TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Siim Tišler 213891IACB

IoT Toataime Seiresüsteem

Bakalaureusetöö

Juhendaja: Peeter Ellervee
PhD

Tallinn 2024

Autorideklaratsioon

Kinnitan, et olen koostanud antud aruande 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.

Autorid: Siim Tišler

16.04.2024

Annotatsioon

Lõputöö käigus loodi IoT toataimede seiresüsteem, mis lihtsustaks toataimede kasvatamist. Süsteem loeb taimedele eluks vajalikke parameetreid ning kasutajal on neid reaalajas võimalik vaadata läbi veebiliidese graafikutelt ja infokaartidelt, et saada hea visuaalne ülevaade oma toataime tervisest.

Tervikliku süsteemi toimeks tuli arendada kolm erinevat osa milleks olid: trükkplaadi disan koos riistvara komponentide analüüsi ja valikuga, sardtarkvara arendamine ning veebiliidese arendamine.

Igale eelnevalt mainitud osale on eraldi seatud eesmärgid ning nendest eesmärkidest tulenevad nõuded.

Teoreetiline töö koosneb võimalike alternatiivsete riistvara komponentide ja tarkvarade analüüsidest, võrdlustest ja põhjendustest miks mõni riistvara komponent või tarkvara valituks osutus.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 48 leheküljel, 13 peatükki, 15 joonist, 6 tabelit.

Abstract

IoT Indoor Plant Monitoring System

The trends of a greener lifestyle is increasingly prevalent, ideas such as recycling, reducing CO2 emissions, and self-sustainability are more important to people than before because people have become more environmentally conscious. As urbanization continues, people seek to bring nature into their homes by growing both edible and decorative plants indoors.

Furthermore, home automation and smart home solutions are currently trending. Smart homes typically incorporate various devices and sensors to make daily life more comfortable and provide insightful data about ones home.

By addressing these trends, the project aims to offer a solution that simplifies plant care while providing users with valuable insights into their plants' needs and overall health.

The system collects vital parameters for plant health, allowing users to monitor them in real-time through a web interface, providing a comprehensive visual assessment of their plants' well-being.

The development of the monitoring system required the creation of three distinct components: hardware design with component analysis and selection, embedded software development, and web interface design. Each component got set specific objectives and requirements. The theoretical part of the thesis includes analyses, comparisons, and justifications for the selection of hardware and software components.

The thesis, is written in Estonian, contains 48 pages of text, 13 chapters, featuring 15 figures and 6 tables.

Lühendite ja mõistete sõnastik

ADC	*Analog to Digital Converter* analoog digitaal konverter
API	*Application Programming Interface,* rakendusliides
BLE	*Bluetooth Low Energy* väikese voolutarbega juhtmevaba võrgutehnoloogia
CSS	*Cascading Style Sheets,* kujunduskeel
Default Gateway	Võrgusõlm, ükdiselt ruuter, mis kontrollib välisvõrkude ja konkreetse sisevõrgu vahelisi pakette
DHCP	*Dynamic Host Configuration Protocol* protokoll mis laseb serveril või ruuteril dünaamiliselt hallata IP aadresse
DNS	*Domain Name Server* teenus, mis teisendab domeeninimed internetis või sisevõrgus kasutatavateks IP-aadressideks
DOM	*Document Object Model,* dokumendi objektimudel on platvormist ja keelest sõltumatu HTML-dokumentidega suhtlemise liides.
Flash	Välkmälu on püsimälu, kus andmed peale toite väljalülitamist säilivad
Frontend	Esiosasüsteem, kasutajaliidese osa, mida kasutaja näeb ning millega omab interaktsioonivõimet
getter, setter	Programmi funktsioonid, millest esimene tagastab teatud atribuudi väärtuse ning teine seab selle väärtuse.
GPIO	*General Purpose Input Output,* üldkasutatav sisend/väljund väljaviik
HTML	*Hypertext Markup Language*, tekstipõhine markeerimiskeel, mis defineerib veebilehe struktuuri
Hotspot	Kuumkoht, WiFi juurdepääsu punkt
IoT	*Internet of Things,* asjade internet ehk nutistu
IP	*Internet Protocol* võrgus oleva seadme identifikaator
I2C	*Inter-Integrated Circuit* mitme võimaliku meiser seadmega jadasiin
JSON	*JavaScript Object Notation,* standardne failiformaat, mis koosneb inimloetavast tekstist ja on keelest sõltumatu
LED	*Light Emitting Diode,* valgust kiirgav diood
LiPo	*Lithium Polymer* liitium ioon akude tehnoloogia
Makefile	Viis tarkvara kompileerimisprotseduuri automatiseerimiseks

MU-MIMO	*Multi-user multiple-input and multiple-output* traadita kommunikatsiooni tehnoloogia, mis kasutab mitut antenni nii saatja kui vastuvõtja otsas
NVS	*Non-Volatile Storage* mälu, mis toite puudumisel jääb alles, püsimälu
OFDMA	*Orthogonal Frequency-Division Multiple Access* on digitaalse andmeedastuse modulatsioonimeetod
OTA	*Over The Air,* juhtmevaba tarkvara uunedamise viis
PWM	*Pulse-Width Modulation* nelinurga kujuline digitaalsignaal
SSID	*Service Set Identifier,* võrgu nimi
TWT	*Target Wake Time* tagab, et seadmed veedavad ooterežiimis rohkem aega ja vahendavad andmeid energiatõhusamalt.
USB	*Universal Serial Bus*, universaalne jadasiin

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Üha enam propageeritakse rohelist eluviisi, põhilisteks ideedeks on taaskasutus, CO2 jalajälje vähendamine, *self-sustainabilty,* parim eestikeelne vaste on jätkusuutlik isemajandamine. Trendid näitavad, et inimesed on sellest mõjutatud ja üldiselt ka rohelisemad kui varem [1]. Jätkusuutliku isemajandamise kui ka suure urbaniseerimise tulemuseks on, et inimesed tahavad tuua loodust enda kodudesse, kasvatades toas nii söögi- kui ilutaimi.

Samuti on üheks trendiks praegu kodude automatiseerimine ja targakodu lahendused. Targad kodud koosnevad tavaliselt erinevatest seadmetest ja sensoritest, mis teevad igapäeva elu kergemaks ja annavad ülevaate kodu „olekust“.

Käesoleva projekti tulemusega proovitakse leida lahendust kahele eelnevalt mainitud probleemile. Eesmärk oli luua targa kodu seade, mille kasutamine lihtsustab taimede eest hoolitsemist ja annaks selge ning visuaalse ülevaate taimele eluks vajalikke parameetrite kohta.

Järgnevates peatükkides tuuakse välja millised süsteemi spetsiifilisemad eesmärgid ja nõuded on, näidatakse analüüsi tulemusena välja selgunud riistvara ja tarkvara valikuid. Milline on süsteemi terviklik arhitektuur. Seletatakse milline on süsteemi kui ka veebiliidese juhtloogika ja lõpuks projekti edasiarenduse võimalused.

2 Süsteemi ülesanne ja nõuded

Kuna füüsiline süsteem jaotub kolmeks suuremaks plokiks, milleks on riistvara disain, sardtarkvara arendus ning veebiliidese arendus, on ka iga osa eesmärgid ja eesmärkidest tulenevad nõuded erinevad.

Üldised süsteemi nõuded:

1. Süsteem peab suutma lugeda mulla niiskust, õhuniiskust ja -temperatuuri ning valgust.
2. Süsteem peab töötama aku/patarei toitel ning olema laetav
3. Süsteem ei tohi häirida või halvendada toataime elu.
4. Süsteemi installeerides saab läbi veebiliidese taime keskkonna parameetritest ülevaate.

2.1 Riistvara nõuded

Riistvara plaadil peavad olema kindlad komponendid, et üldise süsteemi nõuded oleksid täidetud. Et 1. nõue oleks täidetud peavad olema trükkplaadil järgnevad sensorid: pinnase niiskuse sensor, õhu niiskuse sensor, temperatuuri sensor ning valguse sensor.

2. nõude täitmiseks tuleb välja töötada aku toite lahendus. Aku laadimise protsess peab olema lahendatud turvaliselt, sest kasutatakse LiPo (Lithium Polymer) akut, mis võivad olla ohtlikud kui neid valesti käsitseda.

Aku kasutus peab olema optimiseeritud, mis tähendab, et aku ei tohi tühjeneda ebavajalikult kiiresti, ühe täislaadimise kohta peab aku kestma vähemalt 70 ööpäeva.

Et oleks üldse võimalik loetud andmeid läbi sensorite kuhugi edastada, siis peab süsteemil olema mingisugune juhtmevaba andmeedastus funktsionaalsus, olgu selleks WiFi, Bluetooth, Zigbee vms.

Et süsteemi oleks võimalik arendada ja sardtarkvara koodi mikrokontrollerile laadida peab olema samuti selle jaoks võimalus, olgu selleks nt OTA (*Over The Air*) või USB (*Universal Serial Bus*) vms.

Arendatav trükkplaat peab olema sobiva suurusega, et esiteks trükkplaat mahuks potti, teiseks poleks taimele kahjulik, nt ei tohi olla metallist osasid kokkupuutes mullaga, et vältida korrosiooni, mis on kahjulik mullale ja trükkplaadile. Trükkplaadi gabariidid peavad olema lahendatud selliselt, et trükkplaadile oleks võimalik ümber korpus paigaldada. Komponentide asukohad peavad trükkplaadil olema sellised, et korpuse paigaldamisel korpus ise ei takistaks süsteemi tööd, nt ei tohi piirata valguse levikut valguse sensorisse või koguneda soojust korpuse sisemusse, mis mõjutakse temperatuuri sensorit.

2.2 Sardtarkvara nõuded

Sardtarkvara on vabavaraline seega väga tähtis, et sardtarkvara oleks loetav ja jagatud suuresti iseseisvatesse moodulitesse, et ta oleks lihtsasti loetav ning ka kõrvaliste isikute poolt integreeritav enda projektidesse.

Lõpp-programm peab olema ajaliselt optimiseeritud, ehk aeg, mis kulub kõikide koodi ülesannete täitmiseks, peab olema võimalikult lühike.

Kuna sardtarkvara laetakse otse mikrokontrolleri mällu, siis süsteemi pikaajalise töötamise eesmärgil tuleb võimalikult suures mahus lähtutada laialt levinud sardtarkvara kirjutamise printsiibidest ja põhimõtetest [2]. Põhimõtted, millele on eriti rõhku pandud:

- Vältida dünaamilist mälujaotust

  Kuna C ja C++ funktsioonid nagu ***malloc*** ja ***free*** võivad olla ettearvamatud ning mitte igakord suuta mälu tagastada, võib juhtuda, et programm läheb katki. Lisaks mikrokontrollerite puhul, kus mälumahud on väiksed, on mälu fragmenteerumise oht samuti suurem, staatilise mälu kasutus väldib seda probleemi.

- Programmitsüklitel peab olema ülemine piir, näiteks et vältida mõnda lõputusse tsüklisse jäämist, peab olema mõni kontroll muutuja või tingimus, et seda ei juhtuks.

  Selle näiteks on Joonis 1. peal näha, üldist loogikat, olukorral kui WiFi ühendust ei suudeta luua 10 sekundi jooksul, väljutakse tsüklist.

```
seconds_max = 10
while wifi == disconnected and timespent < seconds_max
   TryConnectWifi()
```

Joonis 1. Pseudokoodi näide ülemisest tsükli piirist

- Üks programmi funktsioon täidab ühte konkreetset ülesannet.

  See muudab koodi üldiselt loetavamaks ja arusaadavamaks. Lisaks on võimalik erinevaid funktsioone mitu korda kasutada, et vältida koodi kordamist.

- Vältida mäluviitade (*pointer)* kasutamist

  Mäluviidad hägustavad programmikoodi, muutujate ümberkirjutamine on risk ning mäluviitadega juhtub kaasa tulema ka eelnevalt mainitud dünaamilise mälujaotuse probleem sardsüsteemides. Mäluviitasi kasutades on lihtsam teha vigu.

Eelnevad põhimõtted on kasulikud, et oleks reeglid, millest sardtarkvara kirjutamisel lähtuda, see aga ei tähenda, et igast reeglist/põhimõttest on absoluutselt kinnipeetud kogu koodi puhul, on ka erandeid.

2.3 Veebiliidese nõuded

Läbi URL taibutaim.web.app on võimalik veebiliidesele ligipääseda. Veebiliides peab olema turvaline, seega kasutajal peab olema võimalus registreerida ja sisse logida. Sisse loginud kasutaja näeb ainult oma seadmete andmeid.

Kasutaja saab algseadistada süsteemi vastavalt nõutud vajadustele, näiteks peab kasutaja saama edastada juhtmevaba andmeedastus infot süsteemile, nagu WiFi võrgu SSID (*Service Set Identifier*) ja parool.

Kasutaja peab nägema andmeid reaalajas, kohe kui süsteem mõõtetulemusi saadab, on veebiliidesest seda võimalik vaadata. Lisaks saab kasutaja vaadata mõõtetulemusi ajas muutudes. Mugav lahendus selleks on graafikud, millel saab kasutaja muuta ajavahemikke. Näiteks mõõtetulemused viimase 3 tunni, 12 tunni, 7 päeva või 30 päeva jooksul.

3 Pinnase niiskussensori analüüs

Üks kõige kriitilistemast küsimustest projekti juures on: kuidas lahendada mulla niiskuse mõõtmine?

Neutron sensor – kasutatakse kiiritust, sensori ots kiirgab pinnasesse neutroneid ning tuvastatakse kiirgatud neutronite kokkupõrkeid vesiniku aatomitega.

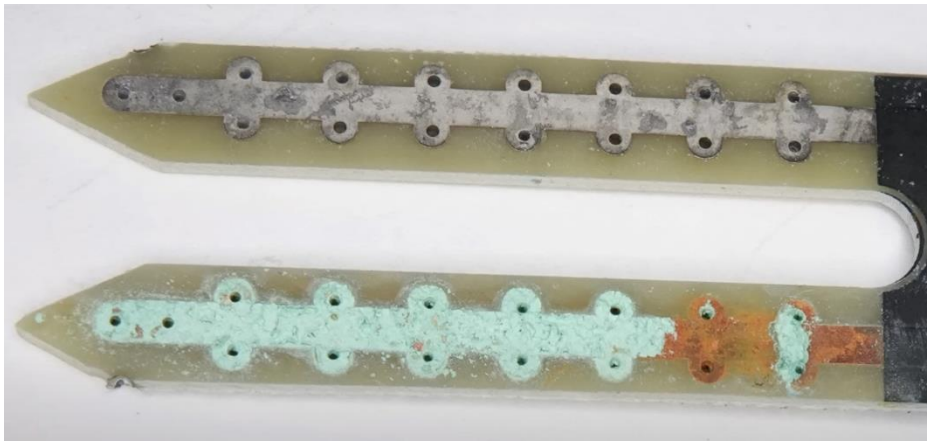
Refaktromeeria sensorid – alternatiivne variant neutron sensorile. Jaguneb kahte liiki, sagedus-domeen refaktromeeria – SDR ja aja-domeen refaktromeetria – ADR. SDR sensorid tekitavad mullas elektrimagnetvälja laineid ja analüüsivad välja minevate ja peegeldunud lainete sagedusnihet. ADR puhul mõõdetakse kahe mullas oleva mõõteotsa vahel olevate pinge impulsside peegeldumisi. [3] [4].

Takistus sensorid – üldiselt kahe elektroodidga mõõteots asetatakse mulda, üks elektroodidest pingestatakse, kahe elektroodi vahele tekib potentsiaalide erinevus. Kui lisada vett, siis takistus väheneb, mõõdetakse muutuvat pinget, et määrata pinnase niiskussisaldust.

Mahtuvus sensorid – kasutatakse ära mahtuvust ja mõõdetakse mulla dielektrilist konstanti. Mulla dielektriline konstant on väiksem kui veel, seega kui mulda lisada vett, siis dielektriline konstant muutub ning muutub ka mahtuvus.

Neutron sensorid sensorid on küll täpsed, kuid väga kallid, nad peavad ka olema litsenseeritud, kuna tegemist on nõrka radioaktiivsust kasutatavate seadmetega. Refaktromeetria sensorid on natuke odavamad, kuid ikkagi suhteliselt kallid. Mõlema puhul on keeruline integreerida mõõtetulemusi ja vajavad tihti selleks lisa seadmeid, tegemist on industriaalselt kasutatavate mõõteriistadega, seega on nad kindlasti välja arvatud käesoleva projekti skoobist.

Takistus sensorid on turul laialt levinud ning nad on ka väga odavad. Kuid takistus sensorite puhul esineb palju erinevaid probleeme. Takistuslikud pinnase niiskussensorid on väga tundlikud pinnase soolasusele. Igal mullal on natuke erinev soolasus, eriti kui mõnele on lisatud väetisi jne, on mõõtetulemus ebatäpne. Kuna takistusliku sensori töötamiseks peab üks elektrood olema pingestatud alalisvooluga ning elektroodid puutuvad kokku veega on elektrolüüsi teke paratamatu. Elektrolüüsi tõttu mõõteotsa elektroodid korrodeeruvad ja ajapikku muutuvad täielikult kasutuskõlbmatuks. Selline olukord peab olema täielikult välistatud käesolevas projektis. Korrodeerunud mõõteotsadest eritub mulda taimedele mürgiseid osakesi ja muudab taime elukeskkonna mitte kõlbulikuks, vt Joonis 2.



Joonis 2. Korrodeerunud takistuslik niiskussensor [5]

Mahtuvuslik pinnase niiskussensor on tundlik sooladele, kui soolade kontsentratsioon pinnases on juba suur [4]. Mahtuvusliku variandi puhul puudub galvanaaniline kontakt ja elektrolüüsi ei teki. Mõõteskeem baseerub tavaliselt ostsillaatoril, mille resonantsvõnkeringi mahtuvuse moodustab mõõteelektrood. Seega lõplikuks valikuks kasutatakse mahtuvusliku andurit. Ostsillaator saab lisada trükkplaadile integraallülitina ning resonantsvõnkeringi mahtuvuse moodustava mõõtelektroodi saab tekitada ise. Selleks on vaja tekitada suure laiusega vaserada ja tema alla jääv maa kihi vahele tekib parasiitmahtuvus, sisuliselt väikese mahtuvusega kondensaator.

4 Süsteemi arhitektuur ja tehnoloogiad

Peatükis kirjeldatakse milliseid tööriistu ja tehnoloogiad kasutatakse ja miks. Seejärel näidatakse kuidas nende valitud tehnoloogiate järgi kujunes süsteemi arhitektuur.

4.1 Arenduseks kasutatud tarkvara

KiCad – Trükkplaadi projekteerimise tarkvara. Kuna tegemist on esimese trükkplaadiga, mille autor on loonud, siis on KiCad sobilik, sest see on tasuta ja algajasõbralik, võrreldes näiteks Altium Designeriga, mis on mõeldud suuremate projektide jaoks, koostöö võimalusega, ning üldiselt tasulise litsentsiga. EasyEDA on aga lihtsam õppida, kuid ta pole nii võimekas kui KiCad, sest EasyEDA baseerub veebitehnoloogiatel, projekteerimine ja terve projekt tehakse pilveteenuste kaudu. Selle pärast valiti kesktee ning KiCad. Kasutati versiooni v7.0.10.

Visual Studio Code – Koodiredaktor. Valiti, sest kogemus on sellega kõige suurem ning on võimalus lisada, integreerida erinevaid laiendus programme ja arenduskeskkondi

PlatoformIO – Mikrokontrollerite arenduskomplekt. Tarkvara on laiendusena lisatav Visual Studio Code külge ning võimaldab laadida programme mikrokontrollerisse. Teeb kogu sardtarkvara arenduse protsessi lihtsaks, arenduskomplekti on integreeritud jadaliidese monitor, kust saab andmeid saata ja lugeda, tehes silumist kergemaks. Samuti on kompileerimisprotseduur automatiseeritud, kõik lisatud teegid kompileeritakse automaatselt, ilma, et oleks vaja ise mõni Makefile või CMake fail luua [6].

4.2 Programmeerimiskeeled ja raamistikud

C/C++ – Mikrokontrolleri sardtarkvara on arendatud C/C++ keeles. C – perekonda kuuluvad keeled on nö „riistvara lähedased“, kuna nendega on võimalik otse seadme mälu alasi kontrollida, nad on optimaalne keele valik, sest nad on ka mälu poolest tõhusad, kuna kirjutatud programm kompileerub otse Assembleriks ja masinkoodi. Kuna resursse mikrokontrolleritel on vähe, siis on C ja C++ väga laialt levinud keeled, millega sardtarkvara kirjutada.

HTML ja Bootstrap CSS – Veebilehe komponentide loomiseks ja kujunduse jaoks mõeldud keeled. Need aitasid teha lehekülje visuaalselt ilusaks. Bootstrap on CSS (Cascading Style Sheets) raamistik, mis võimaldab HTML elementide kujundust muuta kiiremini kui tavalise CSS kasutamisel. Sellega hoitakse aega kokku.

Vue JavaScript – Veebilehele interaktiivsuse lisamine ning dünaamiliselt komponentide loomine, sündmuste jälgimine, andmete vahendamine andmebaasiga. Vue ***frontend*** raamistik teeb JavaScript arenduse loetavamaks, kasutatakse komponentarhitektuuri, kus eri funktsionaalsusega osad on eraldiseisvad. Kui klassikalise JavaScripti puhul iga kasutajaliidese elemendi muudatuse korral peab DOM-i (Document Object Model) muutma, siis raamistikega kasutatakse virtuaalset DOM-i, et reaalset DOM-i uuendada. Virutaalses DOM-is tehtud muudatusi võrreldakse päris DOM-iga ja uuendatakse vaid muudetud osi, selle asemel, et tervet DOM-i uuendada [7]. See tagab parema jõudluse ja sujuvama kasutajaliidese kasutaja jaoks. Kasutatakse Vue raamistiku versiooni 3.

4.3 Pilveteenused

Firebase – Google poolt loodud pilveteenud, mis on mõeldud arendajatele. Firebase valiti pilveteenusena, sest ta sisaldab endas palju veebilehe rakenduse jaoks vaja minevaid funktsionaalsusi, nagu näiteks tasuta veebimajutus, NoSQL reaalaja andmebaas, autenteimine, teavituste saatmine kasutajatele jne. Firebase on ideaalne valik antud projekti suurust ja skoopi arvestades, sest läbi ühe teenuse saab enamus vajalikest funktsionaalsustest, sest Firebase pakub eelnevalt mainitut alam pilveteenustena.

Autentimine – Kasutajatel peab olema võimalus end registreerida. Seade ise töötab sisuliselt ka nagu registreeritud kasutaja, sest ka seadmel peab olema luba, et andmebaasi saata ja sealt pärida.

Reaalaja andmebaaas – Kõik vajalikud andmed on JSON kujul pilve andmebaasis.

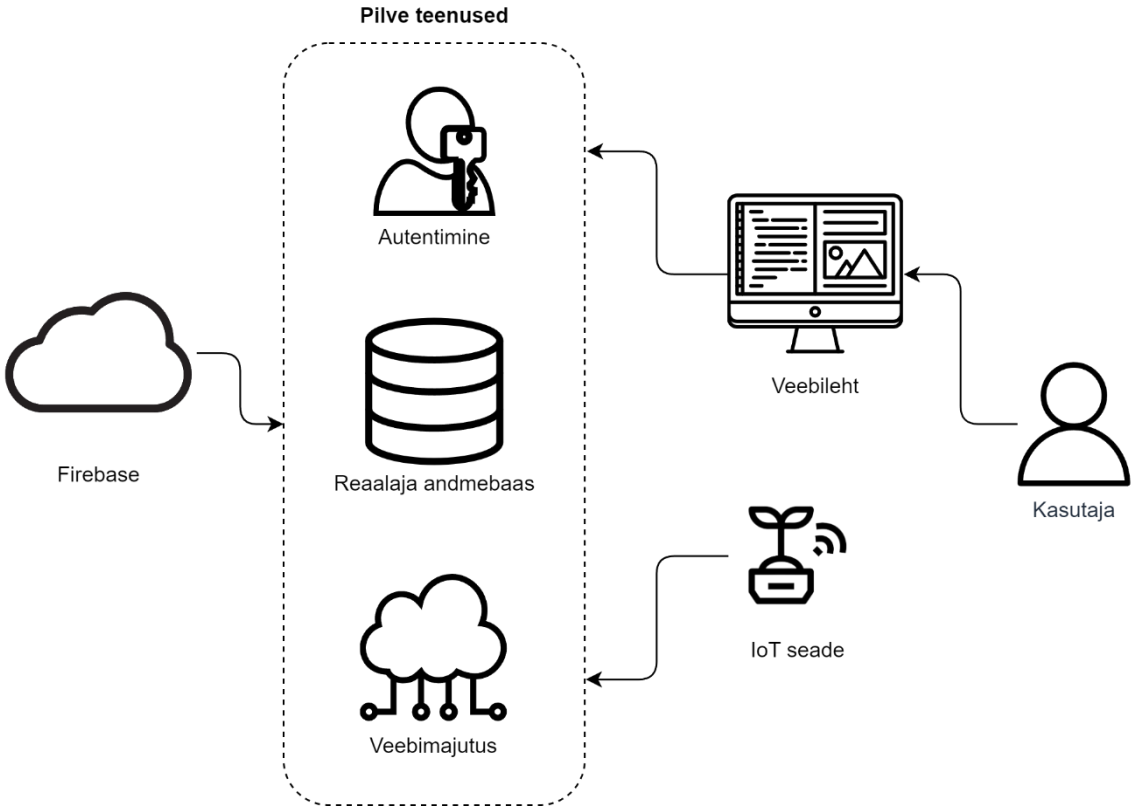
Veebimajutus – Veebilehe eest vastutab Firebase Hosting teenus.

4.4 Versioonihaldus

Git – Projektide versioonihalduse tarkvaraks valiti Git, kuna projekt koosneb riistvara-, sardtarkvara- kui ka veebiarendusest, siis on mõistlik terve projekti koodi hoida ühes repositooriumis.

GitHub – Repositooriumit ise hoiustatakse GitHub-is, kuna autor on GitHubiga kõige rohkem kokku puutunud.

4.5 Arhitektuuri plokkskeem



Joonis 3. Süsteemi üldine arhitektuur

Jooniselt 3 on näha üldist diagrammi mille järgi terve süsteem on ülesse ehitatud. Firebase vastutab kasutajate autentimise, andmebaasi ning veebimajutuse eest. IoT seade ehk lõplik riistvara lahendus käitub kui autenditud kasutaja ning saab saata andmeid reaalaja andmebaasi. Veebileht on Google pilve serveritele majutatud. Kasutaja saab endale läbi veebilehe luua konto ja saada ligipääsu, et läbi veebilehe pärida andmebaasist andmeid, mida veebileht hakkab siis kasutajale kuvama.

5 Riistvara analüüs ja komponentide valik

Peatükis tehakse analüüsid, et välja selgitada millised riistvara komponendid – sensorid, mikrokontrollerid, muud integraallülitid jne trükkplaadile panna. Kõikide komponentide kohta käivad nõuded on järgmised:

- Energiatõhusus
- Suurus
- Teekide/draiverite olemasolu
- Skoop – kui palju ja mis funktsionaalsust on vaja
- Jõudlus/täpsus

Järgnevalt analüüsitakse spetsiifilisi komponente ja antakse põhjused miks mõni komponent valiti üle teiste.

5.1 Mikrokontrolleri valik

Mikrokontrolleri valik oli esimene riistvara komponent, mida analüüsiti, kuna sellega hakatakse juhtima kõiki teisi sensoreid. Seega pidi mikrokontroller täitma kõiki üldisi nõudeid ja lisaks võiks mikrokontrollerisse olla integreeritud juba juhtmevaba andmeedastus funktsionaalsus. See tagab, et trükkplaadile jääb rohkem ruumi ja oleks vähem liidestamist mõne kontrolleri ja juhtmevaba andmeedastus seadme vahel.

Väga tähtis valikukriteerium on teekide ja draiverite olemasolu. Aega peab üle jääma, et ka sardtarkvara ja kasutajaliidese tarkvara kirjutada, seetõttu annab olemasolevate teekide kasutamine suure ajalise eelise.

Tabel 1. Mikrokontrollerite võrdlus [8] [9]

Mikrokontroller/ Funktsionaalsus	Esp32	STM32	Nordic Semiconductors
Energitõhusus	Keskmine	Hea	Väga hea
Jõudlus	Keskmine	Väga hea	Hea
Wireless	BLE, WiFi, Zigbee, Matter, Thread	Zigbee, BLE jm WiFi eraldi, nõuab lisa mooduleid	Zigbee, BLE jm WiFi eraldi, nõuab lisa mooduleid
Hind	Väga odav	Kallis	Kallis
Kättesaadavus	Väga hea	Hea	Keskmine
SDK-d	ESP-IDF, Arduino, Micropython	HAL teek, STMCubeIDE	nRF MDK
Teekide olemasolu	Väga hea	Keskmine	Halb
Kättesaadav info internetis	Väga hea	Hea	Keskmine

Tabel 1. pealt on näha, et STM32 ja Nordic Semiconductors mikrokontrollerid on kõige energiatõhusamad, neil on suurem jõudlus. Suur eelis on aga Esp32 perekonda kuuluvatel kontrolleritel selle poolest, et turul on väga palju variante, kus on nii BLE (Bluetooth Low Energy) 5.0 kui ka WiFi integreeritud ühe mooduli peale. [10] STM32 ja Nordicu puhul on tavaliselt ainult üks neist variantidest korraga. Mõlema jaoks on vaja lisa mooduleid. Üldiselt on STM32 ja Nordic arendamine võrreldes Esp32

keerulisem. STM32 programmeeritakse tavaliselt ST-Link seadmega, Nordic kontrollerid on aga üldiselt kallimad ning nende turusuurus on väiksem. Kuna STM32 ja Nordic mikrokontrollereid kasutatakse rohkem ärilahendustes ja Esp32 rohkem hobiprojektides, leiab internetist rohkem abistavat materjali Esp32 seotud probleemide kohta. Kuna Esp32 kontrolleritele on laialt levinud Arduino.h teek, siis on paljudele turul eksiteerivatele seadmetele kirjutatud teegid, mis on lihtsasti integreeritavad Esp32 projektidesse.

Arvestades projekti skoopi ja ajalist piirangut, on selge eelis Esp32 mikrokontrolleritel. Kuid Esp32 mikrokontrollerid jagunevad veel omakorda eri seeriatesse. Järgnevalt on vaja valida konkreetne Esp32 mikrkontroller. Kuna trükkplaadil on tähtis hoida ruumi kokku, võrreldakse omavahel väikeseid versioone “ESP32-XX-MINI”.

Tabel 2. Esp 32 mikrokontrollerite võrdlus [11] [12] [13] [14]

	S2-MINI	S3-MINI	C3-MINI	C6-MINI
Ilmus aastal	2020	2021	2021	2023
Taktsagedus [MHz]	240	240	160	160
Flash [MB]	4	Kuni 8	4	4
Tuumade arv	1	2	1	1
Suurus [mm]	15.4×20.0×2.4	15.4×20.0×2.4	13.2×16.6×2.4	13.2×16.6×2.4
Bluetooth	-	-	BLE 5.0	BLE 5.3
WiFi	WiFi 4	WiFi 4	WiFi 4	WiFi 6
Zigbee, Thread, Matter	-	-	-	Olemas
Unerežiim, RTC mälu + timer ON [µA]	25	8	5	7

Tabel 2. peal on näha Esp 32 mikrokontrollerite võrdlusi. Võimekuse poolest on ilmselge võitja S seeria kontrollerid, kuid käesoleva projekti raames on 160 MHz taktsagedus täiesti piisav ning mitmetuumalisus pole oluline. Põhjuseks on, et süsteem

hakkab töötama aku toitel ning mõõtma ainult teatud aja intervallide tagant ja suurt andmetöötlust toimuma ei hakka. Üldiselt mida kõrgem on taktsagedus, seda suurem on ka energia kulu. Kõikidel kontrolleritel on WiFi funktsionaalsus, kuid S-Seeria kontrolleritel puudub Bluetooth.

Bluetoothi olemasolu on vajalik, et saaksime pärida kasutaja esimesel ühendamisel koduvõrgu andmeid.

Lisaks on S seeria kontrollerid suuremad kui C seeria omad, seega saame välistada S seeria kontrollerid lõplikust valikust ning alles jäävad vaid ESP32-C3-MINI ja ESP32-C6-MINI.

C3-l on võrreldes C6-ga on vanem BLE versioon, BLE 5.0 aastast 2016 ja BLE 5.3 aastast 2021 samuti toetab C6 IEEE 802.11a/x standardit võrreldes WiFi 4 IEEE 802.11b/g/n-ga, kuigi C6 WiFi raadiosagedused on maksimaalselt 2.4 GHz, 5 GHz toetust pole, on täidetud mõned teised IEE 802.11a/x standardile vastavad funktsionaalsused nagu OFDMA (*Orthogonal Frequency-Division Multiple Access*), MU-MIMO (*multi-user, multiple input, multiple output*), kuid mis projekti kontekstis kõige tähtsam on TWT (*Target Wake Time*), mida WiFi 4 ja C3 mikrokontrolleril pole. Kuna mikrokontrolleril kõige energiakulukam operatsioon on läbi võrguliidese andmete saatmine [13] [14], siis TWT võimaldab seadmel raadioliidese sisse lülitada ainult siis, kui tal on vaja võrguga suhelda. Lisaks on ruuter võimeline kokku leppima aja seadmetega, millal nad andmeid vahetavad. TWT pakubki just kõige rohkem kasu väiksema aku mahuga nuti seadmetele, suurendades nende aku kestvust [15] [16].

Võib tunduda, et ESP32-C6-MINI on igatpidi parem kui ESP32-C3-MINI, kuid C6 on väga uus toode, mis üldiselt on hea näitaja, kuid tarkvaraarenduses on tihti lugu vastupidine. Siiamaani puudub Arduino *stable release,* ehk stabiilne raamistiku toetus ESP32-C6-MINI jaoks, mis tähendab, et nii süsteemi taseme kuid ka liideste draiverid või teegid võivad olla vigased ja mitte alati töötada nii nagu peavad. Espressifi arendajate sõnul on toetus tulemas koos järgneva tarkvaraga: Arduino-ESP32 v3.0.0

koos ESP-IDF v5.1, kuid Arduino-ESP32 v3.0.0 arendus lõputöö kirjutamise ajal oli veel staatusega „*Pre-release*“ ehk eelväljalase. Seetõttu osutus mikrokontrolleri lõplikuks valikuks ESP32-C3-MINI, isegi, et C6 on igatpidi parem.

5.2 Sensorite valik

Kuna kokku on vaja 4 erinevat sensorit, valguse sensor, õhuniiskuse sensor, temperatuuri sensor ja patarei parameetreid jälgimiv sensor. Eesmärk oli leida sensorid, mis kõik on liidestatavad I2C (Inter-Integrated Circuit) protokolliga, et nad saaksid kõik olla ühel siinil ning andmete lugemine on mugavalt teostatav. Valgussensorite võrdlust on näha Tabel 3. peal.

Tabel 3. Valgussensorite võrdlus [17] [18] [19]

	MAX44009	VEML7700-TR	LTR-F216A
ADC resolutsioon [bitti]	22	16	16-20 efektiivne
Mõõtmisvahemik [lx]	0.045 – 188 k	0 – 140 k	* kuni 39 k
Unerežiimi toide [µA]	0.65	2	1
Hind 1 ühik [20] [€]	-	1.89	12.05

MAX44009-t enam ei toodeta, valiti VEML7700, kuna ta oli kõige odavam.

Niiskussensoriks valiti SHTC3, kuna ta oli ainuke niiskussenor, mis Teval elektroonika poes ostetav oli. Ikkagi vastab ta kõikidele kriteeriumitele, unerežiimi tarbimine on vaid

* Andmelehes pole märgitud, maksimaalset mõõtevahemikku, kindlasti on see tegelikkuses suurem kui 39 klux

0.3 µA, tal on I2C liides, väikest mõõtu, lisaks suudab ta mõõta ka temperatuuri. Mõõtmise täpsused: niiskus ±2 %RH ja temperatuur ±0.2 °C [21].

LiPo aku parameetrite jälgimise sensorit Teval elektroonika poes polnud müügil, seega see tuli tellida koos trükkplaadiga Hiinast, JLCPCB tütarfirmast LCSC-st. LiPo aku jälgimissensorite võrdlust on näha Tabel 4. peal.

Tabel 4. LiPo aku jälgimissenorite võrdlus [22] [23] [24]

	MAX17048	BQ27441	MAX17260
Kuloni luger	Ei	Ei	Jah
Väljaviigute arv	8	12	14
Unerežiimi toide [µA]	0.5	9.1	5.1
Teekide olemasolu	Jah	Jah	Ei
Hind 1 ühik [25] [€]	1.15	1.38	3.50

LiPo aku jälgimissensoriks valiti MAX17048. Tootel on kaua aega AdaFruiti poolt arendatud teek, väike energiatarve, suurus on väiksem ning hind on ka kõige parem.

5.3 Muud komponendid

Selles peatükis seletatakse ülejäänud komponentide otstarbed, nt pingeregulaator, aku laadimise integraallüliti, mõni kaitse otstarbega komponent jm. Need komponendid on vajalikud, et kogu süsteem üldse töötaks.

1. **Pingeregulaator**

   Vajalik, et nii akust 3,7 V kuni 4,2 V või läbi USB tulev 5 V oleks reguleeritud 3,3 V peale, mis on ühtlasi Esp32, kui ka teiste seadmete toitepingeks.

2. **USB komponendid**

   Et läbi USB oleks võimalik programmeerida, on vaja USB-jadaliidese muundurit ja transistoride paari, et programmeerimine oleks automaatne ja ei peaks käsitsi mikrokontrollerit programmerimise režiimi lülitama näiteks nuppudega. Lisaks on vaja ka kaitset elektrostaatilise laengu vastu ja lõpuks USB konnektorit.

3. **Patarei laadimise komponendid**

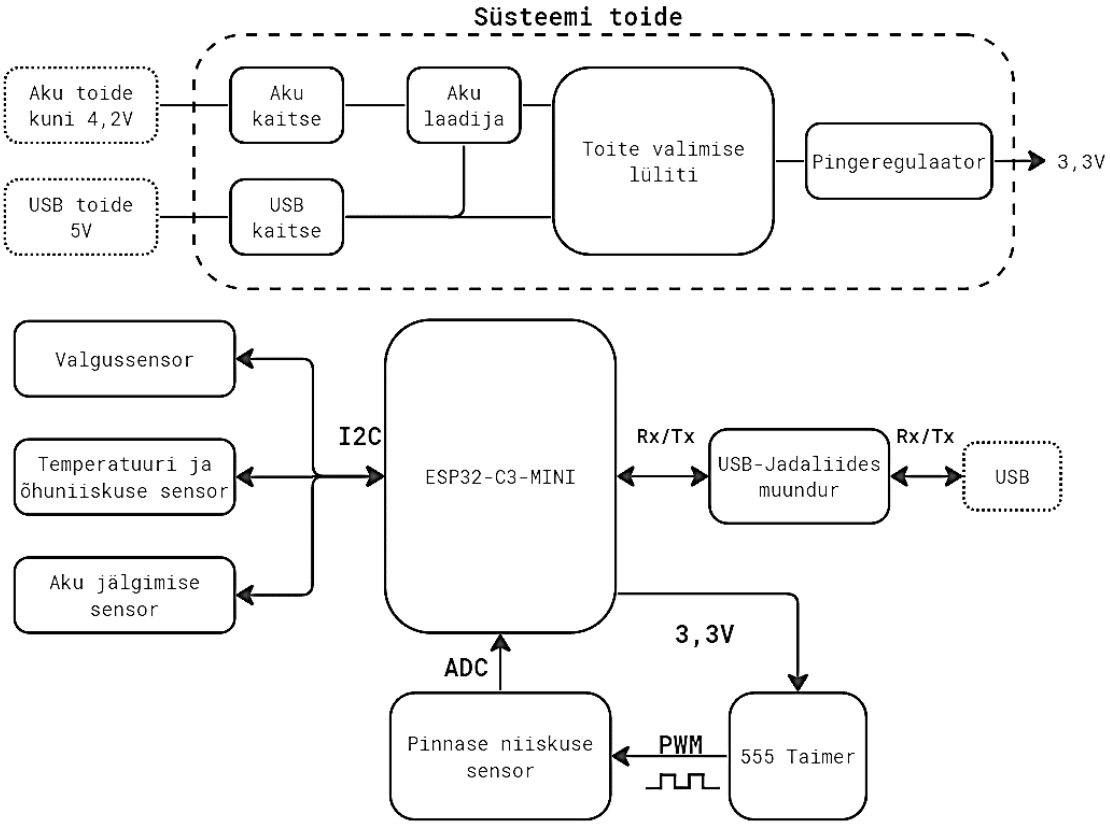
   Aku laadimise integraallüliti laeb akut, kui USB on ühendatud. Lisaks laadimise ja tühjaks laadimise kaitse lülitid ja MOSFET-id.

4. **555 Taimer lüliti**

   Taimer lüliti eesmärk on tekitada PWM (Pulse-Width Modulation) signaal, mida kasutatakse ära, et lugeda pinnaseniiskuse sensori väljundit. Pinnaseniiskuse sensori tööpõhimõtet koos 555 taimeriga on täpsemalt seletatud peatükis 6.2.

6 Riistvara lahendus

Joonisel 4. on üldistatud riistvaraline plokkskeem, mis on tehtud lõpliku elektriskeemi alusel, mida on näha LISA 2 peal.



Joonis 4. Riistvara plokkskeem

Süsteem võib saada toidet kas akust või läbi USB. Lõpptootes on mõeldud, et akut kasutaja enam lahti ei ühenda ja aku juhe on konstantselt ühendatud. Ainult arenduse käigus on "võimalik" lahti/kinni ühendada. Kui aku on piisavalt laetud ning ühendatud, lubab LISA 2. "Power path and LDO" osas Q1 P-MOS transistor pingeregulaatori sisendisse vaid aku toite, seega süsteem saab ka toite läbi aku. Kui on ühendatud vaid USB, siis saab süsteem toite läbi USB. Kui mõlemad on ühendatud, ei lase P-MOS akut pingeregulaatori sisendisse ja süsteem saab enda toite läbi USB, samaaegselt laetakse "Battery charging/protection" osas läbi IC2 TP4056 akut ~800mA-ga. Laadimisevool on reguleeritav R6 takistiga ning takisti laadimisvoolude konfiguratsiooni leiab samast osast sinisest kastist. Eelnev tagab, et süsteem töötaks samamoodi edasi, kui seadet on vaja laadida.

Kõik sensorid, va pinnnase niisksussensor on ühendatud mikrokontrolleriga läbi I2C liidese. I2C oli eelistatav liidestamise viis, sest ühe siini peal saab kasutada kõiki sensoreid. Sensorid saavad samuti oma toite pingeregulaatori 3,3V väljundist. Aku jälgimise sensor peab saama enda toite otse akult, muidu poleks võimalik sensoril aku parameetreid lugeda.

Osas "USB Power and USB-Serial converter" on transistorite paar Q3 ja Q4 vajalik, kui mikrokontrollerile tahetakse läbi USB laadida peale uus programm, siis transistorid lülitavad mikrokontrolleri "programmerimiserežiimi". Sisuliselt tehakse kontrollerile reset ja käivitatakse alglaadimine, mis lubab mikrokontrolleri programmimälu üle kirjutada.

6.1 Trükkplaadi disain

Trükkplaadi disaini puhul prooviti võimalikult palju kinni hoida teatud hea disaini tavadest:

1. Komponentide paigutus

   Trükkplaadi disaini puhul, on tähtis ruumi kokku hoida ja paigutada komponendid võimalikult kompaktselt. Konkreetset funktsionaalsust täitvad komponendid peaksid võimalusel asuma üksteisele lähestikku. Ühendatavad konnektorid, nagu USB ja aku peaksid olema trükkplaadi äärel. Mürale tundlikud osad, nt trükkplaadi antenn peaks olema samuti üle ääre, et teised komponendid ja trükkplaadi vasekiht interferentsi ei tekitaks.

2. Radade laiused vastavalt funktsioonile

   Toite rajad peaksid olema laiemad, sest laiemate radade puhul on takistus, pinge lang ja võimsuse kadu väiksemad. Signaali rajad võivad olla kitsamad.

3. Radade nurgad

   Rajad võiksid pöörata maksimaaltselt 45 kraadiste nurkade all, et vähendada signaali peegeldumist – tootja poolt võivad teravad nurgad põhjustada trükkplaadi lahti koorumist. Tervad nurgad võivad käituda kui antennid ja tekitada seetõttu elektromagnetilist interferentsi.

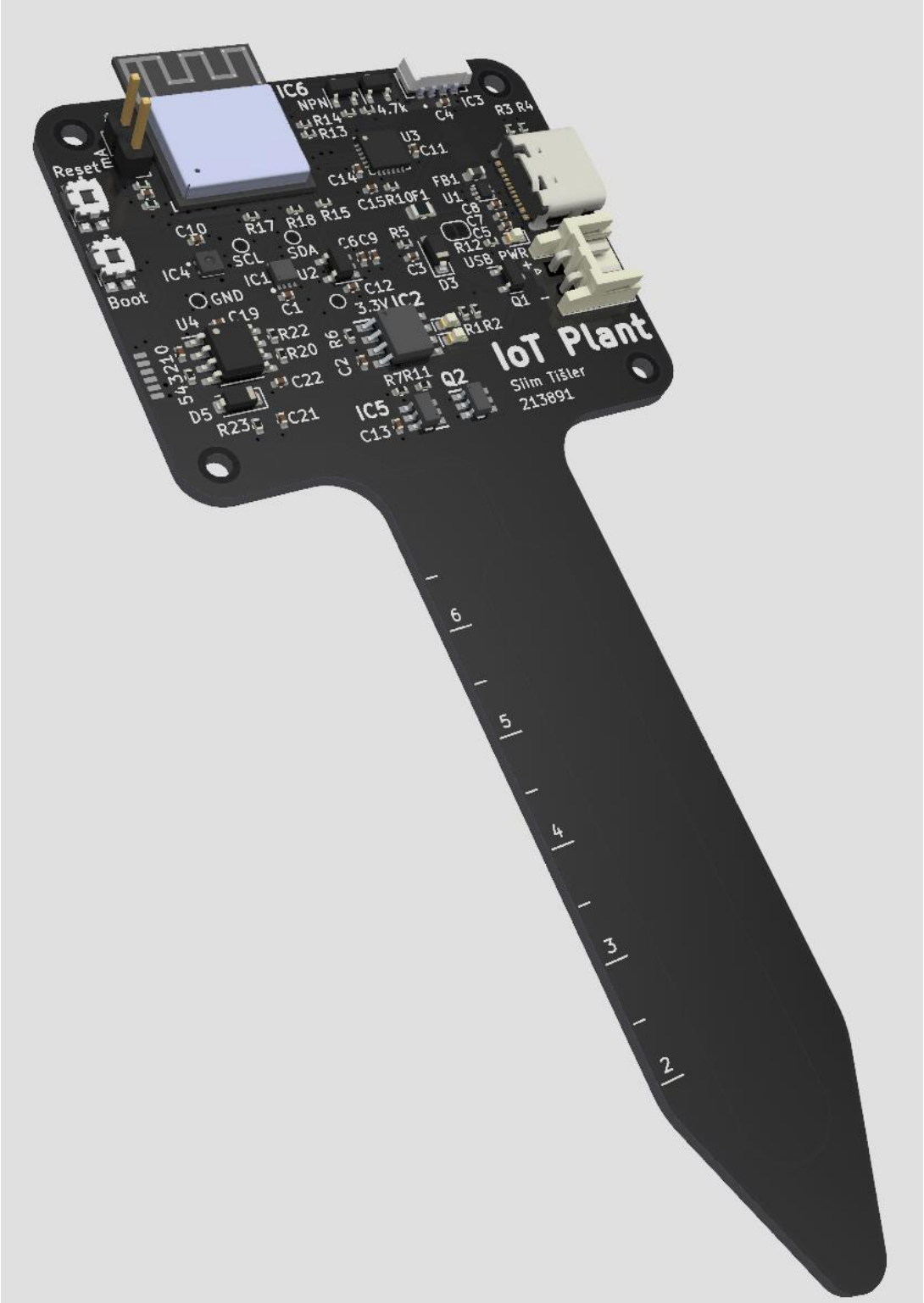
4. Radade ja komponentide vahelised kaugused ja vaba liikumisruum

   Kaugused peavad olema kooskõlas trükkplaadi tootmis-ettevõtte miinimum nõuetega, sellega välditakse lühiseid ja tagatakse tootmise kvaliteet.

5. Testitavus ja parandamine

Testimise eesmärgil on mõistlik lisada trükkplaadile mõõtepunkte. Paranduse jaoks radadele paigutada läbilõike kohti ja potentsiaalselt asendatavad komponentid võiksid olla suuremad, et soovi korral oleks ümber jootmine lihtsam. Lisaks kasutamata väljaviigud, nt mikrokontrolleri ***GPIO***-d (General Purpose Input Output) teha kättesaadavaks,

Trükkplaat on tellitud Hiina trükkplaatide tootmis ettevõtte kaudu JLCPCB. Läbi JLCPCB on võimalik valida teatud tootmise tulemused, näiteks trükkplaadi värv, täitematerjal, millist jootematerjali kasutatakse pinnaviimistluseks jne. Tellitud trükkplaat on 2 kihiline, baasmaterjal on FR-4, plaat on 1,6mm paks ja kasutatud tinavaba jootematerjali. JLCPCB pakub ka komponentide jootmise teenust. Projekti käigus on jootmise teenust kasutatud enamus passiivkomponentide jaoks, lisaks mõned LED-id ning teised üksikud komponendid. Projekti jaoks valitud komponentide seast pole JLCPCB jaoks kõik nö “standard komponendid”. Iga mitte standard komponendi jootmine läheb maksma lisa raha, selleks on need komponendid tellitud Euroopa laost ja joodetud ise, et vähendada kulusi. Lõplikku trükkplaadi renderdust on võimalik näha Joonis 5. pealt.



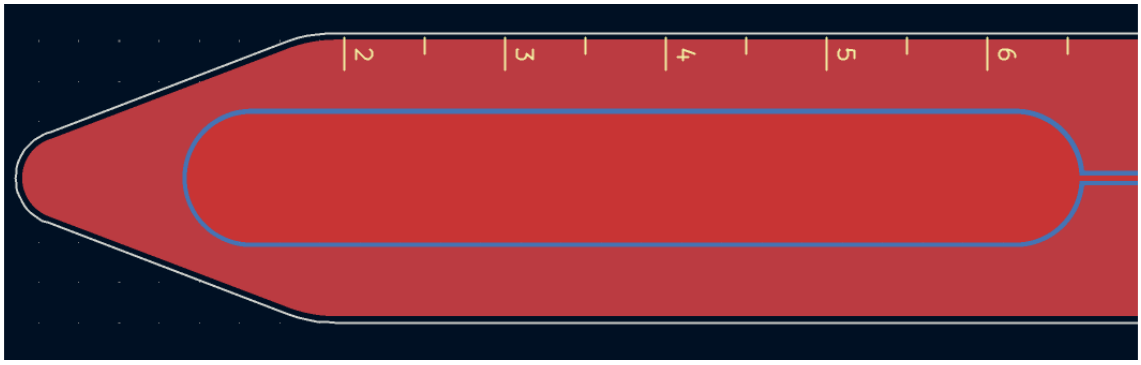
Joonis 5. Trükkplaat koos komponentidega

6.2 Pinnase niiskussensor

Pinnase niiskussensor ei ole osetetud, see luuakse eri komponentide ning trükkplaadi mõõteotsa koostoimel. Pinnase niiskussensor töötab sisuliselt keskkonna dielektrilise läbitavuse konstandi muutumise lugemisel.

$$C = \varepsilon_0 \varepsilon_r \frac{A}{d}$$

Valemis on ε_r suhteline läbitavus, mis muutub vastavalt keskkonnale. Terve mõõteots ongi kondensaator mida suhteline läbitavus mõjutama hakkab. Kuna mööda mõõteotsa liigub suhteliselt lai signaali raja, mis on ümbritsetud maa kihiga, tekib laia raja ja maa kihi vahele parasiitmahtuvus vt Joonis 6.



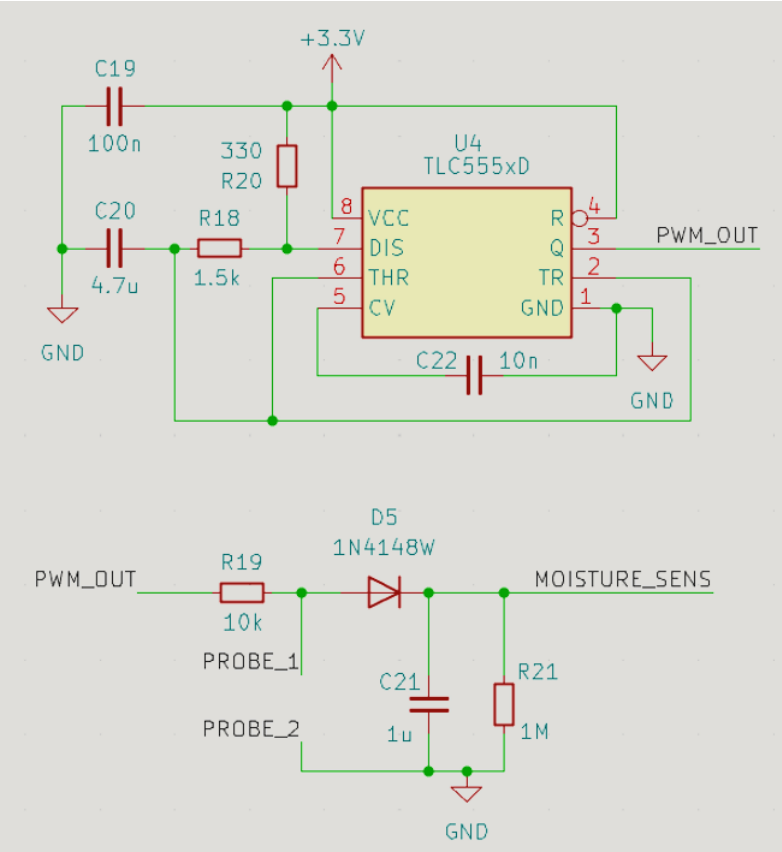
Joonis 6. Parasiitmahtuvuslik mulla niiskuse mõõteots

Trükkplaadi küljed pole üle värvitud ja sealt võib vesi sisse imbuda ja süsteemi kahjustada. Trükkplaadi mõõteots tuleb teha veekindlaks, selleks võib kasutada näiteks küünelakki või mõnda elektroonika jaoks kõlbulikku veekindlat aerosooli.

Tabel 5. Suhtelised dielektrilsied läbitavused [26] [27]

Materjal	Suhteline dielektriline läbitavus [F/m]
Õhk	1
Mulla mineraalid	3-7
Orgaanilised materjalid	2-5
Kuiv mulla pinnas	5
Kuiv liiva pinnas	4-6
Vesi	80

Nagu näha Tabel 5. peal, siis mulla sees olevate mineraalide ning orgaaniliste materjalide dielektrilised läbitavused on vee dielektrilise läbitavuse suhtes väikesed, seega vee kontsentratsioon mõõteotsa ümbritsevas keskkonnas muudab mõõtetulemust ka kõige rohkem. Muidugi ei saa välistada, et inimesed kasutavad erinevaid mullasegusi, seega pole mõõtetulemused reaalse 0% vee kontsentratsiooni juures samad. Kuid see erinevus on minimaalne, sest üli täpne vee kontsentratsiooni mõõtmine ei ole projekti seisukohast kriitiliselt oluline.



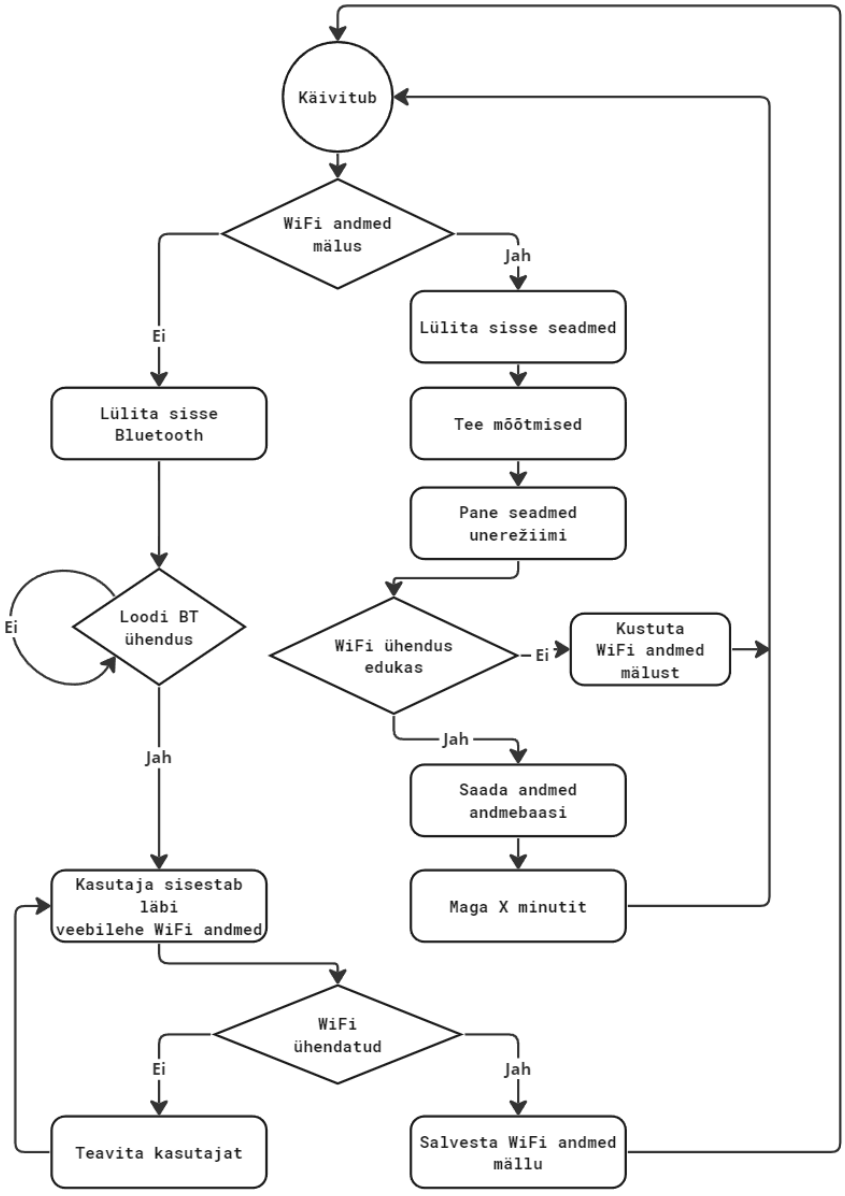
Joonis 7. Pinnase niiskussensori elektriskeem

Joonis 7. peal 555 taimer seadet kasutatakse ostsillaatorina, mis genereerib nelinurk signaali sagedusel ca 500 kHz. Üldiselt on mahtuvussensorid vähe tundlikud soolale võrreldes takistuslike sensoritega, kuid siis peab ostsilaatori genereeritud signaal olema üle 50 MHz [28]. Kuna 555 taimer nii suure sagedusega signaali väljundisse anda ei suuda, siis on sensor samuti natukene rohkem tundlik sooladele.

Joonis 7. Alumises osas võib kujutada ette, et “PROBE_1” ja “PROBE_2” vahel on kondensaator, sest sisuliselt ongi. R19 ja kondensaator moodustavad RC ahela. Nelinurk signaali kõrge osa ajal laetakse kondensaator täis ja madala signaali osa ajal tühjaks. Kondensaatori mahtuvus suureneb, kui suureneb dielektriline läbitavus, mis tähendab, et RC ahela lõikesagedus väheneb ja kondensaator laeb aeglasemini täis, selle tagajärjel RC ahela väljundsignaali tipust tipuni väärtus väheneb. RC ahela väljund läheb seejärel läbi alaldi, mis omakorda silutakse läbi kondensaatori, alles jääb alalissignaal. See alalissignaal signaal läheb lõpuks mikrokontrolleri ADC (*Analog to Digital Converter)* sisendisse “MOISTURE_SENS”, et loetud alalispinge väärtuse mulla niiskuse sisalduseks saaks konverteerida.

7 Seadme juhtloogika

Peatükis on kirjeldatud ning visuaalselt näidatud seadme käitumise diagramm ehk juhtloogika, et saaks selge arusaama, seadme üldisest töövoost vt Joonis 8. Diagramm on lihtsustatud kujul ning ei sisalda endas veakontrolle jm äärejuhte.



Joonis 8. Seadme üldistatud juhtloogika diagramm

Kui esmalt seade käivitub ja nö “tehasest tulnud”, siis on tal püsimälu tühi ning kasutajal on vaja lisada seadmele WiFi seadmed. Selle jaoks tuleb sisse lülitada Esp32 sisse ehitatud BLE moodul. Läbi veebiliidese on võimalik kasutajal luua seadmega Bluetooth ühendus ning seejärel saata seadmele WiFi SSID, ehk nimi ja WiFi parool. Kui WiFi ühendust luua ei suudeta, siis ei juhtu midagi ja kasutajal on võimalik andmed korrektselt uuesti sisetada. Kui sisestatud andmed on õiged, luuakse WiFi ühendus ja salvestatakse WiFi andmed püsimällu, et järgmine kord saaks ühenduse luua automaatselt.

Seega kui WiFi andmed on mälus, siis lülitatakse sisse/äratatakse kõik sensorid, tehakse nendega mõõtmised ning seatakse tagasi unerežiimi. Nüüd luuakse ühendus WiFi ja andmebaasiga ja mõõdetud andmed saadetakse andmebaasi. Kui kõik on ära saadetud pannakse ka Esp mikrokontroller teatud ajaks unerežiimi.

8 Andmemudel

Joonis 9. pealt on näha illustreerimise eesmärgil vaid väikest tükki tervest andmemudelist, et joonis ei oleks liiga kirju ja andmemudelist oleks kergem aru saada.

Andmemudel koosneb JSON (*JavaScript Object Notation*) objektidest. Esimene JSON objekt on „user“, neid objekte on sama palju kui on kasutajaid. Ning iga kasutaja saab omada mitut erinevat IoT taime sensori seadet, milleks on objektid „device“.

Iga seade vastutab vaid endale konkreetse „device“ objekti eest, ehk üks seade vastab konkreetsele JSON objektile andmebaasis. Iga device objekti poolt saadetavad andmed on järgmised:

- **wifi_last_connected**

  Tegemist on WiFi ajatempliga, mida saadetakse teatud aja tagant, et teada millal WiFi viimati ühenduses oli. Hiljem saab seda kasutada, et kasutajale teada anda, kas seade on WiFi võrku veel ühendatud.

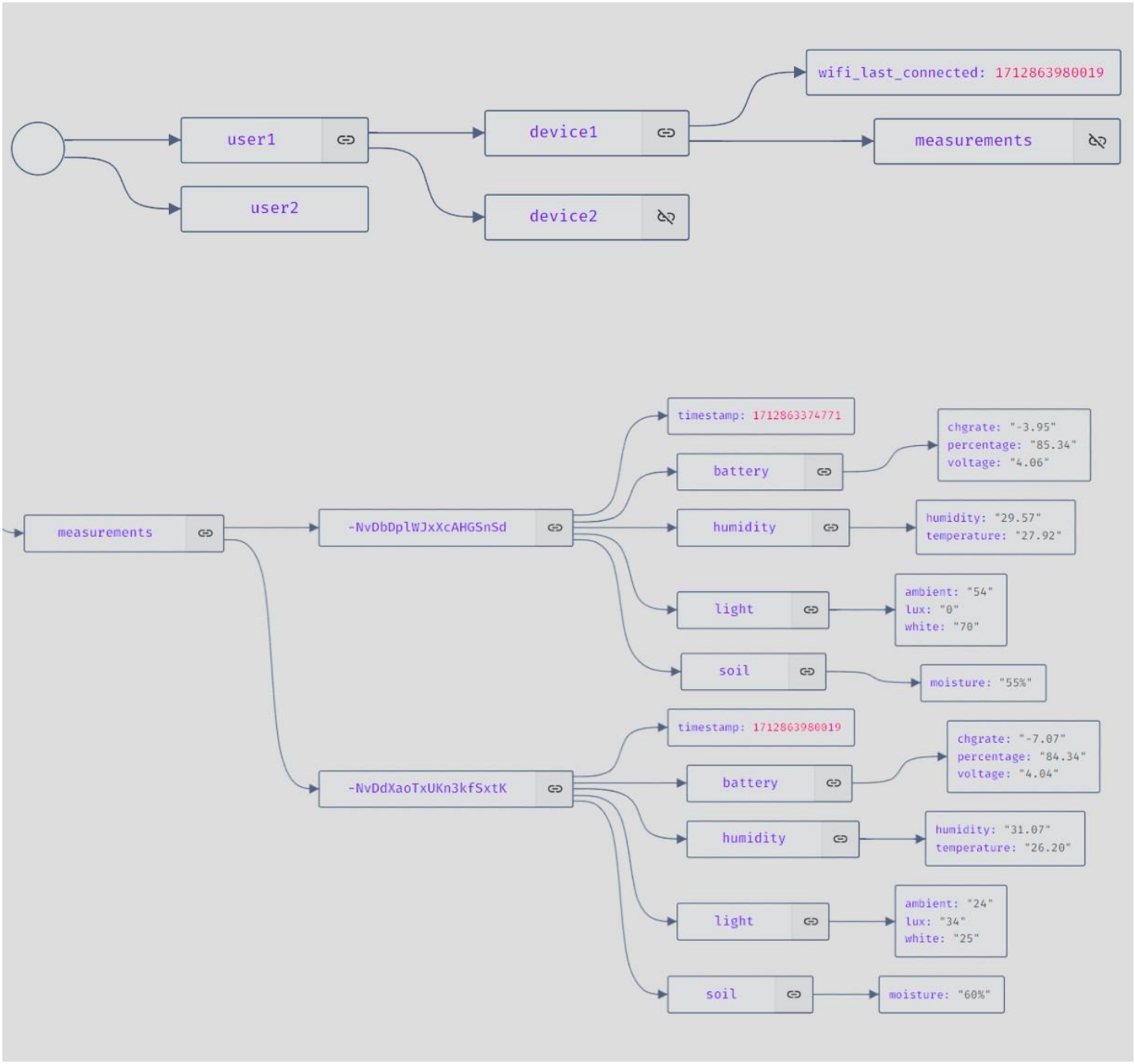
- **measurements**

  See objekt sisaldab endas mitmeid alamobjekte, millel igalühel on oma unikaalne identifikaator, nt „NvDdXaoTxUKn3kfSxtK“. Üks selline objekt sisaldab ühe tsükli mõõtmistulemusi.

  - **timestamp –** Ajatempli arvuline väärtus, et veebilehel saaks konkreetsel ajal tehtud mõõtetulemuse graafikul kuvada.

  - **battery –** Aku sensori objekt, mis sisaldab endas aku sensori mõõdetud väärtusi kujul. Mõõtetulemused on arvulisel kujul. Objektist saab mõõdetud aku pinge, aku laetuse protsendi ja tühjaks/täislaadimise määra.

- **humidity** – Õhuniiskuse ja temperatuuri sensori objekt, mis sisaldab temperatuuri ja õhuniiskuse väärtusi arvulisel kujul
- **light** – Valgussensori objekt, mis sisaldab hajus- ja valge valguse mõõtmisetulemusi ning ka valgus intensiivsust ühikus lux, ehk luumenit ruutmeetri kohta.
- **soil** – Pinnase niiskuse sensori poolt mõõdetud pinnase veesisaldus protsent.



Joonis 9. Andmemudel graafidena

9 Sardtarkvara lahendus

Selles peatükis kirjeldatakse, milliseid väliseid teeke kasutati ja millised ise loodi, milline on üldine koodi struktuur, kuidas sardtarkvara koodi kirjutamise põhimõtteid rakendati. Lisaks kus tekkis raskusi, vigu ning kuidas need parandati ja optimeeriti.

9.1 Struktuur

Kood on jagatud eraldiseisvatesse komponentidesse/moodulitesse, kus iga komponent vastutab vaid enda valdkonda kuuluvate ülesannete eest. Näiteks eraldiseisvad komponendid on valgussensor ja õhuniiskussensor. Selline struktureertitud lähenemine teeb koodi loetavamaks, kergemini hallatavamaks, korduvkasutatavaks ning vigu on kergem leida. Ühe konkreetse komponendi all mõeldakse üldiselt ühte teeki. Võttes jälle näiteks valgussensori, oleks tema teek “lightsensor.h”, kus on defineeritud kõik tavamõistes *public* funktsioonid, muutujad, struktuurid, mille skoopi peavad teised teegid, failid nägema. Teegid mis on “lib” kaustas lingitakse PlatformIO raamistiku poolt automaatselt.

9.2 Komponendid ja teegid

Järgnevalt kirjeldatakse iga komponendi tööd, millistest lisa teekidest ta sõltub ning mis on tema tööpõhimõte.

9.2.1 Mälu

Teek “nvsmemory.h” võimaldab andmeid salvestada ja kätte saada mikrokontrolleri põhimälust. Andmete hoiustamise jaoks kasutatakse Flash mälu piirkonda. Tegemist on *NVS (Non-Volatile Storage)* mäluga, seega mikrokontrolleri mälu ei kustu, kui seade välja lülitatakse. Andmed salvestatakse võti-väärtus paaridena. Teek sõltub veel omakorda kolmandast teegist “Preferences.h”, mis vastutab madalama kihi *Flash* mälu liidestamisega.

Eesmärk seisneb, et saada kätte ja salvestada eelnevalt päritud WiFi andmed. Sisuliselt on tegu *getter* ja *setter*-itega, nt “memoryGetSSID”, mis lihtsalt tagastab mällu salvestatud SSID. “memorySetSSID” funktsiooni sees on staatiliselt võtme nimi, nt “wifi_ssid” ja parameetrina antakse mis väärtus võtmega seostada. Sellised funktsioonid vähendavad vigu, koodi kordust ja parandavad loetavust.

9.2.2 Sensorid

Kuna sensorite moodulid jälgivad üldiselt väga sarnast loogikat, siis on nende kirjeldused kõik ühe alapeatüki all.

Kõik autori poolt kirjutatud sensori teegid sõltuvad omakorda madalama kihi draiver teekidest, mis on kirjutatud varasemalt kolmandate isikute poolt ja käesoleva projekti koodi integreeritud.

Aku sensor, õhuniiskus/temperatuuri sensor ning valgussensor sõltuvad vastavalt “Adafruit_MAX1704X.h”, "Adafruit_SHTC3.h", "Adafruit_VEML7700.h" teekidest. Viimased Adafruiti teegid veel omakorda sõltuvad “Adafruit_BusIO_Register.h”, “Adafruit_BusIO_I2CDevice.h” ja “Adafruit_BusIO_I2CDevice.h” teegist.

Iga ülemisekihi sensori API (*Application Program Interface*) roll on seadme sisse ja välja lülitamine, unerežiimi sisse/välja lülitamine ning mõõtmiste tegemine ja andmete kättesaamine. Isegi, et iga teegi sisemine osa, mis on nö *private*, ehk osad mis API-st välja ei paista on sensoritel üldiselt erinevad.

```
typedef struct{
    uint16_t lux;
    uint16_t white;
    uint16_t ambient;
} light_sens_measurements_t;

boolean initLightSensor();
boolean powerOnLightSensor();

void lightSensorPowerSaverEnable(boolean);

uint16_t getLux();
uint16_t getWhite();
uint16_t getAmbient();

light_sens_measurements_t getLightMeasurements();
```

Joonis 10. Valgussensori API teegi näide C-keeles

Joonis 10. peal on „lightsensor.h“ sisu, et näidata, millist käekirja peab jälgima iga senori API. Seega samamoodi on implementeeritud ka teised sensorid.

9.2.3 Andmebaas

Teek vastutab kõikide Firebase reaalaja andmebaasi päringute ja saatmiste eest ja muude võrgu ülesannete eest. Omakorda sõltub teek veel „WiFi.h“ ning „Firebase_ESP_Client.h“ teekidest.

Läbi API on võimalik luua WiFI ühendus ja seejärel ühendada Firebase andmebaasi. Teek kasutab sisemiselt ära mälu API-t, et vastavalt esimesel edukal WiFi ühendamisel salvestada WiFi andmed mälusse või unerežiimist ärgates, võtta WiFi andmed mälust ning nende abil siis WiFi ühendus luua. Kui WiFi ühendus loomine peaks ebaõnnestuma, järeldatakse, et seadme asukoht võib olla muutunud ning asub mõnes teises võrgus või on lihtsalt võrgu andmed muutunud. Seega on vaja sisestada WiFi andmed uuesti sisestada – selleks teeb süsteem taaskäivituse ja siseneb WiFi andmete pärimise režiimi.

Kui seade on WiFi andmete pärimise režiimis, eksisteerib API funktsioon, et pärida kõik tuvastatavad WiFi SSID-d. Bluetooth API kasutab seda funktsiooni, et BLE-ga saata veebilehele kõik SSID-d, siis saab kasutaja mugavalt valida enda koduvõrgu SSID ja sisestada ainult salasõna.

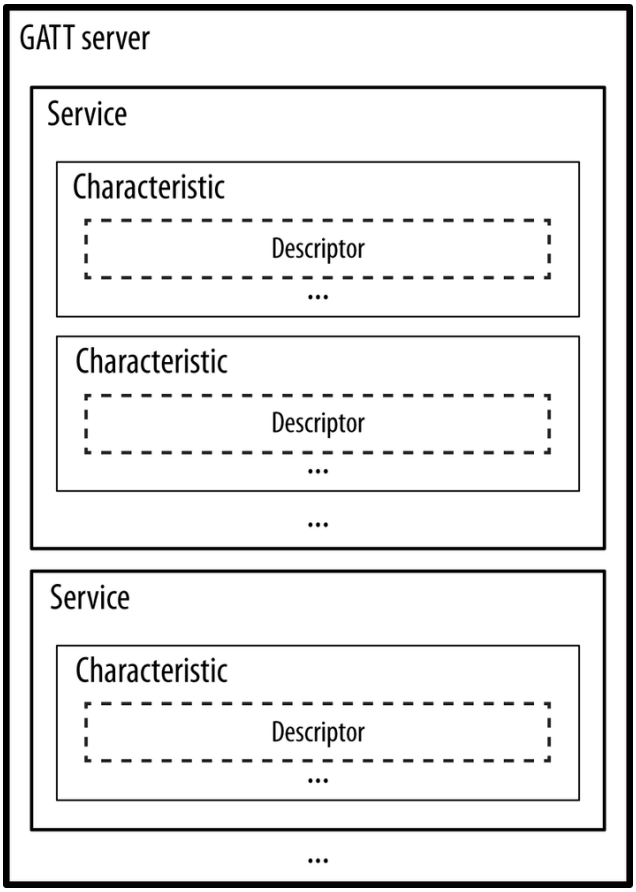
Andmebaasi API funktsioonideks on vaid sensorite andmete saatmine andmebaasi. Sisemiselt teevad funktsioonid sensorite andmestruktuurid samsugusteks JSON objektideks nagu on näha andmemudelil Joonis 9. ning need objektid saadetakse andmebaasi.

9.2.4 Tsükli ülesanne

See API kasutab enamus eelnevalt nimetatud API-de funktsioone, et ühe ärkveloleku tsükli ajal saaks kõik ülesanded tehtud. Eesmärgiks on kõik funktsionaalsused kokku panna ja hoida „main,cpp“ lühike ja lihtne. API funktsioonideks on vaid „powerOnDevices” ja „mainTask“, millest viimase ülesanne on teha esmalt ära kõik mõõtmised sensoritega, seejärel ühendada WiFi ja andmebaasiga ning saata andmed andmebaasi, siis lülitada WiFi moodul välja. Kui funktsioonist väljutakse pannakse mikrokontroller magama 10-ks minutiks, kuni korratakse tsüklit.

9.2.5 Bluetooth

Kolmandad teegid millest Bluetooth API sõltub on: „BLEDevice.h“, „BLEServer.h“, „BLEUtils.h“ ja „BLE2902.h“. API ainukesed funktsioonid on „initBleSerial“ ja „bleSerialTask“. Neid funktsioone kasutatakse vaid siis kui toimub esimene seadme konfigureerimine või kui WiFi ühendus on ära kadunud, eesmärgiga, et kasutaja käest WiFi andmeid pärida. Algul pannakse Bluetooth moodul käima, ning luuakse BLE teenus, kuhu külge pannakse kaks karakteristikut. Seejärel on BLE seade leitav nimega „TaibuTaim“ ning ühendamiseks on vaja teada teenuse ning karakteristikute unikaalseid identifikaator koode. Koodide väärtused on genereeritud kasutades lehekülge www.uuidgenerator.net.



Joonis 11. BLE GATT hierarhia [29]

Joonis 11. peal on näha kuidas BLE teenused, karakteristikud ja nende kirjeldused jagunevad. BLE GATT (Generic Attribute Profile) on Bluetoothi spetsifikatsion, mis määrab kuidas BLE seadmed andmeid vahetavad, kasutades hierarhilist struktuuri, mis koosneb teenusetest, karakteristikutest ja deskriptoritest. GATT töötab klient/server põhimõttel, kus seade ise on serveri rollis. GATT defineerib reeglid kuidas BLE seadmed üksteisega ühenduvad, saadavad, pärivad jne [30]. Süsteemis on 1 teenus, mille küljes on 2 karakteristikut ning teenus tervikuna töötab nagu jadaliides. Võimalik on saata ja vastuvõtta baitidena andmeid. Üks karakteristik vastutab saatmise ja teine vastuvõtmise eest. Lõpuks töötavad mõlemad karakteristikud koos, moodustades lihtsa olekumasina. Olekumasin käitub järgmiselt – kui luuakse esimene ühendus saadetakse kõik SSID-d veebilehele. Kui kasutaja on ühendatud ja läbi BLE jadaliidese saatnud WiFi andmed, proovib süsteem WiFi ühenduse luua. Edukal ühendusel antakse

kasutajale läbi BLE jadaliidese märku, lülitatakse Bluetooth moodul välja, salvestatakse WiFi andmed ning tehakse restart. Peale restarti algab tsükli ülesanne vt peatükk 9.2.4. Kui WiFi ühendust luua ei suudeta, antakse jällegi läbi BLE jadaliidese sellest teada kasutajale ning kasutaja saab WiFi andmed soovi korral uuesti sisestada või BLE ühenduse katkestada.

9.3 Aku kasutuse optimeerimine

Valdav enamik aku toitel töötavate seadmete puhul on väga hea kui aku peab vastu võimalikult pika aja. Kõige lihtsam viis kuidas aku vastupidavust saab tarkvaraliselt optimeerida, on unerežiimis veedetud aeg teha võimalikult pikaks ja ärkvel olemise aeg võimalikult lühikeseks. Seega on tarvis, et programm teeb enda kõik üksikud ülesanded võimalikult kiiresti ära, sest unerežiimis oldud aeg on konstantselt sama ning parandada on võimalik vaid ärkvel oleku aega.

Esimene asi mida tuleb kindlasti vältida on viitefunktsioonid, mille ülesanne nagu nimi ka ütleb, on oodata teatud aeg. Selle aja jooksul mikrokontroller enamus muid ülesandeid täita ei saa, mis tähendab, et funktsioon on olemuselt blokeeriv. Viitefunktsioonid on mõningates kohtades vajalikud, näiteks sensori käivitamise ja andmete lugemise vahel olevad viited, kuid valitud viited on üldiselt minimaalsed.

Teine optimeerimise põhimõte seisnes selles, et jagati kõik ühe tsükli ülesanded alamülesanneteks, näiteks WiFi ühenduse loomine, valguse mõõtmine, temperatuuri mõõtmine, andmete saatmine jne ning vaadati milline alamülesande täitmiseks kulub kõige rohkem aega, et saaks ajakulukaid ülesandeid kiiremaks teha.

Tabel 6. Alamülesannete jaoks kulunud aeg

Mulla niiskus	Valguse sensor	Õhuniiskus ja temp	Aku parameetrid	WiFi ühendus *	Firebase ühendus *	Andmete saatmine *	Kokku
200 ms	700 ms	25 ms	5 ms	Kuni 4500 ms	Kuni 900 ms	Kuni 600 ms	Kuni 6930 ms

Tabel 6. peal on näha alamülesanneta jaoks kulunud aega. Mõnele mõõtmisele tabelis on * juurde lisatud. Kõik need mõõtmised on kuidagi seotud võrguühendustega. Mõõtmistulemuseks on võetud sellepärast ka maksimaalne mõõdetud aeg, mis konkreetse alamülesande jaoks kulus. Ülejäänud alamülesannetele kulunud aeg on väga väikeste kõikumistega võrreldes võrgu ülesannetega. Sensorite mõõtmiste lugemise puhul on tegemist ainult trükkplaadil toimuvate digitaalsete signaalidega, mis tähendab, et sama ülesande täitmiseks kulub peaaegu alati sama palju aega.

Mulla niiskuse mõõtmise puhul peab ootama teatud aja enne konkreetset lugemist, selle vajadus tuleb RC ahela aja konstandist *tau*.

$$\tau = RC = \frac{1}{2\pi f_c}$$

Kuna trükkplaadil paksu rajaga tekitatud kondensaatori parasiitmahtuvust on keeruline mõõta ja täpset *tau*-d arvutada, siis pandi ooteajaks 200ms enne ADC-ga mulla niiskuse lugemist. Selle numbrini jõuti puhtalt katse-eksitus meetodil. Peale 200ms pinge enam ei kasvanud, kui mullasensor oli staatiliselt samas keskkonnas. Seega 200ms on nö süsteemi nõue, sest peale seda on mulla niiskuse mõõtmine kõige täpsem.

Valguse sensoriga on saranane olukord, kus iga mõõtmise vahel on integratsiooni aeg, mis tagab kõige täpsemad mõõtetulemused [18]. Kui integratsiooni aja vajadus välja

lülitada, mõõdetakse valgus umb 5ms kiirusega. Täpsuse eesmärgil on integratsiooni aeg sisse jäetud.

Kõige ajakulukam ülesanne oli WiFi ühenduse loomine, mis võis võtta aega kuni 4,5 sekundit. See on üsna kriitiline ülesanne, sest WiFi mooduli voolutarbe tipud on kuni 350mA [13] ja WiFi moodul tarbib keskmiselt kõige rohkem süsteemis, ehk WiFi ühendamine peaks ideaalis kiirem olema ning WiFi moodul peaks töötama võimalikult lühikest aega.

Et WiFi moodul oleks kasutuses võimalikult vähe, tuleb teha sensoritega mõõtmised ära ja alles siis lülitada sisse moodul. Kui kõik andmed on edukalt ära saadetud, lülitatakse moodul kohe välja.

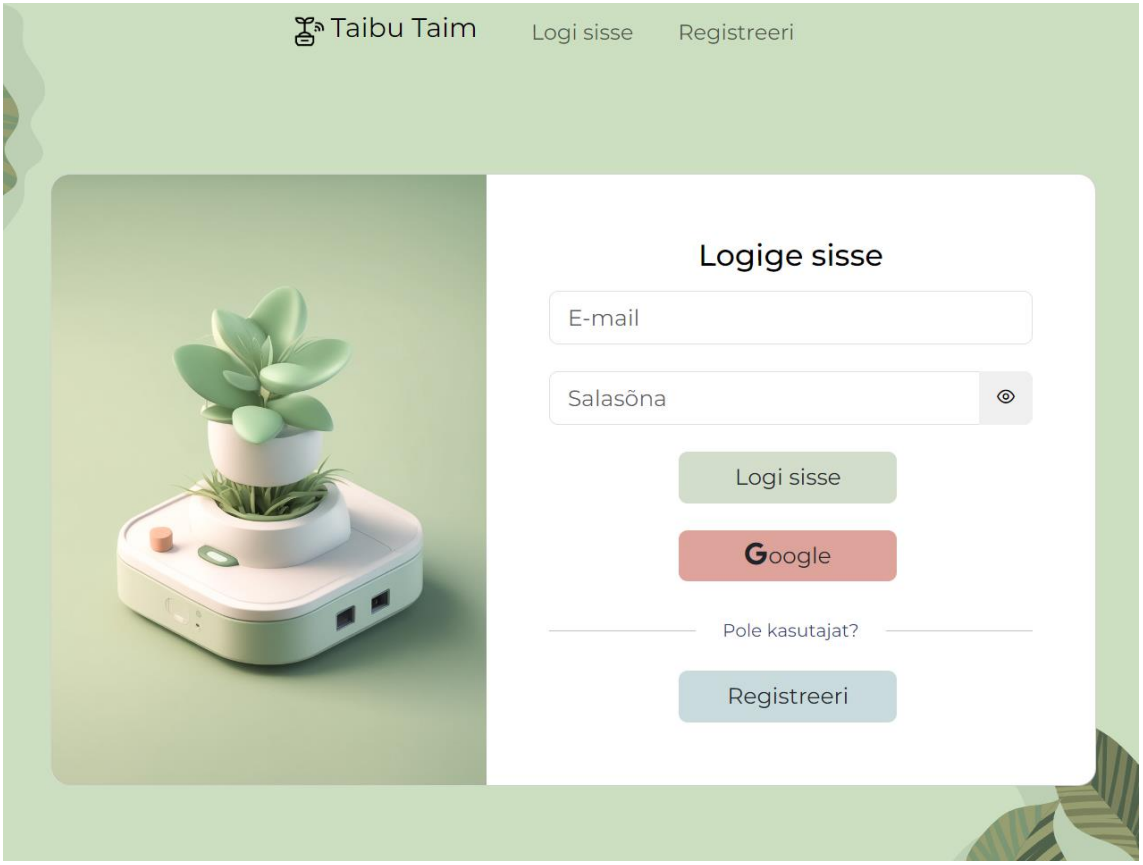
WiFi ühendamiseks kasutatakse dünaamilist IP (*Internet Protocol*) aadressi määramist läbi DHCP (*Dynamic Host Configuration Protocol*). Igakord kui süsteem ülesse ärkab, siis WiFi ühendamisel pöördutakse DHCP poole, mis üldiselt on ruuterites automaatselt olemas, ehk peab pärima läbi ruuteri parameetrid nagu DNS (*Domain Name Server*), alamvõrgu maski ning seadmele vaba IP aadressi.

Lahenduseks on, et kui süsteemi mälus WiFi andmeid ei salvestatud ei ole, küsitakse DHCP-lt alamvõrgu mask, DNS-i IP aadress, *Default Gateway* IP ja lokaalne IP aadress seadmele endale. Andmed salvestatakse Flash püsimällu. Järgmine kord kui seade ärkab unerežiimist, kasutatakse eelnevalt päritud andmed WiFi konfigureerimiseks, DHCP-d kasutamata. Seekord WiFi ühenduse loomiseks kulub maksimaalselt 150ms aega, mis tähendab, et esialgse ca 4500ms võrreldes paranes tulemus 96,7%.

10 Veebilehe lahendus

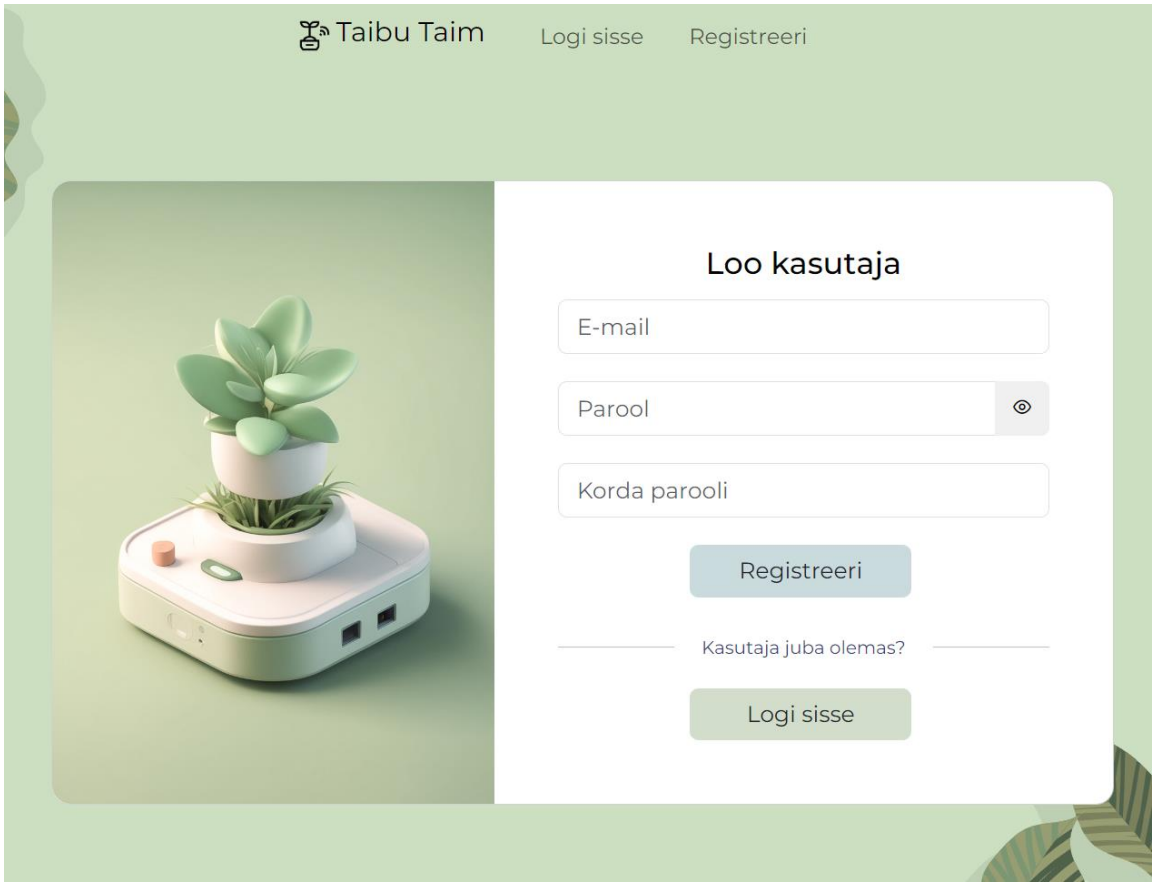
Selles peatükis kirjeldatakse milline on loodud veebilehe struktuur, veebilehe funktsionaalsused ja mida on kasutajal võimalik läbi veebilehe teha.

Peamine veebilehe ülesanne on näidata kasutajale seadme mõõdetud andmeid. Kuid igale kasutajale on mõeldud vaid tema seadmete poolt mõõdetud andmed, seega on esmalt vaja tegeleda kasutajate autentimisega. Selleks on võimalik igal kasutajal luua endale veebilehe konto või logida sisse olemasoleva Google kasutajaga. Joonis 12. peal on näha sisse logimise vaadet veebilehel.



Joonis 12. Sisse logimise vaade

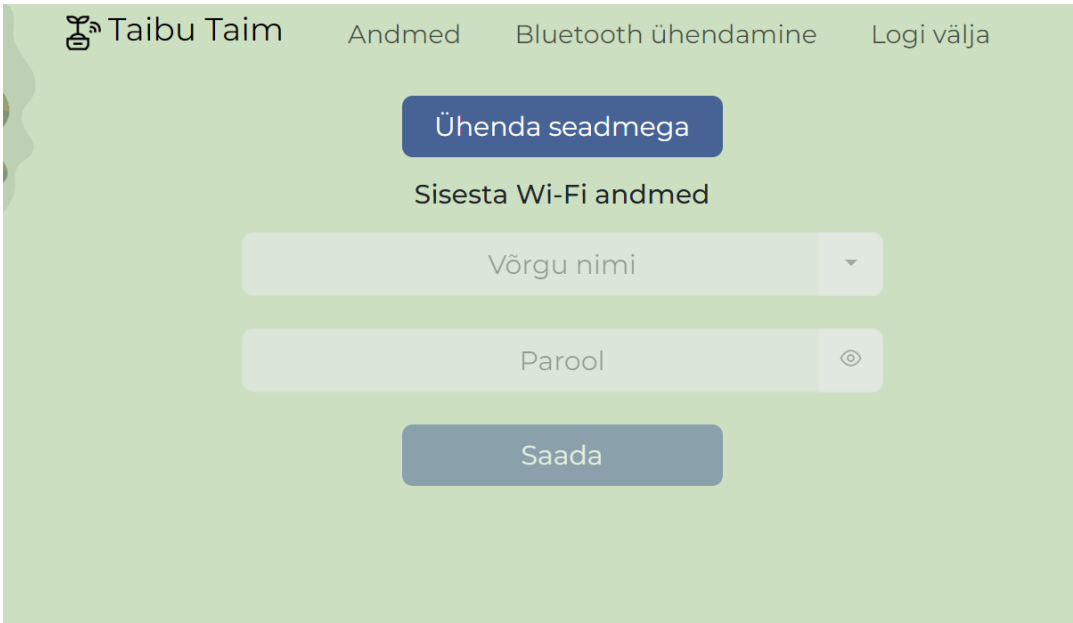
Joonis 13. peal on näha registreermise vaadet veebilehel.



Joonis 13. Registreerimise vaade

Kui kasutaja on endale konto loonud on, avaneb tal võimalus kasutada autenditud kasutajatele mõeldud funktsionaalsust.

Joonis 14. peal on näha Bluetooth-i ühendamise vaadet veebilehel. Alguses peab kasutaja andma seadmele WiFi andmed. Seda tehakse lehelt Bluetooth ühendamine.



Joonis 14. Bluetooth ühendamise leht

Kui vajutada „Ühenda seadmega“ nuppu leitakse kõik seadmed nimega TaibuTaim, millel on õige BLE teenuse identifikaator. Vahepeal on inimestel kodus WiFi nimed keerulised – sisaldavad numbreid ja halvasti meeldejäävad. Sellele vastu aitamiseks, saadab seade kõik leitud WiFi võrgu SSID-d veebilehele ja need kuvatakse nimekirjas „Võrgu nimi“, et kasutajal oleks kerge valida enda koduvõrgu nimi ning üle jääb vaid parool sisestada. WiFi ühenduse tulemusest antakse kasutajale märku.

Lehel andmed on kõik kasutaja seadme poolt mõõdetud parameetrid, mida on võimalik graafikutelt ja infokaartidelt vaadata. Kui seade teeb mõõtmise ja saadab andmed andmebaasi, siis veebliehel uuenevad infokaardid ja graafikud reaalajas.

Joonis 15. Andmete vaade

Kõige ülemisel paneelil infokaartidelt on näha kõige viimati mõõdetud andmeid, vasakult paremale vastavalt paterei laetuse protsent, õhu temperatuur, mulla niiskus, õhuniiskus, valgus intensiivsus. Keskmisel paneelil on graafikud. Ülemiste nuppudega on võimalik valida kuvatavate graafikute vahel ning alumised nupud panevad paika mõõtetulemuste ajavahemiku. Kõige alumisel paneelil on näidatud ajas toimunud muutusi ja valitud ajavahemiku keskmist. Joonis 15. peal on parasjagu kuvatud aku laetuse muutus viimase 24 tunni jooksul. Alumiselt paneelilt selgub, et 24 tunni jooksul keskmine aku protsent on 89.8%, viimase 3, 6, 12, 24 tunni jooksul on aku protsendi muutused olnud vastavalt 0,0%, -0,2%, -0,5% ja -1,1%.

11 Testimine

Kui kogu sardtarkvara ja veebilehe tarkvara oli valmis kirjutatud, oli vaja testida kas süsteem tervikuna töötab järjest nii, et vigu ei esineks ja seadme programm ning veebileht ei hanguks ega kokku ei jookseks

Alates viimasest sardtarkvara uuendusest on süsteem olnud aktiivselt töökorras ilma vigadeta ja kokku jooksmiseta alates praeguse peatüki kirjutamisest 30 päeva. Testimise metoodikaks oli seadme iga funktsionaalsuse läbi testimine eri olukordades ja eri aegadel. Metoodika lubas kontrollida igas äärejuhus süsteemi õiget käitumist.

Näiteks seadme ühendamine koduvõrkku kasutades koduvõrgu andmeid. Kui läbi ruuteri WiFi parool ära vahetada, ei suuda seade end koduvõrkku enam ühendada ning seade läheb WiFi andmete päringu režiimi, seejärel on võimalik läbi veebiliidese uued WiFi andmed anda. Võrkku ühendamist testiti ka telefonist loodud WiFi *hotspot*-iga, et emuleerida olukorda, kus seadme omanik viib seadme teisse võrkku.

Samuti testiti, et sensorite mõõdetud tulemused ei oleks absurdsed. Andmelehtede järgi on sensorid kindlasti piisavalt täpsed, sest taimed taluvad muudatusi suuremates vahemikes, mis on sensorite enda mõõtemääramatused. Valgussensori mõõtetulemuste puhul on veebiliidese graafikutelt näha selgeid tõuse kui päike tõuseb või tuled põlevad. Niiskussensori puhul on 30 päeva arvutatud keskmine 30%, mis võiks toa sisekliima kontekstis olla suurem, kuid ka loogiline, sest külmade ilmade ajal, õhuniiskus on madalam [31]. Temperatuuri sensorit sai testida asetades seade keskkonda, mille temperatuuri me teame ca 2 °C vahemikus, nt külmkappi (kui külmkapp näitab kraade ekraanil) või õue ja vaadata mõnest ilmateate portaalist õhutemperatuuri.

12 Projekti võimalikud edasiarendused

Projekti kõik nõuded ja eesmärgid said täidetud, sellegipoolest eksisteerib võimalusi, kuidas saaks teha projekti paremaks.

Nõuete järgi loodi trükkplaat konkreetse gabariidi ja komponentide paigutustega, selleks, et talle oleks võimalik luua korpus. Korpuse lahenduseni aga lõputöö käigus ei jõutud. Tulevikus tuleks uurida ja testida erinevate korpustega. Korpus võiks olla hüdrofoobne, mahutada ära ka LiPo aku ja ei tohiks piirata sensorite tööd.

Kuna iga taime nõuded on erinevad, mõned vajavad rohkem vett, valgust soojust kui teised. Selleks tuleks lisada veebilehele kasutaja funktsionaalsus, et seada mõõtmiste alam- ja ülempiire. Näiteks kui mullaniiskus on alla 20%, antakse kasutajale sellest kuidagi kohe teada. Märguande lahendus riistvaraliselt näiteks sumisti piiks või tarkvaraliselt sõnum, milleks Firebase pakub samuti pilvelahendust Firebase Cloud Messaging.

Praegu ei ole ühtegi visuaalset indikaatorit trükkplaadi peal, mis näitaks seadme omanikule millises olekus süsteem füüsiliselt on, siis kui trükkplaat parasjagu ei lae. Näiteks Bluetooth ja WiFi ühendustest võiks mõne LED-ga sellest indikatsiooni anda.

13 Kokkuvõte

Antud lõputöö eesmärgiks oli luua ideest tooteni IoT seade, mis aitaks inimestel kasvatada enda kodudes toataimi. Seade loeb taime eluks vajalikke parameetreid, mida on kasutajal võimalik vaadata läbi veebiliidese.

Et eesmärgini jõuda tuli esmalt valida konkreetsed tehnoloogiad ja tööriistad mida eesmärkide saavutamiseks kasutatakse. Seejärel tuli algusest lõpuni arendada riistvara, sh analüüside käigus valida välja sobilikud komponendid, disainida trükkplaat, siis arendada sardtarkvara, et see oleks tõhus ja veakindel ning lõpuks luua veebiliides. Arendusprotsessi käigus tehtud valikud on töö käigus põhjendatud ja analüüsitud.

Lõpptulemus on riistvara, sardtarkvara ja veebiliidese seatud nõuetele igati vastavuses ning kõik jõuti sellisel määral arendatud, et lõpptoode on reaalselt kasutatav, töötav ja täidab endale seatud eesmärgid. Seade töötab aku toitel, annab ülevaate taime elust, andmed on kasutaja liidesest jälgitavad ning seadme lisamine mulda ei kahjusta seal oleva taime elu.

Lõpus tuuakse välja ka projekti puudujäägid, kuid pakutakse neile välja võimalikud lahendused.

Kasutatud kirjandus

[1] S.-K. &. PARTNERS, „Recent Study Reveals More Than a Third of Global Consumers Are Willing to Pay More for Sustainability as Demand Grows for Environmentally-Friendly Alternatives,“ 14 10 2021. [Võrgumaterjal]. Available: https://www.businesswire.com/news/home/20211014005090/en/Recent-Study-Reveals-More-Than-a-Third-of-Global-Consumers-Are-Willing-to-Pay-More-for-Sustainability-as-Demand-Grows-for-Environmentally-Friendly-Alternatives. [Kasutatud 16 4 2021].

[2] G. J. Holzmann, NASA/JPL Laboratory for Reliable Software, Juuni 2006. [Võrgumaterjal]. Available: The Power of 10: Rules for Developing Safety-Critical Code. [Kasutatud 17 4 2024].

[3] V. Cherlinka, „Soil Moisture Sensors: Smart Tool For Precision Farming,“ 6 2 2023. [Võrgumaterjal]. Available: https://eos.com/blog/soil-moisture-sensor/. [Kasutatud 11 4 2024].

[4] „Soil moisture sensors — How they work. Why some are not research grade.,“ [Võrgumaterjal]. Available: https://metergroup.com/measurement-insights/soil-moisture-sensors-how-they-work-why-some-are-not-research-grade/. [Kasutatud 11 4 2024].

[5] A. Spiess, 20 8 2023. [Võrgumaterjal]. Available: https://youtube.com/clip/UgkxBCH9l_0xvx0dZalWSbdCdr2N-4E3ikE9?si=FsB9DAvWZBS3MQ-j.

[6] „PlatformIO Core (CLI),“ [Võrgumaterjal]. Available: https://docs.platformio.org/en/latest/core/index.html.

[7] „Vue.js Rendering Mechanism,“ [Võrgumaterjal]. Available: https://vuejs.org/guide/extras/rendering-mechanism.

[8] D. D. Gesualdo, „BLE: Let’s Compare the Best Market Solutions,“ EEWeb, 22 2 2021. [Võrgumaterjal]. Available: https://www.eeweb.com/ble-lets-compare-the-best-market-solutions/. [Kasutatud 18 4 2024].

[9] „Nordic Product Guide,“ 2023. [Võrgumaterjal]. Available: https://www.nordicsemi.com/-/media/Publications/WQ-Product-guide/Product-Guide_Nordic_2023.pdf. [Kasutatud 18 4 2024].

[10] „Espressif Modules,“ [Võrgumaterjal]. Available: https://www.espressif.com/en/products/modules. [Kasutatud 18 4 2024].

[11] espressif, „ESP32-S2-MINI-1,“ 2023. [Võrgumaterjal]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-s2-mini-1_esp32-s2-mini-1u_datasheet_en.pdf. [Kasutatud 18 4 2024].

[12] espressif, „ESP32-S3-MINI-1,“ 2024. [Võrgumaterjal]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-s3-mini-

1_mini-1u_datasheet_en.pdf. [Kasutatud 18 4 2024].

[13] espressif, „ESP32-C3-MINI-1,“ 2022. [Võrgumaterjal]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-c3-mini-1_datasheet_en.pdf. [Kasutatud 18 4 2024].

[14] espressif, „ESP32-C6-MINI-1,“ 2024. [Võrgumaterjal]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-c6-mini-1_mini-1u_datasheet_en.pdf. [Kasutatud 18 4 2024].

[15] A. R. Gupta, „Leveraging Wi-Fi 6 Features for IoT Applications,“ Medium, 14 6 2023. [Võrgumaterjal]. Available: https://blog.espressif.com/leveraging-wi-fi-6-features-for-iot-applications-c23cc6a548aa. [Kasutatud 18 4 2024].

[16] D. Huang, „802.11ax fundamentals: Target Wake Time (TWT),“ Commscope, 9 2018. [Võrgumaterjal]. Available: https://www.commscope.com/blog/2018/802.11ax-fundamentals-target-wake-time-twt/. [Kasutatud 18 4 2024].

[17] „MAX44009 Industry’s Lowest-Power Ambient Light Sensor,“ [Võrgumaterjal]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/max44009.pdf.

[18] „VEML7700 High Accuracy Ambient Light Sensor With I2C Interface,“ [Võrgumaterjal]. Available: https://www.vishay.com/docs/84286/veml7700.pdf.

[19] „LTR-F216A Optical Sensor Product Data Sheet,“ [Võrgumaterjal]. Available: https://optoelectronics.liteon.com/upload/download/DS86-2019-0016/LTR-F216A_Final_DS_V1.4.PDF.

[20] „Teval elektroonika,“ [Võrgumaterjal]. Available: https://www.teval.ee/shop.

[21] „SHTC3 Datasheet,“ [Võrgumaterjal]. Available: https://www.mouser.ee/datasheet/2/682/seri_s_a0003561073_1-2291167.pdf.

[22] „MAX17048/MAX17049 3µA 1-Cell/2-Cell Fuel Gauge,“ [Võrgumaterjal]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/max17048-max17049.pdf.

[23] „bq27441-G1 System-Side Impedance Track™ Fuel Gauge,“ [Võrgumaterjal]. Available: https://www.ti.com/lit/ds/symlink/bq27441-g1.pdf?ts=1713539030099&ref_url=https%253A%252F%252Fwww.google.com%252F.

[24] „MAX17260 5.1µA 1-Cell Fuel Gauge,“ [Võrgumaterjal]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/max17260.pdf.

[25] „LCSC,“ [Võrgumaterjal]. Available: https://www.lcsc.com/.

[26] „Dielectric Parameters,“ [Võrgumaterjal]. Available: https://eo-college.org/topic/dielectric-parameters/. [Kasutatud 9 5 2024].

[27] „Impacts of Internal Curing on the Performance of Concrete Materials in the Laboratory and the Field,“ 11 2017. [Võrgumaterjal]. Available:

https://www.researchgate.net/publication/323792836_Impacts_of_Internal_Curing_on_the_Performance_of_Concrete_Materials_in_the_Laboratory_and_the_Field#pf23. [Kasutatud 9 5 2024].

[28] A. A. R. J. F. C. C. D. A. d. S. T. Clarissa Pereira dos Santos, „Performance of the capacitive moisture sensor under different saline conditions,“ 2022.

[29] C. C. A. R. D. Kevin Townsend, „Getting Started with Bluetooth Low Energy,“ 2014. [Võrgumaterjal]. Available: https://www.oreilly.com/library/view/getting-started-with/9781491900550/ch04.html.

[30] Silicon Labs, „Bluetooth® LE Fundamentals,“ [Võrgumaterjal]. Available: https://www.silabs.com/documents/public/user-guides/ug103-14-fundamentals-ble.pdf.

[31] „Indoor air temperature and relative humidity measurements in Finnish schools and day-care centres,“ 1 12 2023. [Võrgumaterjal]. Available: https://www.sciencedirect.com/science/article/pii/S0360132323009964?dgcid=rss_sd_all. [Kasutatud 9 5 2024].

[32] „“platformio.ini” (Project Configuration File),“ PlatformIO Labs, [Võrgumaterjal]. Available: https://docs.platformio.org/en/stable/projectconf/index.html. [Kasutatud 4. detsember 2023].

[33] Google, „API Reference,“ 8. august 2023. [Võrgumaterjal]. Available: https://firebase.google.com/docs/reference/js. [Kasutatud 15. detsember 2023].

[34] „STM32 32-bit Arm Cortex MCUs,“ [Võrgumaterjal]. Available: https://www.st.com/en/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus.html. [Kasutatud 18 4 2024].

[35] „ESP-C6 support,“ GitHub, 16 1 2023. [Võrgumaterjal]. Available: https://github.com/espressif/arduino-esp32/issues/7713#issuecomment-1383776442. [Kasutatud 18 4 2024].

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

Mina, Siim Tišler

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose "IoT toataime seiresüsteem", mille juhendaja on Peeter Ellervee

   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.

12.05.2024

[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 – Elektriskeem