensorite kohta ning andmed nendelt tulnud mõõtetulemuste kohta. Seega oli vaja tekitada 2 andmemudelit ning siduda need rakenduses SQLAlchemyga. Selle tulemusena tekib andmebaasi tabel nimega „*Sensors*“ ja tabel nimega „*Readings*“ , nende illustreerimiseks on Joonis 10.



Joonis 10: Andmebaasi tabelid

Andmehalduseks, st andmebaasi tekitamiseks ja sellega suhtlemiseks kasutatakse moodulit Flask-SQLAlchemy, mis lihtsustab baasi päringute tegemist – need on

kapseldatud ja kõik toimub läbi SQLAlchemy pakutava liidese. See hoiab ära võimaluse suure hulga vigade tekkeks – andmebaas ja päringud on abstraheeritud, nende poole saab pöörduda nagu tavalistegi objektide poole. SQL päringuid tehakse siiski, aga juba automaatselt SQLAlchemy poolt. Lisaks on hoolitsetud ka andmebaasi kontrolleriga – antud töös SQLite – liidestamise eest [36].

4.3.2 Kasutusjuhud ja päringud

Moodul Flask-RESTful võimaldab siduda URI külge soovitud funktsionaalsus. Vastaval *endpointil* peab olema defineeritud funktsioon, mille nimi on päringutüübiga sama, näiteks GET päringu korral funktsionaalsuse tagamiseks peab olema defineeritud funktsioon „get“. Sellega on võimalik kontrollida, milliseid päringuid teha saab ja milliseid mitte. Väärtus, mille funktsioon tagastab, on ka päringu vastuseks, õnnestunud päringu korral tagastatakse vaikimisi ka kood 200. Päringu vastus, selle kood ja vastuse tüüp on kõik lihtsasti muudetavad. Kõikide päringu vastuseks on sõnastike massiivid informatsiooni mugavamaks väljasõelumiseks. Kasutusel olevad päringud on välja toodud Tabel 5.

Tabel 5: API päringud

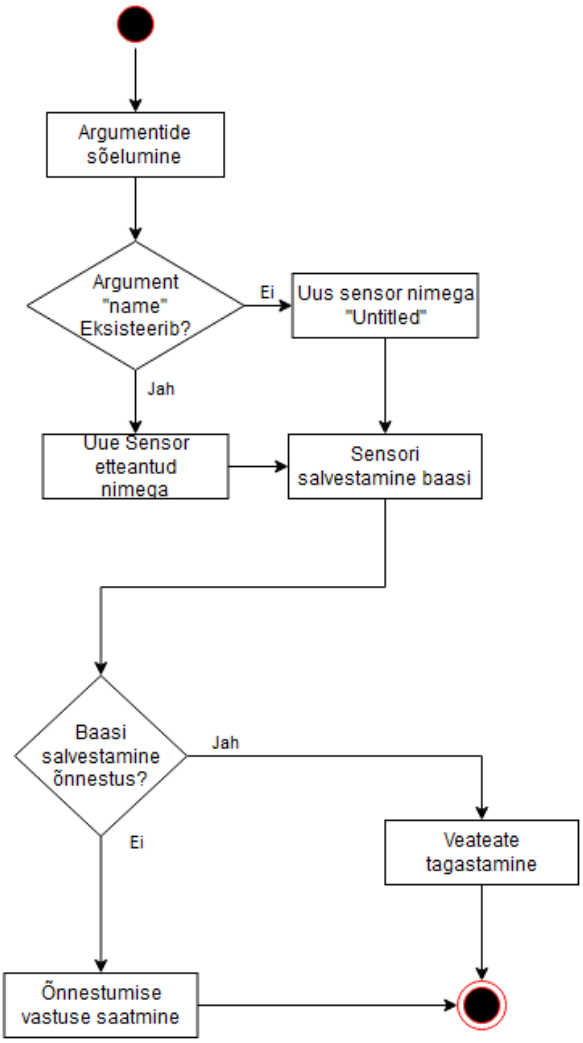
Eesmärk	Päring	Argumendid, formaat	Õnnestumise korral päringu tagastatav info

Sensori registreerimine	POST /sensors	[name], URL-encoded	[
			{
			'message': 'Sensor created',
			'id': 3
			}
			]

Mõõtetulemuse salvestamine	POST /reading	<id> <value>, URL-encoded	[
			{
			'message': 'Reading saved: 15 kW'
			}
			]

Ühe sensori tulemuste pärimine	GET /sensors/<id>		[{ 'id': 23, 'reading': 3.5 'time': '13.12.2011 10:09' }, { 'id': 22, ... }, ...]
Kõigi sensorite ja nende viimase tulemusega	GET /sensors		[{ 'id': 1, 'reading': 3.5 'time': '13.12.2011 10:09' }, { 'id':2, ... }, ...]

Sensori lisamine kujutab endast uue rea lisamist tabelisse „*Sensors*“, milleks ettenähtud API *endpoint* nõuab HTTP POST päringut URI-le „/sensors“. Argumendina võib soovi korral kaasa anda ka vabalt valitud nime sensorile. Õnnestunud sensori lisamise korral tagastatakse sõnumi sees id, mille sensor salvestab ja edaspidi saadab tulemusi sama *id*-ga. Sensori lisamise protsess on kujutatud Joonis 9:

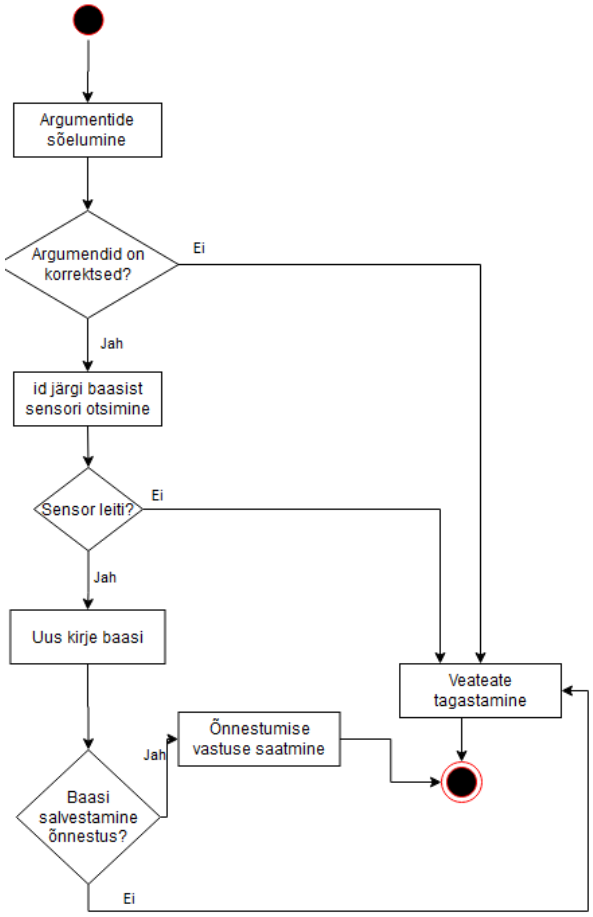


Joonis 11: Uue sensori loomise tegevusdiagramm

Päringu saamisel salvestatakse argumendina kaasa antud nimega või nimega „*Untitled*“ sensor andmebaasi. Selle ebaõnnestumise korral tagastatakse päringu vastusena veateade, õnnestumise korral aga sõnum õnnestumise kohta ja loodud kirje *id* (vt Tabel 5, lk 33).

Mõõtmistulemused saadetakse HTTP POST päringuga URI-le „/reading“, kuhu argumentideks kaasa pidid olema antud sensori id ja tulemuse väärtus. Pärast päringu

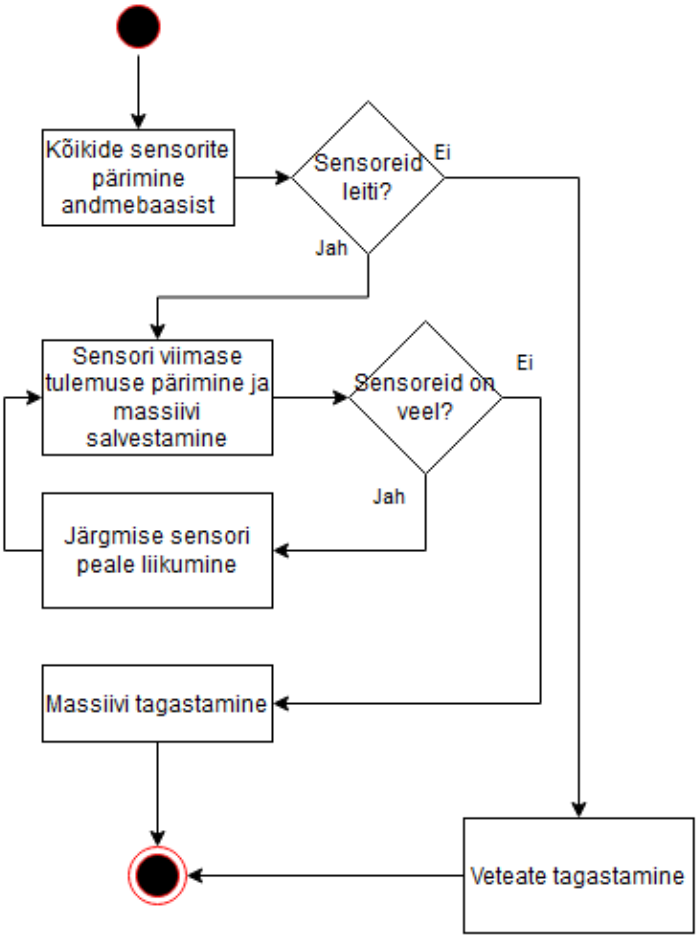
saamist luuakse uus kirje tulemuste tabelisse ja salvestatakse see andmebaasi. Protsess on näha Joonis 12:



Joonis 12: Tulemuse salvestamise tegevusdiagramm

Päringu saamisel argumendid sõelutakse ning korrektsete mõõteseadme *id* ja mõõtetulemuse korral otsitakse andmebaasist id järgi vastav sensor. Ebaõnnestumise korral tagastatakse veateade. Kui baasist leitakse otsitav sensor, siis tekitatakse sellega seotud kirje andmebaasi, tulemuste tabelisse, ebaõnnestumise korral tagastatakse veateade. Kui andmebaasi salvestamine õnnestub, saadetakse päringule vastus õnnestumise teatega, vastasel juhul aga veateade. (vt Tabel 5, lk 33)

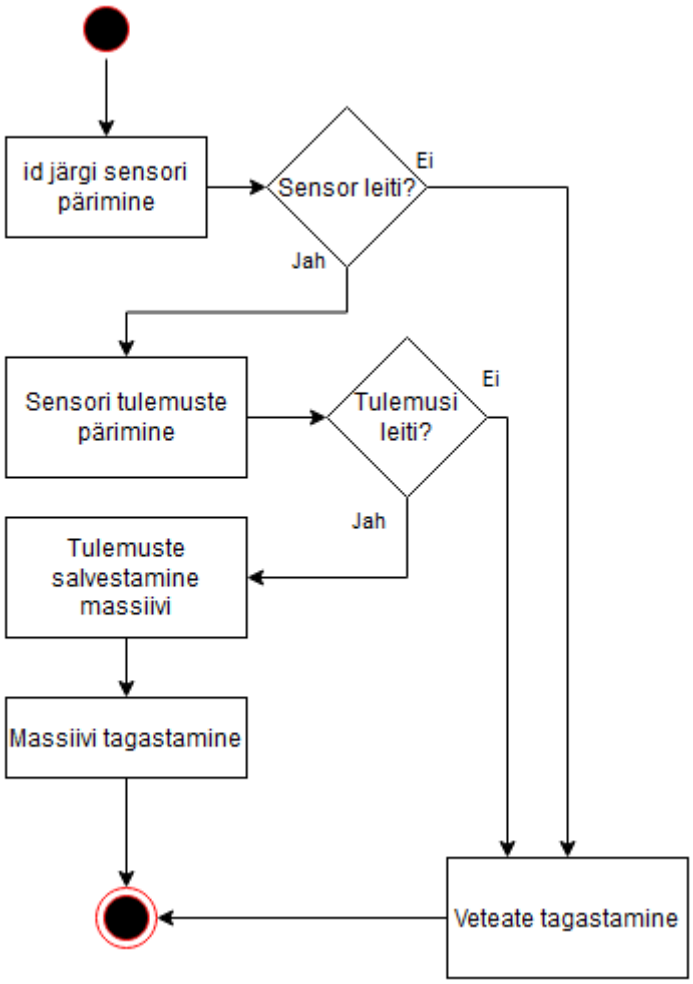
Kõikidest sensoritest hetkeülevaate saamiseks on vajalik pärida kõik sensorid ja kõigi nende viimased mõõtetulemused. Tegevusele vastas API *endpoint* „/sensors“, millele on vaja teha GET päring. Õnnestunud päringu korral tagastatakse sõnastike massiiv, mille võtmeteks on „*id*“, „*reading*“ ja „*time*“ ning neile vastavad baasist tagastatud väärtused. Protsessi kirjeldab Joonis 13:



Joonis 13: Kõikide sensorite viimase info pärimine

HTTP päringu saamisel päritakse andmebaasist sensorid, kui neid ei leita, vastatakse päringule veateatega. Kui sensoreid leitakse, siis tsüklis käiakse need ükshaaval läbi ja tehakse päring baasi iga sensori viimase mõõtetulemuse kohta. Tulemused salvestatakse massiivi; juhul, kui tulemusi ei leitud salvestatakse massiivi küll sensori *id*, kuid tühjade andmeväljadega. Tsükli lõppedes tagastatakse päringu vastusena tulemustega massiiv. (vt Tabel 5, lk 33)

Kindla sensori ja selle mõõtmistulemuste ajaloo pärimiseks on vaja teha GET päring aadressile „/sensors/<id>“, kus <id> asemele käib soovitava sensori id. Õnnestunud päringu korral tagastatakse sõnastike massiiv, mille võtmeteks on „*id*“, „*reading*“ ja „*time*“ ning neile vastavad baasist tagastatud väärtused. Joonis 14 illustreerib päringu tulemusena läbiviidavat protsessi.



Joonis 14: Ühe sensori tulemuste päring

HTTP päringu peale päritakse andmebaasist *id* põhjal sensor. Kui seda ei leita, siis tagastatakse veateade, vastasel juhul päritakse selle saadetud tulemused. Kui tulemusi ei leita, tagastatakse veateade, kui leitakse siis tagastatakse päringu vastusena ülalmainitud massiiv. (vt ka Tabel 5, lk 33)

4.4 Kuvar

Ekraan peab olema võimeline API-st pärima informatsiooni sensorite kohta ja seda kuvama, see eeldab kolme põhilist kasutusjuhtu:

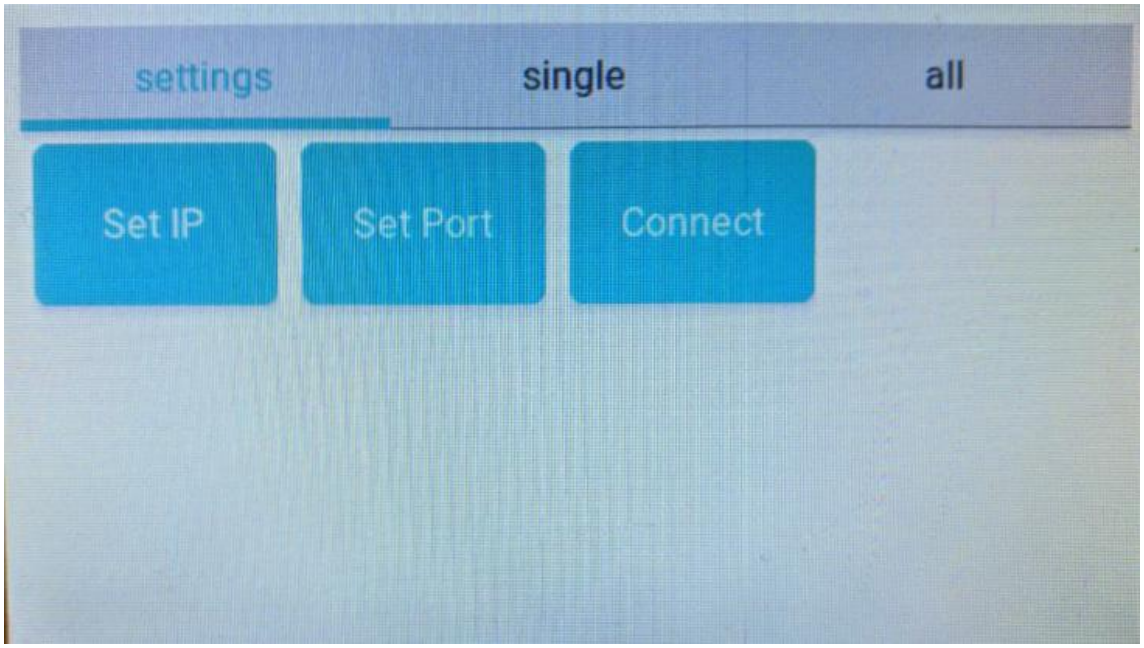
- Kõigi sensorit andmete pärimine.
- Ühe sensori tulemuste pärimine.

- Ühenduse parameetrite seadistamine.

Kasutatud riistvara 4,3’’ ekraan on kogu funktsionaalsuse korraga näitamiseks võrdlemisi väike, seega on kasutajaliides jaotatud kasutusjuhtude järgi eraldi vaadeteks. Kõik vaated on omavahel ühendatud *navbar*-i ehk navigeerimisriba kaudu ekraani ülaosas ning vaated ise moodustavad selle paanid. Kasutajaliidese põhimõte on hoida see võimalikult lihtne, ent samas võimalikult mugav ja hea välimusega. Link lähtekoodi repositooriumile on Lisa 4, lk 53.

4.4.1 Seaded

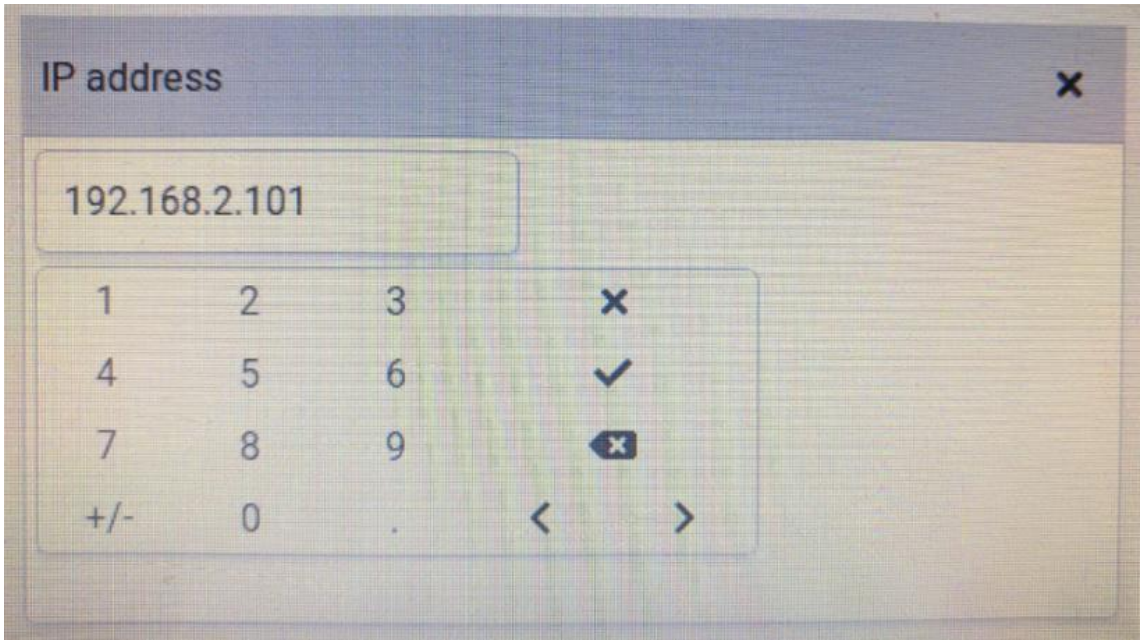
Enne, kui süsteem on võimeline andmeid pärima, peab see olema ühendatud Etherneti kaabliga ning samasse LAN-i, kus asub ka API server. Selleks on vaja seadmele märku anda, kui kaabel on ühendatud, et see end ka tarkvaras ühendaks ja ruuterilt IP aadressi päriks. Lisaks peab kasutajal võimaldama seada API IP aadressi ja porti sellega ühendamiseks.



Joonis 15: Seadete paani vaade diagramm

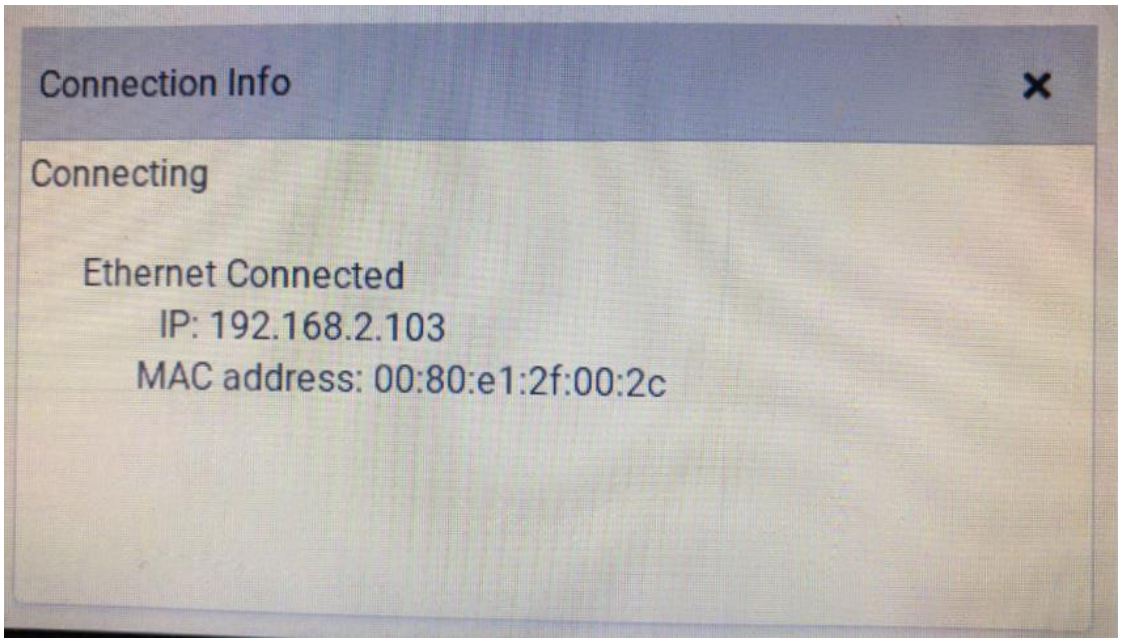
Joonis 15 on näidatud seadete vaadet kasutajaliideses. *Navbar* võimaldab liikuda vaatesse „*Settings*“, mille osa see ka ise on. Lisaks on vaatel veel 3 nuppu, „*Set Port*“, „*Set IP*“ ja „*Connect*“. Esimesed 2 nuppu avavad sarnased aknad: tekstikast vastavalt kas pordi või IP aadressi sisestamiseks, vaikimisi loetakse välkmälust salvestatud parameetrid ning neid on näha tekstikastis, kus *numpad*-i abil neid muuta saab. Vajutades *numpad*-i *enter*-

nuppu, salvestatakse tekstikastis olev kas siis IP või pordi muutujasse, vastavalt aknale. Nupu „*Set IP*“ vajutusel tekkivat akent visualiseerib Joonis 16.



Joonis 16: IP aadressi seadmise vaade

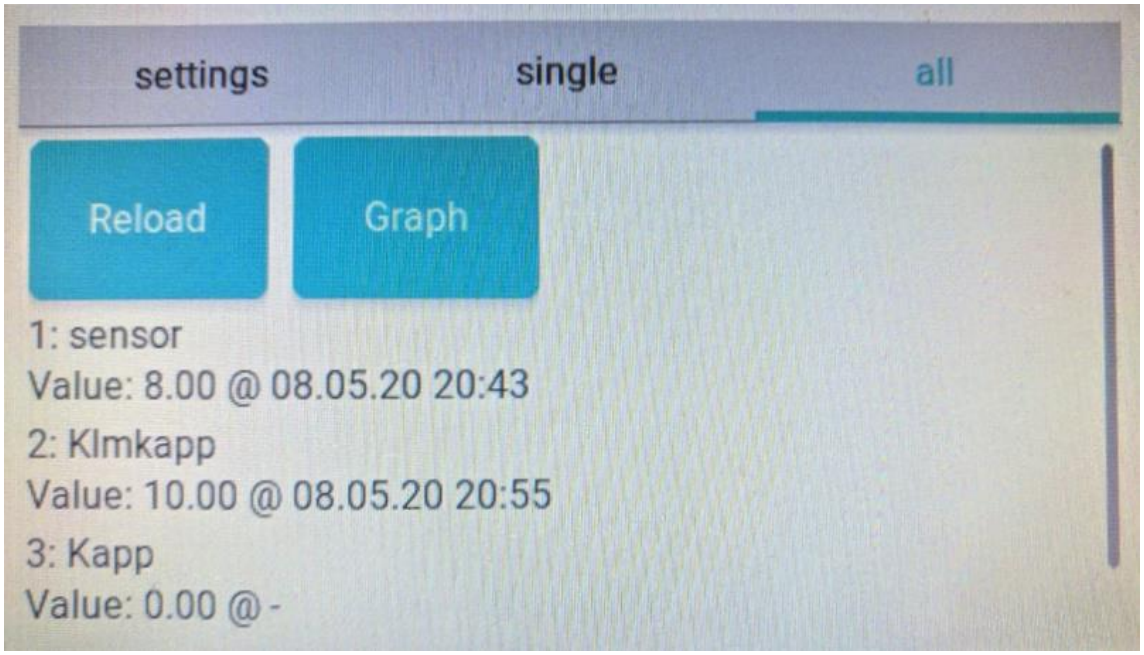
„*Connect*“-nupp käivitab funktsiooni, mis ühendab operatsioonisüsteemi võrguliidese kaudu riistvaralise Etherneti pesa tarkvaraga ning seejärel pärib IP aadressi. Pärast ühendumisprotsessi lõppu avatakse aken, kus on kuvatud IP ja MAC aadressid, mida on näha Joonis 17. Protsessi ebaõnnestumise korral kuvatakse veateade (vt Lisa 2, lk 51).



Joonis 17: Ühenduse info vaade

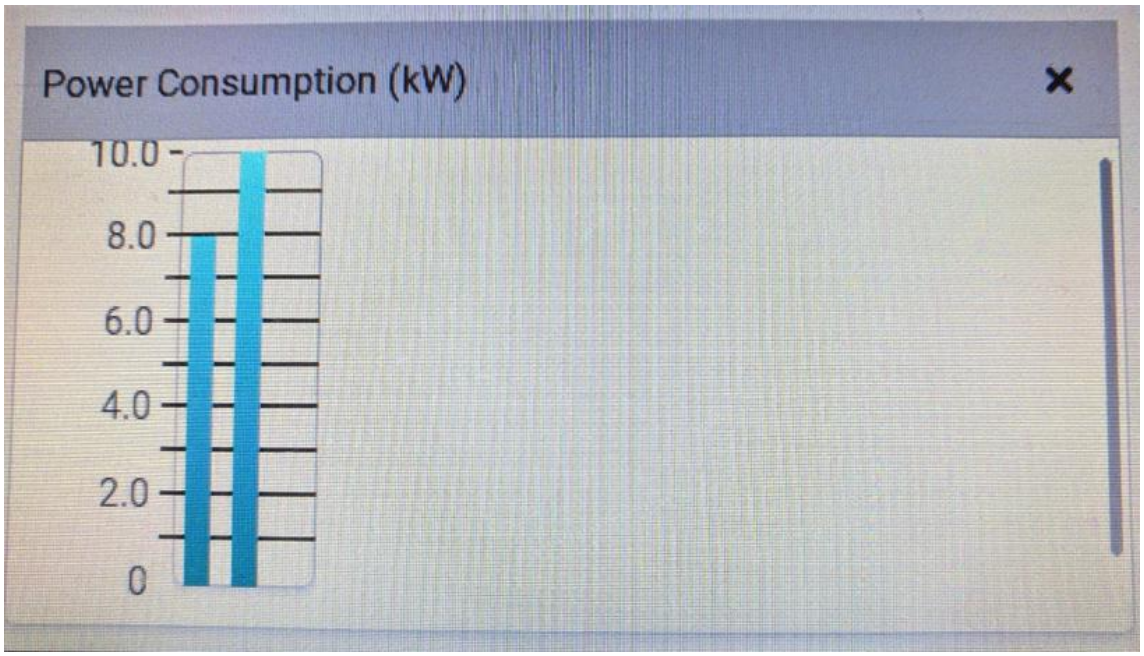
4.4.2 Kõigi sensorite vaade

Vaade on mõeldud kõikide sensorite ning nende viimaste mõõtetulemuste kuvamiseks. Need päritakse API-lt, töödeldakse ja saadud andmed antakse edasi UI (kasutajaliides) elementidele näitamiseks. Andmete põhjal on võimalik kuvada ka tulpdiagramm, kus on peal erinevate sensorite viimased mõõtetulemused.



Joonis 18: Kõikide sensorite vaade

Joonis 18 kuvatakse kõikide sensorite vaadet. *Navbar* võimaldab liikuda vaatesse „*All*“, kus asuvad 2 nuppu „*Reload*“ ja „*Graph*“. „*Reload*“ nupp teeb sensori kohta päringu API-sse, kui võrguühendus on olemas, kui seda ei ole, avatakse uus aken, kus kuvatakse veateade (vt Lisa 2, lk 51). Vajutades aknal sulgemise nuppu, liigutakse tagasi kõikide sensorite vaatesse. Pärast eduka päringu tegemist töödeldakse andmed ja salvestatakse massiivi, mille põhjal luuakse kirjetele vastav arv tekstielemente, mille tekstiks saab vajalik informatsioon andmetest: sensori *id*, nimi, viimane mõõtetulemus ja selle aeg.

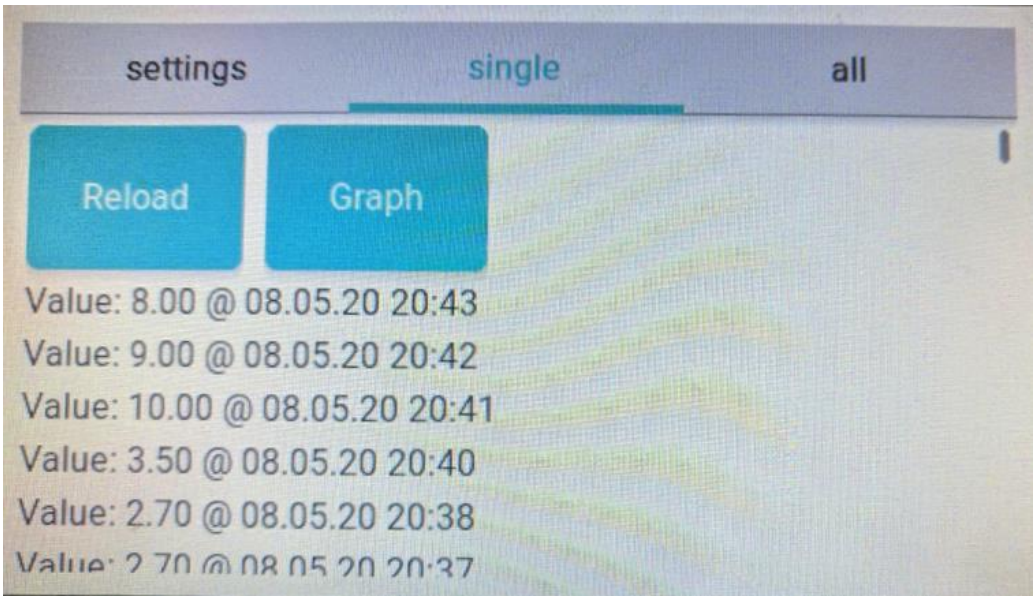


Joonis 19: Kõikide sensorite graafik

Nupuvajutusega „*Graph*“ arvutatakse andmete põhjal graafiku punktide koordinaadid ning luuakse uus aken „*Graph*“, mis on näidatud Joonis 19. Aknas on tulpdiagramm sensorite viimaste mõõtetulemuste kohta ja nupp aknast lahkumiseks, mis viib tagasi vaatesse „*All*“. Tekstielementidele vajutamine teeb vastava *id*-ga sensori kohta päringu API-sse ja viib vaatele „*Single*“.

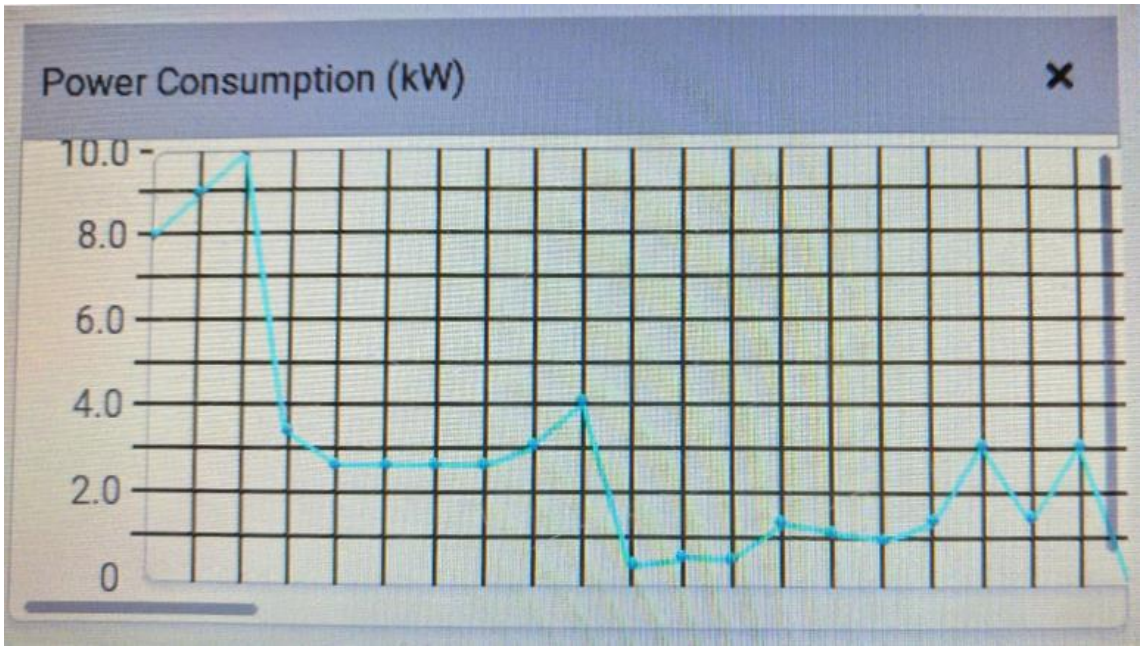
4.4.3 Ühe sensori vaade

Vaade on mõeldud konkreetse sensori viimaste mõõtetulemuste kuvamiseks. Need päritakse API-lt, töödeldakse ja saadud andmed antakse edasi UI elementidele näitamiseks. Andmete põhjal on võimalik kuvada ka joondiagramm, kus on peal sensori viimased mõõtetulemused.



Joonis 20: Ühe sensori vaade

Joonis 20 illustreerib ühe sensori vaadet, kus *navbar* võimaldab liikuda vaatesse „*Single*“, kus asuvad 2 nuppu „*Reload*“ ja „*Graph*“. „*Reload*“ nupp teeb mõõtetulemuste kohta päringu API-sse, kui võrguühendus on olemas, kui seda ei ole, avatakse uus aken, kus kuvatakse veateade. Vajutades aknal sulgemise nuppu, liigutakse tagasi konkreetse sensori vaatesse. Pärast eduka päringu tegemist töödeldakse andmed ja salvestatakse massiivi, mille põhjal luuakse kirjetele vastav arv tekstielemente, mille tekstiks saab vajalik informatsioon andmetest: mõõtetulemus ja selle tegemise aeg.



Joonis 21: Ühe sensori graafik

Nupuvajutusega „*Graph*“ arvutatakse andmete põhjal graafiku punktide koordinaadid ning luuakse uus aken „*Graph*“. Aknas on joondiagramm mõõtetulemuste kohta ja nupp aknast lahkumiseks, mis viib tagasi vaatesse „*Single*“. Aken ja sellel olev graafik on välja toodud Joonisel 20.

5 Kokkuvõte

Antud lõputöö eesmärgiks oli luua energia mõõtmise lahendus, millega saaks jälgida energiatarvet ilma internetiühenduse ja lisaseadmeteta. Arenduse alguses seati nõuded, et süsteem oleks turul olevate seadmetega konkurentsivõimeline, neist lähtuvalt sai ülesandeks arendada neile vastav lahendus.

Arenduse jooksul jagati probleem osadeks ning arendati alamprobleeme lahendavad seadmed: voolumõõtja, graafiline liides ja keskseade. Voolumõõtja kujutab endast juhtme ümber kinnitatavat voolumõõtjat ja mikrokontrollerit, mille kaudu tehakse vajalikud mõõtmised ja arvutused ning saadetakse tulemused edasi. Andmete vastuvõtmiseks, salvestamiseks ja haldamiseks loodi miniarvutil põhinev server, kuhu saab HTTP päringute kaudu andmeid salvestada ja neid pärida. Andmete hoidmiseks kasutatakse samas serveris paiknevat andmebaasi. Informatsiooni kuvamine toimub graafilise liidese kaudu, milleks on arendusplaat, kuhu on ühendatud ekraan ja mikrokontroller. Selle abil on võimalik vaadata hetketarvet ja tarbimisajalugu ning luua visualiseerimiseks graafikuid.

Kasutatud kirjandus

[1] Amazon.com, Inc, „Amazon.com: Efergy Engage hub kit: Home Audio & Theater,“ Amazon.com, Inc, 2020. [Võrgumaterjal]. Available: https://www.amazon.com/dp/product/B0128Z0K1I/. [Kasutatud 4 May 2020].

[2] Amazon.com, Inc, „Sense Energy Monitor – Track Electricity Usage in Real Time and Save Money – Meets Rigorous ETL/Intertek Safety Standards -- Amazon.com,“ Amazon.com, Inc, 2020. [Võrgumaterjal]. Available: https://www.amazon.com/dp/product/B075K6PHJ9/?th=1. [Kasutatud 4 May 2020].

[3] Emporia Energy , „Emporia Energy Smart Home Energy Monitor Devices,“ Emporia Energy, 2020. [Võrgumaterjal]. Available: https://shop.emporiaenergy.com/. [Kasutatud 4 May 2020].

[4] „Efergy Engage Hub Kit,“ 05 2019. [Võrgumaterjal]. Available: https://efergy.com/wp-content/uploads/2019/05/ENGAGE-HUB-KIT-3Mini.jpg. [Kasutatud 18 05 2020].

[5] „AliExpress,“ [Võrgumaterjal]. Available: https://www.aliexpress.com/w/wholesale-sct013.html. [Kasutatud 17 05 2020].

[6] „esp32 - Buy esp32 with free shipping on AliExpress version,“ AliExpress.com, 26 April 2020. [Võrgumaterjal]. Available: https://www.aliexpress.com/w/wholesale-esp32.html. [Kasutatud 26 April 2020].

[7] ST Microelectronics, „32F746GDISCOVERY,“ 2020. [Võrgumaterjal]. Available: https://www.st.com/en/evaluation-tools/32f746gdiscovery.html. [Kasutatud 17 05 2020].

[8] AliExpress.com, „AliExpress,“ AliExpress.com, 2020. [Võrgumaterjal]. Available: https://www.aliexpress.com/w/wholesale-raspberry-pi-4-b.html. [Kasutatud 17 05 2020].

[9] AliExpress.com, „AliExpress,“ AliExpress.com, 2020. [Võrgumaterjal]. Available: https://www.aliexpress.com/popular/micro-sd-card.html. [Kasutatud 17 05 2020].

[10] AliExpress.com, „AliExpress,“ AliExpress.com, 2020. [Võrgumaterjal]. Available: https://www.aliexpress.com/category/711005/usb-flash-drives.html. [Kasutatud 17 05 2020].

[11] „Stormwandler Zeichnung - Current transformer - Wikipedia,“ 15 04 2009. [Võrgumaterjal]. Available: https://en.wikipedia.org/wiki/Current_transformer#/media/File:Stromwandler_Zeichnung.svg. [Kasutatud 17 05 2020].

[12] Flipkart, „Flipkart,“ [Võrgumaterjal]. Available: https://rukminim1.flixcart.com/image/832/832/jfyaf0w0/learning-toy/s/e/u/30a-sct-013-030-non-invasive-ac-current-sensor-sunrobotics-original-imaf4b9dywccvegn.jpeg?q=70. [Kasutatud 3 may 2020].

[13] MCI Electronics, „Split core current transformer model SCT-013,“ MCI Electronics.

[14] Espressif Systems, „ESP32 Series Datasheet,“ Espressif Systems, Shaghai, 2020.

[15] „1 - 7Kolmefaasiline.pdf,“ [Võrgumaterjal]. Available: http://www.ene.ttu.ee/leonardo/elektro_alused/7Kolmefaasiline.pdf. [Kasutatud 17 05 2020].

[16] R. P. Foundation, „Raspberry Pi 4 Model B specifications - Raspberry Pi,“ Raspberry Pi Foundation, [Võrgumaterjal]. Available: https://www.raspberrypi.org/products/raspberry-pi-4-model-b/specifications/. [Kasutatud 1 May 2020].

[17] raspberrypi, „Github,“ Raspberrypi Foundation, 2019. [Võrgumaterjal]. Available: https://github.com/raspberrypi/documentation/blob/master/raspbian/README.md. [Kasutatud 1 may 2020].

[18] ST Micrelectronics, „UM1907 - Discovery kit for STM32F7 Series with STM32F746NG MCU,“ ST Micrelectronics, 2020.

[19] The PiHut, „Raspberry Pi Screens & Displays,“ [Võrgumaterjal]. Available: https://thepihut.com/collections/raspberry-pi-screens. [Kasutatud 17 05 2020].

[20] [Võrgumaterjal]. Available: https://docs.zephyrproject.org/latest/_images/stm32f746g_disco.jpg. [Kasutatud 07 05 2020].

[21] Raspberry Pi Foundation, „Download Raspbian for Raspberry Pi,“ 13 02 2020. [Võrgumaterjal]. Available: http://downloads.raspberrypi.org/raspbian/. [Kasutatud 4 5 2020].

[22] The Apache Software Foundation, 2020. [Võrgumaterjal]. Available: https://httpd.apache.org/. [Kasutatud 17 05 2020].

[23] G. Dumpleton, 2020. [Võrgumaterjal]. Available: https://modwsgi.readthedocs.io/en/develop/. [Kasutatud 17 05 2020].

[24] Wikipedia Foundation Inc., „Flask (web framework) - Wikipedia,“ Wikipedia Foundation Inc., 19 04 2020. [Võrgumaterjal]. Available: https://en.wikipedia.org/wiki/Flask_(web_framework). [Kasutatud 20 05 2020].

[25] Pallets, „Foreword - Flask Documentation (1.1.x),“ Pallets, 2010. [Võrgumaterjal]. Available: https://flask.palletsprojects.com/en/1.1.x/foreword/. [Kasutatud 05 05 2020].

[26] SQLite Consortium, „Most Widely Deployed SQL Engine,“ SQLite Consortium, [Võrgumaterjal]. Available: https://www.sqlite.org/mostdeployed.html. [Kasutatud 05 05 2020].

[27] Arm Limited, „Introduction - Mbed OS 5 | Mbed OS 5 Documentation,“ 2020. [Võrgumaterjal]. Available: https://os.mbed.com/docs/mbed-os/v5.15/introduction/images/Mbed_OS_diagram_for_intro.png. [Kasutatud 05 05 2020].

[28] Arm Limited, „Development boards | Mbed,“ Arm Limited, 2020. [Võrgumaterjal]. Available: https://os.mbed.com/platforms/. [Kasutatud 05 05 2020].

[29] Arm Limited, „Releases | Mbed,“ Arm Limited, 2020. [Võrgumaterjal]. Available: https://os.mbed.com/mbed-os/releases/. [Kasutatud 05 05 2020].

[30] G. Kiss-Vámosi, „Welcome to LittlevGL's documentation - LittlevGL 6.1.2 documentation,“ 2019. [Võrgumaterjal]. Available: https://docs.littlevgl.com/en/html/index.html. [Kasutatud 06 05 2020].

[31] [Võrgumaterjal]. Available: https://code.visualstudio.com/. [Kasutatud 17 05 2020].

[32] PlatformIO, „A new generation ecosystem for embedded development - PlatformIO,“ [Võrgumaterjal]. Available: https://platformio.org/. [Kasutatud 17 05 2020].

[33] Arm Limited, 2020. [Võrgumaterjal]. Available: https://os.mbed.com/studio/. [Kasutatud 17 05 2020].

[34] H. Ikasamo, „Overview - AutoConnect for ESP8266/ESP32,“ 2020. [Võrgumaterjal]. Available: https://hieromon.github.io/AutoConnect/index.html. [Kasutatud 09 05 2020].

[35] capella-ben, „emon_esp32/Power_Monitor.ino at master . capella-ben/emon-esp32,“ 11 12 2016. [Võrgumaterjal]. Available: https://github.com/capella-ben/emon_esp32/blob/master/Power_Monitor.ino. [Kasutatud 09 05 2020].

[36] Pallets, „Quickstart - Flask-SQLAlchemy Documentatin (2.x),“ 2020. [Võrgumaterjal]. Available: https://flask-sqlalchemy.palletsprojects.com/en/2.x/quickstart/. [Kasutatud 10 05 2020].

[37] [Võrgumaterjal]. Available: https://www.raspbian.org/. [Kasutatud 04 05 2020].

[38] M. Grinberg, „Flask-Migrate - Flask-Migrate documentation,“ 2019. [Võrgumaterjal]. Available: https://flask-migrate.readthedocs.io/en/latest/. [Kasutatud 08 05 2020].

[39] Z. Stone, 2 10 2014. [Võrgumaterjal]. Available: https://www.electricalpereview.com/square-root-three-3-electrical/. [Kasutatud 17 05 2020].

Lisa 1 Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

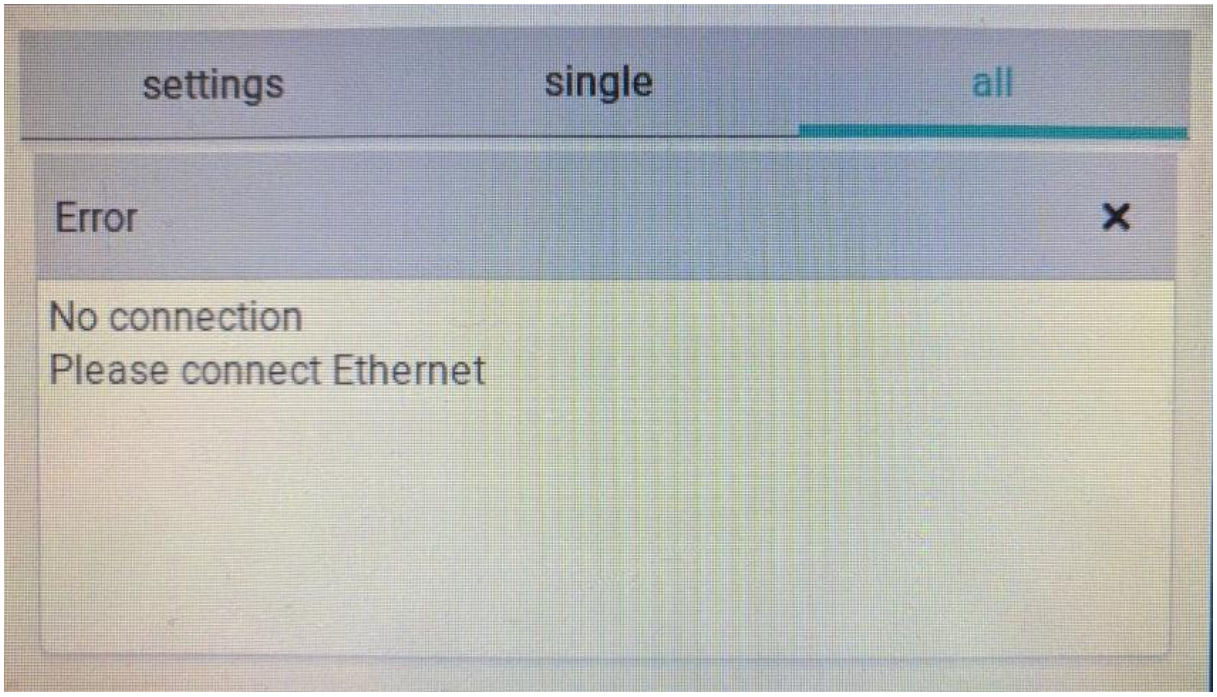
Mina, Paul Ööbik

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose „Energiatarbe mõõtmise ja visualiseerimise lahendus“, mille juhendajad on Mairo Leier ja Karl Laanemets
   1.1. reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
   1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.
3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

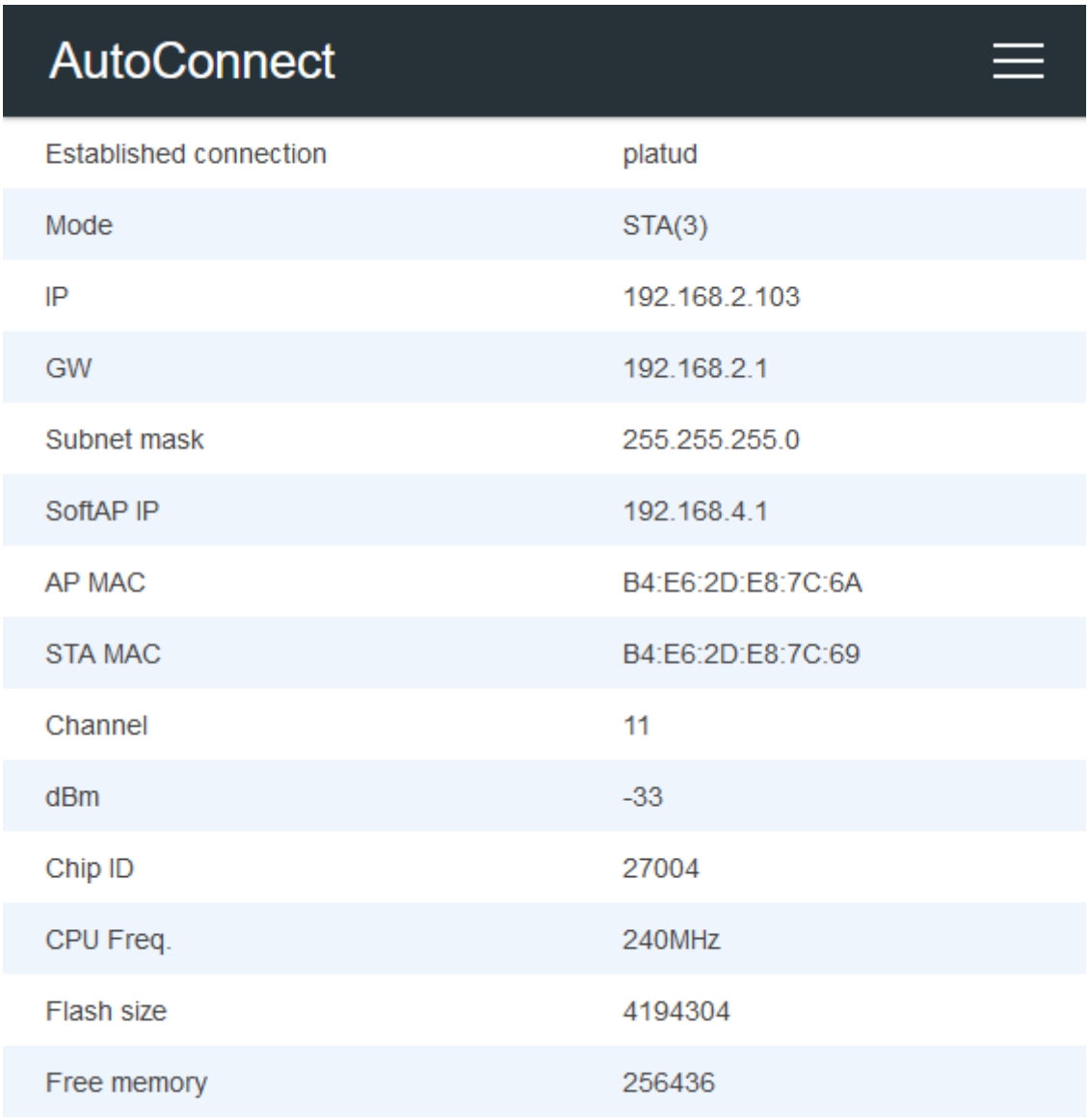
08.01.2021

1 Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või enam isikut oma ühise loomingulise tegevusega ning lõputöö kaas- või ühisautor(id) ei ole andnud lõputööd kaitsvale üliõpilasele kindlaksmääratud tähtajaks nõusolekut lõputöö reprodutseerimiseks ja avalikustamiseks vastavalt lihtlitsentsi punktidele 1.1. ja 1.2, siis lihtlitsents nimetatud tähtaja jooksul ei kehti.

Lisa 2 Veateade



Lisa 3 Mõõteseadme veebikeskkonna avaleht



Lisa 4 Lähtekood

Lähtekood on kättesaadav *Gitlab*-i repositooriumitest, mis asuvad alljärgnevatel aadressidel:

- Sensor – https://gitlab.pld.ttu.ee/thesis/2020_paul_oobik/sensors.git
- Kasutajaliides – https://gitlab.pld.ttu.ee/thesis/2020_paul_oobik/remote_display.git
- Keskseade – https://gitlab.pld.ttu.ee/thesis/2020_paul_oobik/server_api.git