EESTI MAAÜLIKOOL
Metsanduse ja inseneeria instituut

Jaan Jalas

MIKROKONTROLLERITE JA RASPBERRY PI KASUTAMINE NUTIKATE KODUDE ENERGIAKASUTUSE OPTIMEERIMISEL

USING MICROCONTROLLERS AND RASPBERRY PI TO OPTIMIZE THE ENERGY USAGE IN SMART HOMES

Bakalaureusetöö

Tehnika ja tehnoloogia päevaõppe õppekava

Juhendaja: vanemlektor Janar Kalder, *PhD*

Tartu 2024

www.emu.ee
Eesti Maaülikool
Estonian University of Life Sciences

Eesti Maaülikool Kreutzwaldi 1, Tartu 51014		Bakalaureusetöö lühikokkuvõte	
Autor: Jaan Jalas		Õppekava: Tehnika ja Tehnoloogia	
Pealkiri: Mikrokontrollerite ja Raspberry Pi kasutamine nutikate kodude energiakasutuse optimeerimisel			
Lehekülgi: 63	Jooniseid: 23	Tabeleid: 0	Lisasid: 8
Osakond / Õppetool: Energiakasutuse õppetool ETIS-e teadusvaldkond ja CERC S-i kood: 4. Loodusteadused ja tehnika, T140 Energeetika Juhendaja: Janar Kalder, *PhD* Kaitsmiskoht ja -aasta: Tartu 2024			

Kodumajapidamiste elektrienergia lõpptarbimine on viimase paarikümne aastaga tõusnud, elektriseadmete arvulise kasvu ning elektrifitseerimise tulemusena. Energiatarbimise vähendamiseks on vaja luua innovaatilisi lahendusi. Bakalaureusetöö raames uuriti nutikaid kodusid, Raspberry Pi-d, ESP8266 mikrokontrollereid, Home Assistant tarkvara ning kodumajapidamiste energiatarbimist. Saadud teadmistega loodi Raspberry Pi ja mikrokontrollerite baasil toimiv nutika kodu süsteem, mille eesmärgiks on kodumajapidamise energiakasutuse optimeerimine. Nutika kodu süsteem loodi mudelina, mitte reaalse kodu paigaldusena. Sellegi poolest imiteerib mudel reaalset nutikat kodu. Nutika kodu süsteem juhib automaatselt küttesüsteemi ning ühte elektriseadme pistikupesa. Küttesüsteemi juhtimisel loodi lahendus, mis säästab energiat sellega, et kodust eemal olles soojendatakse ruume tavalisest vähem. Elektriseadme pistikupesa juhitakse vastavalt elektrihinnale. Tarkvaraliselt saadakse Nord Pool börsihinnast kõige madalam elektrihind, mille ajal lülitab süsteem pistikupesa aktiivseks. Lisaks loodi voolutrafot kasutades lahendus, mis võimaldab reaalajas jälgida kodu elektritarbimist. Lõpetuseks katsetati loodud süsteemi, mille käigus selgus, et loodud süsteem toimis korrektselt ning eesmärkidele vastavalt. Edaspidistes uuringutes võiks pakutud lahenduse energia optimeerimise tõhusust analüüsida.

Märksõnad: nutikas kodu, energiakasutus, mikrokontroller, Raspberry Pi, Home Assistant

www.emu.ee
Eesti Maaülikool
Estonian University of Life Sciences

Estonian University of Life Sciences Kreutzwaldi 1, Tartu 51014		Abstract of Bachelor's Thesis	
Author: Jaan Jalas		Curriculum: Engineering	
Title: Using microcontrollers and Raspberry Pi to optimize the energy usage in smart homes			
Pages: 63	Figures: 23	Tables: 0	Appendixes: 8
Department / Chair: Chair of Energy Application Engineering Field of research and (CERC S) code: 4. Natural Sciences and Engineering, T140 Energy research Supervisor: Janar Kalder, *PhD* Place and date: Tartu 2024			

The usage of electrical energy in residential households has increased within the last decades because of the increase in electrical devices and electrification. To decrease the amount of energy used, innovative solutions are needed. In this bachelor's thesis, research was done on smart homes, Raspberry Pi, ESP8266 microcontrollers, Home Assistant software and energy usage in residential households. With this newly gained knowledge, a smart home system was created by using Raspberry Pi and microcontrollers to optimize energy usage in residential households. The smart home system was created as a model that imitates an actual smart home. The smart home system automatically controls a heating system and a socket for electrical devices. Control of the heating system was created as a solution that saves energy by keeping the rooms cooler than usual when nobody is home, this way less heating is needed. Control of the socket for electrical devices was created so that the smart home software can recognize the cheapest hour from the Nord Pool electricity exchange and activate the socket only during that hour. A current transformer was used to enable real-time monitoring of the energy usage in the smart home. Finally, tests were done to ensure that the system was functioning properly, which it did. In the future research can be done to analyze the energy savings of the proposed system.

Keywords: smart home, energy management, microcontroller, Raspberry Pi, Home Assistant

SISUKORD

SISSEJUHATUS 6
1. NUTIKAD KODUD 7
1.1. Nutikas kodu 7
1.2. Asjade internet IoT 7
1.3. Andurid ja täiturid 8
1.4. Kontrollerid 9
1.5. Keskseadmed 9
1.6. Kommunikatsioon 10
1.7. Nutika kodu kasutajaliides 10
2. RASPBERRY PI, ESP8266 JA HOME ASSISTANT 11
2.1. Raspberry Pi 11
2.2. ESP8266 11
2.3. Home Assistant 13
3. ENERGIAKASUTUS KODUMAJAPIDAMISTES 14
3.1. Energia tarbimine kodumajapidamistes 14
3.2. Elektrienergia hind 16
4. NUTIKA KODU SÜSTEEMI PLANEERIMINE 17
4.1. Nutika kodu energiakasutuse optimeerimise ülesannete määramine 17
4.2. Nutika kodu süsteemi arhitektuuri disainimine 18
4.3. Nutika kodu süsteemi seadmete valimine 19
4.3.1. Raspberry Pi valimine 20
4.3.2. Mikrokontrollerite valimine 20
4.3.3. Releede valimine 22
4.3.4. Vooluanduri valimine 23
4.3.5. Temperatuurianduri valimine 24
5. MUDELI LOOMINE JA KATSETAMINE 25
5.1. Home Assistant operatsioonisüsteemi üles seadmine 25
5.2. ESP8266 arendusplaatide ühendamine Home Assistant-iga 26
5.3. ESP8266 arendusplaatide ühendamine releede ja anduritega 28
5.4. ESP8266 arendusplaatide konfigureerimine ja funktsioonide katsetamine 29
5.5. Mudeli püsipaigalduse tegemine 31

6. AUTOMATISEERIMISED JA JUHTPANEEL 33
6.1. Küttesüsteemi automatiseerimine 33
6.2. Pistikupesa automatiseerimine 34
6.3. Juhtpaneeli vaated 36
6.4. Automaatika katsetamine 38
KOKKUVÕTE 41
KASUTATUD KIRJANDUS 42
LISAD 45
Lisa 1. Küttesüsteemi NodeMCU v3 YAML konfiguratsioon 46
Lisa 2. Pistikupesa NodeMCU v3 YAML konfiguratsioon 48
Lisa 3. Andurite Wemos D1 R1 YAML konfiguratsioon 50
Lisa 4. Mudeli alalisvoolu ühendusskeem koos voolutrafoga 52
Lisa 5. Mudeli vahelduvvoolu ühendusskeem 53
Lisa 6. Küttesüsteemi automatiseerimiste YAML konfiguratsioonid 54
Lisa 7. templates.yaml 58
Lisa 8. Pistikupesa automatiseerimiste YAML konfiguratsioonid 60
LIHTLITSENTS 63

SISSEJUHATUS

Elektrienergia lõpptarbimine Eesti kodumajapidamistes on viimase paarikümne aasta jooksul teinud suure tõusu [1]. Tegu on paratamatu tõusuga, mille on põhjustanud meie ühiskonna areng. Märkimisväärselt on suurenenud elektriseadmete arv kodumajapidamistes ning elektrifitseerimise tulemusel on paljud kodud elektriküttel. Paraku on elektrienergia saamine hetkel veel keskkonda kahjustav tegevus.

Keskkonnamõjude kahandamiseks saab igaüks enda kodus elektrienergia tarbimist vähendada. Kodumajapidamiste elektrienergia lõpptarbimise vähendamiseks on kaks peamist võimalust. Esimeseks võimaluseks on elektriseadmete tööaja vähendamine. Teiseks võimaluseks on seadmete efektiivsemaks muutmine. Seadmete efektiivsemaks muutmine tähendab siinkohal, et kodumajapidamises kasutatakse parema energiamärgisega tooteid. Eriti tähtis on see seadmete puhul mis töötavad pidevalt, näiteks külmikud.

Elektriseadmete tööaega on võimalik juhtida kas käsitsi või automaatselt. Kodumajapidamistes olevate elektriseadmete automaatse juhtimise süsteeme nimetatakse nutika või targa kodu lahendusteks. Nutika kodu lahendused võimaldavad luua süsteemi, millega optimeeritakse kodust energiakasutust.

Antud bakalaureusetöö eesmärgiks on uurida Raspberry Pi ja mikrokontrollerite kasutamise võimalusi nutikates kodudes ning luua nutika kodu lahendus mis, nimetatud seadmed kasutades, aitab kodumajapidamise energiakasutust optimeerida.

1. NUTIKAD KODUD

1.1. Nutikas kodu

Nutikas või tark kodu on kodumajapidamine, mis kasutab omavahel suhtlevaid nutiseadmeid, et võimaldada erinevate kodus olevate seadmete ja süsteemide kaugjuhtimist ning automatiseerimist. Nutikate kodude algusaegades oli tegu pigem mugavust pakkuva lahendusega, nüüdseks pakuvad nutikad kodud veel lisaks võimalusi energiakasutamise optimeerimisel ja turvalisuse tagamisel. Viimaste aastate jooksul on asjade internetil ehk IoT tehnoloogial põhinevad nutikate kodude lahendused saanud kõige populaarsemaks. Nutiseadmete ühendamine läbi interneti võimaldab nutika kodu omanikel enda kodu jälgida ja juhtida ka kodust eemal olles läbi nutitelefoni või arvuti. [2]

Nutikad kodud töötavad rakendades kolme põhilist protsessi. Esiteks kogutakse informatsiooni läbi erinevate andurite. Kogutud informatsioon võib olla näiteks toatemperatuur, õhuniiskus ja seadmete enda staatused. Teiseks salvestatakse ja analüüsitakse kogutud informatsioon nutika kodu keskseadmes. Keskseade suudab analüüsitud informatsiooni põhjal luua juhised vajalikeks tegevusteks. Siia lisanduvad veel juhised, mis kasutaja ise kaugjuhtides annab. Viimaseks juhitakse erinevaid seadmeid keskseadme poolt tulnud juhiste alusel. [2]

1.2. Asjade internet IoT

Omavahel ühendatud süsteem eristatavalt aadressitavatest seadmetest ja masinatest, millel on erinevad töötlemise, tajumise ja täituri võimalused ning mis jagavad võimekust koostoimimiseks ja kommunikatsiooniks läbi interneti, nimetatakse asjade internetiks (*Internet of Things* ehk IoT). Tegu on suhteliselt uue tehnoloogiaga, mis kasvab kiiresti ning esimesena kasutas terminit *Internet of Things* artvutiteadlane Kevin Ashton aastal 1999. [3]

IoT seadmeid on maailmas arvatavasti juba miljardeid ning prognoositakse, et pideva kasvu järel on juba mõne aasta pärast nende hulk kümnetes miljardites [3, 4]. Tegelikult kasutame me

selliseid seadmeid sisuliselt kõik, sest nende hulka kuuluvad näiteks nutitelefoniga ühendatavad nutikellad ja WiFi põhised valvekaamerad. Lisaks võib nutitelefone ning lauaarvuteid samuti lugeda IoT seadmete hulka, kuigi on ka neid, kes selliseid seadmeid IoT seadmeteks ei nimetaks [5].

Kordumatu tunnus (*unique identifier* ehk UID) on IoT seadmete puhul väga tähtis, iga seade omab mingisugust tunnust mille abil on võimalik teda tuvastada isegi suuremas võrgus [3]. Kõige tuttavam näide siinkohal oleks IP-aadress.

1.3. Andurid ja täiturid

Nutikate kodude süsteemis on tähtsad seadmed, mis midagi mõõdavad ehk andurid ning seadmed, mis midagi füüsiliselt liigutavad ehk täiturid. Toome mõningad näited nutikates kodudes enim kasutatavates anduritest: [2]

- Infrapunaandurid – mõõdavad objektidelt kiirgavat infrapunakiirgust ning võimaldavad temperatuuri mõõtmist ja liikumise tuvastamist.
- Temperatuuriandurid – eri tüüpi lahendused, kuid kõik mõõdavad keskkonna temperatuuri. Enamasti töötavad temperatuuriandurid põhimõttel, kus temperatuuri järgi muutub mõõdetav pingelang.
- Niiskuseandurid – mõõdavad õhus olevat niiskuse ja veeauru kogust.
- Gaasiandurid – mõõdavad kindla gaasi sisaldust õhus. Tegu on pigem turvalisuse tagamiseks kasutatavate anduritega. Jälgitakse ohtlike gaaside nagu vingugaasi, propaani, metaani ja süsihappegaasi sisaldust õhus. Edastav signaal võib olla mõõtehulgana või alarmeeriv.
- Suitsuandurid – tuvastavad suitsu enda läheduses. Samuti turvalisuse tagamiseks kasutatavad andurid. Teatud suitsu kogusest alates saadab alarmeeriva signaali.
- Leegiandurid – tuvastab leeki enda läheduses. Samuti turvalisuse tagamiseks kasutatav andur, mis edastab alarmeeriva signaali.
- Ultraheliandurid – mõõdavad objektide kaugust andurist, kasutades ultrahelilaineid.
- Valguse tuvastamise andurid – mõõdavad valgustihedust teatud asukohas.
- Ukseandurid – tuvastavad ukse avamist ning sulgemist.

Täituritest kasutatakse nutikates kodudes kõige rohkem releesid. Relee on elektrooniline seade, mis sisendpinge olemasolul muudab oma kontaktide asendit. Sisuliselt kõikides täitursüsteemides on kasutusel releed, näiteks küttesüsteem käivitatakse releega, digitaalselt juhitavat valgustust lülitatakse sisse ja välja kasutades releesid, ruloode mootorid käivitatakse releega. Releede kõrval teisteks täituriteks ongi enamasti erinevad elektrilised ajamid ning solenoidlukud ja -klapid.

1.4. Kontrollerid

Protsessori põhised kontrollerid on nutikas kodus lahutamata osa andurite ja täiturite kasutamiseks. Kontrollerid koosnevad tavaliselt ühest või mitmest protsessorist, mälust ning sisenditest ja väljunditest ühtsel trükkplaadil. Muidugi on kontrolleri trükkplaadil veel mitmeid väiksemaid elektroonilisi komponente. [2]

Kontrollereid kasutatakse andurite ja täiturite juhtimiseks ja informatsiooni haldamiseks. Näiteks nutikas kodus oleva anduri enda ainuke väljund võib olla lihtsalt analoogsignaal pingena, ühendades anduri kontrolleriga suudab kontroller analoogsignaali muuta ümber digitaalsignaaliks ja tarkvara abiga väljendada anduri mõõtetulemust kraadides. Sellisel kujul on võimalik kontrollerist saata temperatuuri andmed edasi keskseadmesse.

1.5. Keskseadmed

Kontrollerite puudujääk on nende väike jõudlus ning mälumaht, ehk puudub võimekus paljude protsesside üheaegseks juhtimiseks. Seetõttu on nutikas kodus vajalik kasutada keskseadet, mis on nutika kodu süsteemi südameks.

Keskseadmed on miniatuursed arvutid, mis on kontrolleritest palju võimekamad. Nende protsessorite jõudlus ja mälumaht lubavad luua keerukaid süsteeme. Keskseade kogub kontrolleritelt tuleva informatsiooni kokku ning analüüsib seda. Vastavalt vajadusele võib keskseade informatsiooni endasse talletada. Selliselt on võimalik kõiki majas olevaid seadmeid jälgida nii reaalajas kui ka ajaloo järgi.

Keskseadme valimisel tuleb arvestada süsteemi keerukust ja vajadusi, et oleks tagatud piisav võimekus ja vajalikud funktsioonid. Iga keskseade vajab töötamiseks tarkvara, turul olevad keskseadme lahendused tulevad tavaliselt koos kindla tootjapoolse tarkvaraga. Koduomanikul on lihtne paigaldada valmis seadmeid nagu Amazon Alexa, Apple Home, Google Home ja Samsung SmartThings [6]. Iseehitajatel on võimalik kasutada avatud lähtekoodiga Home Assistant tarkvara, et muuta näiteks oma vana arvuti nutika kodu keskseadmeks.

1.6. Kommunikatsioon

Nutika kodu seadmete omavaheliseks ühendamiseks on olemas erinevaid kommunikatsiooni lahendusi ja protokolle. Kommunikatsiooni viise võib liigitada ühenduse järgi kolmeks: juhtmega, juhtmevaba, hübriid. [7]

Juhtmega ühendatud protokollidest on enim kasutust leidvad Ethernet, KNX, X10. Juhtmevabadest protokollidest on populaarseimad WiFi, Bluetooth, Zigbee ja Z-Wave. Lihtsamate lahenduste jaoks lisanduvad juhtmega ühendatud protokollide juurde veel I^2C, SPI ja UART. [7]

1.7. Nutika kodu kasutajaliides

Kasutajaliides võimaldab omanikul enda nutikat kodu konfigureerida, jälgida ning juhtida. Enamasti on kasutajaliidese kuvamiseks vaja ühendada keskseade mõne arvuti, nutitelefoni või tahvelarvutiga. Kasutajaliidesed võivad olla rakenduse või veebilehe põhised. Kasutajaliideses on võimalik luua juhtpaneel, eriti kiireks ligipääsuks nutikas kodus toimuvale. [2]

2. RASPBERRY PI, ESP8266 JA HOME ASSISTANT

2.1. Raspberry Pi

Raspberry Pi on ühel trükkplaadil olev miniarvuti. Raspberry Pi jõudlus on piisav, et seda saab klaviatuuri, hiire ja monitori ühendamisel kasutada ka personaalarvutina. Jõudluse poolest on see siiski nõrgem laua- ja sülearvutitest, kuid selle elektrienergia tarbimine on see-eest kordades väiksem. Sellest tulenevalt kasutatakse Raspberry Pi-d kõige rohkem erinevate projektide jaoks. [8]

Operatsioonisüsteemi Raspberry Pi jaoks on võimalik installeerida microSD-kaardile mõnes teises arvutis, kasutades tootja poolt väljastatud programmi Raspberry Pi Imager. Operatsioonisüsteemiga microSD-kaardi sisestamisel käivitab Raspberry Pi oma käivituse ajal operatsioonisüsteemi. Raspberry Pi jaoks optimeeritud vabavaras saadaval olev operatsioonisüsteem on Linuxi põhjal loodud Raspberry Pi OS, tuntud ka kui Raspbian.

Raspberry Pi versioone on tänaseks juba mitmeid ning nende riistvara on aja jooksul muutunud. Iga versiooni trükkplaat on varustatud üldkasutatavate sisendite ja väljunditega (*general-purpose input/output* ehk GPIO), millest mõned on kasutatavad veel I^2C, SPI ning UART kommunikatsiooniks. Nimetatud protokollide kasutamise jaoks on need esmalt vaja konfiguratsioonist sisse lülitada, sest vaikimisi on need välja lülitatud. Üldkasutatavate sisendite ja väljundite juures on veel 3,3 V ning 5 V ühendused koos maandavate ühendustega. Siinkohal on tähtis mainida, et kõik üldkasutatavad sisendid ja väljundid töötavad 3,3 V binaarsignaalidega, analoogsisendid Raspberry Pi-l puuduvad. Raspberry Pi 4 mudel B näitel on üldkasutatavate sisendite loogilise nulli pinge 0 – 0,8 V ja loogilise ühe pinge 2 – 3,3 V, väljundite loogiline null on maksimaalselt 0,4 V ning loogiline üks minimaalselt 2,9 V [9].

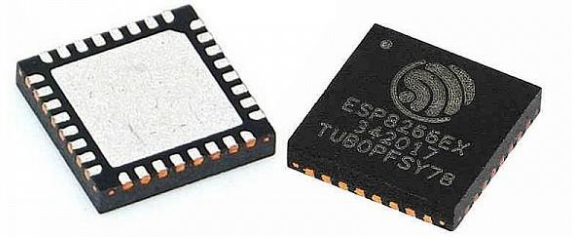
2.2. ESP8266

ESP8266 on madala hinnaga, kuid hea jõudlusega WiFi mikrokontroller/süsteemikiip, toodetud Espressif Systems poolt (joonis 1) [10]. ESP8266 mikrokontrolleri ajuks on 32 bitine, Tensilica L106 protsessor, mis töötab vaikimisi 80 MHz juures. Mikrokontrolleri muutmälu suurus on 160 KB, millest 64 KB on kasutav juhiste jaoks. Välkmälu mikrokontrolleril endal ei ole, kuid

toetab kuni 16 MB suurust välist välkmälu. WiFi protokollidena on toetatud 802.11 b/g/n (HT20), ning selle sageduseks on 2,4 GHz kuni 2,5 GHz. 5 GHz sagedusega WiFi-t ei ole antud mikrokontrolleriga võimalik kasutada. [11]

Mikrokontrolleril on kokku 32 erinevat ühendust, millest 17 on üldkasutatavad sisendid ja väljundid. Tegelikult saab neist kasutada ainult 11, sest 6 neist on kasutatavad välkmälu ühendamiseks. Osad nendest üldkasutatavatest ühendustest on kasutatavad veel UART, SPI, I^2C kommunikatsiooni jaoks. Lisaks eelnevalt mainitud ühendustele on mikrokontrolleril veel 1 analoogsisend, millel on analoog-digitaalmuundur. Kõik üldkasutatavad sisendid ja väljundid toetavad pulsilaiusmodulatsiooni ning töötavad 3,3 V juures, analoogsisendi töövahemik on 0 – 1 V. Ülejäänud ühendused on kasutatavad raadioside antenni ühendamiseks, toiteallikatena, maandustena ja muude signaalide jaoks. [11]

Üldkasutatavate sisendite loogilise nulli pinge väärtused on vahemikus -0,3 – 0,825 V ja loogilise ühe väärtused vahemikus 2,475 – 3,3 V. Väljundite puhul on loogiline null maksimaalselt 0,33 V ning loogiline üks minimaalselt 2,64 V. [11]



Joonis 1. ESP8266 mikrokontroller [12].

Antud mikrokontrollerit kasutatakse, mitmete erinevate tootjate poolt, et luua mooduleid ja arendusplaate. Enamus ESP8266 põhjal loodud arendusplaadid sarnanevad rohkem tuntud Arduino arendusplaatidega. Arvutusjõudluse poolest on Arduino arendusplaadid ESP8266 arendusplaatidest nõrgemad ja nende tavamudelitel puudub sisseehitatud WiFi ühendus. Seetõttu on paljudes projektides eelistatud just ESP8266 arendusplaadid

ESP8266 arendusplaate on lihtne programmeerida kasutades Arduino IDE tarkvara, ehk neid on võimalik programmeerida läbi Arduino programmeerimise tarkvara, nii nagu tehakse seda Arduino arendusplaatidega. WiFi ühenduse korral on võimalik arvutist saata kirjutatud püsivara failid ESP8266 arendusplaatidele juhtmevabalt.

2.3. Home Assistant

Home Assistant on vabavaraline nutika kodu keskseadme tarkvara. Antud tarkvara abil on võimalik kõik oma nutika kodu seadmed kuvada koondatult ühele juhtpaneelile. Home Assistant toetab väga suurt hulka erinevaid seadmeid ja süsteeme ning selle põhiline eelis konkurentide ees on suur kasutajate kogukond, kes loovad ise integratsioone juurde. Hetke seisuga toetab Home Assistant 2778 erinevat integratsiooni ning selles numbris ei ole kogukonna integratsioonid isegi kajastatud [13].

Integratsioonid on tarkvaralised lahendused, mis lubavad Home Assistant-il ühenduda teiste tarkvarade ja platvormidega. Lisaks integratsioonidele on võimalik installeerida lisamooduleid, mis on Home Assistant-i sisesed rakendused, lisafunktsioonaalsuste tekitamiseks [14].

Lihtsustatud automatiseerimise redaktor võimaldab luua lihtsamaid funktsioone ilma koodi kirjutamise vajaduseta, kuid keerulisemaid automatiseerimisi on võimalik kirjutada koodis. Selliselt on Home Assistant algajatele väga kasutajasõbralik, aga samas võimekas tarkvara.

Installeerimiseks on Home Assistant olemas neljas erinevas variandis: Home Assistant OS ehk operatsioonisüsteem, Home Assistant Container, Home Assistant Core ja Home Assistant Supervised. Home Assistant OS on nendest neljast variandist algajate jaoks kõige lihtsam nii kasutuse kui ka installatsiooni poolest, sisaldades seehulgas kõige rohkem funktsioone. [15]

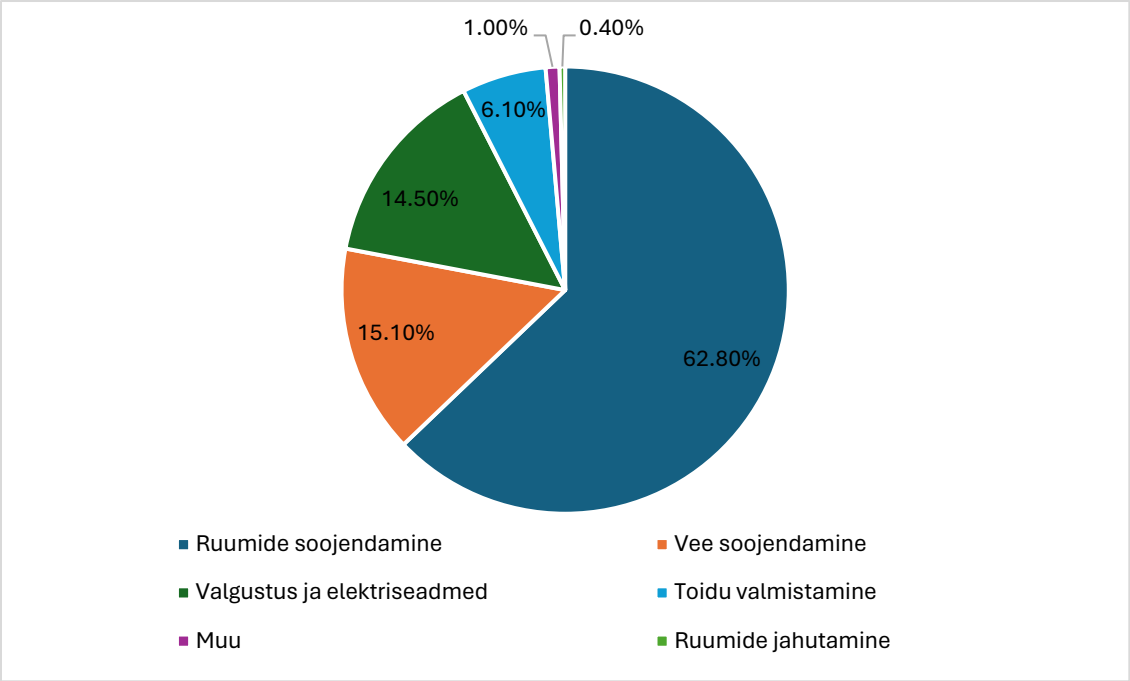
3. ENERGIAKASUTUS KODUMAJAPIDAMISTES

3.1. Energia tarbimine kodumajapidamistes

Eesti Statistikaameti statistika andmebaasi järgi on aastal 2022 Eestis elektrienergia tegelik lõpptarbimine olnud 25710 TJ ning soojusenergia tegelik lõpptarbimine 20934 TJ [1]. Siinkohal on lõpptarbimine kodumajapidamistest olnud elektrienergiana 7081 TJ ja soojusenergiana 14000 TJ [1]. Mainitud energiahulgast arusaamiseks võime näitena kasutada Tšornobõli tuumaelektrijaama kunagist tootlust, kus elektrijaamas töötas neli RBMK-1000 reaktorit, mille väljundvõimsused elektrienergiana olid kokku 4000 MW [16]. Eesti kodumajapidamiste aastase soojus- ja elektrienergia lõpptarbimise hulga tootmiseks elektrienergiana peaks selline elektrijaam töötama, võrgukadusid arvestamata, järjest ligikaudu 61 päeva.

Tähtis on mainida, et energia lõpptarbimine kodumajapidamistes oli aastal 2022 kokku lausa 40357 TJ, kui eelmistele energialiikidele lisada juurde veel kütustena primaarsed tahked biokütused (17102 TJ), maagaas (2041 TJ), gaasiõli ja diisliõli ilma biokütuse osata (42 TJ) ja vedelgaasid propaan ning butaan (91 TJ) [1]. Asendades kõik see energia puhtalt elektrienergiaga peaks eelneva näite elektrijaam töötama järjest ligikaudu 117 päeva võrgukadusid arvestamata.

Energiakasutuse optimeerimiseks peame esmalt aru saama, kuidas jaotub kodumajapidamiste energiatarbimine liigiti. Esitame 2020 aastal kogutud Euroopa Liidu majapidamiste energiatarbimised liigiti graafikul (joonis 2) [17].



Joonis 2. Energiatarbimine Euroopa Liidu kodumajapidamistes liigiti aastal 2020 [17].

Graafikult on näha, et suurim osa kodumajapidamistes tarbitud energiast kulub ruumide soojendamiseks. Seejärel juba umbes 4 korda vähem energiat kulub vee soojendamiseks ning umbes sarnane hulk energiat kulub veel valgustuseks ja elektriseadmete kasutamiseks. Toidu valmistamisele kulub juba väiksem osa, ainult 6,1%. Ruumide jahutamine ja muu tarbib kokku vaid 1,4%. Seega on kõige tähtsam koht kodumajapidamiste energiakasutuse optimeerimisel just ruumide kütmise protsessi juhtimisel, millele järgneb boilerite ja elektriseadmete juhtimine.

Elektriautode populaarsuse kasvuga võib tulevikus tõusta kodumajapidamiste elektrienergia tarbimine elektriautode kodulaadijatega laadimise tõttu. Aasta 2024 1. mai seisuga on Transpordiameti andmetel liiklusregistris vaid 6592 elektrisõidukit, seega hetkel on nende aastane elektrienergia lõpptarbimine vaid mõnekümne TJ juures [18]. Suur hulk sellest laadimiseks kuluvast elektrienergiast ei tarbita isegi kodumajapidamistes, sest elektriautosid laetakse lisaks kodulaadijatele veel teenusepakkujate laadimisvõrkudes.

3.2. Elektrienergia hind

Elektrienergia tarbimisel on tarbija jaoks tähtis elektrihind. Tihti ei ole energiakasutuse optimeerimise eesmärk pelgalt tarbimise vähendamine, vaid raha säästmine. Eestis on võimalik tarbijal sõlmida elektrimüüjaga igakuiselt fikseeritud hinnaga või muutuva börshinnaga leping. Muutuva börsihinnaga lepingu puhul muutub elektrienergia hind iga tund vastavalt Nord Pool avatud elektrituru hinnale, kuhu kuuluvad peale Eesti ka Leedu, Läti, Soome, Rootsi, Norra ja Taani. Piirkonniti võivad riikide elektrihinnad erineda, kuid elektribörsil kujuneb hind nõudluse ja pakkumise tulemusena. Lisaks sõltub börsihind toodetavast energiast, hind on soodne, kui nõudlus on kaetud taastuva- ja tuumaenergiaga. Hind on kallim, kui turule pääsevad muud fossiilkütustel töötavad elektrijaamad, sest kütuse ja CO_2 emissioonitasud tõstavad hinda. [19]

Elektriarvele lisanduvad võrgutasud, taastuvenergia tasu ja elektriaktsiis. Võrgutasu võib sõltuda peakaitsme suurusest, pingest, kellaajast ning kuutasust. Aastal 2024 on taastuvenergia tasu 1,28 senti kWh kohta ja elektriaktsiis 0,18 senti kWh kohta, mõlemal juhul on tasu esitatud koos käibemaksuga [20]. [21]

Tarbijal on seega võimalik enda elektriarvet tarbimise ajastamisega kõige rohkem mõjutada juhul, kui sõlmitud on börsipakett sellise võrgupaketiga, kus öö ja päeva elektri edastamise hinnad on erinevad. Näiteks soojendades vett ainult madalama hinnaga elektrienergia ajal öösel, kui edastamise hind on soodsam on tarbitud energia sama, kuid kordades odavam võrreldes päevase tarbimise tipuajaga. Tänapäevased veeboilerid on piisavalt hea isolatsiooniga, et soe vesi püsib soojana mitmeid tunde või lausa päevi peale viimast soojendamist ja seetõttu ei ole vaja veeboilerit pidevalt töös hoida [22].

4. NUTIKA KODU SÜSTEEMI PLANEERIMINE

4.1. Nutika kodu energiakasutuse optimeerimise ülesannete määramine

Raspberry Pi ja mikrokontrollerite kasutamiseks nutika kodu energiakasutuse optimeerimisel peame esmalt pakkuma välja funktsioonid, mida nimetatud seadmetega saavutada soovitakse. Nutika kodu seadmeid võib juba varasemaid olla, samuti võib nutika kodu nullist disainida. Antud juhul lähtume pigem viimasest, kuid üritame tekitada võimaluse vajadusel varasemate seadmete ühendamiseks. Süsteem luuakse lõputöö raames mudelina, mitte reaalsele hoonele installatsioonina, kuid mudel peab imiteerima reaalset nutikat kodu.

Energiakasutuse optimeerimise esimeseks lahenduseks juhime suurimat energiakulu allikat, ehk kodus olevat küttesüsteemi. Küttesüsteemi juhtimisel üritame energiakasutust optimeerida luues lahenduse, kus saame automatiseerida kütmise vajadust. Automatiseerimiseks on vaja tagasisidet keskkonnalt temperatuurianduri kujul, mille mõõdetud õhutemperatuuri saame kasutada küttesüsteemi käivitamiseks või peatamiseks. Nutika kodu kasutaja saab määrata temperatuurid, mis peavad olema juhul, kui keegi viibib kodus ja temperatuurid, kui kodus kedagi ei viibi. Sellisel juhul ei soojendata kodust eemal olles kodu otstarbetult soojaks ning hoitakse energiat kokku.

Teiseks üritame energiakasutust optimeerida juhtides mingit elektriseadet. Selleks on meil vaja juhtida pistikupesa, kuhu elektriseade ühendatakse. Erinevalt küttesüsteemi juhtimisest, kus optimeeritakse energia tarbimise kogust sõltumata elektrihinnast, üritame pistikupesa juhtimist korraldada arvestades majanduslikku energiakasutuse optimeerimist ehk juhtimist lähtuvalt elektrienergia hinnast. Lõputöö raames me otseselt ühtegi elektriseadet pistikupessa ei määra, reaalses nutikas kodus võib sinna ühendada näiteks veeboileri või mingisuguse seadme laadija.

Kolmandaks üritame energiakasutust optimeerida kasutajale energiatarbimise teadvustamisega. Kasutades energiatarbimist mõõtvaid andureid saab kasutaja reaalajas teavet kodu tarbimisest. Aja möödudes tekib kasutajal arusaam milline on elektrienergia tarbimine normaaltingimustes ning põhjendamatult suure tarbimise korral tekib võimalus kontrollida, kas mingid elektriseadmed on jäänud tahtmatult tööle.

4.2. Nutika kodu süsteemi arhitektuuri disainimine

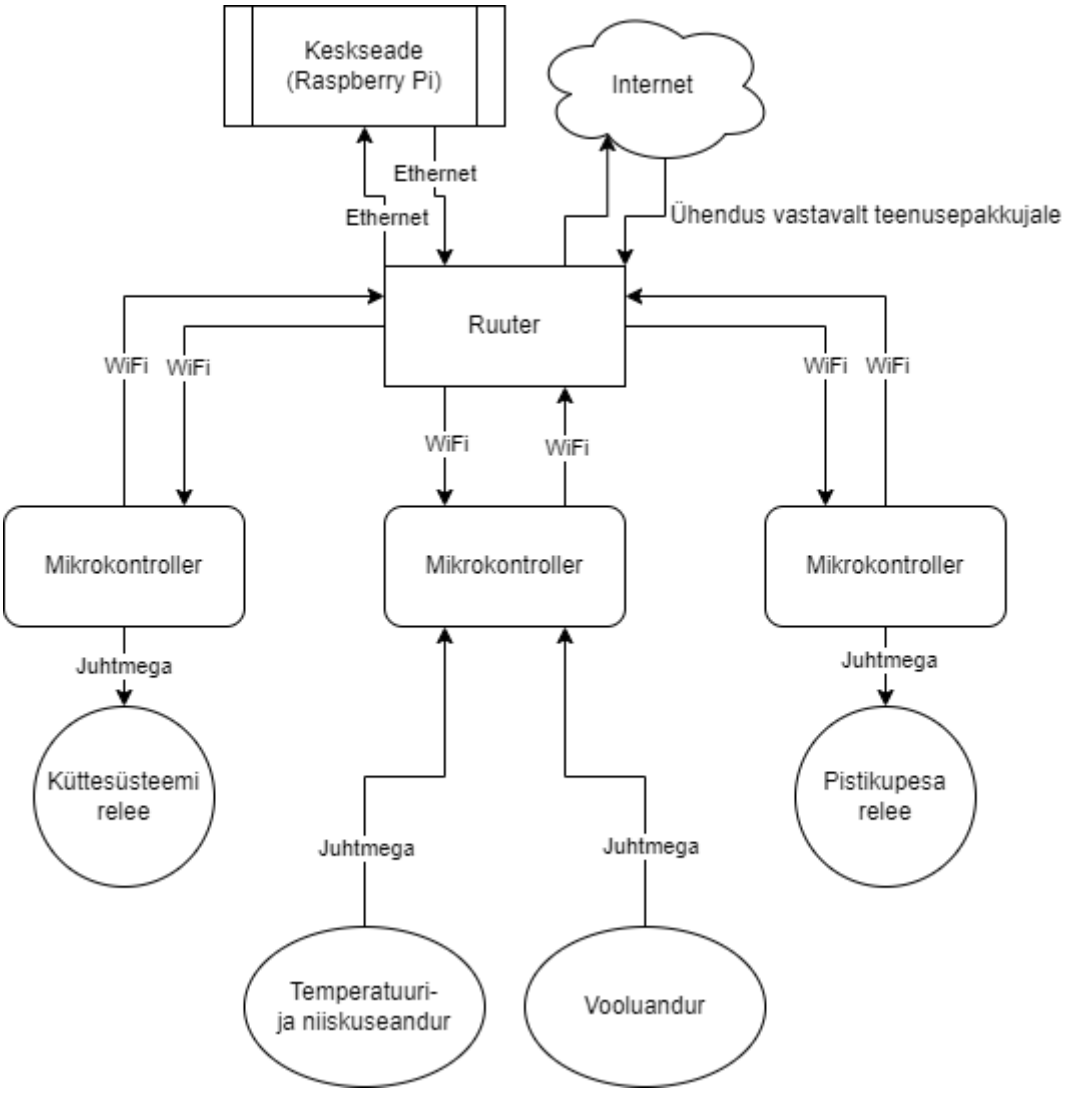
Lähtuvalt eelmises peatükis kirjeldatud ülesannetele koostame nutika kodu mudeli süsteemi diagrammi, mis kirjeldab süsteemi terviklikku toimimist (joonis 3). Esmalt kirjeldame kõiki süsteemi osasid.

Keskseadme ja mikrokontrollerite vaheline kommunikatsioon toimib juhtmevabalt üle WiFi ja saab toimida tegelikult kohtvõrguna. Nutika kodu süsteemi ühendamisel internetiga tekib võimalus mitmete uute integratsioonide jaoks ehk vähemalt keskseade on vaja ühendada internetiga. Seega kasutame kommunikatsiooni tagamiseks ruuterit.

Süsteemi kõrgeima taseme seadmena võtame kasutusele Raspberry Pi, ehk sellest saab meie süsteemi keskseade. Keskseadme tarkvarana kasutame varasemalt mainitud Home Assistant vabavaralist operatsioonisüsteemi. Madalamate seadmetega kommunikatsiooniks ühendatakse keskseade ruuteriga Ethernet kaabliga.

Lülitusprotsesside täituriteks on vastavate mikrokontrolleritega ühilduvad releed, mis võimaldavad lülitada ümber 230 V 50 Hz vahelduvvoolu. Mikrokontrolleriteks, mis vastutavad lülitusprotsesside täiturite kontrollimise eest, valime WiFi ühendust võimaldavad mikrokontrollerid.

Mõõtmisteks kasutame õhutemperatuuri jaoks temperatuuriandurit, mis võimaldab lisaks veel mõõta suhtelist õhuniiskust. Elektrienergia tarbimist mõõdame vooluanduriga. Andurid ühendatakse ühe mikrokontrolleriga, seega neid tuleb valida vastavalt ühilduvusele. Mikrokontroller peab võimaldama WiFi ühendust.



Joonis 3. Nutika kodu süsteemi arhitektuuri diagramm.

4.3. Nutika kodu süsteemi seadmete valimine

Vastavalt määratud ülesannetele ja koostatud süsteemi arhitektuurile, pidi valima nutika kodu toimimiseks seadmed. Seadmete valimisel tuli arvestada vajalike ühenduste olemasoluga. Ruuterina kasutati autori kodus olevat ruuterit.

4.3.1. Raspberry Pi valimine

Keskseadmena kasutatav Raspberry Pi valiti vastavalt hinnale ja jõudlusele. 2023 aasta oktoobris tuli müügile kõige uuem Raspberry Pi mudel Raspberry Pi 5 ning kuna varasem mudel Rasberry Pi 4 oli samas hinnaklassis, valiti keskseadmeks Raspberry Pi 5 4 GB muutmäluga versioon (joonis 4).

Raspberry Pi 5 pakendilt loetud tehnilised andmed:

- 64 bitine, neljatuumaline Cortex-A76 protsessor
- 4 GB LPDDR4X muutmälu
- 2 micro HDMI pesa, 4Kp60 toega
- 2 USB 3.0 pesa
- 2 USB 2.0 pesa
- Gigabit Ethernet pesa
- 802.11b/g/n/ac WiFi ühendus
- Bluetooth 5.0 ühendus
- PoE võimeline (lisaseadmega)
- PCIe ühendus (lisaseadmega)
- 5 V 5 A või 5 V 3 A USB-C toiteallikas vastavalt jõudluse soovile (lisaseadmena)

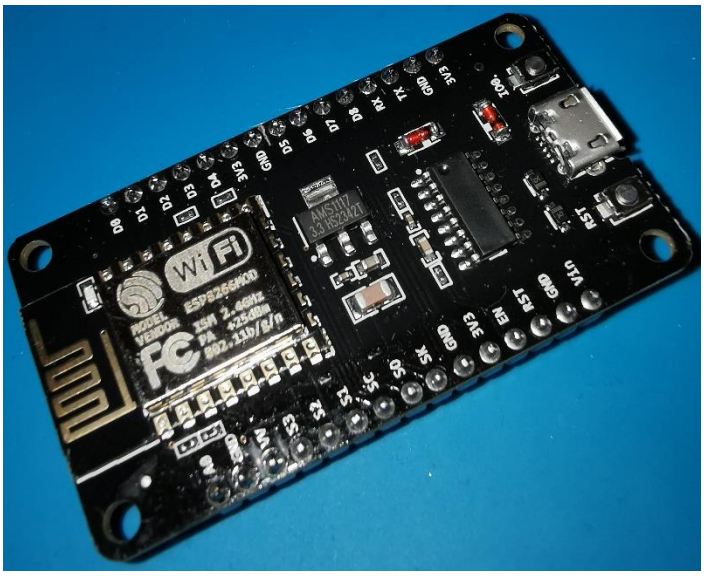
Joonis 4. Raspberry Pi 5 4 GB versioon koos korpusega.

4.3.2. Mikrokontrollerite valimine

Mikrokontrollerite valikul otsustati ESP8266 kasuks. Releede kontrollimiseks kasutatakse kahte ESP8266 NodeMCU v3 CH340 arendusplaati (joonis 5).

NodeMCU v3 CH340 tehnilised andmed: [23]

- ESP-12E moodul
- 4 MB välkmälu
- 64 KB muutmälu
- Micro USB pesa
- 11 binaarsisendit ja väljundit
- 1 analoogsisend
- I^2C, SPI, UART ühendused
- Analoog-digitaalmuunduri tööpiirkond 0 – 3,3 V
- Tööpinge 3,3 V
- Toitepinge 4,5 – 10 V

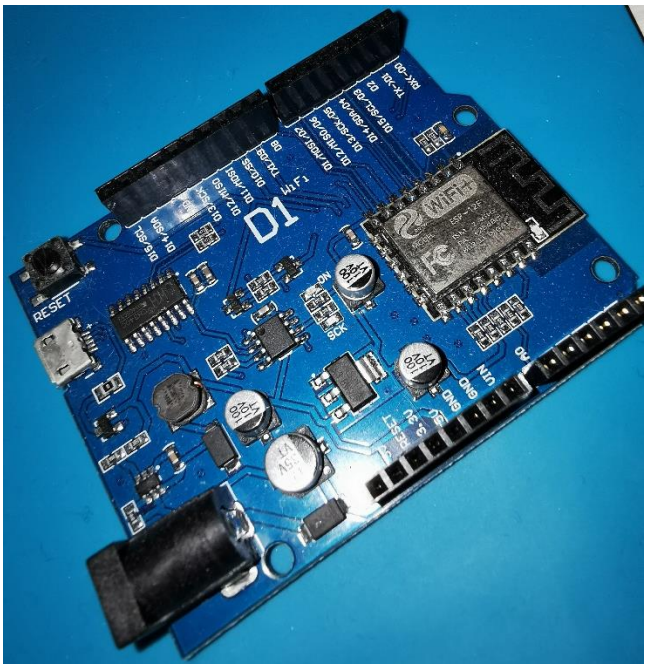


Joonis 5. ESP8266 NodeMCU v3 CH340 arendusplaat.

Andurite lugemiseks kasutatakse ühte ESP8266 Wemos D1 R1 arendusplaati (joonis 6).

Wemos D1 R1 tehnilised andmed: [24]

- ESP-12F moodul
- 4 MB välkmälu
- 80 KB muutmälu
- Micro USB pesa
- 11 binaarsisendit ja väljundit
- 1 analoogsisend
- I^2C, SPI, UART ühendused
- Tööpinge 3,3 V
- Toitepinge 9 – 24 V



Joonis 6. ESP8266 Wemos D1 R1 arendusplaat.

Kõik kolm valitud arendusplaati on tegelikult väga sarnased, kuid Wemos D1 R1 arendusplaat on tehtud Arduino UNO R3 järgi.

4.3.3. Releede valimine

Releede valimisel pidi arvestama, et NodeMCU v3 binaarühenduste loogiline 1 on väärtusega 3,3 V ning loogiline 0 väärtusega 0 V. Valitud releeks osutus ühe kanaliga Bestep JQC3F-03VDC-C releemoodul (joonis 7). Releemoodulil on normaalselt suletud ja avatud kontaktid, mis võimaldavad primaarpoolel lülitada maksimaalselt 10 A 250 V vahelduvvoolu. Sekundaarpoolel on vaja 3,3 V alalisvoolu toitepinget, maandust ja 3,3 V sisendsignaali. Lisaks on releemoodulil roheline LED, mis tähistab toite olemasolu ning punane LED, mis tähistab sisendsignaali olemasolu ja lülituse aktiivset olekut.

Joonis 7. Bestep JQC3F-03VDC-C releemoodul.

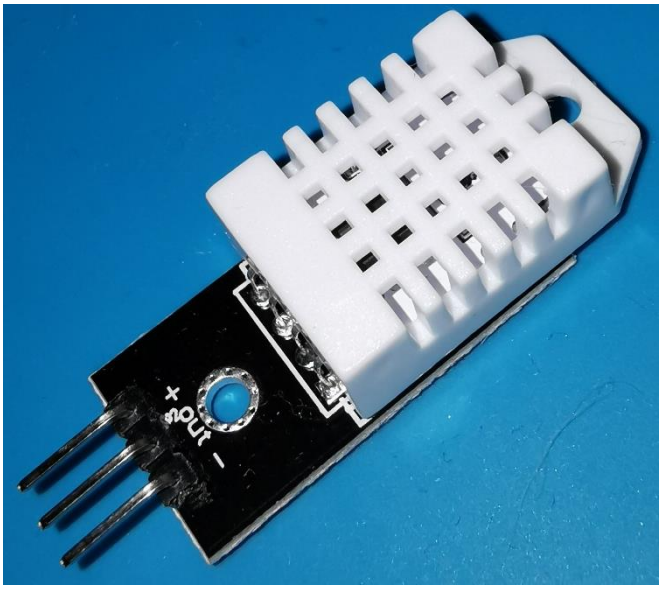
4.3.4. Vooluanduri valimine

Wemos D1 R1 analoogsisendi maksimaalne pinge on 3,3 V. Seega oli meil vaja vooluandurit, mille väljundiks oleks maksimaalselt 3,3 V. Vooluanduriks valiti YHDC SCT013-30 voolutrafo, mille saab ümber faasijuhtme kinnitada (joonis 8). Voolutrafo nominaalvool on 30 A, mille juures on väljundiks 1 V. Pinge jaoks on voolutrafol sisseehitatud takisti, mida läbib voolutrafo vool ning väljundpingeks ongi takisti pingelang.

Joonis 8. YHDC SCT013-30 voolutrafo.

4.3.5. Temperatuurianduri valimine

Temperatuuriandurina kasutati juba olemasolevat DHT22 temperatuuriandurit koos mooduliga (joonis 9). Antud andur võimaldab mõõta õhutemperatuuri koos suhtelise õhuniiskusega. Anduril on kolm ühendust, 3,3 – 5,5 V alalisvoolu toitepinge sisend, maanduse ühendus ja digitaalse väljundsignaali ühendus. Väljundsignaali stabiliseerimiseks on moodulil sisseehitatud takisti. Temperatuuri mõõtemääramatus on ±0.5 °C ning õhuniiskuse ±1%. [25]



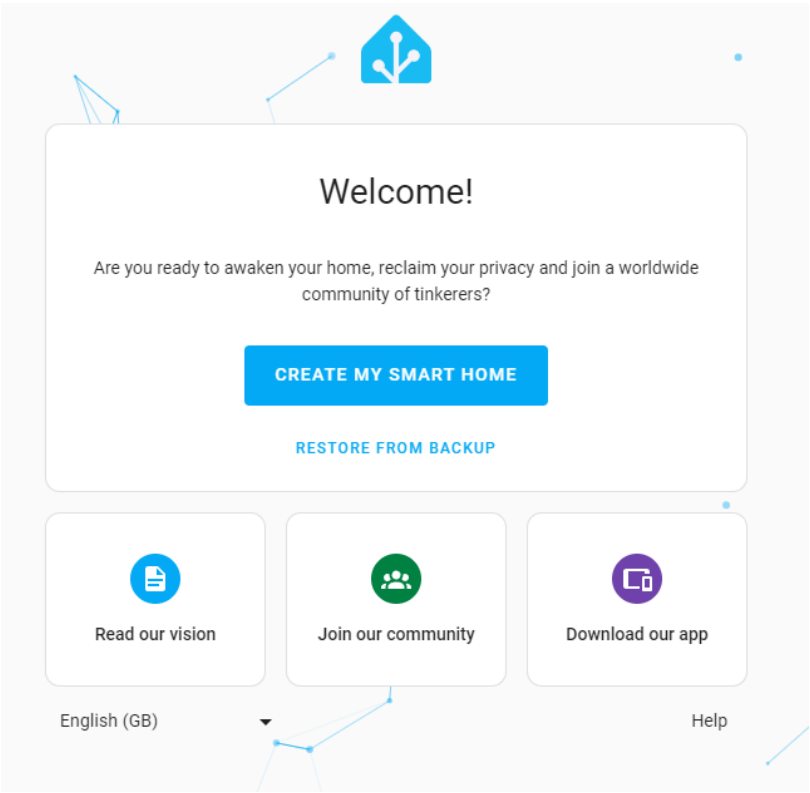
Joonis 9. DHT22 temperatuuri- ja niiskuseandur koos mooduliga.

5. MUDELI LOOMINE JA KATSETAMINE

5.1. Home Assistant operatsioonisüsteemi üles seadmine

Nutika kodu mudeli loomise esimese sammuna pidi Raspberry Pi peale paigaldama Home Assistant operatsioonisüsteemi. Operatsioonisüsteem installeeriti 64 GB suurusele microSD-kaardile kasutades Raspberry Pi Imager tarkvara. Seejärel sisestati microSD-kaart Raspberry Pi sisse ja miniarvuti ühendati Ethernet kaablit kasutades ruuteriga. Toiteploki ühendamise järel käivitus Raspberry Pi ning juba mõne minuti pärast oli võimalik, läbi teise arvuti brauseri, pääseda ligi Home Assistant kasutajaliidesele, sisestades brauseri aadressiribale *http://homeassistant.local:8123/* [26].

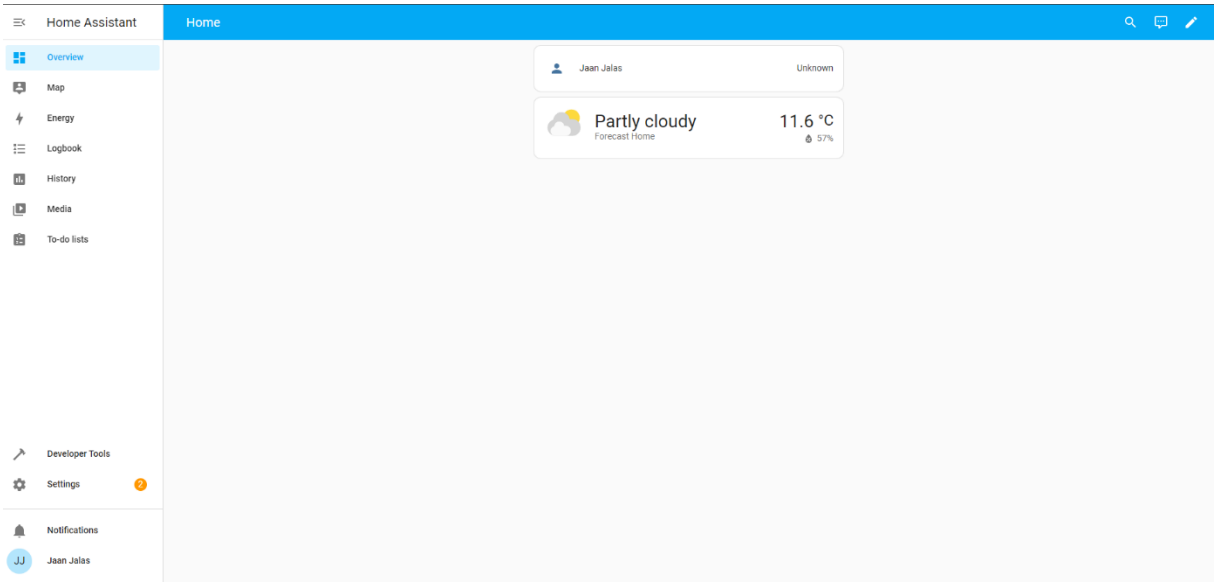
Esimene kasutajaliidesel kuvatud ekraan andis võimaluse alustada tarkvaras nutika kodu loomist nullist või taastada varukoopia abil juba olemasolev nutika kodu (joonis 10). Hetkel oli vaja alustada nutika kodu loomist nullist, seega valiti esimene variant.



Joonis 10. Home Assistant kasutajaliidese esimene ekraan.

Järgnevalt tuli luua kasutaja, sisestades enda nime, kasutajanime ja salasõna. Seejärel pidi valima nime nutika kodu installatsioonile, märkima kaardil kodu asukoha, valima riigi, keele, ajatsooni, kõrguse merepinnast, ühikusüsteemi ja valuuta.

Home Assistant tuvastas ja kuvas kohtvõrgus olevad seadmed ja teenused, mis on integratsioonidena süsteemi poolt toetatud [27]. Tuvastatud integratsioone oli võimalik koheselt üles seada, kuid seda saab teha ka hiljem. Antud juhul ühtegi integratsiooni ei lisatutud ja üles seadmise lõpetamisel kuvati ekraanile juhtpaneel, kus oli näha ainult kasutajat ja ilmateadet (joonis 11).



Joonis 11. Esialgne juhtpaneel.

5.2. ESP8266 arendusplaatide ühendamine Home Assistant-iga

Home Assistant-i funktsionaalsust on võimalik suurendada lisamoodulite installeerimisega [28]. Navigeerides sätetest lisamoodulite juurde kuvatakse kõik installeeritud lisamoodulid ning alumises paremas nurgas on viide lisamoodulite poele. Lisamoodulite poes on kuvatud nii ametlikud lisamoodulid kui ka lisamoodulid, mis on kogukonna arendatud. Vajutades soovitud lisamoodulile on võimalik installeerida see lisamoodul Home Assistant-ile. Ennem lisamooduli käivitust on võimalik valida, kas lisamoodul hakkab tulevikus käivituma automaatselt koos

süsteemi käivitusega või kui lisamoodul jookseb kokku ning, kas lisamoodul uuendab end automaatselt uue versiooni olemasolul.

Üheks selliseks ametlikuks lisamooduliks on ESPHome, mis võimaldab ESP8266 ja ESP32 mikrokontrollereid programmeerida ja juhtida otse läbi Home Assistant-i. Programmeerimiseks ei ole vaja kirjutada C++ programme, nagu seda peab tegema Arduino IDE tarkvaraga, vaid piisab YAML konfiguratsiooni failide muutmisest. YAML konfiguratsiooni järgi loob ESPHome C++ koodi, mis seejärel kompileeritakse masinkeelde ja saadetakse mikrokontrollerile.

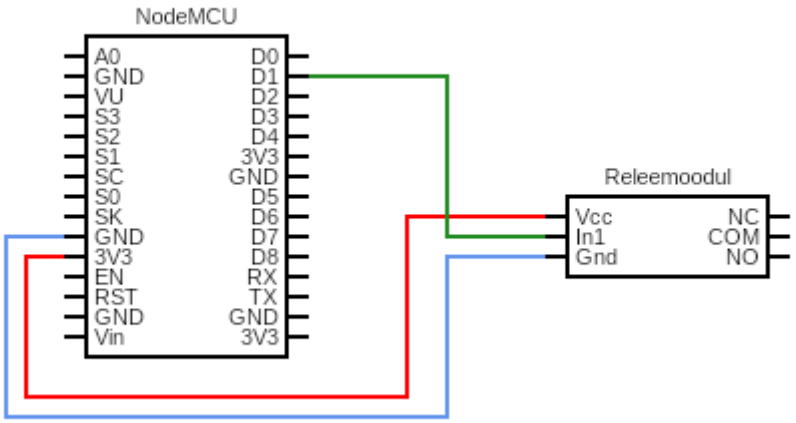
Esmalt installeeriti ja käivitati ESPHome lisamooduli. Seejärel avati lisamooduli rakenduse ja vajutati seadme lisamise viitele, mis viis meid ESP seadmete ühendamise kuvale. ESP8266 seadme üles seadmiseks ühendati arendusplaat arvutiga kasutades micro USB juhet, valiti õige kommunikatsiooni port ning määrati WiFi parameetrid. Üles seadmise lõppedes eemaldati micro USB juhe, sest nüüd oli võimalik ESPHome rakenduses kuvatud ESP8266 seadme IP aadressiga integreerida seade Home Assistant-iga. Sama protsessi korrati veel kahe teise ESP8266 seadme jaoks ja nüüd olid meie kaks NodeMCU v3 ja üks Wemos D1 R1 arendusplaadid Home Assistant-iga integreeritud.

Ühendus ESP seadmete ning Home Assistant-i vahel toimub antud lahenduse puhul kasutades ESPHome lisamooduli spetsiaalselt rakendusliidest ehk API-t. Vaikimisi seatakse seadmele ülesse ka lihtne veebiserver, mille abil pääseb seadmele ligi Home Assistant-it kasutamata (joonis 12) [29, 30].

Joonis 12. ESPHome poolt loodud veebiserver NodeMCU v3 arendusplaadil.

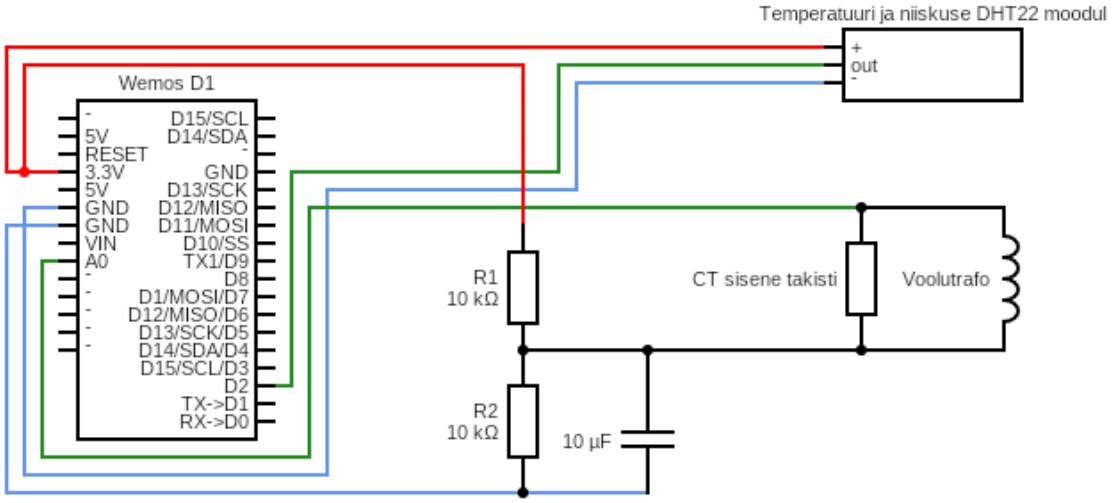
5.3. ESP8266 arendusplaatide ühendamine releede ja anduritega

Kahe NodeMCU v3 arendusplaadi ning Bestep JQC3F-03VDC-C releemooduli ühendused tehti tüüplahendusena. Selliselt on võimalik YAML konfiguratsioonis mõlemad ESP8266 muudatused teha samamoodi. Tüüplahenduse ühendusskeem on toodud joonisel 13.



Joonis 13. NodeMCU v3 arendusplaadi ja Bestep JQC3F-03VDC-C releemooduli ühendusskeem.

Wemos D1 R1 ühendus temperatuuri ja niiskuse DHT22 mooduliga oli lihtne, kuid voolutrafo ühendusel pidi arvestama asjaoluga, et voolutrafost väljuv pinge omab hetkel vahelduvvoolust tulenevalt ka negatiivset piirkonda. Teadaolevalt on arendusplaadil pingejagur, mis võimaldab analoogsisendisse sisestada voolu vahemikus 0 – 3,3 V (ESP8266 mikrokontrolleril on vahemik 0 – 1 V) [19]. Seega tuli muuta voolutrafost analoogsisendisse minev pinge, mis oli hetkel tippudega -1,414 V ja 1,414 V, selliseks et seda oleks võimeline mikrokontroller lugema. Selleks tuli tõsta siinuslaine keskpunkti, mis oli hetkel 0 V, kasutades Wemos D1 R1 arendusplaadil olevat 3,3 V toiteallikat. Ühendades voolutrafo üks väljund pingejaguriga, mis asub arendusplaadi 3,3 V toiteallika ning maanduse vahel ning mille takisti väärtused on mõlemad 10000 Ω, nihutasime siinuslaine keskpunkti 1,65 V peale. Pinge analoogsisendisse muutub nüüd maksimaalselt vahemikus 0,236 – 3,064 V ning RMS väärtustena vahemikus 0,65 – 2,65 V, ehk on loetav arendusplaadi analoogsisendi poolt. Maanduse ja voolutrafo väljundi vahel luuakse ennem pingejagurit lisaühendus läbi 10 µF suuruse kondensaatori, et kompenseerida voolutrafo põhjustatud äkilisi lainemuudatusi ja pinge tõuse. Voolutrafo ning temperatuuri ja niiskuse mooduli ühendusskeem arendusplaadiga on toodud joonisel 14. [31]



Joonis 14. Andurite ja Wemos D1 R1 arendusplaadi ühendusskeem.

Esialgu ühendati voolutrafo jaoks vajalikud komponendid voolutrafo ja arendusplaadiga kasutades maketeerimislauda.

5.4. ESP8266 arendusplaatide konfigureerimine ja funktsioonide katsetamine

Peale ühenduste tegemist tuli ESP8266 seadmete YAML failides teha vajalikud lisad, et ühendatud andurid ja releed toimiksid korrektselt. Nagu eelnevalt mainitud ühendati NodeMCU v3 arendusplaadid releedega tüüplahendusena, seega konfiguratsioonis tehtud muudatused on muidu identsed, ainult releedele määratud nimed erinevad. Releede juhtimiseks defineeritakse kõigepealt kontrollplokk, antud juhul *switch*, seejärel määratakse juhtimiseks kasutatav platvorm, antud juhul *gpio*, pärast mida tuleb määrata juhtimiseks kasutatav ühendus, antud juhul D1 ehk üldkasutatav ühendus number 5 [23]. Lisaks määratakse kas lüliti on vaikimisi väljas, selleks peab koodi lisama inverteerimise ja selle aktiveerima. Lõpetuseks antakse loodud lülitile nimi, ühe puhul küttesüsteem ja teisel pistikupesa, salvestatakse konfiguratsioon ning laetakse juhtmevabalt arendusplaadile. NodeMCU v3 konfiguratsioonid on leitavad Lisades 1 ja 2. [32]

Uue konfiguratsiooni eduka üleslaadimise järel lisati releede juhtimiseks lülitid Home Assistant juhtpaneelile. Juhtpaneelil lülitati releesid sisse ja välja, et katsetada korrektset toimimist. Kontaktide asendite tuvastamiseks mõõdeti multimeetriga kontaktide vahelisi takistusi. Releed toimisid korrektselt, ehk ilma toiteta olid releede normaalasendis avatud kontaktid avatud, toite olemasolul olid samuti avatud ning lülitussignaali saatmisel lülitusid vastavalt suletuks või avatuks.

Wemos D1 R1 arendusplaadi konfigureerimisel loodi kontrollplokk *sensor* ja kaks erinevat platvormi, *dht* ning *ct_clamp*. Esimene platvorm on temperatuuri- ja niiskuseanduri jaoks, kus defineeriti ühendus seekord arendusplaadi füüsiliste ühenduste järjekorras, ehk ühendus D2 on numbriga 16. Seejärel määrati anduri mudel, antud juhul DHT22 ning uuenduste saatmise intervall, antud juhul 5 sekundit. Lõpetuseks määrati anduri temperatuuri ja niiskuse komponendid, vastavalt *temperatuure* ning *humidity*. Mõlema komponendi juurde tuli lisada ka soovitud nimed, antud juhul DHT22 Temperatuur ja DHT22 Niiskus. [33, 34]

Teine platvorm on voolutrafo jaoks, ning esmalt defineeriti voolu mõõtmise sisend andurina. Selleks loodi kontrollplokile juurde veel uus platvorm *adc*, ehk analoog-digitaalmuundur, kuhu märgiti arendusplaadi ainuke A0 analoogsisend ja tunnus mille järgi saab konfiguratsioonis loodud andurile viidata, antud juhul *adc_sensor*. Peale lisa anduri loomist liiguti tagasi voolutrafo konfiguratsiooni juurde, kus määrati voolutrafo anduri nimeks Mõõdetud Vool. Taaskord lisati uuenduste saatmise intervalliks 5 sekundit. Wemos D1 R1 arendusplaadi uus konfiguratsiooni salvestati ning saadeti seadmele. [34]

Arendusplaadilt saadetud temperatuuri ja niiskuse andmed olid koheselt korrektsed, voolu mõõtmised mitte. Vooluanduri kalibreerimiseks tuli YAML konfiguratsioonis defineerida filter, mis kalibreerib andurit lineaarselt. Selleks defineeriti, milline oli mõõtetulemus vooluta olukorras, antud juhul 0 A juures 0 A ning milline oli mõõtetulemus teadaoleva tarbimisega elektriseadme lisamisel. Antud juhul kasutati veekeetjat, mille voolu tarbimine on 10 A, mõõtetulemus oli siinkohal 0,1 A. Uue konfiguratsiooniga oli mõõdetud voolu täpsus antud lõputöö raames rahuldav. Wemos D1 R1 konfiguratsioon on leitav Lisas 3.

5.5. Mudeli püsipaigalduse tegemine

Seadmed töötasid korrektselt, ning kõik funktsionaalsed muudatused sõltusid nüüd ainult Home Assistant-is loodud tarkvaralistest lahendustest. Seega võis süsteemi kohtkindlalt paigaldada mudeli alusele. Ennem paigalduse alustamist tuli voolutrafo maketeerimislaual olev ahel joota püsivalt universaalsele trükkplaadile.

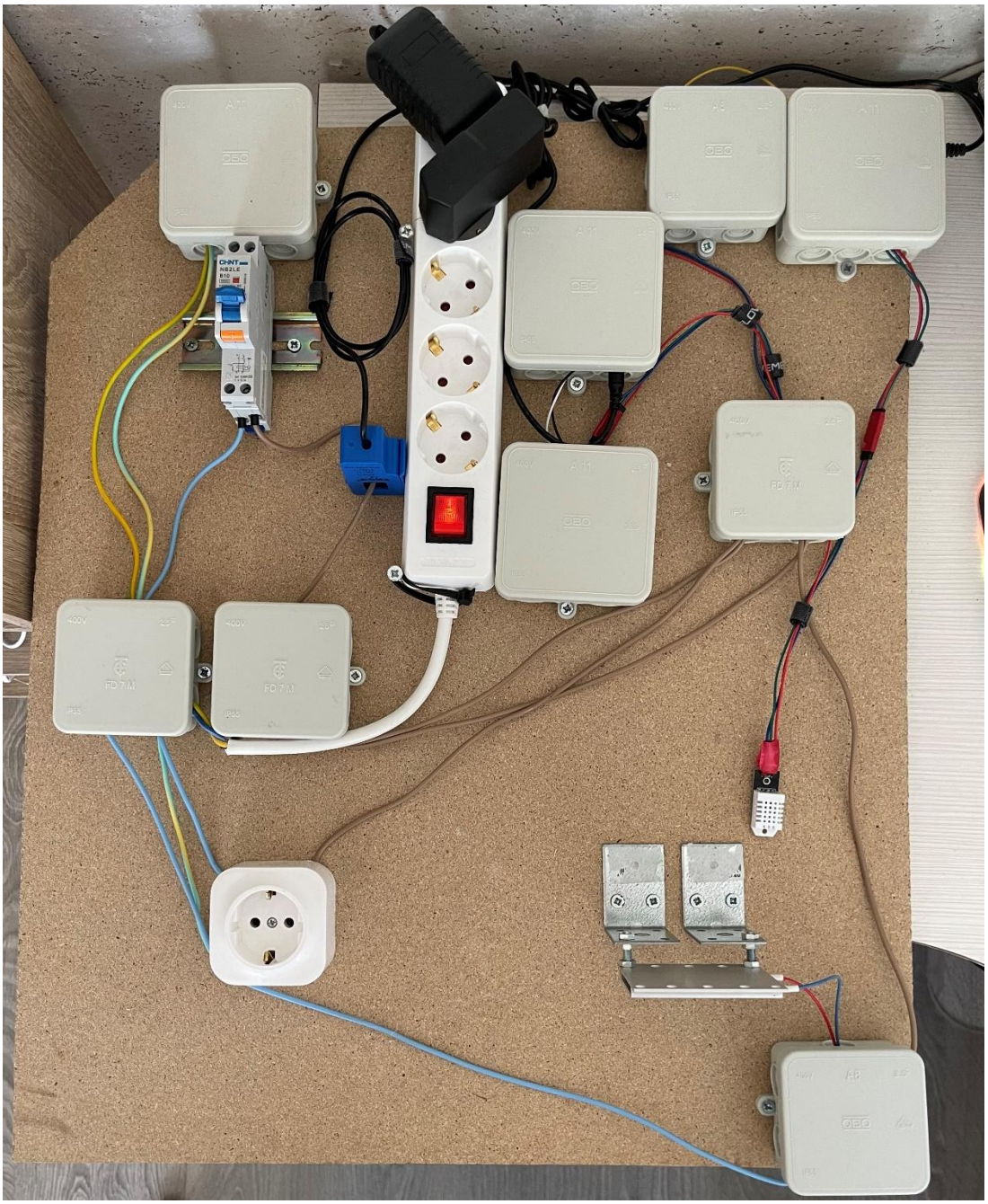
Aluseks kasutati vineerplaati, kuhu kinnitati harukarbid, mille sisse paigaldati releed, arendusplaadid, juhtmeühendused, voolutrafo ahel. Lisaks kinnitati alusele DIN liist kaitselüliti paigaldamiseks, pistikupesa, kütteelemendi kinnitusalus, pikendusjuhe.

Mudeli vahelduvvooluahel algab pistikust, mida on võimalik sisestada pistikupessa. Pistikust liigub mudeli esimesse harukarpi kolmesooneline kaabel, kust väljuvad kaablisooned eraldatuna. Kaitsemaanduse juhtmed ühendatakse kokku selles harukarbis, kaitsemaandatud on pikendusjuhtme kõik pesad ning pistikupesa. Peale harukarpi asub DIN liistul ohutuse tagamiseks CHINT NB2LE kaitselüliti, mis pakub ülekoormuse, lühise ning rikkevoolu kaitset. Kaitselülitisse ühendati neutraal ja faas. Tegu on ühefaasilise B tüüpi kaitselülitiga, mille nimivool on 10 A ning nimirikkevool 30 mA. Kaitselülitist väljuvad faas ja neutraal eraldi harukarpidesse, kus toimub nende jagunemine mudeli peale laiali. Voolutrafo kinnitati ümber faasi juhet ennem harukarpi.

Neutraali harukarbist viiakse ühendused pikendusjuhtmele, pistikupessa ning kütteelemendi harukarpi, kus ühendatakse see kütteelemendi neutraaliga. Faasi harukarbist viiakse ühendused pikendusjuhtmele ning mõlema relee COM ühendusse. Kütteelemendina kasutatakse PTC termistori põhist küttekeha, mis kinnitati küttelemendi kinnitusalusele. Kütteelementi juhtiva relee normaalasendis avatud kontaktist väljub faasijuhe küttelemendi harukarpi, kus ühendatakse see kütteelemendi faasiga. Pistikupesa juhtiva relee normaalasendis avatud kontaktist ühendatakse faas pistikupesa faasi ühendusega.

Pikendusjuhtmesse ühendatakse NodeMCU v3 arendusplaatide toiteks üks 5 V 2 A toiteallikas ning üks 9 V 2 A toiteallikas Wemos D1 R1 toiteks. Wemos D1 R1 ühendatakse toiteallikaga kasutades arendusplaadil olevat toitepesa, NodeMCU v3 arendusplaatide jaoks jagatakse toiteallika kaablist positiivne ja negatiivne juht laiali, vastavalt mõlema arendusplaadi Vin ja GND ühenduste sooritamiseks.

Mudeli alalisvoolu ja vahelduvvoolu ühendusskeemid on kuvatud Lisades 4 ja 5, mudel on kuvatud joonisel 15. Ühendusskeemidelt ja jooniselt on välja jäetud Raspberry Pi koos ruuteriga, sest neid ei paigaldatud mudeli alusele.



Joonis 15. Koostatud nutika kodu mudeli püsipaigaldus.

6. AUTOMATISEERIMISED JA JUHTPANEEL

6.1. Küttesüsteemi automatiseerimine

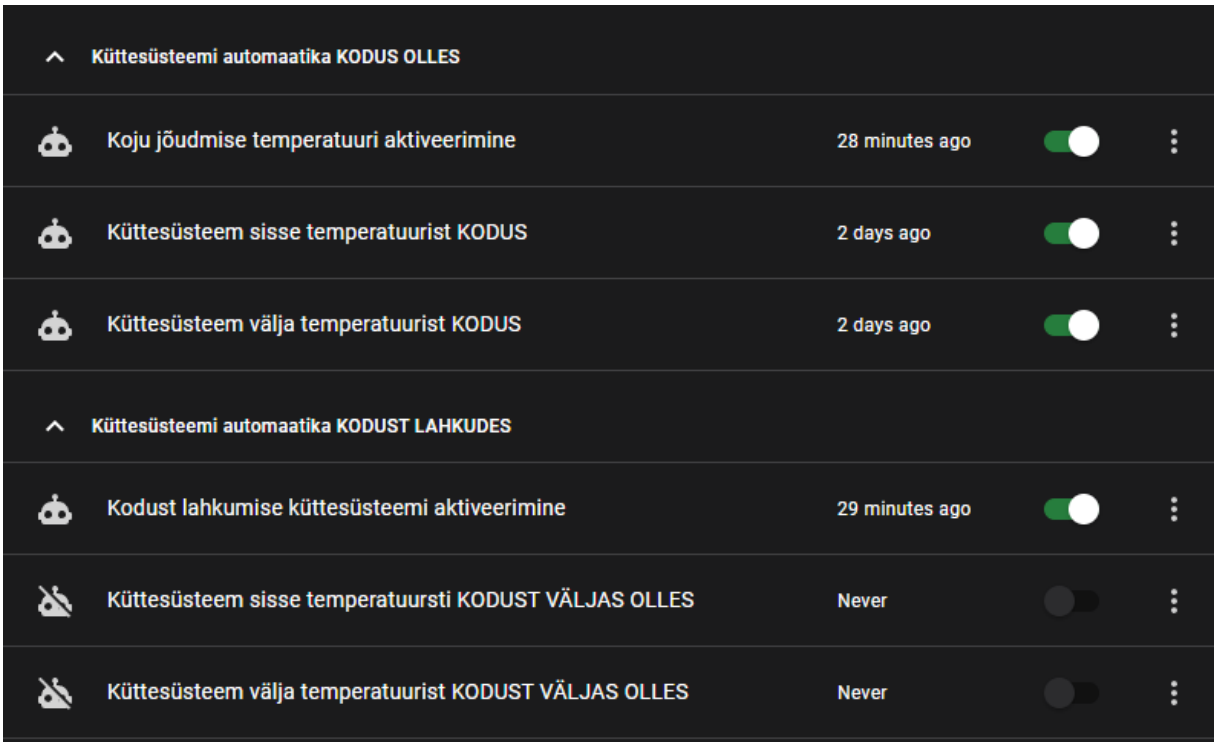
Vastavalt süsteemi planeerimises esitatud küttesüsteemi automatiseerimise eesmärgile, koostati Home Assitant-is vajalikud lahendused. Automatiseerimise jaoks tuleb kindlasti aru saada seadmete ja üksuste vahelisest erinevust, kuigi tegelikult oleme neid kasutanud juba varem. Home Assistant-i üksused on andurid, täiturid või funktsioonid, mida kasutatakse andmete jälgimiseks. Loogiline grupeering ühest või mitmest üksusest on tuntud kui seade. Kasutatud DHT22 anduri näitel on andur ise seade, mis sisaldab kahte üksust, niiskus ja temperatuur. Temperatuur ja niiskus andmetena on siinkohal tuntud kui üksuste olekud, ehk 23,7 on Home Assistant-i mõttes üksuse temperatuur olek. [14]

Küttesüsteemi automatiseerimised koostati täies mahus Home Assistant automatiseerimise redaktoris, kuid esmalt loodi lisa üksused, millega on võimalik määrata soovitud temperatuurid, juhul kui viibitakse kodus ja juhul kui ollakse kodust eemal. Kasutaja asukoha määramiseks installeeriti nutitelefonile Home Assistant rakendus ning tehti ühendus läbi virtuaalse privaatvõrgu. Home Assistant-i rakendus lubab kuvada nutika kodu kasutajaliidest telefonis ning lisaks on võimalik telefonist andureid luua. Antud juhul loodi uus asukohaandur, mis kasutab nutitelefoni asukohta.

Nutika kodu ümber asub 100 meetri raadiusega ring, mille sees olevat ala loetakse koduks. Asukohaanduriga ühendatud nutitelefoni sisenemisel antud ringi, loetakse kasutajat kodus olevaks ja väljumisel vastupidi. Antud olekut kasutades loodi automatiseerimised, mis hoiavad temperatuuri vastavalt kasutaja sisestatud väärtustele ja nutitelefoni asukohale. Küttesüsteemi automaatika, kodus viibimise ajal, käivitub kui temperatuurianduri näit on 5 sekundit olnud allapoole määratud temperatuuri. Seejärel kontrollitakse, et relee oleks avatud asendis ning suletakse kontaktid. Automaatika küttesüsteemi väljalülitamiseks toimub vastupidiselt.

Küttesüsteemi automaatika, kodus mitte viibimise korral, käitub sarnaselt kodus viibimise ajal olevale automaatikale, erinevuseks on vaid teise määratud temperatuuri kasutamine. Kokku on kontrollivaid automatiseerimisi seega neli, kuid automatiseerimiste enda juhtimiseks loodi veel kaks automatiseerimist. Esimene automatiseerimine lülitab kodus oleku tuvastamisel välja kodus mitte viibimise automaatika ning sisse kodus viibimise automaatika. Teine

automatiseerimine lülitab kodust lahkumise tuvastamisel välja kodus olemise automaatika ja sisse kodus mitte olemise automaatika. Küttesüsteemi automatiseerimiste vaade Home Assistant automatiseerimise redaktoris on kuvatud joonisel 15.



Joonis 16. Küttesüsteemi automatiseerimised Home Assistant automatiseerimise redaktoris.

Kõikide loodud küttesüsteemi automatiseerimiste YAML konfiguratsioon on kuvatud Lisas 6.

6.2. Pistikupesa automatiseerimine

Börsihinna jälgimiseks on võimalik kasutada Nord Pool integratsiooni, kuid kuna tegu on mitteametliku integratsiooniga, peab tegema mõned lisasammud [35]. Järgnevalt kirjeldame mitteametlike integratsioonide võimaldamist läbi HACS ehk Home Assistant Community Store integratsiooni.

Kasutaja profiili alt peab esmalt sisse lülitama täiustatud režiimi, seejärel tuleb installeerida Terminal & SSH lisamoodul ning sisestada installeeritud terminalis järgnev koodirida: *wget -q -O - https://install.hacs.xyz | bash -*. Sisestatud koodirida laeb alla HACS integratsiooni ja

laadimise lõppedes tuleb sooritada Home Assistant-i taaskäivitus. Peale taaskäivitust on integratsioonide loetelust võimalik üles seada HACS integratsioon. Läbi HACS integratsiooni saab Home Assistant-i lisada Nord Pool integratsiooni [36].

Nord Pool integratsioon tekitab juurde ühe üksusega seadme, mille olekusse kuvatakse hetkene Nord Pool börsihind. Oleku omaduste YAML konfiguratsioon sisaldab lisaks veel kõiki tänaseid ja homseid elektrihindasid koos kellaegadega. Kasutades Nord Pool anduri oleku omadusi saame luua YAML malli, et luua kaks uut andurit odavaima hinna ja aja kuvamiseks. Malle kasutatakse Home Assistant-is olemasolevate andmete haldamiseks.

Börsihinnale soovime lisada võrguteenuse öised ja päevased tasud. Selleks loodi üksused, kuhu on kasutajal võimalik sisestada soovitud väärtused. Töö käigus kasutasime elektrilevi Võrk 2 paketi hindasid, kus öine hind oli 6,16 senti kWh kohta ning päevane 10,59 senti kWh kohta [21]. Lisaks loodi lisa üksused viimase odavaima aja salvestamiseks ning odavaima aja salvestamiseks.

Esmalt laeti alla Studio Code Server lisamoodul, et pääseda ligi Home Assistant-i konfiguratsioonifailidele. Seejärel loodi uus YAML fail nimega templates ning Home Assistant-i konfiguratsiooni YAML faili lisati loodud failile viide, et seda käsitletaks kui mallide konfiguratsioonifaili: *template: !include templates.yaml*.

Loodud mallis defineeriti esmalt uue andurina kõige odavama hinna andur, siis määrati üksusele unikaalne tunnus. Selleks genereeriti anduri identifitseerimiseks universaalselt unikaalne identifikaator, ehk juhuslikult genereeritud 128 bitine number. Määrati veel üksuse ikoon ja mõõteühik.

Üksuse oleku defineerimisel loodi loogika odavaima hinna leidmiseks, arvestades sealhulgas kasutaja sisestatud parameetreid. Loogikas defineeriti esmalt muutujad, millest ooteaeg (24 tundi), hetkene aeg ja järgmine lubatud aeg on tsüklite välised ning odavaim hind ja tunni loendur tsüklite sisesed. Esimene tsükkel defineerib tänast ja homset. Teine tsükkel loodi nii, et Nord Pool anduri omadustest minnakse järjest tsükli kaupa läbi tänase ja homse päeva andmed. Andmete hulgast kasutatakse hinna alguse aega ning hinda. Alguse aega võrreldakse hetkese ajaga ja viimase kõige odavama ajaga. Alguse aeg peab hetkesest ajast olema vähemalt tund aega ees, et käivitada tunniloendur. Viimasest kõige odavamast ajast peab olema möödunud 6 tundi, et alguse aja järgi lisatakse hinnale juurde võrguteenuse tasud ning võrreldakse võrgutasuga börsihinda tsüklis eelnevalt määratud odavaima hinnaga. Juhul kui uus hind on eelnevast odavam, kirjutatakse vana hind üle. Tsükkel kestab seni, kuni tunni loendur

on suurem või võrdne määratud ooteajaga. Tsüklist väljas ümardatakse odavaim hind teise kümnendkohani.

Mallis defineeriti seejärel kõige odavama aja andur, mille loomisel võeti aluseks esimene andur. Muudatused tehti unikaalses tunnuses ning loogikas. Loogikas oli vaja nüüd lisada juurde veel tsükli sisene muutuja odavaima aja jaoks, mis pannakse võrduma odavaima hinna korral oleva algusajaga. Tsüklist väljas esitatakse seekord odavaim aeg tundides ja minutites. Kogu templates YAML faili konfiguratsioon on kuvatud Lisas 7.

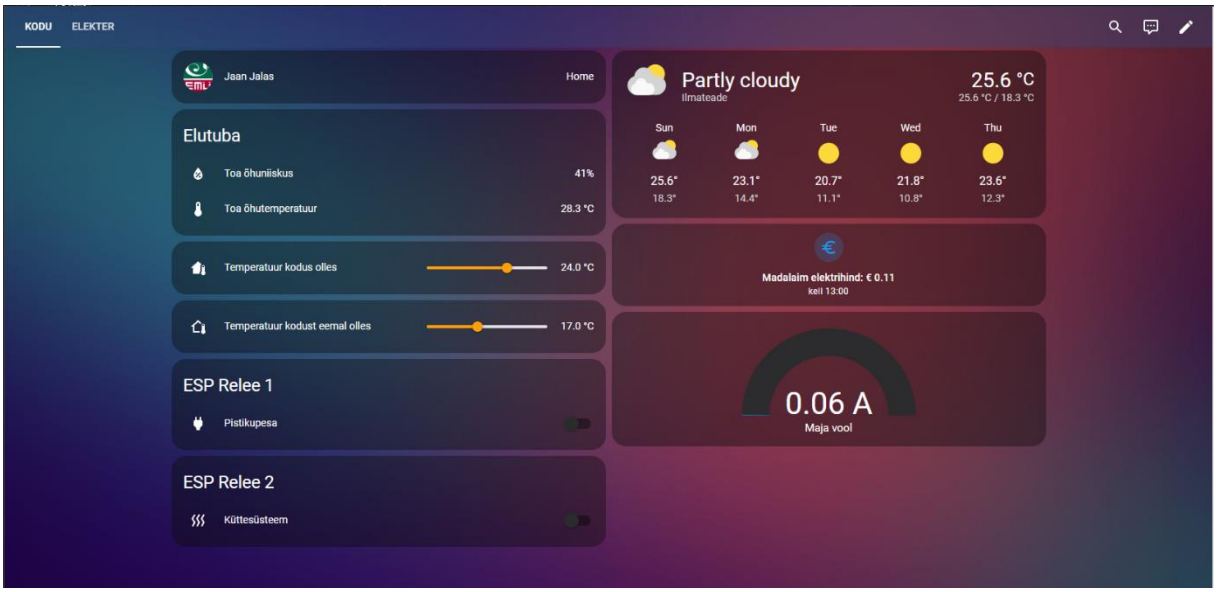
Lisaks mallile loodi pistikupesa juhtimiseks veel, Home Assistant-i automatiseerimise redaktoris, viis automatiseerimist. Esimene automatiseerimine salvestab kell 00:00 päeva kõige odavama aja koos hinnaga. Teine automatiseerimine võrdleb salvestatud aja tundi süsteemiajaga. Tundide võrdumise korral oodatakse viis sekundit ning käivitatakse automatiseerimine, mis vaatab et pistikupesa juhtiv relee oleks avatud ning seejärel lülitab kontaktid ümber. Kolmas automatiseerimine lülitab relee kontaktid avatuks samal põhimõttel, siis kui tunnid ei võrdu.

Neljas automatiseerimine on loodud automaatika väljalülitamiseks manuaalse sisselülituse korral. Selleks on loodud juhtpaneeli uus lüliti, mille lülitamine aktiveerib automatiseerimise mis deaktiveerib Nord Pool odavaima aja järgi toimivad automatiseerimised ning lülitab pistikupesa relee kontaktid suletuks. Lüliti väljalülitamisel aktiveeritakse taaskord automaatika ning avatakse pistikupesa relee kontaktid. Viies automatiseerimine on loodud viimase kõige odavama aja uuendamise jaoks, aeg kirjutatakse üle kui pistikupesa on tund aega aktiveeritud.

Kõigi viie loodud automatiseerimise YAML konfiguratsioonid on kuvatud Lisas 8.

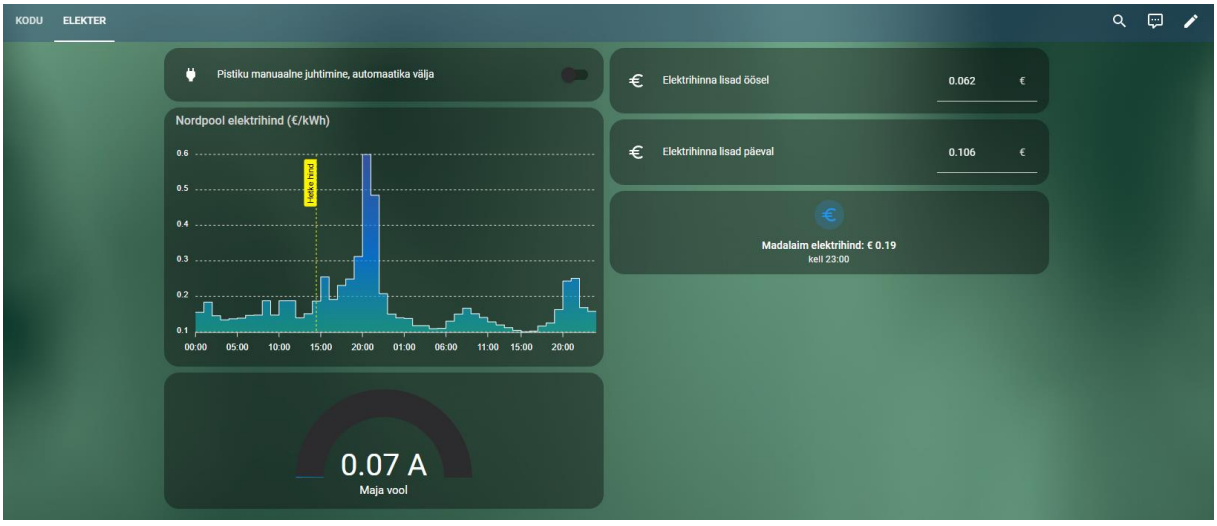
6.3. Juhtpaneeli vaated

Home Assitanti kasutajaliideses loodi kaks juhtpaneeli. Esimene ja vaikimisi kuvatud juhtpaneel KODU sisaldab endas: kasutajat koos asukohaga, õhuniiskuse ja õhutemperatuuri näitusid, kahte liugurit temperatuuride määramiseks, releede oleku lüliteid, ilmateadet, madalaima elektrihinna ja kellaaja näitu, voolutrafo poolt mõõdetud voolu mõõdikut (joonis 17).



Joonis 17. Juhtpaneel nimega KODU.

Teine juhtpaneel nimega ELEKTER sisaldab endas: pistikupesa kolmanda automatiseerimise lülitit, Nord Pool elektri börsihinna graafikut, voolutrafo poolt mõõdetud voolu mõõdikut, võrguteenuse hinna määramise väljasid nii öö kui päeva jaoks, madalaima elektrihinna ja kellaaja näitu (joonis 18).

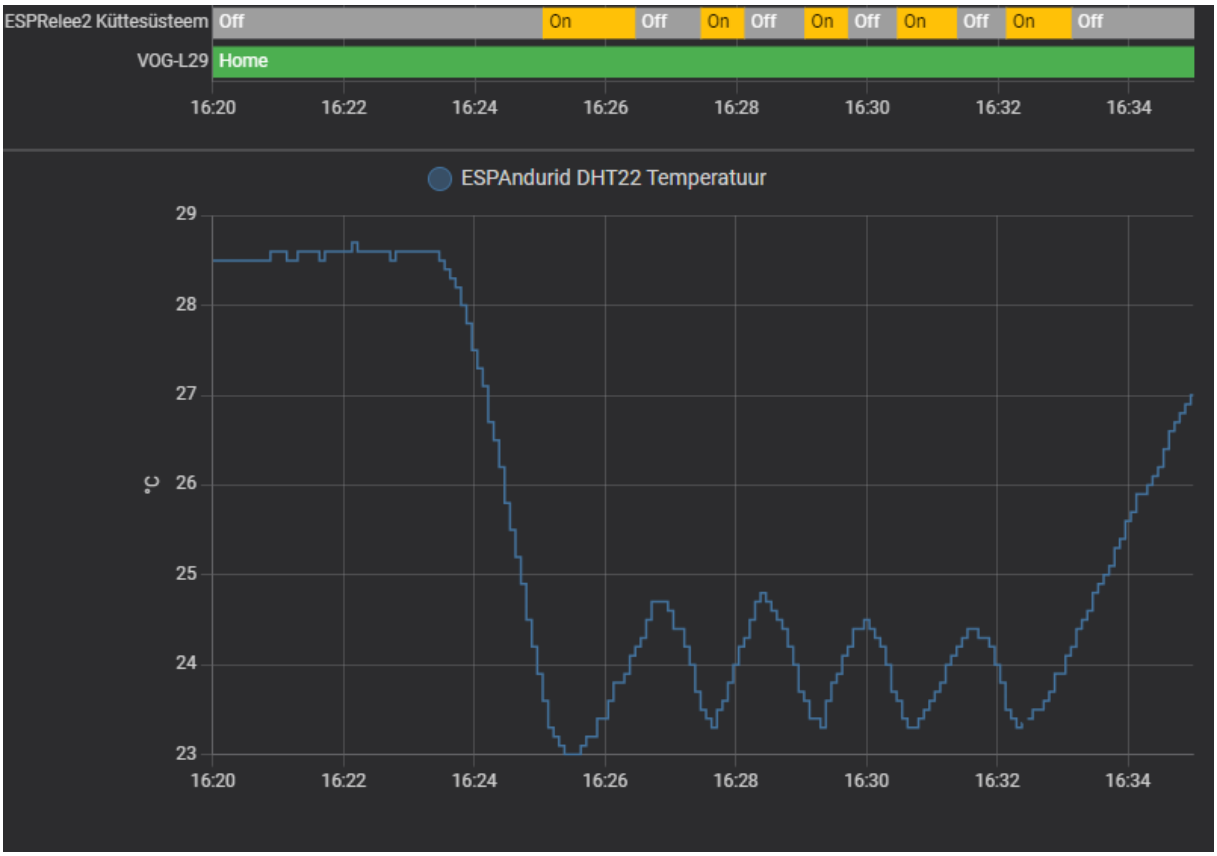


Joonis 18. Juhtpaneel nimega ELEKTER.

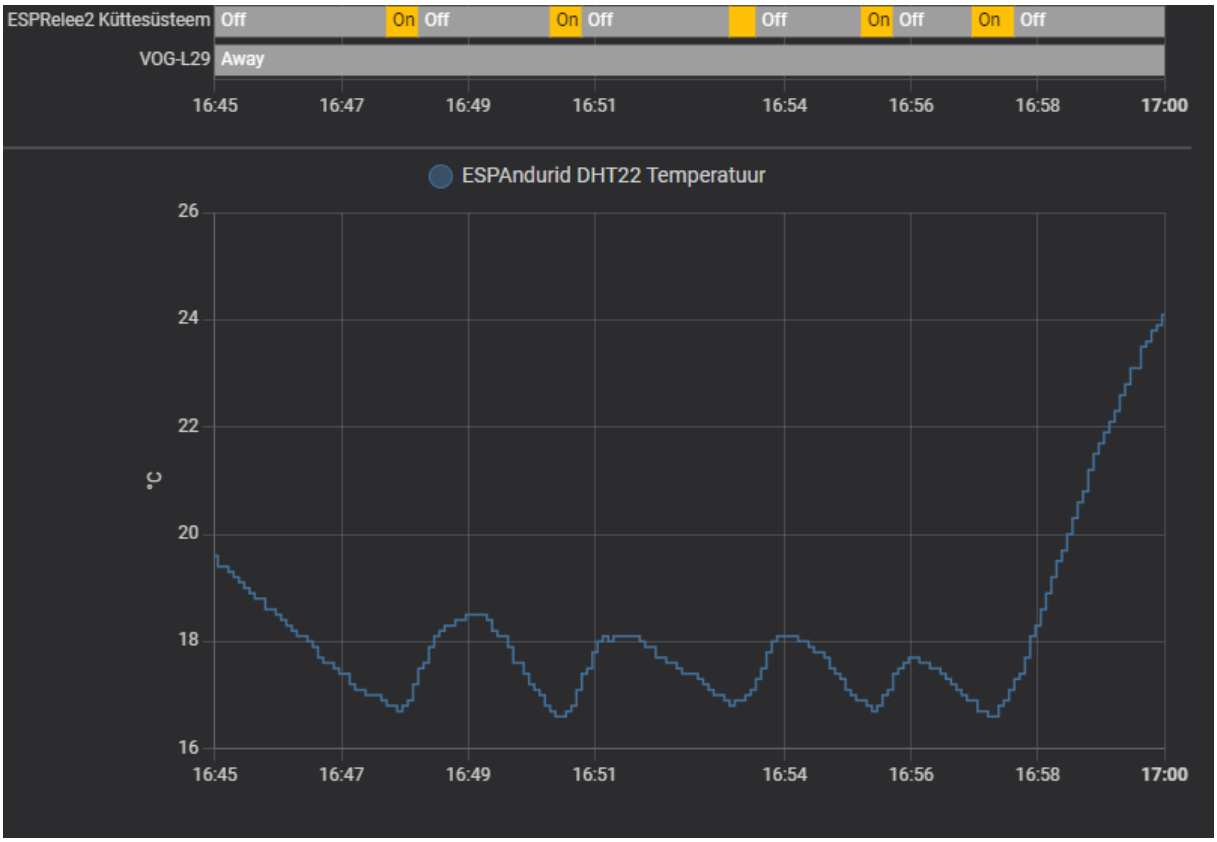
6.4. Automaatika katsetamine

Küttesüsteemi automaatika katsetamiseks viidi läbi järgnevad katsed. Kodus viibimise automaatikat katsetati, nutitelefoni koduse asukohaga, määrates temperatuuriks 24 °C. Kasutades külmakotti, jahutati anduri temperatuuri näit alla 24 °C ning toimus automaatne küttesüsteemi aktiveerimine, mille järel eemaldati külmakott, et temperatuur saaks küttesüsteemi soojusega tagasi üle 24 °C tõusta. Temperatuuri 24 °C ületamisel lülitus küttesüsteem automaatselt välja.

Kodus mitteviibimise automaatikat katsetati sarnaselt eelnevalt mainitule, erinevuseks oli nutitelefoni asukoha muutmine kodust väljapoole ning temperatuuri määramine 17 °C peale. Mõlemal juhul katsetati automaatika toimimist 5 korda ja automaatika töötas korrektselt (joonised 19, 20). Katsetega toimusid asukoha muutumisega seotud automaatika lülitused samuti korrektselt.

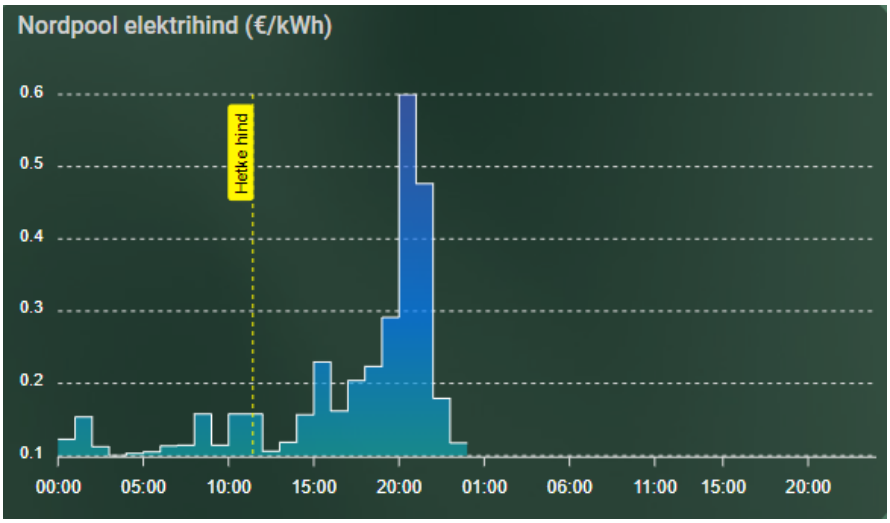


Joonis 19. Küttesüsteemi automaatika katsetamine kodus olekus, kuvatud on temperatuuri, küttesüsteemi staatuse ja nutitelefoni asukoha staatuse sõltuvus ajast.



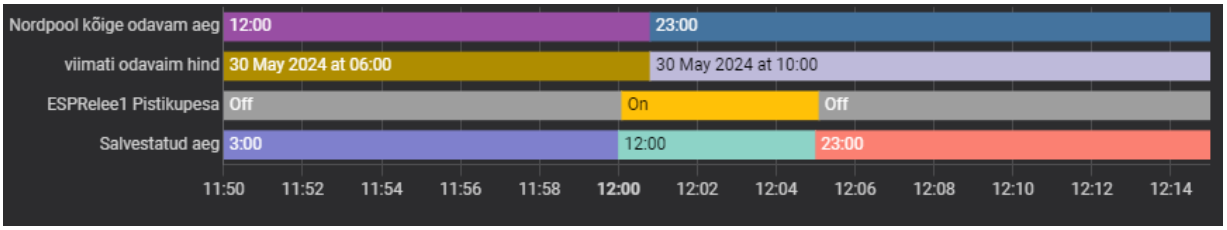
Joonis 20. Küttesüsteemi automaatika katsetamine kodust eemal olekus, kuvatud on temperatuuri, küttesüsteemi staatuse ja nutitelefoni asukoha staatuse sõltuvus ajast.

Börsihinna järgi pistikupesa lülitamise automaatikat katsetati salvestamise automatiseerimise ja viimase odavaima hinna aja manipuleerimisega. Viimaseks odavaimaks ajaks määrati 06:00, salvestuse käivituse ajaks 12:00. Katse viidi läbi 11:50 kuni 12:15, mille ajal oli järgmise 24 tunni kõige odavam börsihind kell 12:00, milleks oli 0,12 eurot kWh kohta (joonis 21). Mõlemad lisatasud määrati katse ajaks 0,062 eurot kWh kohta.



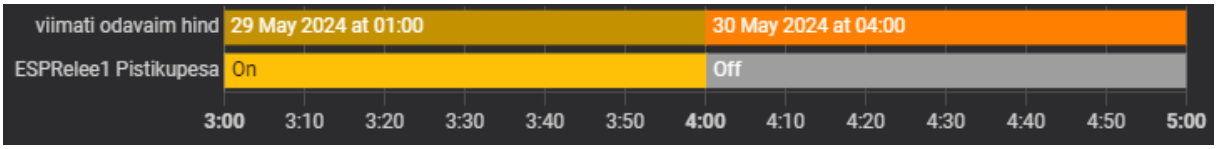
Joonis 21. Nord Pool börsihinnad katse ajal.

Katse ajal viimase odavaima aja muutmisel ning salvestuse automatiseerimise käivitusel tuvastas automaatika, et hetkel on kõige odavam elektrihind ning lülitas pistikupesa aktiivseks. Seejärel määrati viimaseks odavaimaks ajaks 10:00 ning salvestuse automatiseerimise käivitus ajaks 12:05. Ootuspäraselt lülitas automaatika pistikupesa välja, kuvades korrektselt, et odavaim elektrihind on kell 23:00 (joonis 22).



Joonis 22. Pistikupesa automaatika katsetamine.

Viimati odavaima hinna aja uuenduse automaatikat katsetati eraldi ning see töötas samuti korrektselt (joonis 22). Lõpetuseks prooviti pistikupesa manuaalse juhtimise lülitit, mille vajutamisel lülitusid pistikupesa Nord Pool automatiseerimised plaanipäraselt välja.



Joonis 23. Pistikupesa viimase odavaima hinna aja katsetamine.

KOKKUVÕTE

Antud bakalaureusetöö eesmärgiks oli uurida Raspberry Pi ja mikrokontrollerite kasutamise võimalusi nutikates kodudes ning luua nutika kodu lahendus mis, nimetatud seadmeid kasutades, aitab kodumajapidamise energiakasutust optimeerida.

Eesmärgi saavutamiseks tutvuti esmalt nutikate kodudega ja nutika kodu süsteemide komponentidega. Seejärel tehti ülevaade Raspberry Pi-st, ESP8266 mikrokontrollerist ning Home Assistant nutika kodu tarkvarast.

Uuriti Eesti kodumajapidamiste energiatarbimist lähtudes Statistikaameti andmebaasi andmetest ja kodumajapidamiste suurimate energiatarbijate väljaselgitamiseks tutvuti Eurostat analüüsiga. Analüüsist selgus, et Euroopa Liidu kodudes tulevad suurimad energiakulud ruumide soojendamisest. Lisaks kulub palju energiat vee soojendamisele ning valgustusele ja elektriseadmetele.

Teoreetiliselt kogutud materjali abil planeeriti nutika kodu süsteem, kus juhitakse küttesüsteemi ja elektriseadme pistikupesa, kasutades energiakasutuse optimeerimise võtteid. Küttesüsteemi energiakasutust optimeeriti lahendusega, kus ruumide soojendamine viiakse madalamale temperatuurile, siis kui kedagi kodus ei viibi. Elektriseadme pistikupesa muudeti elektri börsihinna järgi aktiveeritavaks. Kodu omanikule tarbimise teadvustamiseks lisati süsteemi voolutrafo. Nutika kodu keskseadmena kasutati Raspberry Pi 5-te, andurite ja täituritega ühendati ESP8266 mikrokontrollerid, mis ühendati juhtmevabalt keskseadmega. Keskseadmes ja mikrokontrollerites tehti vajalikud tarkvaralised muudatused, et süsteem toimiks ja ühilduks planeeringule vastavalt.

Lõpetuseks loodi nutika kodu mudel püsipaigaldusena ning tekitati erinevad automatiseerimised süsteemi juhtimiseks. Loodud süsteemi katsetati ning süsteem töötas ootuspäraselt ning korrektselt vastavalt planeeringule.

Edaspidistes uuringutes soovitab autor uurida pakutud lahenduse energia optimeerimise tõhusust nii energiatarbimise vähendamisel kui ka rahaliste kulude kokkuhoiul.

KASUTATUD KIRJANDUS

[1] KE0240: Energiabilanss, kütuse või energia liigi järgi, teradžauli (EUROSTATI metoodika). (andmed uuendatud 18.10.2023). – *Eesti Statistika andmebaas*. https://andmed.stat.ee/ (20.03.2024)

[2] **Chakraborty, A., Hasan, M., Islam, M., Islam, S., Shahriyar, F., Zaman, H., U.** (2023). Smart Home System: A Comprehensive Review. - Hindawi [e-ajakiri] https://www.hindawi.com/journals/jece/2023/7616683/ (20.03.2024)

[3] Coursera. What is the Internet of Things (IoT)? [veebileht] https://www.coursera.org/articles/internet-of-things (23.03.2024)

[4] IBM. What is the Internet of Things (IoT)? [veebileht] https://www.ibm.com/topics/internet-of-things (23.03.2024)

[5] Networkworld. 8 Internet things that are not IoT. [veebileht] https://www.networkworld.com/article/926302/8-internet-things-that-are-not-iot.html (23.03.2024)

[6] The Verge. How to pick a smart home platform. [veebileht] https://www.theverge.com/23751295/smart-home-platform-google-amazon-apple-samsung (23.03.2024)

[7] **Chen, Y., Lashkari, B., Musilek, P.** (2019) Energy Management for Smart Homes—State of the Art. - MDPI [e-ajakiri] https://www.mdpi.com/2076-3417/9/17/3459 (25.03.2024)

[8] ElectronicWings. Raspberry Pi introduction. [veebileht] https://www.electronicwings.com/raspberry-pi/raspberry-pi-introduction (29.03.2024)

[9] Raspberry Pi Datasheets. Raspberry Pi 4 datasheet. [veebileht] https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-datasheet.pdf (29.03.2024)

[10] Fabacademy. Networking ESP8266. [veebileht] https://fabacademy.org/2015/doc/networking-esp8266.html (05.04.2024)

[11] Espressif. ESP8266EX Datasheet. [veebileht] https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf (05.04.2024)

[12] Flyrobo. ESP8266 QFN-32 WIFI IC Chip. [veebileht] https://www.flyrobo.in/esp8266-qfn-32-wifi-ic-chip (05.04.2024)

[13] Home Assistant. Integrations. [veebileht] https://www.home-assistant.io/integrations/ (10.04.2024)

[14] Home Assistant. Concepts and terminology. [veebileht] https://www.home-assistant.io/getting-started/concepts-terminology/ (10.04.2024)

[15] Home Assistant. Installation. [veebileht] https://www.home-assistant.io/installation/ (10.04.2024)

[16] Nuclear Energy Agency. Cherobyl: Chapter I. The site and accident sequence. [veebileht] https://www.oecd-nea.org/jcms/pl_28271/chernobyl-chapter-i-the-site-and-accident-sequence (15.04.2024)

[17] Eurostat. Energy use in households in 2020. [veebileht] https://ec.europa.eu/eurostat/web/products-eurostat-news/-/ddn-20220617-1 (15.04.2024)

[18] Transpordiamet. Sõidukite statistika. [veebileht] https://www.transpordiamet.ee/soidukite-statistika (15.04.2024)

[19] Enefit. Elektriturg. [veebileht] https://www.energia.ee/era/elekter/elektriturg (19.04.2024)

[20] Elering. Taastuvenergia tasu. [veebileht] https://elering.ee/taastuvenergia-tasu (19.04.2024)

[21] Elektrilevi. Hinnakirjad. [veebileht] https://elektrilevi.ee/et/hinnakirjad (19.04.2024)

[22] Centre for sustainable energy. Energy saving myths. [veebileht] https://www.cse.org.uk/advice/energy-saving-myths/ (19.04.2024)

[23] Make It. NodeMCU ESP8266 Detailed Review. [veebileht] https://www.make-it.ca/nodemcu-details-specifications/ (25.04.2024)

[24] Dialedin. Wemos D1 ESP-12F. [veebileht] https://shop.dialedin.com.au/products/wemos-d1-esp-12f (25.04.2024)

[25] Components 101. DHT22 pinout specs datasheet. [veebileht] https://components101.com/sensors/dht22-pinout-specs-datasheet (25.04.2024)

[26] Home Assistant. Raspberry Pi. [veebileht] https://www.home-assistant.io/installation/raspberrypi (27.04.2024)

[27] Home Assistant. Onboarding. [veebileht] https://www.home-assistant.io/getting-started/onboarding/ (27.04.2024)

[28] Home Assistant. Home Assistant Add-ons. [veebileht] https://www.home-assistant.io/addons/ (27.04.2024)

[29] Native API Component. [veebileht] https://esphome.io/components/api.html (27.04.2024)

[30] Web Server Api. [veebileht] https://esphome.io/web-api/index.html (27.04.2024)

[31] **Viorel, M., A.** Comperative Study Regarding Measurements of Different AC Current Sensors. - Researchgate [e-ajakiri] https://www.researchgate.net/publication/312038242_Comparative_study_regarding_measurements_of_different_AC_current_sensors (30.04.2024)

[32] ESPHome. GPIO Switch. [veebileht] https://esphome.io/components/switch/gpio.html (30.04.2024)

[33] ESPHome. DHT Temperature+Humidty Senor. [veebileht] https://esphome.io/components/sensor/dht.html (30.04.2024)

[34] ESPHome. CT Clamp Current Sensor. [veebileht]
https://esphome.io/components/sensor/ct_clamp.html (30.04.2024)
[35] Github. Nordpool. [veebileht] https://github.com/custom-components/nordpool/ (05.05.2024)
[36] The automation blog. What is HACS and how to use it. [veebileht]
https://www.thehomeautomationblog.com/what-is-hacs-and-how-to-use-it/ (05.05.2024)

LISAD

Lisa 1. Küttesüsteemi NodeMCU v3 YAML konfiguratsioon

```yaml
substitutions:
  name: esphome-web-247b14
  friendly_name: ESPRelee1

esphome:
  name: ${name}
  friendly_name: ${friendly_name}
  name_add_mac_suffix: false
  project:
    name: esphome.web
    version: '1.0'

esp8266:
  board: esp01_1m

# Enable logging
logger:

# Enable Home Assistant API
api:

# Allow Over-The-Air updates
ota:

# Allow provisioning Wi-Fi via serial
improv_serial:

wifi:
  # Set up a wifi access point
  ap: {}

# In combination with the `ap` this allows the user
# to provision wifi credentials to the device via WiFi AP.
captive_portal:

# Relee kontroll
switch:
  - platform: gpio
    pin: 5
    inverted: true
    name: "Pistikupesa"

dashboard_import:
  package_import_url: github://esphome/example-configs/esphome-
web/esp8266.yaml@main
```

```
  import_full_config: true

# To have a "next url" for improv serial
web_server:
```

Lisa 2. Pistikupesa NodeMCU v3 YAML konfiguratsioon

```
substitutions:
  name: esphome-web-8d91af
  friendly_name: ESPRelee2

esphome:
  name: ${name}
  friendly_name: ${friendly_name}
  name_add_mac_suffix: false
  project:
    name: esphome.web
    version: '1.0'

esp8266:
  board: esp01_1m

# Enable logging
logger:

# Enable Home Assistant API
api:

# Allow Over-The-Air updates
ota:

# Allow provisioning Wi-Fi via serial
improv_serial:

wifi:
  # Set up a wifi access point
  ap: {}

# In combination with the `ap` this allows the user
# to provision wifi credentials to the device via WiFi AP.
captive_portal:

# Relee kontroll
switch:
  - platform: gpio
    pin: 5
    inverted: true
    name: "Küttesüsteem"

dashboard_import:
  package_import_url: github://esphome/example-configs/esphome-
web/esp8266.yaml@main
```

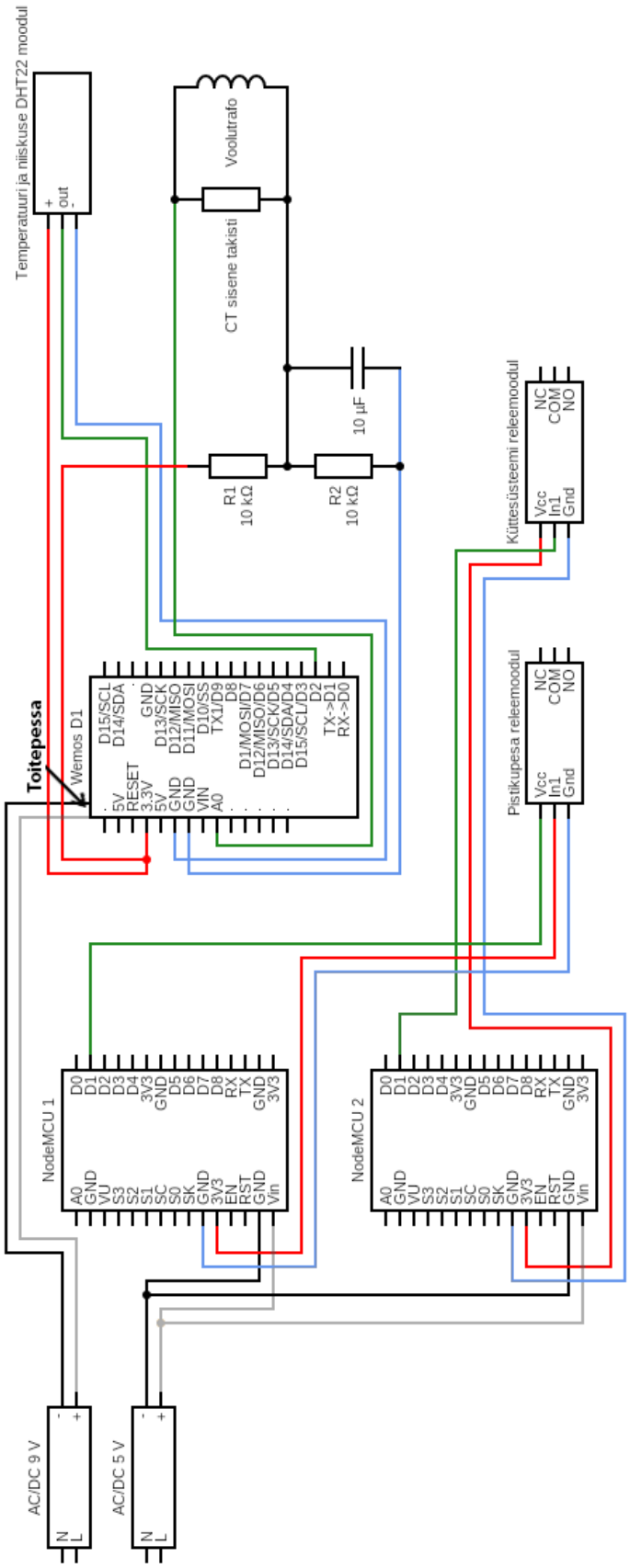
```
  import_full_config: true

# To have a "next url" for improv serial
web_server:
```

Lisa 3. Andurite Wemos D1 R1 YAML konfiguratsioon

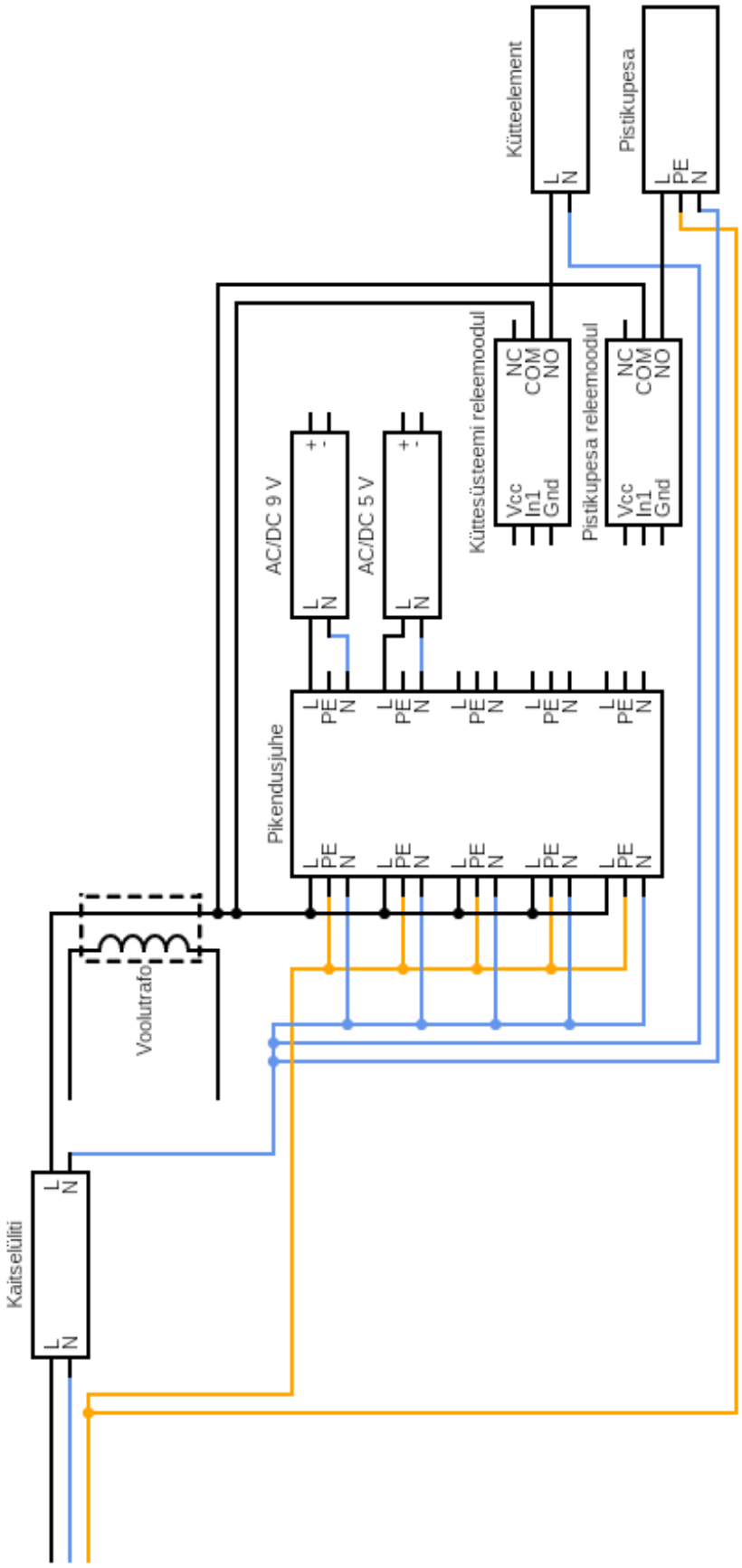
```yaml
substitutions:
  name: esphome-web-f37434
  friendly_name: ESPAndurid

esphome:
  name: ${name}
  friendly_name: ${friendly_name}
  name_add_mac_suffix: false
  project:
    name: esphome.web
    version: '1.0'

esp8266:
  board: esp01_1m

# Enable logging
logger:

# Enable Home Assistant API
api:

# Allow Over-The-Air updates
ota:

# Allow provisioning Wi-Fi via serial
improv_serial:

wifi:
  # Set up a wifi access point
  ap: {}

# In combination with the `ap` this allows the user
# to provision wifi credentials to the device via WiFi AP.
captive_portal:

#Kood DHT22 sensori jaoks
sensor:
  - platform: dht
    pin: 16
    model: DHT22
    update_interval: 5s
    temperature:
      name: "DHT22 Temperatuur"
    humidity:
      name: "DHT22 Niiskus"
```

```
  # Kood voolutrafo (CT) sensori jaoks
  - platform: ct_clamp
    sensor: adc_sensor
    name: "Mõõdetud Vool"
    update_interval: 5s
    filters:
      - calibrate_linear:
          # Voolu puudumisel on väljund 0
          - 0 -> 0
          # Tuli kasutada teadaoleva tarbimisega tarbijat (Antud juhul
veekeetja)
          # Logis näidatud vool vs teadaolev vool
          - 0.1 -> 10
  # ADC pini defineerimine
  - platform: adc
    pin: A0
    id: adc_sensor


dashboard_import:
  package_import_url: github://esphome/example-configs/esphome-
web/esp8266.yaml@main
  import_full_config: true

# To have a "next url" for improv serial
web_server:
```

Lisa 4. Mudeli alalisvoolu ühendusskeem koos voolutrafoga



Lisa 5. Mudeli vahelduvvoolu ühendusskeem



Lisa 6. Küttesüsteemi automatiseerimiste YAML konfiguratsioonid

```yaml
- id: '1715005092791'
  alias: Küttesüsteem sisse temperatuurist kodus
  description: Temperatuuri järgi küttesüsteemi sisselülitus kodus
  trigger:
  - platform: numeric_state
    entity_id:
    - sensor.esphome_web_f37434_dht22_temperatuur
    for:
      hours: 0
      minutes: 0
      seconds: 5
    below: input_number.temperatuur_kodus_olles
  condition:
  - condition: state
    entity_id: switch.esphome_web_8d91af_k_ttes_steem
    state: 'off'
    for:
      hours: 0
      minutes: 0
      seconds: 0
  action:
  - service: switch.turn_on
    metadata: {}
    data: {}
    target:
      entity_id: switch.esphome_web_8d91af_k_ttes_steem
  mode: single
- id: '1715006034346'
  alias: Küttesüsteem välja temperatuurist kodus
  description: Temperatuuri järgi küttesüsteemi väljalülitus kodus
  trigger:
  - platform: numeric_state
    entity_id:
    - sensor.esphome_web_f37434_dht22_temperatuur
    for:
      hours: 0
      minutes: 0
      seconds: 5
    above: input_number.temperatuur_kodus_olles
  condition:
  - condition: state
    entity_id: switch.esphome_web_8d91af_k_ttes_steem
    state: 'on'
    for:
      hours: 0
```

```
      minutes: 0
      seconds: 0
  action:
  - service: switch.turn_off
    metadata: {}
    data: {}
    target:
      entity_id: switch.esphome_web_8d91af_k_ttes_steem
  mode: single
- id: '1715019826107'
  alias: Koju jõudmise temperatuuri aktiveerimine
  description: ''
  trigger:
  - platform: zone
    entity_id: person.jaan_jalas
    zone: zone.home
    event: enter
  condition: []
  action:
  - service: automation.turn_on
    target:
      entity_id:
      - automation.kuttesusteem_sisse_temperatuurist
      - automation.kute_valja
    data: {}
  - service: automation.turn_off
    data:
      stop_actions: true
    target:
      entity_id:
      - automation.kuttesusteem_sisse_kodust_valjas
      - automation.kuttesusteem_valja_temperatuurist_kodust_valjas_olles
  mode: single
- id: '1715020069478'
  alias: Küttesüsteem sisse temperatuurist kodust väljas
  description: Temperatuuri järgi küttesüsteemi sisselülitus kodust väljas
  trigger:
  - platform: numeric_state
    entity_id:
    - sensor.esphome_web_f37434_dht22_temperatuur
    for:
      hours: 0
      minutes: 0
      seconds: 5
    below: input_number.temperatuur_kodust_valjas_olles
  condition:
  - condition: state
    entity_id: switch.esphome_web_8d91af_k_ttes_steem
    state: 'off'
```

```
    for:
      hours: 0
      minutes: 0
      seconds: 0
  action:
  - service: switch.turn_on
    metadata: {}
    data: {}
    target:
      entity_id: switch.esphome_web_8d91af_k_ttes_steem
  mode: single
- id: '1715020292861'
  alias: Küttesüsteem välja temperatuurist kodust väljas
  description: Temperatuuri järgi küttesüsteemi väljalülitus kodust väljas
  trigger:
  - platform: numeric_state
    entity_id:
    - sensor.esphome_web_f37434_dht22_temperatuur
    for:
      hours: 0
      minutes: 0
      seconds: 5
    above: input_number.temperatuur_kodust_valjas_olles
  condition:
  - condition: state
    entity_id: switch.esphome_web_8d91af_k_ttes_steem
    state: 'on'
    for:
      hours: 0
      minutes: 0
      seconds: 0
  action:
  - service: switch.turn_off
    metadata: {}
    data: {}
    target:
      entity_id: switch.esphome_web_8d91af_k_ttes_steem
  mode: single
- id: '1715020472295'
  alias: Kodust lahkumise küttesüsteemi aktiveerimine
  description: ''
  trigger:
  - platform: zone
    entity_id: person.jaan_jalas
    zone: zone.home
    event: leave
  condition: []
  action:
  - service: automation.turn_on
```

```
    metadata: {}
    data: {}
    target:
      entity_id:
      - automation.kuttesusteem_sisse_kodust_valjas
      - automation.kuttesusteem_valja_temperatuurist_kodust_valjas_olles
  - service: automation.turn_on
    metadata: {}
    data: {}
    target:
      entity_id:
automation.kuttesusteem_valja_temperatuurist_kodust_valjas_olles
  - service: automation.turn_off
    metadata: {}
    data:
      stop_actions: true
    target:
      entity_id:
      - automation.kuttesusteem_sisse_temperatuurist
      - automation.kute_valja
  mode: single
```

Lisa 7. templates.yaml

```
sensor:
  - name: "NordPool kõige odavam hind"
    unique_id: fe888f35-a4fd-487d-b7d8-13de415d6928
    icon: mdi:currency-eur
    unit_of_measurement: "€"
    state: >
      {% set ooteaeg_tundides = 24 | int %}
      {% set viimati_odavaim_hind =
states("input_datetime.viimati_odavaim_hind") | as_timestamp | round %}
      {% set hetkene_aeg = as_timestamp(now()) %}
      {% set ns = namespace(odavaim_hind=100, tunni_loendur=0) %}
      {% set jargmine_lubatud_aeg = viimati_odavaim_hind + 6 * 3600 - 5 %}

      {% for paev in ["raw_today", "raw_tomorrow"] %}
        {% for tunnid in state_attr("sensor.nordpool_kwh_ee_eur_3_10_02",
paev) %}
          {% if ns.tunni_loendur >= ooteaeg_tundides %}
            {% break %}
          {% endif %}
          {% set saadud_aeg = as_timestamp(as_local(tunnid.start)) %}
          {% if saadud_aeg > hetkene_aeg - 3600 %}
            {% set ns.tunni_loendur = ns.tunni_loendur + 1 %}
            {% if saadud_aeg >= jargmine_lubatud_aeg %}
              {% set hinna_lisad = 0 %}
              {% if saadud_aeg|timestamp_custom("%H:%M") >= "07:00" and
saadud_aeg|timestamp_custom("%H:%M") < "22:00" %}
                {% set hinna_lisad =
states("input_number.elektri_lisa_paeval") | float(0) %}
              {% else %}
                {% set hinna_lisad = states("input_number.elektri_lisa_oosel")
| float(0) %}
              {% endif %}
              {% set saadud_hind = (tunnid.value | float(0) + hinna_lisad) %}
              {% if saadud_hind < ns.odavaim_hind %}
                {% set ns.odavaim_hind = saadud_hind %}
              {% endif %}
            {% endif %}
          {% endif %}
        {% endfor %}
      {% endfor %}
      {{ ns.odavaim_hind | round(2) }}

  - name: "NordPool kõige odavam aeg"
    unique_id: 97c3cbf2-4d98-4a79-8897-f44794cfa74d
    icon: mdi:clock
```

```
    state: >
      {% set ooteaeg_tundides = 24 | int %}
      {% set viimati_odavaim_hind =
states("input_datetime.viimati_odavaim_hind") | as_timestamp | round %}
      {% set hetkene_aeg = as_timestamp(now()) %}
      {% set ns = namespace(odavaim_hind=100, odavaim_aeg=now(),
tunni_loendur=0) %}
      {% set jargmine_lubatud_aeg = viimati_odavaim_hind + 6 * 3600 - 5 %}

      {% for paev in ["raw_today", "raw_tomorrow"] %}
        {% for tunnid in state_attr("sensor.nordpool_kwh_ee_eur_3_10_02",
paev) %}
          {% if ns.tunni_loendur >= ooteaeg_tundides %}
            {% break %}
          {% endif %}
          {% set saadud_aeg = as_timestamp(as_local(tunnid.start)) %}
          {% if saadud_aeg > hetkene_aeg - 3600 %}
            {% set ns.tunni_loendur = ns.tunni_loendur + 1 %}
            {% if saadud_aeg >= jargmine_lubatud_aeg %}
              {% set hinna_lisad = 0 %}
              {% if saadud_aeg|timestamp_custom("%H:%M") >= "07:00" and
saadud_aeg|timestamp_custom("%H:%M") < "22:00" %}
                {% set hinna_lisad =
states("input_number.elektri_lisa_paeval") | float(0) %}
              {% else %}
                {% set hinna_lisad = states("input_number.elektri_lisa_oosel")
| float(0) %}
              {% endif %}
              {% set saadud_hind = (tunnid.value | float(0) + hinna_lisad) %}
              {% if saadud_hind < ns.odavaim_hind %}
                {% set ns.odavaim_aeg = saadud_aeg %}
                {% set ns.odavaim_hind = saadud_hind %}
              {% endif %}
            {% endif %}
          {% endif %}
        {% endfor %}
      {% endfor %}
      {{ ns.odavaim_aeg | timestamp_custom("%H:%M", true) }}
```

Lisa 8. Pistikupesa automatiseerimiste YAML konfiguratsioonid

```yaml
- id: '1715102158387'
  alias: Pistiku manuaalne juhtimine, automaatika välja
  description: ''
  trigger:
  - platform: state
    entity_id:
    - input_boolean.pistiku_manuaalne_juhtimine_automaatika_valja
  condition: []
  action:
  - service: automation.toggle
    target:
      entity_id:
      - automation.salvestus_pistik
      - automation.salvestus_pistik_sisse
    data: {}
  - service: switch.toggle
    metadata: {}
    data: {}
    target:
      entity_id:
      - switch.esphome_web_247b14_pistikupesa
  mode: single
- id: '1716123293746'
  alias: Viimati odavaim hind
  description: ''
  trigger:
  - platform: state
    entity_id:
    - switch.esphome_web_247b14_pistikupesa
    from: 'off'
    to: 'on'
    for:
      hours: 1
      minutes: 0
      seconds: 0
  condition: []
  action:
  - service: input_datetime.set_datetime
    metadata: {}
    data:
      timestamp: '{{ as_timestamp(now()) }}'
    target:
      entity_id: input_datetime.viimati_odavaim_hind
  mode: single
- id: '1717023197359'
```

```
  alias: Nord pool hinna ja aja salvestamine
  description: Salvestamine
  trigger:
  - platform: time
    at: '12:05:00'
  condition: []
  action:
  - service: input_number.set_value
    data_template:
      entity_id: input_number.salvestatudhind
      value: '{{ states(''sensor.nordpool_odavaim_hind'') }}'
  - service: input_datetime.set_datetime
    data_template:
      entity_id: input_datetime.salvestatudaeg
      time: '{{ states(''sensor.nordpool_odavaim_aeg'') }}'
  mode: single
- id: '1717054092447'
  alias: Nord Pool pistikupesa välja
  description: ''
  trigger:
  - platform: template
    value_template: '{{ states("input_datetime.salvestatudaeg")[:2] !=
now().strftime("%H")
      }}'
    for:
      hours: 0
      minutes: 0
      seconds: 5
  condition:
  - condition: state
    entity_id: switch.esphome_web_247b14_pistikupesa
    state: 'on'
    for:
      hours: 0
      minutes: 0
      seconds: 5
  action:
  - service: switch.turn_off
    metadata: {}
    data: {}
    target:
      entity_id:
      - switch.esphome_web_247b14_pistikupesa
  mode: single
- id: '1717054227392'
  alias: Nord Pool pistikupesa sisse
  description: ''
  trigger:
  - platform: template
```

```
    value_template: '{{ states("input_datetime.salvestatudaeg")[:2] ==
now().strftime("%H")
      }}'
    for:
      hours: 0
      minutes: 0
      seconds: 5
  condition:
  - condition: state
    state: 'off'
    for:
      hours: 0
      minutes: 0
      seconds: 5
    entity_id: switch.esphome_web_247b14_pistikupesa
  action:
  - service: switch.turn_on
    metadata: {}
    data: {}
    target:
      entity_id:
      - switch.esphome_web_247b14_pistikupesa
  mode: single
```

LIHTLITSENTS

Lihtlitsents lõputöö salvestamiseks ja üldsusele kättesaadavaks tegemiseks ning juhendaja(te) kinnitus lõputöö kaitsmisele lubamise kohta

Mina, Jaan Jalas,

sünniaeg 11.04.2001,

1. annan Eesti Maaülikoolile tasuta loa (lihtlitsentsi) enda koostatud lõputöö

Mikrokontrollerite ja Raspberry Pi kasutamine nutikate kodude energiakasutuse optimeerimisel,

mille juhendaja on Janar Kalder, *PhD*

   1.1. salvestamiseks säilitamise eesmärgil,
   1.2. digiarhiivi DSpace lisamiseks ja
   1.3. veebikeskkonnas üldsusele kättesaadavaks tegemiseks

kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile;
3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Lõputöö autor Jaan Jalas
/Allkirjastatud digitaalselt/

Tartu, 30.05.2024

Juhendaja kinnitus lõputöö kaitsmisele lubamise kohta

Luban lõputöö kaitsmisele.

Janar Kalder
/Allkirjastatud digitaalselt/ (30.05.2024)