Opinnäytetyö (AMK)

Sulautetut järjestelmät

2018

Mikael Nyberg

# LASER-LIIKESUUNTATUNNISTIN

– lähiverkkoon liitettävä kävijälaskuri myymälään

TURKU AMK
TURKU UNIVERSITY OF
APPLIED SCIENCES

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Sulautetut järjestelmät

2018 | 46 sivua

Mikael Nyberg

# LASER-LIIKESUUNTATUNNISTIN

## - lähiverkkoon liitettävä kävijälaskuri myymälään

Opinnäytetyön tavoitteena oli toteuttaa Storm Motors Oy:lle kävilaskurijärjestelmä. Kävijälaskurin tavoitteena oli perinteisen laskennan lisäksi myös mahdollisuus seurata statistiikkaa etänä verkon välityksellä, sekä mahdollistaa tiettyjen ajanhetkien vierasmäärien selvitys. Toteutuksen oli tarkoitus korvata perinteinen kävijälaskuri, joka liikkeissä sillä hetkellä toimi, ja tuoda verkkoliitettävyydellä enemmän mahdollisuuksia seurata vierailijamääriä hallitusti eri puolilta Suomea.

Projektissa tutkitaan ja kokeillaan useita eri toteutusvaihtoehtoja. Alustoina toimivat pääosin Googlen tarjoama Analytics-rajapinta, NodeMCU-mikro-ohjain sekä Arduino-mikro-ohjain. Windows-käyttöjärjestelmälle toteutettiin myös Arduino-pohjaisessa laitemallissa isäntäsovellus, jonka tarkoitus oli ohjata NRF24-protokollan kautta kulkevaa tietoa pilvipalveluihin. Verkkoratkaisuina kokeiltiin NRF24-moduuleja sekä NodeMCU:n omaa Wifi-adapteria.

Liiketunnistintekniikka toteutettiin lasereilla ja LDR-vastuksilla. Lasereita ja vastaanottimia tuotettiin kaksi paria. Ensimmäinen laukaistu pari kertoo henkilön kulkusuunnan ja mahdollistaa vieraiden kokonaismäärän sekä liikkeen vierasmäärän seurannan tietyn aikavälin sisällä. Laitteiden kotelointi ja ripustusratkaisut toteutettiin 3D-tulostetuilla muoviosilla, jotka suunniteltiin räätäliratkaisuna projektissa kehitetyille elektroniikkakokonaisuuksille. Storm Motors ilmaisi myös kiinnostusta säätietojen sisällytykseen tietoja säilöttäessä. Tarkoitus oli pystyä tutkimaan korrelaatiota sään ja vierasmäärän välillä. Windows-isäntäsovellukseen tuotettiin näkymä, joka mahdollisti säätietojen sisältämisen Analyticsiin lähtevään dataan. Projekti jäi toistaiseksi proof of concept-tasolle eikä sitä ole otettu tuotantokäyttöön Storm Motorsilla. Yksittäiset kokonaisuuden osat olivat kuitenkin toimivaksi asti jalostettuja.

Tässä opinnäytetyössä käydään lyhyesti läpi alustat, ratkaisut ja tekniikat, joita projektissa on käytetty. Esittelyn jälkeen käydään läpi tavat, joilla kyseisiä asioita on hyödynnetty juuri tässä projektissa.

ASIASANAT:

Arduino, laser, nodemcu, nrf24, 3d-tulostus, kulunvalvonta, mikrokontrolleri

BACHELOR´S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Embedded systems

2018 | 46 pages

Mikael Nyberg

# MOVEMENT DIRECTION SENSORS BASED ON LASERS

## - network enabled visitor counter for store use

The objective of this thesis is to design and create a visitor counter for Storm Motors Oy. The counter is supposed to replace an existing solution with a new version that allows exporting data to external services, which then allows the user to see data from all stores located across Finland. The project explores and tries out different alternatives for the solution. Platforms used include: Google Analytics, NodeMCU, Arduino and Windows. Windows is primarily used for displaying data and relaying information from the NRF24 radios in the alternative version that uses those for communication.

NodeMCU’s integrated WIFI and NRF24-radios represent the alternatives for radio communication in the project. Motion sensor technology is represented by laser-LDR-pairs that allow tracking of movement direction by means of determining which pair was triggered first. Having movement data also allows displaying data about visitor amounts currently in the store at certain times. Device encasings and mounting solutions are modeled and prototyped with 3D-printers.

Storm Motors also expressed interest in incorporating weather data into the results that are sent to Google Analytics. The main interest in weather data is to map a correlation between weather conditions and customer traffic. The project remained at a proof of concept-stage where most individual components are working, but the solutions hasn’t been taken into production use.

This thesis explores the techniques utilized in the process of creating the system and offers a brief introduction into the technology being used at each stage.

KEYWORDS:

Arduino, lazer, nodemcu, nrf24, 3d-printing, access control, microcontroller

# SISÄLTÖ

# KÄYTETYT LYHENTEET

API Application program interface, rajapinta järjestelmässä jolla voidaan syöttää tai hakea tietoa

FDM Fused deposition modelling, pursotusmallinen ratkaisu

GND Ground, maa (elektroniikka)

GPIO General purpose input output, yleiskäyttöinen tulo- ja lähtöliitäntä

HTTP Hypertext transfer protocol, internet-sivujen lataamiseen käytetty verkkoprotokolla

IDE Integrated development environment, kehitysympäristö

IoT Internet of things, moderni termi pienille, mahdollisesti paristokäyttöisille järjestelmille, jotka yhdistyvät internettiin

JSON Javascript object notation, javascript-ohjelmointikielen merkkijonotyyppinen esitystapa

LDR Light dependent resistor, valosta riippuvainen vastus

LED Light emitting diode, valoa hohtava komponentti

LUA ei akronyymi, tarkoittaa sanaa ”Kuu” portugaliksi, ohjelmointikieli

MOSFET Metal-oxide-semiconductor field-effect transistor, eristehilatransistori

MYSQL “My” on kehittäjän tyttären nimi, SQL on Structured query language, tietokannoissa käytetty ohjelmointikieli tiedon käsittelyssä, palvelintyyppinen tietokantaratkaisu

NRF24 Nordic Semiconductor Radio 2.4GHz ~

PIR passive infrared sensor, liiketunnistin

RC (piiri) resistor–capacitor circuit (RC circuit), vastuksesta ja kondensaattorista koostuva sähköpiiri

SPI Serial peripheral interface, sarjaliikenneliitäntä

SQLite Structured query language lite (lite = light = kevyt), tietokannoissa käytetty ohjelmointikieli tiedon käsittelyssä, paikallistiedostossa toimiva tietokantaratkaisu

STL stereolithography, stereolitografia, 3D-tulostustekniikassa käytetty tiedostomuoto mallille

TCP Transmission control protocol, tiedonsiirtoprotokolla

USB Universal serial bus, yleiskäyttöön soveltuva sarjaportti

WIFI nimitys langattomalle teknologialle, wireless local area network (WLAN)

# 1 JOHDANTO

Storm Motors Oy lähestyi Turun ammattikorkeakoulua kävijälaskurijärjestelmän toteutuspyynnöllä. Toimeksianto ilmoitettiin sulautettujen järjestelmien tunneilla, jolloin halukkaita tekijöitä haettiin suorittamaan järjestelmän rakennus opinnäytetyönä.

Kävijämäärien seurannalla, sekä muulla statistiikalla, on nykyään paljon kiinnostuneita yrityksiä jotka haluavat kehittää omia sovelluksia ja laitteitaan tähän tarkoitukseen. Storm Motors Oy halusi kehittää uusia tapoja, joilla vierasmäärien ja vieräsmäärien suhdetta muihin tekijöihin, olisi mahdollista seurata tehokkaammin. Alkuperäinen vierasseuranta liikkeessä toimi PIR-sensorilla, jonka toiminta oli hyvin yksinkertaisella tasolla, ja johon haluttiin uudistusta.

Keskeisimmät asiat projektissa olivat,

- kustannustehokkuus muihin valmisratkaisuihin verrattuna
- kyky nähdä päivittäinen vieralijamäärä
- kyky seurata vierasmääriä liikkeessä tietyin ajankohdin
- kyky lähettää tieto keskitysti yhteen paikkaan monesta eri liikkeestä
- seurata sään vaikutusta vierasmääriin

Alustoiksi harkittiin Raspberry Pi -tietokonetta, Arduino-mikrokontrolleria sekä NodeMcu-kehitysalustaa. Raspberry Pi karsittiin pois prototyyppivaiheesta, koska laitteiston ylläpidon monimutkaisuus ja virhealttius, nähtiin liian suureksi riskiksi pitkäaikaiselle huoltovapaalle järjestelmälle.

Arduinolla sekä NodeMCU:lla kehitettiin prototyypit, joita kumpaakin testattiin ja kehitettiin erillisinä järjestelminä, mutta ainoastaan Arduinolla tehty ratkaisu päätyi tuotantojärjestelmän ehdokkaaksi.

Molemmat ratkaisut valmistettiin komponenteista ja moduuleista, täysin räätäliratkaisuna, juuri tähän projektiin. Projektin aikana hyödynnettiin valmiita kehitysalustoja logiikan toteutukseen, elektroniikkakomponenttien yhdistelmiä sensori- ja liitettävyysratkaisuihin, sekä moduuleita muun muassa radioliikenteen mahdollistamiseksi. Kotelointi ja liitäntäratkaisut mallinnettiin tarpeen mukaan ja valmistettiin 3D-tulostusta hyödyntäen. PIR-sensorit korvattiin lasereilla sekä valoherkillä vastuksilla.

# 2 LAITTEISTO

## 2.1 Mikro-ohjaimet

### 2.1.1 Arduino

Arduino on vuonna 2005 Italiassa kehitetty mikro-ohjain, jonka on kehittänyt ryhmä ohjelmoijia sekä opettajia, Ivrea Interaction Design -instituutissa. Alkuperäinen motiivi tuotteen kehitykselle oli auttaa opiskelijoita prototyyppauksessa. (Alice Mela, 2013 [1])

Kiinalaiset versiot Arduinosta ovat hyvin suosittuja hintansa puolesta, mutta niissä on tehty hieman muutoksia alkuperäisiin versioihin. Kopioversiossa on muun muassa käytetty eri USB-ohjainpiiriä alkuperäisiin Arduinoihin verrattuna (Kuva 1).

Kuva 1. Arduino Nano V3.0 kopiomalli, Core electronics [2].

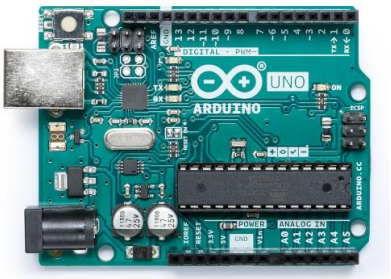
Arduinosta on valmistettu useita eri versioita ja malleja, joista tunnetuimmat ovat: Uno, Leonardo, 101, Esplora, Micro, Nano ja Mini. Lisäominaisuuksia on myös tarjolla ”shield”-lisäkorteilla, jotka ovat suunniteltu sopimaan suoraan Arduinon päälle asetettavaksi ilman juotoksia. Edellä mainitut lisäkortit tuovat lisäominaisuuksia kuten wifi-liitettävyys ja kyky hallinnoida voimavirtapiirejä releillä. (arduino.cc, 2018 [3])

Arduino laskee elektroniikasta kiinnostuneiden aloituskynnystä merkittävästi helppokäyttöisyydellään. Alusta koostuu ATMEL-piiristä, joka kytkeytyy kehitysalustan I/O-pinneihin, ja mahdollistaa helpon logiikan käskytyksen muutamalla rivillä koodia.

Tyypillinen projekti ohjelmoidaan C++ -ohjelmointikielellä Arduino IDE:ssä. Lähdekooditiedostoja kutsutaan nimellä ”Sketch”, eli hahmotelma. Tavallisesti Arduino sketch koostuu setup-osiosta ja loop-osiosta. Setup-osio sisältää kerran ajettavaa

koodia, eli sarjaporttien ja I/O-lähtöjen määrittelyä. Loop-osio sisältää ohjelman varsinaisen työjonon jonne, voidaan esimerkiksi määritellä sarja käskyjä, joilla luodaan valosarja digitaalilähtöihin kytketyillä LED:illä.

Tässä opinnäytetyössä käytetään kahta Arduinoa. Liikkeen portti muodostaa yhden kokonaisuuden joka sisältää laserit, LDR-vastukset, Arduinon sekä NRF24-radion. Vastaanottava pää, eli toinen Arduino toisella NRF24-radiolla, liitetään myyntitiskin kassakoneeseen USB-yhteydellä. USB-yhteyden välityksellä voidaan luoda sarjaliikennettä kassakoneella sijaitsevaan C#-ohjelmaan. Alkuperäinen Arduino voidaan kytkeä Arduinon omalla ajurilla suoraan tietokoneeseen, jolloin ajuri luo tietokoneelle virtuaalisen sarjaportin, jota hyödyntämällä voidaan lähettää tietoa molempiin suuntiin (Kuva 2). Kiinalaiset kopiomallit tarvitsevat eri ajurin, koska ne käyttävät eri piiriä USB-yhteyteen.




Kuva 2 Alkuperäinen Arduino UNO, arduino.cc [4]

Arduinon vahvuus tässä opinnäytetyössä on sisäänrakennettu 5 V:n muuntaja sekä laaja skaala sisääntulojännitteitä. USB-tekniikka toimii myös 5 V:n jännitteellä, joten laitteiden käyttö tietokoneiden ja puhelinlaturien läheisyydessä sujuu ilman lisämuuntajia.

### 2.1.2 ESP8266 NodeMCU

ESP8266 on kiinassa valmistettu matalakustanteinen mikrosiru, joka tukee TCP/IP-protokollaa ja johon on sisäänrakennettu wifi-ominaisuus (Kuva 3).

Kuva 3. NodeMCU variaatio, marginallyclever.com [5]

ESP-mikrosirut valimisti alun perin Espressif Systems kiinassa, mutta suurin osa teknisestä dokumentaatiosta oli saatavilla vain kiinankielellä. Laitteen koko, ominaisuudet ja hinta houkutteli länsimaalaisia harrastajia vuonna 2014 kun ESP-01-siru levisi länsimaiseen tietoon. Harrastajat aloittivat dokumentaation kääntämisen Englanniksi ja ESP-sarjan mikrosirut levisivät nopeasti maailmanlaajuiseen käyttöön. (wikipedia.org, 2018 [6])

NodeMCU on avoimeen lähdekoodiin perustuva IoT, eli Internet Of Things, kehityslauta jonka ytimenä toimii ESP8266-mikrosiru. Kehityslauta tarjoaa helpomman liitettävyyden ESP8266-sirulle samalla tavalla kuin Arduino tarjoaa Atmel-siruille.

Pääasiallinen ohjelmointikieli NodeMCU:lle on normista poikkeavasti Lua-skriptikieli, mutta kehitysalustalle on myöhemmin lisätty tuki myös Arduino IDE -ohjelmistoon, joka sallii ohjelmoinnin perinteisellä C:llä. Käyttö ja ohjelmointi on siis tuotu hyvin lähelle Arduinon kehitystapaa.

Arduinon logiikka toimii 5 V:n jännitteellä, joka kuten edellä on mainittu, on hyvin yleinen jännite esimerkiksi USB-liikennöinnissä. ESP8266 toimii 3,3 V:n logiikalla, joka vaatii hieman suunnittelua, kun sitä yhdistetään muihin laitteisiin. Ympäristöt eivät siis ole aivan täysin keskenään yhteensopivia kaikissa tilanteissa, mutta logiikkajännitteen muuntajia käyttämällä on nämäkin haasteet ratkaistavissa.

Vahvin ominaisuus NodeMCU:ssa Arduinoon vertailtuna tässä projektissa on sisäänrakennettu langaton verkkoyhteys ja sitä tukevat TCP-protokollat, jotka sirulta löytyvät. Arduinolla toteutettu ratkaisu vaatii kaksi kehityslautaa ja kaksi radiota sekä tietokoneen lähettämään tietoja pilvipalveluun. NodeMCU voidaan liittää suoraan liikkeen olemassa olevaan langattomaan verkkoon, jolloin tietojen lähetys suoraan pilvipalvelun rajapintaan onnistuu ilman mitään muita välikäsiä tai laitteita.

## 2.2 Radiot

### 2.2.1 NRF24L01

NRF24L01 on Nordic Semiconductorin valmistama 2.4GHz toimiva mikropiiri jolla on mahdollista toteuttaa kustannustehokkaasti yhteyksiä eri laitteiden välille. Mikropiiriä saa eri piirilevykokoonpanoilla, joista pienempi sisältää suoraan piirilevylle toteutetun antennin, ja suurempi sisältää erillisen liittimen ulkoiselle antennille (Kuva 4). (nordicsemi.com, 2018 [7])

Kuva 4. nRF24L01 lähetin, Makerlab Electronics [8]

Yhteysetäisyydet vaihtelevat ympäristön, kokoonpanon, nopeusasetusten sekä tehoasetusten perusteella 250 metristä kahteen kilometriin. Yleinen käyttökohde on

harrastelijarobotiikka sekä muut käyttökohteet, joihin tarvitaan yhteyksiä järjestelmäkokonaisuuksien välille.

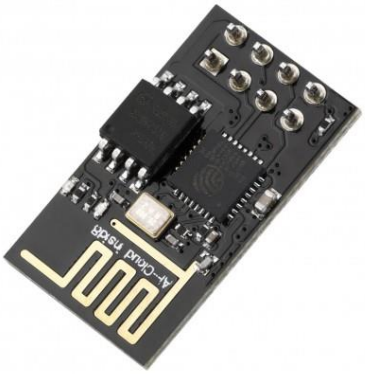
Piiri toimii pääasiallisesti 3,3 V:n logiikalla, mutta piirin sisääntulot sietävät 5 V:n logiikkaa. Radion jännitteensyöttö toteutettiin kaikissa käyttötapauksissa 5 V:n USB-jännitteeltä joka ajettiin regulaattorin läpi 3,3 V:ksi.

### 2.2.2 ESP8266 Wifi

ESP8266 wifi-moduuli on itsenäisesti toimiva järjestelmä mikropiirillä (Kuva 5). Piiri yhdistetään useimmiten osaksi mikrokontrolleria, jolloin kyseinen kontrolleri kykenee hyödyntämään TCP/IP-protokollaa ja wifi-antennia omiin tarpeisiinsa.

Piiriä voidaan ajaa AT, eli ”ATtention”, komennoilla joita käytetään yleisimmin modeemeissa. Piirin yhdistäminen on siis mahdollista moneen eri laitteeseen.

Tässä projektissa piiri on osana NodeMCU-kehityslautaa ja laitetta ei tulla ajamaan matalan tason käskyillä muuten kuin miten valmiit kirjastot mahdollisesti tekevät taustalla.



Kuva 5. ESP8266 Wifi-moduuli, ihmevekotin.fi [9]

ESP8266 toimii pääasiallisesti 3,3 V:n logiikalla, ja se ei siedä ollenkaan yhdistämistä 5 V:n logiikkaan tai käyttöjännitteeseen, joten regulaattori sekä logiikkamuunnin on ehdottoman tärkeä, mikäli tarkoitus on yhdistää laite esimerkiksi Arduinoon.

Piirin yhteysetäisyys riippuu paljon ympäristöstä, mutta käyttäjät ovat testeillä onnistuneet lähettämään tietoa jopa 300 m:n etäisyyksillä pelkällä piirilevyyn toteutetulla antennilla.

## 2.3 Komponentit

### 2.3.1 Laser

Projektissa kokeiltiin muutamia eri laserdiodeja, joiden teho oli maksimissaan 5 mW (Kuva 6). Laserit toimivat 5 V:n jännitteellä ja sisältävät valmiiksi sisäänrakennetun ajurin, eli vakiovirtamuuntimen. Diodia ohjataan MOSFET:n avulla suoraan mikrokontrollerin IO-lähdöstä.

Kuva 6. Laserdiodi, robotpark.com [10]

Laserin pääasiallinen tehtävä on tuottaa kaupan varashälytinportin välikköön kohdennettu valolähde, jonka LDR-vastus havaitsee ja ilmoittaa, kun henkilö on katkaissut laserin kävelemällä sen lävitse.

Valotehoa ja sen vaikutuksia ja haittoja lapsille käsiteltiin projektin aikana useamman kerran. Portin valotekniikka toteutettiin korkeudella joka altistaa helposti lapsen silmät lasersäteille. Lopullinen versio hyödyntää yhden milliwatin laseria joka alkaa ohjelmallisesti ajamaan lasereita pulssilla joka yhdistettynä matalaan valotehoon minimoi silmien altistuksen pitkäaikaiselle valolle.

### 2.3.2 LDR-vastus

LDR tulee englanninkielen lyhenteestä ”Light-dependent resistor”, eli valosta riippuvainen vastus. Komponenttia kutsutaan myös fotovastukseksi. Vastuksen resistanssi muuttuu suhteessa siihen kohdistuvaan valoon. Tässä projektissa vastus altistetaan myymälän valoille ja laserille, jolloin laser on näistä kahdesta valonlähteestä huomattavasti kirkkaampi koska valo saapuu komponentin pinnalle hyvin kohdistuneessa muodossa (Kuva 7).

Kuva 7. LDR-vastus, iprototype.nl [11]

LDR-vastus kytketään tyypillisesti mikrokontrollerin analogituloon jännitteenjakopiirillä. Jännitejakopiiri kytketään mitattavan jännitteen rinnalle ja siihen kytketään sarjaan kaksi tai useampi vastus, jolloin tietty osuus jännitteestä kohdistuu yhteen vastukseen, ja osa toiseen. Jos toinen vastuksista korvataan LDR-vastuksella, niin piirillä voidaan selvittää, kuinka suuri osuus jännitteestä kohdistuu LDR-vastukselle. Tuloksena syntyy osuus alkuperäisestä syöttöjännitteestä, joka voidaan muuntaa lineaariseksi skaalaksi ja josta voidaan päätellä vastukseen kohdistuva hetkittäinen valoteho.

### 2.3.3 Mosfet

Nimitys “MOSFET” on englanninkielen lyhenne, joka tulee sanoista ”metal-oxide-semiconductor field-effect transistor”. Suomeksi tämä on metallioksidi-puolijohdekanavatransistori tai eristehilatransistori (wikipedia.org, 2018 [12]). Arki- ja ammattikielessä komponenttia kutsutaan nimellä ”mosfetti” tai ”fetti” myös suomea puhuttaessa.

Komponentti koostuu kolmesta eri alueesta: hila (**G**ate), lähde (**S**ource) ja nielu (**D**rain). Pääasiallinen käyttötarkoitus mosfetilla tässä projektissa on kyky ohjata käyttöjännitettä laitteille joita mikrokontrollerin omat ulostulot eivät tehon puolesta pysty ajamaan. Esimerkiksi lasereita ohjataan mosfetin hilan ja maan välisen jännitepotentiaalin muutoksilla, jolloin laserin käyttöjännite tulee ulkoisesta jännitelähteestä, ja signaali joka ohjaa milloin laser on päällä tai pois päältä tulee mikrokontrollerin digitaalilähdöstä.

Projektissa käytettiin kahta erilaista mosfettia, joista toinen on suunniteltu 5 V:n logiikalle, ja toinen toimi alemmilla jännitetasoilla.

NodeMCU:ta käytettäessä jouduttiin ottamaan herkemmät mosfetit käyttöön, koska 3,3 V:n logiikkajännite ei kaikissa tapauksissa onnistunut avaamaan mosfettia ja näin ollen laseri ei tullut päälle haluttaessa (Kuva 8).


Kuva 8. 3,3V yhteensopiva mosfet, Addicore [13]

### 2.3.4 Regulaattori

Projektissa hyödynnettiin AMS1117-tyyppisiä regulaattoreita, joilla saatiin 5 V:n käyttöjännitteet tuotua alas 3,3 V:iin tapauksissa joissa oli käytössä moduuleja, jotka sitä vaativat (Kuva 9).

Kuva 9. AMS1117 Moduuli, Robu [14]

AMS1117 regulaattorilla saadaan tuotua 800 mA virtaa käyttökohteelle, joten esimerkiksi NodeMCU:n käyttöjännite tuotiin siihen itseensä sisäänrakennetun regulaattorin lävitse, jonka lähtövirta on riittävät mikrokontrollerin ja wifin virtatarpeisiin.

### 2.3.5 Kondensaattori

Kondensaattori on hyvin yleinen elektroniikassa käytetty passiivikomponentti, jonka tehtävänä on varastoida sähkövarausta. Kondensaattoria käytetään yleisimmin suodattimena ja jännitevaihteluiden tasoittajana (Kuva 10).

Kuva 10. Pari elektrolyyttikondensaattoreita, HTF Electronics [15]

Tässä projektissa kondensaattoreita on näin ollen käytetty myös NRF24L01-piirien jännitetuloissa tasaamassa jännitettä, kun radion virrantarve vaihtelee jyrkästi lähetystilanteissa.

### 2.3.6 Summeri

Piezo summeri on pietsosähköilmiöön perustuva pieni kaiuttimen tapainen elementti, jolla on mahdollista tuottaa ääniä (Kuva 11).

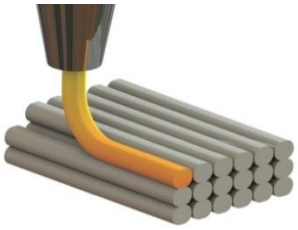
Kuva 11. Pietsosähköinen summeri, Squishy Circuits [16]

Tässä projektissa kyseistä komponenttia käytetään lyhyiden äänisarjojen soittamiseen käyttäjälle.

# 3 KOTELOINTI JA 3D-TULOSTUS

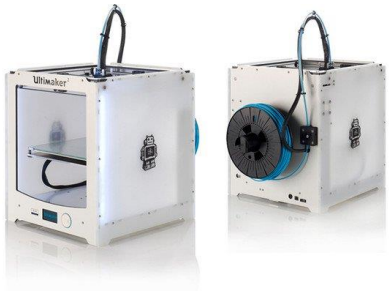
## 3.1 FDM-tekniikka

3D-tulostuksessa on käytössä monia eri tapoja tulostaa monia eri materiaaleja. Näistä yleisin ja tässä projektissa käytetty tekniikka on FDM-tekniikka eli Fused Deposition Modeling. Kyseessä on pursotusmenetelmä jolla kylmälle tai kuumalle lasialustalle pursotetaan sulaa muovia. (additive3d.com, 2018 [17])



Kuva 12. FDM-tekniikka havainnollistettuna, additive3d.com [17]

Projektissa hyödynnettiin Ultimaker 2- ja Wilson 2 -3D-tulostimia (Kuva 13, kuva 14). Toinen on kaupallisesti myynnissä oleva tulostin ja toinen on harrastelijoiden suunnitteleman tulostin jonka mallitiedostot ja rakennusohjeet ovat vapaasti saatavilla verkossa.



Kuva 13. Ultimaker 2 -tulostin, hothardware.com [18]

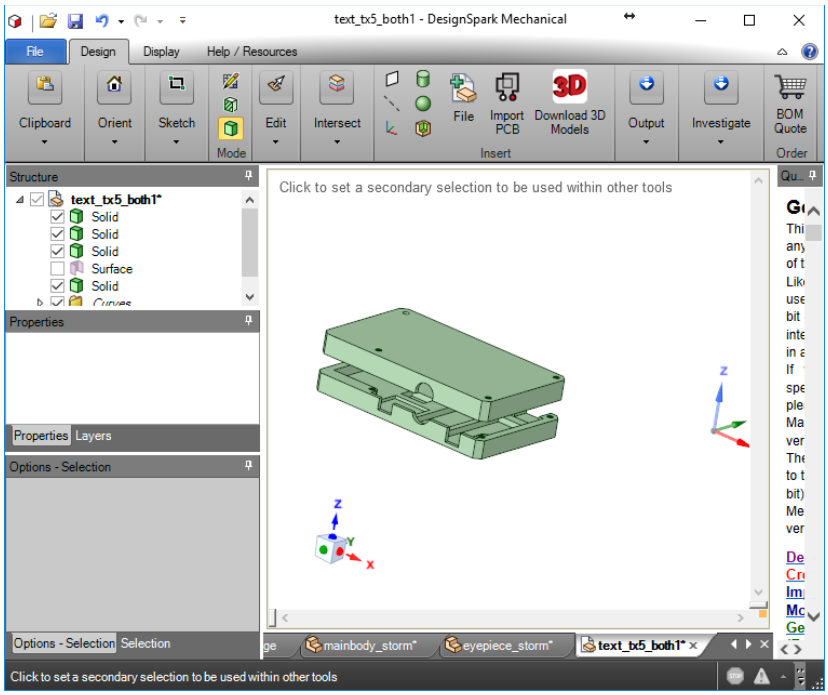
Molempia tulostimia saa ostettua valmiiksi koottuna ja molempien tarkat speksit ja mallit ovat saatavilla verkosta.

Kuva 14. Wilson 2 -tulostin, Tindie [14]

Wilson 2 on Martin Ricen itse suunnittelema tulostin.

## 3.2 3D-mallinnus

Mallinnus tehtiin täysin räätälöitynä ja kaikki suunnittelutyö tehtiin DesignSpark Mechanical-ohjelmalla (Kuva 15).

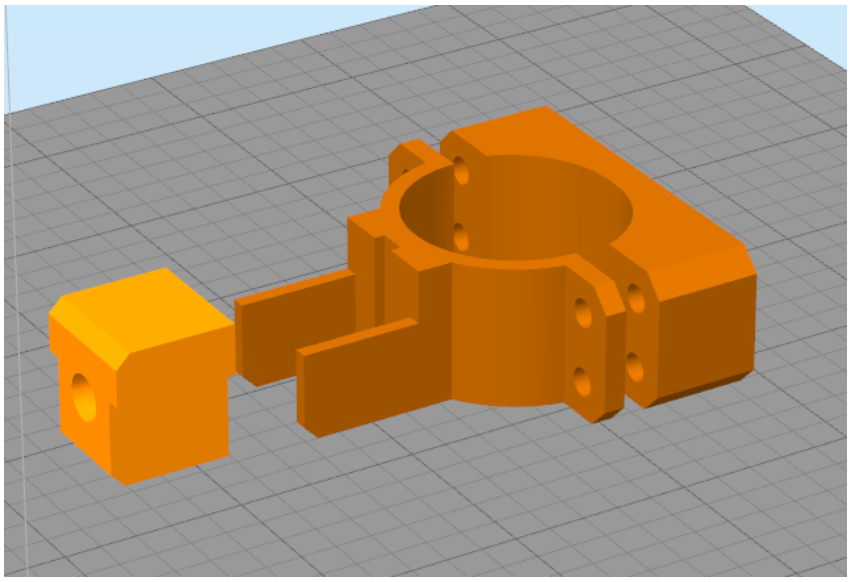



Kuva 15. DesignSpark Mechanical - Arduinon laitekotelo

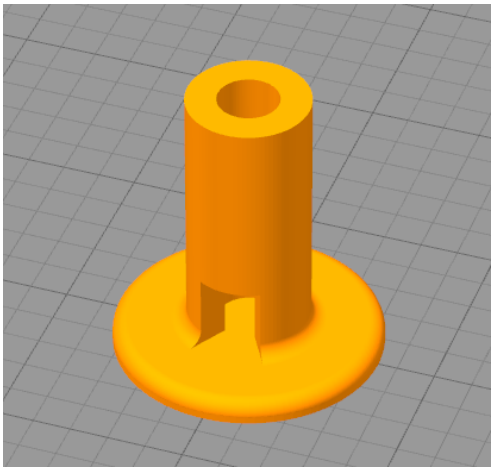
Ohjelma tuottaa STL-tiedostoja, jotka voidaan muuntaa myöhemmässä vaiheessa 3D-tulostimen ymmärtämään ohjetiedostomuotoon.

### 3.3 Laitekotelointi

Lasereille ja LDR-vastuksille suunniteltiin omat koteloinnit, joiden on määrä kiinnittyä 16 mm:n putkiin liikkeen varashälytinportissa (Kuva 16, Kuva 17).



Kuva 16. LDR-vastuksen kotelointi

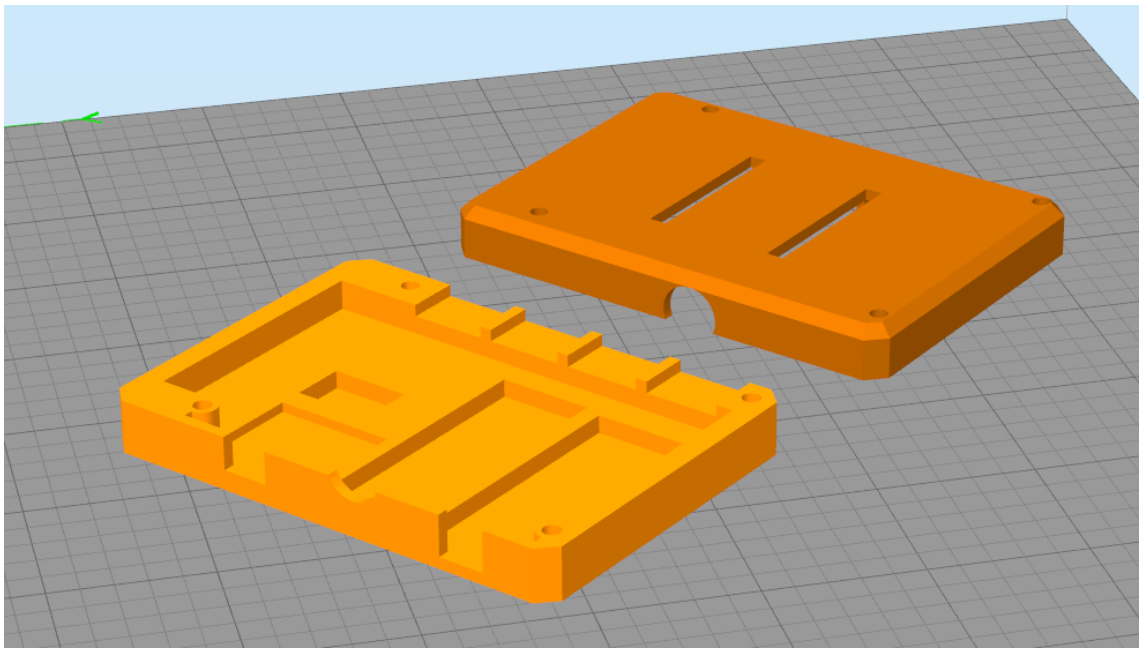


Kuva 17. Laserin pidike, ensimmäinen versio

Arduinolle ja NodeMCU:lle suunniteltiin kotelointi, johon kaikki tarvittavat komponentit ja moduulit mahdutettiin sisälle.

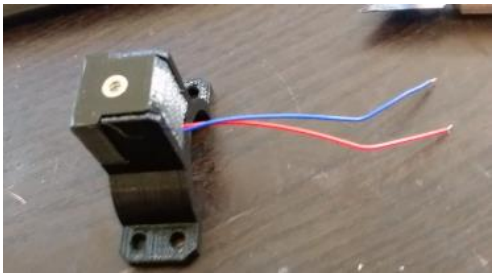
Suunnittelussa pyrittiin minimoimaan kaikki ylimääräinen johdotus niin että kaikki olisi mahdollisimman siistiä ja helposti kytkettävää.

Lopullinen versio toteutettiin USB-liittimillä, joilla saatiin lasereille ja LDR-vastuksille käyttöjännite ja ohjausjännitteet ja signaalit tuotua USB-johdon neljän liittimen avulla kätevästi takaisin Arduinolle tai NodeMCU:lle (Kuva 18).

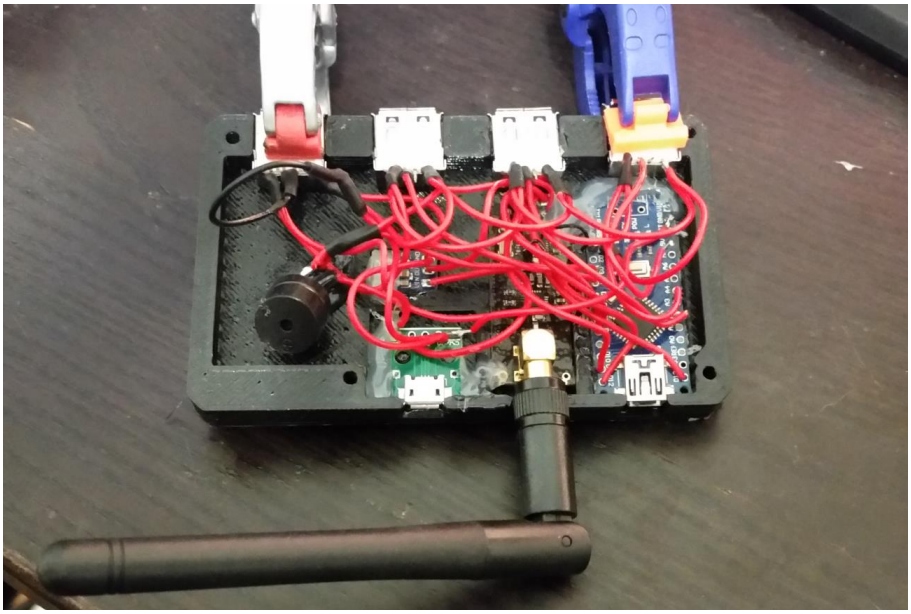


Kuva 18. Lopullinen Arduinon kotelointi

Koteloihin liimattiin komponentit omille paikoilleen ja mahdolliset johdotukset ja juotokset tehtiin kotelon tyhjiin tiloihin. Lopuksi laatikko suljettiin M3-koneruuveilla tai liimalla (Kuva 19, Kuva 20).



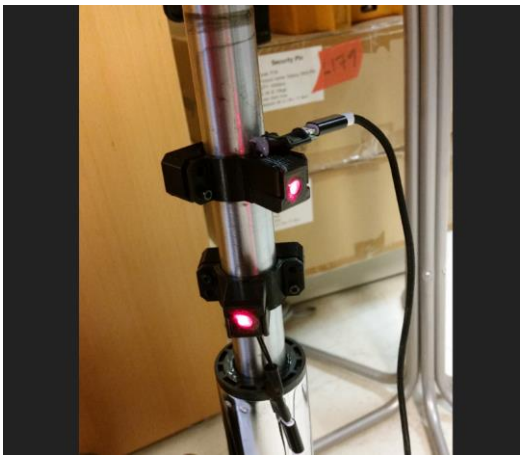
Kuva 19. Lopullinen lasersilmä tulostettuna ja koottuna



Kuva 20. Arduino kotelointi koottuna ja johdotettuna, yllä USB-naarasliittimet

LDR-vastukset ja laserit sijoitetaan haluttuun kohtaan putkea ja kiristetään sitten M3-ruuveilla ja muttereilla.

Laseri ja LDR-vastaus asetetaan vastakkaisille puolille porttia, ja johdotus vedetään maton alta toiselle puolelle, jossa sijaitsee loput laitteistosta. Virransyöttö tapahtuu tavallisella puhelinlaturilla seinäpistokkeesta (Kuva 21).



Kuva 21. LDR-vastukset koteloinneissaan testivaiheessa, laseri osuu vastukseen.

Kohdistus oli äärimmäisen tärkeää järjestelmän toiminnan kannalta. Pieni pinta-ala vastaanottimessa yhdistettynä laserin pieneen valopisteeseen aiheutti paljon hienoviilausta, jotta tärinä ja muut tekijät eivät aiheuttaneet katkoksia toiminnassa.

# 4 TOIMINTAKUVAUS

## 4.1 Arduino ratkaisu

Arduinojärjestelmän toiminta koostuu kahdeksasta eri vaiheesta.

Portille asennettava järjestelmä toimii seuraavanlaisesti:

1. Laitteisto käynnistyy ja myymälän eri osiin sijoitetut laitteet muodostavat yhteyden toisiinsa.
2. Laserit laitetaan päälle ja vastaanottava LDR-vastus tarkistaa nykyisen valotehon ja tekee sen perusteella päätöksen missä tilassa ollaan.
3. Ohjelma tarkistaa tiheällä aikavälillä LDR-vastuksiin kohdistuvan valon määrän ja käynnistää ohjelman seuraavan vaiheen heti kuin toinen kahdesta laservastus parista on laukaistu.
4. Ohjelma tarkistaa kumpi pareista laukaisi ensin ja tekee sen perusteella päätelmän henkilön kulkusuunnasta.
5. Ohjelma muodostaa paketin joka sisältää vastaanottavan radion osoitteen ja havaitun kulkusuunnan.
6. Ohjelma jää silmukkaan mittaamaan LDR-vastuksia, kunnes molempiin vastuksiin osuu taas laseri. Silmukan aikana lasereita ei pidetä yhtäjaksoisesti päällä, vaan valoa säädellään pulssilla, jotta se olisi silmille turvallisempi, jos henkilö sattuu tuijottamaan laseriin.

Kassalle asennettava järjestelmä toimii seuraavanlaisesti, kun sille lähetetty paketti on sille saapunut:

7. Kohdan 5 paketti saapuu kassatiskin vastaanottavalle Arduinolle josta viestin sisältö tulkataan kassan tietokoneen sarjaliikenneporttiin.
8. Kassakoneella sijaitseva C#-sovellus ylläpitää kaikkia sarjaliikenteen kautta tulleita viestejä ja tallentaa niistä historiaa paikalliseen tietokantaansa.

## 4.2 NodeMCU ratkaisu

NodeMCU-järjestelmän toiminta koostuu viidestä eri vaiheesta:

1. Laitteisto käynnistyy ja laite yhdistää myymälän wifi-verkkoon.
2. Laserit laitetaan päälle ja vastaanottava LDR-vastus tarkistaa nykyisen valotehon ja tekee sen perusteella päätöksen missä tilassa ollaan.
3. Ohjelma tarkistaa tiheällä aikavälillä LDR-vastuksiin kohdistuvan valon määrän ja käynnistää ohjelman seuraavan vaiheen heti kuin toinen kahdesta laservastus parista on laukaistu.
4. Ohjelma tarkistaa kumpi pareista laukaisi ensin ja tekee sen perusteella päätelmän henkilön kulkusuunnasta.
   Ohjelma muodostaa paketin joka lähetetään palvelimella sijaitsevalla rajapinnalle.
5. Mahdolliset jatkotoimenpiteet datalle tehdään palvelimelta.

## 4.3 Järjestelmien eroavaisuudet

Arduino-ratkaisu vaatii kolme eri järjestelmää toimiakseen: Arduino portille, Arduino kassalle sekä kassan tietokoneen välittämään dataa rajapintaan. NodeMCU-ratkaisu toimii yhdellä järjestelmällä, joka kytketään liikkeen olemassa olevaan verkkorakenteeseen.

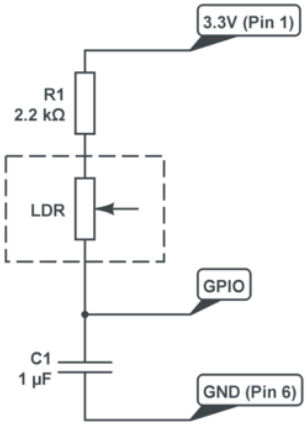
NodeMCU-ratkaisu vaikuttaa laitteiden määrän ja kokonaisrakenteen yksinkertaisuudellaan järkevämmältä ja helpommalta ratkaisulta. Arduino-pohjainen ratkaisu monimutkaisemmalla rakenteellaan toimi kuitenkin varmemmin kuin NodeMCU:lla toteutettu vastine. Suurin osa haasteista tuntui osuvan lasereiden luentavarmuuteen sekä verkkoyhteyden jatkuvaan toimintaan.

Vikoja ei saatu koskaan täysin kitkettyä, joten lopulliseen toteutukseen jätettiin Arduinoilla toimiva versio. Uskottavin selitys NodeMCU:n ongelmille tuntui olevan huono muistinhallinta laitteella joka johti pidemmällä aikavälillä laitteen muistin täyttymiseen ja täten esti laitetta muodostamasta uusia yhteyksiä.

Ohjelmistoon toteutettiin myös aikavälein toimiva ratkaisu, joka digitaalilähtöjä hyödyntämällä, mahdollisti sirun itsestään uudelleen käynnistyksen. Järjestelmä laukaisi siis oman ”reset”-tulonsa jolloin laite käynnistyi uudestaan.

NodeMCU:n 3,3 V:n logiikkajännite aiheutti myös haasteita samoilla komponenteilla joita Arduino käyttää. Tätä projektia varten valittiin mosfetit, joilla pienempi kynnysjännite, jotta lasereita saatiin ohjattua varmemmin.

NodeMCU:ssa on myös vain yksi analogitulo lukemista varten. Molemmat järjestelmät käyttävät kahta LDR-vastuksilla toteutettua ”silmää” jotka vaativat analogitulon vastuksen jännitteen mittaamista varten. LDR-vastuksien jännitettä päädyttiin mittaamaan samalla tavalla, kuin esimerkiksi Raspberry Pi:llä on yleisesti tällaiset ongelmat ratkaistu, eli perinteisellä RC-piirillä, joka hyödyntää kondensaattoria sekä vastusta, ja jolla lasketaan kauanko piirillä kestää nostaa jännite tietylle tasolle (Kuva 22).




Kuva 22. RC-piiri LDR-vastukselle, Matt Hawkins 2012 [20]

# 5 KOODIPOHJA

Projektin aikana tehtiin kaksi eri ohjelmistoversiota. Arduino ja NodeMCU käyttävät eri kirjastoja ja sisältävät pari muuta pientä eroavaisuutta, joka johti siihen, että niitä ei voinut käskyttää samalla ohjelmalla.

Arduino-versio sisältää kahden laitteen ratkaisunsa takia kaksi eri ohjelmaa, yksi portille ja toinen kassalle. NodeMCU-ratkaisu sisältää vain yhden ohjelman.

## 5.1 Arduino sisäänkäynnillä

NRF24L01-moduulien kanssa kommunikointi toteutettiin RF24-kirjastolla. Kirjastot määritellään ohjelman alussa ”#include”-riveillä (Kuva 23).

```
#include <RF24Network.h>
#include <RF24.h>
#include <SPI.h>
#include <Arduino.h>
```

Kuva 23. Arduino IDE, Arduino porttisovellus, kirjastot

Viestintä vaatii myös SPI-protokollan käytön, joten se on sisällytetty kirjastoihin. Radio vaatii määrittelyihinsä myös käytetyt digitaalilähdöt Arduinolta sekä yhdyspisteosoitteet itselleen ja vastaanottavalle toiselle päälle (Kuva 24).

```
RF24 radio(7,8); // Digitaalilähdöt

RF24Network network(radio);

const uint16_t this_node = 01;        // Tämän yhdyspisteen osoite
const uint16_t other_node = 00;       // Toisen yhdyspisteen osoite
```

Kuva 24. Arduino IDE, Arduino porttisovellus, vakioasetukset

Tietoliikenne radioiden välillä vaatii myös toimiakseen erikseen määritetyn pakettirakenteen. Paketti sisältää kaksi kokonaislukua: visitor ja dir. Visitor lähetetään aina ykkösenä ja se toimii pakettityypin tunnisteena. Dir merkitsee henkilön kulkusuuntaa ja se lähetetään ykkösenä tai kakkosena riippuen suunnasta (Kuva 25).

```
struct payload_t {                  // Paketin tietorakenne
unsigned int visitor;
unsigned int dir;
};
```

Kuva 25. Arduino IDE, Arduino porttisovellus, pakettirakenne

Summeri ja laserit määritellään myös vakioksi ohjelman alussa. Määritelmät digitaalilähdöissä määritellään Arduinon vakioista, eli esimerkiksi arvo ”6” ei välttämättä ole digitaalilähtö kuusi. Tämä on olennaisempaa NodeMCU:lla, jossa ilmiö on vahvemmin esillä (Kuva 26).

```
int buzzPin = 6;
int l1 = 3;
int l2 = 4;
```

Kuva 26. Arduino IDE, Arduino porttisovellus, summerin ja lasereiden digitaalilähdöt määriteltynä

Ohjelman ”setup”-osiossa suoritetaan kaikki kerran ajettava koodi (Kuva 27).

```
void setup(void)
{
  pinMode(buzzPin, OUTPUT);
  pinMode(l1, OUTPUT);
  pinMode(l2, OUTPUT);

  Serial.begin(9600);
  Serial.println("Transmitter");

  SPI.begin();

  radio.setPALevel(RF24_PA_LOW);
  radio.setDataRate(RF24_250KBPS);
  radio.setCRCLength(RF24_CRC_16);
  radio.enableDynamicPayloads();
  radio.setRetries (500, 5);
  radio.setAutoAck(true);
  radio.begin();

  Serial.println("Set Radio settings");

  radio.begin();

  Serial.println("Radio started");

  network.begin(/*channel*/ 108, /*node address*/ this_node);

  Serial.println("Network started");

}
```

Kuva 27. Arduino IDE, Arduino porttisovellus, ohjelman setup-osio

GPIO-portit määritellään lähdöiksi, sarjaliikenne tietokoneelle avataan, SPI-protokolla alustetaan, radion tarkemmat yhteysasetukset määritellään ja radio käynnistetään, verkon kanava määritellään ja käynnistetään.

Radion asetuksiin kuuluu: lähetysteho, lähetysnopeus, virheentarkistuksen pituus, dynaamiset paketit, uudelleenyrityskerrat sekä automaattinen paketin kuittaus.

Lähetysteho on määritelty matalaksi, jotta virrankulutus pysyisi mahdollisimman pienenä. Lähetysnopeus on asetettu alhaiseksi, jotta paketit tulisivat luotettavammin perille.

Ohjelman ”loop”-osio käsittää kaiken uudelleen ja uudelleen tapahtuvan logiikan. Ohjelma suorittaa tietyt toimenpiteet ja aloittaa sitten alusta uudestaan. Loop-osiossa käytetään funktioita jotka ovat olennaisia koodin ymmärtämisessä, joten ne tuodaan esille tässä työssä ennen loop-osioon perehtymistä.

Funktio ”waitForClear” tarkastelee analogituloihin kytkettyjä LDR-vastuksia ja tarkistaa, onko niiden arvo kynnysarvon ylittävä tai alittava arvo. Mikäli molemmat arvot ovat yli kynnysarvon, palauttaa funktion arvon ”true”, eli tosi. Mikäli toinen arvoista on alle kynnyksen, palauttaa funktio arvon ”false”, eli epätosi (Kuva 28).

```
bool waitForClear()
{
  int sensorValue1 = analogRead(A2);
  int sensorValue2 = analogRead(A4);
  Serial.print(sensorValue1);
  Serial.print(" Waiting for clear  ");
  Serial.println(sensorValue2);
  bool sta = false;
  if (sensorValue1 > threshold && sensorValue2 > threshold) {sta = true; }
  return sta;
}
```

Kuva 28. Arduino IDE, Arduino porttisovellus, waitForClear-funktio

Funktio ”sendPacket” lähettää viestin toiselle radiolle. Parametriksi sisällytetään henkilön kulkusuunta. Paketin sisältö määritellään. Vastaanottajaksi määritellään aikaisemmin vakioon talletettu toisen radion osoite, ja lopputulos kirjoitetaan verkkoon. Kirjoitustoiminto palauttaa totuusarvomuuttujan lähetyksen onnistumisesta, jonka perusteella kirjoitetaan vianhakua varten tietoa sarjaliikenneporttiin (Kuva 29).

```
void sendPacket(int dir)
{

    Serial.print("Sending...");
    payload_t payload = {1, dir};
    RF24NetworkHeader header(/*to node*/ other_node);
    bool ok = network.write(header,&payload,sizeof(payload));
    if (ok) {
      Serial.println("ok.");
    }
    else
    {
      Serial.println("failed.");
    }

}
```

Kuva 29. Arduino IDE, Arduino porttisovellus, sendPacket-funktio

Ohjelman loop-osiossa laserit laitetaan päälle ja LDR-vastuksien arvot luetaan. Mikäli jompikumpi vastuksista palauttaa arvon, joka on alle kynnysarvon, niin ohjelma siirtyy paketinlähetystilaan (Kuva 30).

```
void loop(void) {

  digitalWrite(l1, HIGH);   // Laser 1 päälle
  digitalWrite(l2, HIGH);   // Laser 2 päälle

  int sensorValue1 = analogRead(A2); //LDR 1 arvo
  int sensorValue2 = analogRead(A4); //LDR 2 arvo

  int which = 0; // Kulkusuunta

  if (sensorValue1 < threshold || sensorValue2 < threshold)
  {
   if (sensorValue1 > threshold)
   {
    which = 1;
   }
   else if (sensorValue2 > threshold)
   {
    which = 2;
   }

    Serial.println("...Buzzing, S1/S2 > " + threshold);
    digitalWrite(buzzPin, HIGH);   // Summeri päälle
    delay(10);                  // Odota 10 millisekunttia
    digitalWrite(buzzPin, LOW);    // Summeri pois päältä

    Serial.println("Sending packet now.");
    sendPacket(which); // Paketin lähetys

    smartDelay(1000);

    while (waitForClear() == false)
    {
        digitalWrite(l1, LOW);   // Laser 1 pois päältä
        digitalWrite(l2, LOW);   // Laser 2 pois päältä
        delay(500);
        digitalWrite(l1, HIGH);   // Laser 1 päälle
        digitalWrite(l2, HIGH);   // Laser 2 päälle
        delay(200);
    }

  }

  network.update();                      // Tarkista verkosta uusia saapuneita paketteja



  Serial.print(sensorValue1);
  Serial.print(" Connection ok ");
  Serial.println(sensorValue2);
}
```

Kuva 30. Arduino IDE, Arduino porttisovellus, loop-osio

Paketinlähetystilassa ohjelma tarkistaa, kumpi vastuksista katkaisi ensin, soittaa lyhyen merkkiäänen ja laukaisee paketin lähetyksen sendPacket-funktiolla. ”smartDelay”-

funktio jättää ohjelman sekunniksi odotustilaan. Tämä on toteutettu omana funktionaan koska Arduinon oma ”Delay”-funktio jättää suorittimen lukkoon sekunnin ajaksi, jolloin esimerkiksi uusien verkkopakettien saapuminen ei onnistu (Kuva 31).

```
void smartDelay(long milliseconds)
{
  unsigned long sDelay = 0;
  sDelay = millis();
    while ((millis() - sDelay) < milliseconds)
    {}
}
```

Kuva 31. Arduino IDE, Arduino porttisovellus, smartDelay-funktio

Kun paketti on lähetetty onnistuneesti, siirtyy ohjelma silmukkaan, joka kyselee jatkuvasti waitForClear-funktiolta totuusarvoa ennen kuin ohjelma voi siirtyä seuraavaan vaiheeseen. Tässä kohdassa odotellaan, että käyttäjä on kokonaan poistunut lasereiden välistä ja seuraava henkilö voidaan havaita. Pienet viiveet ja odotustoiminnat toivat testeissä hyviä tuloksia etenkin tuplalukujen minimoinnissa. Käyttäjä harvemmin, jos koskaan, luettiin portista normaali kävelyvauhdilla läpi mentäessä.

Viimeinen asia jonka ohjelma suorittaa, on tarkistus verkkoliikenteen puolelta, jossa kaikki loop-osion aikana saapunut verkkoliikenne saadaan talteen.

## 5.2 Arduino Kassatiskillä

Tiskille asennettu sisältää samat alkumäärittelyt kirjastojen ja asetusten osalta. Osoite radiolle on eri, mutta muuten koodi on melkein identtinen porttisovelluksen kanssa. Ratkaisun luonteen ansiosta ohjelma on hyvin yksinkertainen. Ohjelma seuraa jatkuvasti verkkoliikennettä uusien pakettien varalta, ja kirjoittaa heti paketin saavuttua tiedon siitä sarjaporttiin, josta kytketty kassakone saa tiedon talteen.

Sekunnin välein kirjoitetaan myös teksti ”VSK” sarjaliikenneporttiin, jotta kassakone tietää vastaanottimen olevan toimintakunnossa. Ilman sekunnin välein tulevaa kuittausta, tapahtuu ainoa päivitys tilanteessa, jossa portilta saapuu paketti (Kuva 32).

```
void loop(void){

    smartDelay(1000);

    Serial.println("VSK");

    network.update();

  while ( network.available() ) {     // Tarkista onko verkossa uusia paketteja

      RF24NetworkHeader header;        // Jos paketti saapunut, käsittele se
      payload_t payload;
      network.read(header,&payload,sizeof(payload));

      if (payload.visitor == 1)
      {
        Serial.print("CNT");
        Serial.println(payload.dir);
      }

    }

}
```

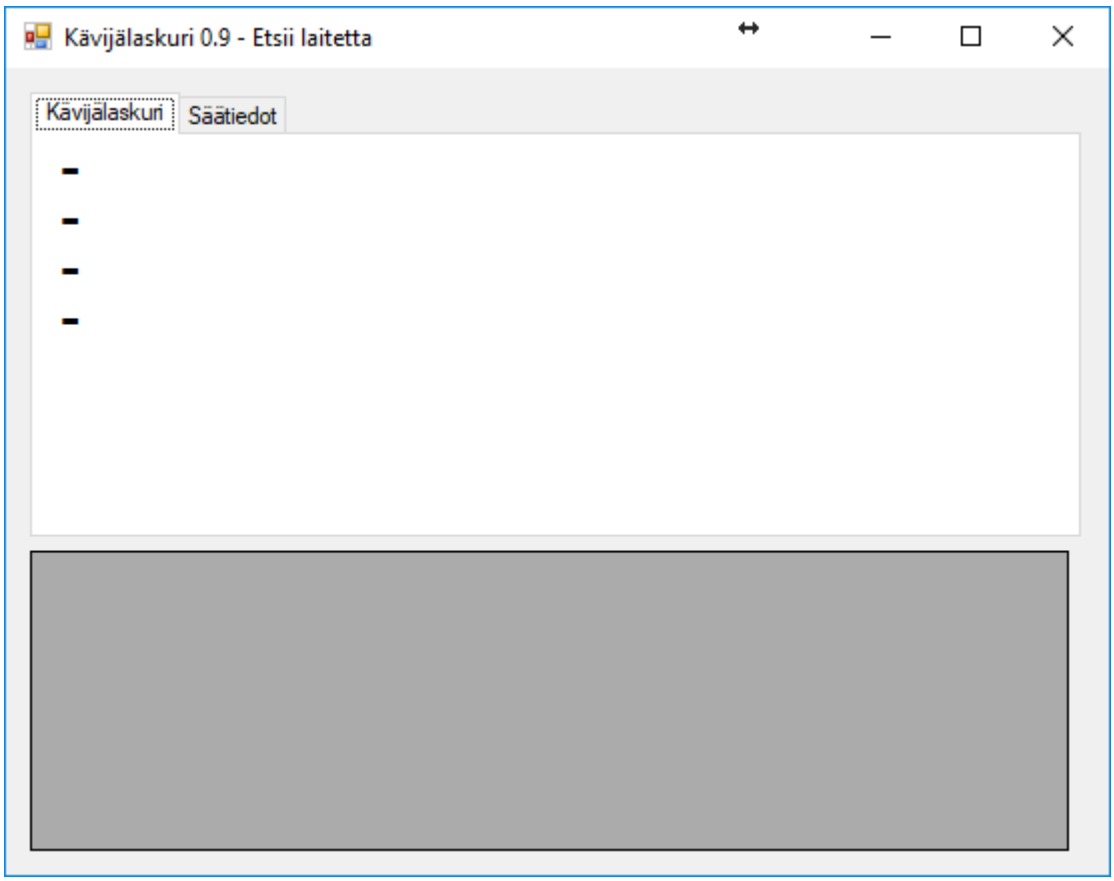
Kuva 32. Arduino IDE, Arduino kassatiskisovellus, loop-osio

Loput järjestelmän toiminnosta siirtyy kassatiskin tietokoneelle jossa c#-sovellus näyttää kokonaismäärän vieraita päivän aikana, tämän hetkisen vierailijatilanteen sekä säätiedot.

## 5.3 C# -sovellus kassatiskillä

Windowsille tehtiin yksinkertainen Windows Forms -ohjelma joka hyödyntää Net Framework 4.0:n sovelluskehystä (Kuva 33).

Ohjelman perustoimintaan kuuluu sarjaporttiliikenteen kuuntelu, tietojen tallentaminen paikalliseen SQLite-kantaan ja säätietojen hakemisen openweathermap.org rajapinnasta. Tarkoitus oli yhdistää kaikki tämä tieto edelleen lähetettäväksi Storm Motors Oy:n hallinnoimaan Google Analytics-rajapintaan. Lopullista toteutusta ei koskaan tehty, ja viimeisin toimitettu versio sisältää siis tiedon vain vastaanottavan kassatiskin tietokoneen tietokannalla.




Kuva 33. Visual Studio 2017, Winforms sovellus käynnistysikkuna

Ohjelma sisältää paljon JSON-merkkijonojen käsittelyä, englanninkielisten säätietojen kovakoodattuja kielimuunnostapauksia sekä muita juuri tähän käyttötarkoitukseen räätälöityjä parannuksia.

Ohjelma osaa etsiä oikeaa sarjaliikenneporttia johon Arduino on kytketty (Arduinon sekunnin välein lähettämää ”VSK” viestiä käytetään tähän) (Kuva 34).

```
void findComPortWithReceiver()
{

    string[] availablePorts = SerialPort.GetPortNames();

    int portsFound = availablePorts.Length;

    int currentPort = 0;



    while ((currentPort < portsFound) && !portFoundSuccessfully)
    {
        serialPort1.PortName = availablePorts[currentPort];
        serialPort1.BaudRate = 57600;

        try
        {
            serialPort1.Open();
        }
        catch (Exception ex)
        {

        }

        int timey = 0;

        while (!portFoundSuccessfully && timey < 20)
        {
            Thread.Sleep(50);
            timey++;
        }

        serialPort1.Close();

        currentPort++;
    }

}
```

Kuva 34. Visual Studio 2017, sarjaliikenneportin valitsijan logiikka

Sarjaliikenteeltä tuleva liikenne vastaanotetaan omassa tapahtumafunktiossa, joten yllä oleva logiikka yhdistyy totuusarvomuuttujan ”portFoundSuccessfully” avulla siihen.

Merkkijono ”VSK” asettaa portin löydetyksi. Merkkijonot ”CNT1” ja ”CNT2” ynnäävät ohjelman laskureita joilla ilmoitetaan muun muassa liikkeen tämänhetkinen vierastilanne. Säätiedot näytetään ohjelman pääikkunassa, mikäli ne ovat onnistuneesti haettu rajapinnasta (Kuva 35).

```
private void serialPort1_DataReceived(object sender, System.IO.Ports.SerialDataReceivedEventArgs e)
{
    string RxString = serialPort1.ReadLine().Replace("\n", "").Replace("\r", "").Replace(".", ",");

    if (RxString.Contains("VSK"))
    {
        arduinoPort = serialPort1.PortName.ToString();
        portFoundSuccessfully = true;
        //MessageBox.Show(arduinoPort);
        lastGateUpdate = DateTime.Now;
    }

    if (RxString.Contains("CNT"))
    {
        lastGateUpdate = DateTime.Now;
        lastVisitorUpdate = DateTime.Now;
        vCount++;
        SetvCountLabel("Laskuri: " + vCount.ToString());

        if (RxString.Contains("CNT1"))
        {
            inwards++;
        }
        else if (RxString.Contains("CNT2"))
        {
            outwards++;
        }

        int total = inwards - outwards;

        SetActiveVisitorsLabel("Liikkeessä vieraita nyt: " + total);
```

Kuva 35. Visual Studio 2017, sarjaliikennevastaanotto – alkuosa

Säätietoja asetettaessa tarkistetaan mahdolliset tehtävät kielikäännökset. Rajapinnan teksti saapuu englanninkielisenä, joten yleisimmille teksteille tehtiin käännökset (Kuva 36, Kuva 37).

```
try
{
    if (temp.Length > 0)
    {


        SetWeatherTypeLabel("Sää: " + Translator(wMain));
        SetWeatherDescLabel("Tarkempi määritys: " + Translator(wDesc));

        SetTempLabel("Lämpötila: " + temp + " Celciusta");
        SetPressureLabel("Ilmanpaine: " + pressure + " hPa");
        SetHumidityLabel("Kosteus: " + humidity + "%");
        SetWindLabel("Tuuli: " + wind + " m/s");
    }
}
catch (Exception ex)
{
    MessageBox.Show(ex.ToString());
}
```

Kuva 36. Visual Studio 2017, säätietojen kirjoitus näytölle

```
replacements = new Dictionary<string,string>();

replacements.Add("Snow", "Lunta");
replacements.Add("snow", "lunta");
replacements.Add("Light", "Kevyttä");
replacements.Add("light", "kevyttä");


replacements.Add("Rain", "Sadetta");
replacements.Add("rain", "sadetta");
replacements.Add("Clouds", "Pilvistä");
```

Kuva 37. Visual Studio 2017, kielikäännöstauluja

Säätietojen hakuun tarvitaan oma API-avain, jonka saa rekisteröimällä ilmaisen tilin openweathermap.org-sivustolle. Funktio ”GetWeatherData” hakee rajapinnasta merkkijonon JSON-muodossa ja ohjelma käsittelee tiedon omiin muuttujiinsa (Kuva 38).

```
public void GetWeatherData()
{
    try
    {
        WebClient c = new WebClient();
        var data = c.DownloadString("http://api.openweathermap.org/data/2.5/weather?q=espoo,fi&units=metric&APPID=cb7cf
        //Console.WriteLine(data);

        var rootObject = JsonConvert.DeserializeObject<WeatherData.RootObject>(data);
        temp = rootObject.main.temp;
        pressure = rootObject.main.pressure;
        humidity = rootObject.main.humidity;
        wind = rootObject.wind.speed;
        name = rootObject.name;


        wMain = rootObject.weather.First().main;
        wDesc = rootObject.weather.First().description;

        //MessageBox.Show(name);
        lastWeatherUpdate = DateTime.Now;




    }
    catch(Exception ex)
    {
        MessageBox.Show(ex.ToString());
    }
}
```

Kuva 38. Visual Studio 2017, säätietojen rajapintahaku muuttujiin

Testauksessa asetettiin kaupungiksi Espoo. JSON-merkkijonojen lukemiseen hyödynnettiin Newtonsoftin JSON-kirjastoja.

## 5.4 NodeMCU sisäänkäynnillä

NodeMCU ohjelma kirjoitettiin Arduino IDE:llä hyvin samankaltaisesti Arduino-ohjelmien kanssa. LUA-kieli on oletus NodeMCU:lla, mutta tätä lähestymistapaa hyödyntämällä koodaus toimii myös C++-kielellä.

Kirjastot sisältävät ESP8266:lle tarvittavat Wifi ja http-kutsujen vaatimat tiedostotuonnit (Kuva 39).

```
#include <Arduino.h>

#include <ESP8266WiFi.h>
#include <ESP8266WiFiMulti.h>

#include <ESP8266HTTPClient.h>
```

Kuva 39. Arduino IDE, NodeMCU portti, kirjastot

Digitaalilähdöt ja Arduinosta poiketen myös LDR-vastusten tulojen numerointi asetetaan vakioarvoihin. LDR-vastukset luetaan RC-piirillä, kuten aikaisemmin on mainittu, joten tämä vaatii analogiluennan sijasta digitaalitulon (Kuva 40).

```
int photocellPin1 = D3;
int photocellPin2 = D4;
int buzzPin = D8;
int l1 = D6;
int l2 = D7;
```

Kuva 40. Arduino IDE, NodeMCU portti, vakioarvojen määrittelyt

Setup-osiossa määritellään sarjaliikenteen asetukset sekä digitaalilähdöt ja tulot. Wifi-kirjastolla luodaan yhteys langattomaan verkkoon verkon nimellä ja tunnistusavaimella.

Laserit vilkkuvat 8 s:n ajan, jotta laitteistolla on aikaa saada wifi-moduuli käyntiin käynnistyksen yhteydessä (Kuva 41).

```
void setup() {
  // put your setup code here, to run once:

  Serial.begin(115200);
   // Serial.setDebugOutput(true);

  pinMode(buzzPin, OUTPUT);
  pinMode(l1, OUTPUT);
  pinMode(l2, OUTPUT);

    Serial.println();
    Serial.println();
    Serial.println();

    for(uint8_t t = 8; t > 0; t--) {
        Serial.printf("[SETUP] WAIT %d...\n", t);
        Serial.flush();
          digitalWrite(l1, HIGH);   // Laser 1 päälle
          digitalWrite(l2, HIGH);   // Laser 2 päälle
        delay(1000);
          digitalWrite(l1, LOW);   // Laser 1 pois päältä
          digitalWrite(l2, LOW);   // Laser 2 pois päältä
    }

    WiFiMulti.addAP("Koti_D3E9", "J84FB78BT7U78");

}
```

Kuva 41. Arduino IDE, NodeMCU portti, setup-osio

Aikaisemmin mainittu RC-piirin hyödyntäminen analogilukemisen korvikkeena on sijoitettu funktioon RCtime. Funktio asettaa mitattavan digitaalilähdön lähettäväksi ja asettaa sen logiikkajännitteen nollaksi.

Ohjelma odottaa 25 ms ja asettaa sitten saman digitaalilähdön vastaanottavaan tilaan ja seuraa kauanko sillä kestää palautua tasolle joka vastaa logiikkatasolla korkeaa signaalia. Mitattu aika nollasta 3,3~ V:iin palautetaan funktiosta (Kuva 42).

```
int RCtime(int RCpin) {
  int reading = 0;  // start with 0

  pinMode(RCpin, OUTPUT);
  digitalWrite(RCpin, LOW);

  delay(25);

  pinMode(RCpin, INPUT);
  while (digitalRead(RCpin) == LOW) {
    reading++;

    if (reading == 30000) {
      break;
    }
  }


  return reading;
}
```

Kuva 42. Arduino IDE, NodeMCU portti, RCtime-funktio

Ohjelma sisältää samankaltaisia toimintoja joita Arduino-ohjelmassa aikaisemmin tarkasteltiin. Funktiot smartDelay ja waitForClear ovat miltei identtiset. Uusia käsitteitä tulee tietojen lähetyksessä, jossa hyödynnetään http-asiakasohjelmaa.

Funktio ”httpHit” ottaa parametriksi henkilön kulkusuunnan ja lähettää tiedon webpalvelimella sijaitsevalle rajapintapalvelulle. Tässä lähestymistavassa menetellään monin tavoin samalla tavalla kuin kassatiskin Arduinon kanssa, mutta sarjaliikenteen sijasta tieto lähetetään http-protokollalla palvelimelle käsiteltäväksi (Kuva 43).

```
void httpHit(int dir)
{
HTTPClient http;

        Serial.print("[HTTP] begin...\n");

        String dire = String(dir);

        String url = "http://pritek.fi/storm/hit.php?dir=" + dire;

        Serial.println(url);

        http.begin(url); //HTTP

        Serial.print("[HTTP] GET...\n");

        int httpCode = http.GET();

        if(httpCode > 0) {

            Serial.printf("[HTTP] GET... code: %d\n", httpCode);

            if(httpCode == HTTP_CODE_OK) {
                String payload = http.getString();
                Serial.println(payload);
            }
        } else {
            Serial.printf("[HTTP] GET... failed, error: %s\n", http.errorToString(httpCode).c_str());
        }

        http.end();

}
```

Kuva 43. Arduino IDE, NodeMCU portti, httpHit-funktio

Kirjaston http.GET()-funktio palauttaa negatiivisen arvon mikäli lähetys on epäonnistunut, muissa tapauksissa lähetys meni perille. Mikäli vastaus on 200, tai vakioarvoon asetettu HTTP_CODE_OK (200), niin lähetys on vastaanotettu onnistuneesti.

Tieto on nyt tallella palvelimen MYSQL-palvelimella.

# 6 LOPUKSI

Opinnäytetyö koostui asiakastapaamisesta, elektroniikkasuunnittelusta, mallintamisesta ja ohjelmistokehityksestä. Ratkaisujen valinnoissa sai käyttää omaa harkintaa ja työ sujui ilman suurempia rajoituksia.

Työssä oli tavoitteena pystyä laskemaan kävijämääriä hetkellisesti ja päiväkohtaisesti, tutkimaan ulkoisten tekijöiden vaikutusta kävijämääriin sekä mahdollistaa tiedon automaattinen vienti muihin järjestelmiin. Kaikissa tavoitteissa onnistuttiin perustasolla ja teknisesti katsoen kaikki haluttu toiminnallisuus oli joko täysin valmis, tai vaihtoehtoisesti hyvin lähellä valmista.

Radioliikenteen ja tunnistuksen osalta onnistuttiin odotettua paremmin, mutta kaapelointi ja ohjelmiston käytettävyys kassallta onnistui odotettua haasteellisemmaksi kassahenkilökunnalle. Laserit ja LDR-vastukset toimivat yllättävän hyvin, mutta ne osoittautuivat samanaikaisesti erittäin vika-alttiiksi vaihtoehdoksi ympäristöön, jossa on mahdoton kiinnittää niitä erittäin tukevasti paikoilleen.

Projektin aikana saatiin kokemusta ja ideoita miten mahdolliset tulevaisuudessa toteutettavat samankaltaiset projektit voisi paremmin toteuttaa. Useiden eri laitekokonaisuuksien väliltä valittiin ratkaisu, jonka tekeminen vaati enemmän aikaa, osia ja suunnittelua. Helpommin toteutettavissa olevasta ratkaisusta luovuttiin laitteessa esiintyneiden muistinhallintapuolen ongelmien takia. Mikäli aikaa olisi ollut enemmän, olisivat nämäkin ongelmat todennäköisesti voitu ratkaista jollakin tapaa.

Jos projekti aloitettaisiin alusta nyt ja kaikki tästä opittu tieto olisi käytettävissä, niin ratkaisuksi valittaisiin mahdollisesti laserit, joissa on viivatyyppinen kohdennus. LDR-vastusten vastaanottava linssi suunniteltaisiin suuremmaksi. Mikrokontrolleriksi valittaisiin ESP8266:n seuraaja ESP32, jossa esimerkiksi muistiin liittyviä ongelmia on korjattu  ja joka on muutenkin nopeampi. Työpöytäsovellukset jätettäisiin kokonaan pois ja kaikki palvelinpuolen kehitys toteutettaisiin verkkopalvelimelle rajapintaan.

Lopullinen tuotekehityksen tila jäi konseptitasolle, mutta toimivaan tuotteeseen on melkein valmiit ratkaisut ja käytännöllisesti katsoen kaikki yksittäiset järjestelmän osat on testattu ja todettu toimiviksi. Google Analytics integraatio on ainoa osa järjestelmää jonka kehitys jäi kokonaan tekemättä.

# LÄHTEET


[1] “A visit to the Arduino factories”, viitattu 7.6.2018. Saatavissa: https://www.domusweb.it/en/design/2013/03/15/a-visit-to-the-arduino-factories.html

[2] “Nano V3.0 Board (Arduino Nano V3.0-Compatible)”, viitattu 31.5.2018. Saatavissa: https://core-electronics.com.au/nano-v3-0-board-arduino-nano-v3-0-compatible.html

[3] “Arduino Products”, viitattu 7.6.2018. Saatavissa: https://www.arduino.cc/en/Main/Products

[4] “ARDUINO UNO REV3”, viitattu 31.5.2018. Saatavissa: https://store.arduino.cc/usa/arduino-uno-rev3

[5] “ELEC-0108 NodeMcu Lua Wifi v3 Development Board (ESP8266)”, viitattu 31.5.2018. Saatavissa: https://www.marginallyclever.com/product/nodemcu-lua-wifi-v3-development-board-esp8266/

[6] “ESP8266”, viitattu 7.6.2018. Saatavissa: https://en.wikipedia.org/wiki/ESP8266

[7] “nRF24L01, Ultra low power 2.4GHz RF Transceiver IC”, viitattu 7.6.2018. Saatavissa: http://www.nordicsemi.com/eng/Products/2.4GHz-RF/nRF24L01

[8] “nRF24L01 Wireless Transceiver Arduino Raspberry Pi”, viitattu 31.5.2018. Saatavissa: https://www.makerlab-electronics.com/product/nrf24l01-wireless-transceiver-arduino-raspberry-pi/

[9] “ESP8266 WiFi-moduuli”, viitattu 31.5.2018. Saatavissa: http://ihmevekotin.fi/product/273_esp8266-wifi-moduuli

[10] “5V LASER DOT DIODE MODULE HEAD”, viitattu 31.5.2018. Saatavissa: http://www.robotpark.com/5V-Laser-Dot-Diode-Module-Head

[11] “LDR Light Dependent Resistor”, viitattu 31.5.2018. Saatavissa: https://iprototype.nl/products/components/sensors/ldr

[12] “MOSFET”, viitattu 7.6.2018. Saatavissa: https://fi.wikipedia.org/wiki/MOSFET

[13] “2N7000 Logic Level N-Channel Mosfet”, viitattu 31.5.2018. Saatavissa: https://www.addicore.com/2N7000-p/ad258.htm

[14] “AMS1117-3.3 LDO 800MA DC 5V to 3.3V Step-Down Power Supply Module”, viitattu 31.5.2018. Saatavissa: https://robu.in/product/ams1117-3-3-ldo-800ma-dc-5v-3-3v-step-power-supply-module/

[15] “CONDENSATOR 10MF 450V DC”, viitattu 31.5.2018. Saatavissa: https://www.htfelectronics.nl/nl/condensator-10f-450v-dc.html

[16] “Piezoelectric Buzzer”, viitattu 31.5.2018. Saatavissa: https://squishycircuits.com/products/piezoelectric-buzzer

[17] “Extrusion deposition: Fused Deposition Modeling (FDM)”, viitattu 31.5.2018. Saatavissa: https://www.additive3d.com/extrusion-deposition-fused-deposition-modeling-fdm/

[18] “Ultimaker Debuts Ultimaker 2 3D Printer With Open Source Cura Software and YouMagine Website”, viitattu 31.5.2018. Saatavissa: https://hothardware.com/news/ultimaker-debuts-ultimaker-2-3d-printer-with-open-source-cura-software-and-youmagine-website

[19] “Reprap Wilson II complete 3D printer kit”, viitattu 31.5.2018. Saatavissa: https://www.tindie.com/products/mjrice/reprap-wilson-ii-complete-3d-printer-kit/

[20] Matt Hawkins 2012, “Reading Analogue Sensors With One GPIO Pin”, viitattu 31.5.2018. Saatavissa: https://www.raspberrypi-spy.co.uk/2012/08/reading-analogue-sensors-with-one-gpio-pin/