Joonas Nislin

Kasvien monitorointi- ja kastelujärjestelmän automatisointi IoT-teknologian avulla

Tietotekniikan
pro gradu -tutkielma
14. maaliskuuta 2024

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Kokkolan yliopistokeskus Chydenius

Tekijä: Joonas Nislin
Yhteystiedot: joonas.nislin@hotmail.com
Puhelinnumero: 040-591 4198
Ohjaaja: Lasse Harjumaa
Työn nimi: Kasvien monitorointi- ja kastelujärjestelmän automatisointi IoT-teknologian avulla
Title in English: Plant monitoring and irrigation system automation using IoT technology
Työ: Tietotekniikan pro gradu -tutkielma
Sivumäärä: 60+2
Tiivistelmä: Kasvien monitorointiin ja kastelun automatisointiin liittyy monia hyötyjä. Automaatio vähentää manuaalista työtä, vedenkulutusta ja mahdollistaa kasvien tilan seuraamisen reaaliaikaisesti. Tämän pro gradu -tutkielman pääkysymys on "Mitä on otettava huomioon kastelun automatisoinnissa IoT-teknologioiden avulla?". Tutkielma on konstruktiivinen, jonka alussa tehdään kirjallisuuskatsaus tutkimuskentän nykytilan ja tyypillisen kastelujärjestelmän toteutuksen selvittämiseksi. Tutkielman tuloksena syntyy kastelujärjestelmän, joka mahdollistaa kasvien monitoroinnin ja kastelun automatisoinnin.
Avainsanat: automatisoitu kastelu, mittauslaite, monitorointijärjestelmä
Abstract: Automation of plant monitoring and watering has many benefits. Automation reduces manual work, water consumption and enables real-time monitoring of the plants condition. The main question of this thesis is "What should be considered in automating irrigation using IoT-technologies". This study is constructive and literature review is used to investigate the current state of the research field and typical implementation of irrigation systems. Outcome of this study is an irrigation system, that enables plant monitoring and automated irrigation.
Keywords: automated irrigation, measuring instrument, monitoring system

Sanasto

AES	Advanced Encryption Standard. Laajalti käytetty lohkosalausmenetelmä.
AMQP	Advanced Message Queuing Protocol. Sovellusprotokolla sovellusten väliseen viestintään.
AP-mode	Access Point mode. Tukiasematila, mahdollistaa muiden laitteiden yhteyden muodostuksen tukiasemaan.
APK	Android Application Package. Sovelluspakettien tiedostomuoto, jota käytetään Android-käyttöjärjestelmässä.
AWS	Amazon Web Services. Amazonin pilvipalvelualusta.
Big data	Massadata. Termi jatkuvasti lisääntyville valtaville tietomassoille.
BLE	Bluetooth Low Energy. Lyhyen kantaman langaton teknologia.
CSB	Chip Select Bar. SPI-väylän pinni, jolla valitaan SPI-laite.
EAS	Expo Application Services. Expo-kehitysalustan tarjoama pilvipalvelu sovellusten kehittämiselle, testaamiselle ja julkaisemiselle.
FMIS	Farm management information systems. Maatalouden hallintatietojärjestelmä.
GND	Ground. Suojamaadoitus, sähkölaitteiden vikasuojausmenetelmä.
GPIO	General Purpose Input/Output. Yleiskäyttöinen pinni, joka voi vastaanottaa tai lähettää tietoa.

HTTPS	Hypertext Transfer Protocol Secure. Protokolla, joka mahdollistaa turvallisen tiedonsiirron verkkosivustojen välillä.
IDE	Integrated development environment. Ohjelmointiympäristö.
IEEE	Institute of Electrical and Electronics Engineers. Kansainvälinen sähkö- ja tietotekniikan tieteellinen ja ammatillinen organisaatio.
I2C	Inter-Integrated Circuit. Yksinkertainen kaksisuuntainen tiedonsiirtoväylä.
ICT	Information and communication technology. Tieto- ja viestintätekniikka.
IaaS	Infrastructure as a service. Infrastruktuuripalvelut, pilvipalvelu tyyppi.
IoT	Internet of Things. Esineiden internet.
LTE	Long Term Evolution. Neljännen sukupolven (4G) langaton tiedonsiirtotekniikka.
LoRa	Long Range. Pitkän kantaman verkkoteknologia.
LoRaWAN	Long Range, Wide Area Network. Pitkän kantaman langaton tiedonsiirtoverkko.
LPWAN	Low Power Wide Area Network. Matalatehoinen laajaverkko.
M2M	Machine-to-Machine. Koneliittymä.
Mesh-topology	Silmukkaverkko, jossa jokainen laite välittää tietoa toisilleen.

MQTT	MQ Telemetry Transport. Kevyt ja tehokas viestintäprotokolla IoT-sovellusten väliseen kommunikointiin.
NB-IoT	Narrow Band Internet of Things. Kapeaa taajuusaluetta hyödyntävä IoT-verkko.
NoSQL	Not only SQL. Käsite, joka kuvaa perinteisestä relaatiomallista poikkeavia tietokantoja.
NPM	Node Package Manager. Paketinhallintajärjestelmä.
OTA	Over-the-air. Langaton ohjelmistopäivitys.
PaaS	Platform as a service. Alustapalvelut, pilvipalvelu tyyppi.
SIG	Bluetooth Special Interest Group. Bluetooth standardointijärjestö.
SSID	Service Set Identifier. Langattoman lähiverkon tunniste.
SPI	Serial Peripheral Interface. Synkroninen sarjaväylä.
SaaS	Software as a service. Sovelluspalvelut, pilvipalvelu tyyppi.
SQL	Structured Query Language. Standardoitu tietokannan kyselykieli.
Star-topology	Tähtiverkko, jossa laitteet välittävät tietoa keskusyksikölle.
TLS	Transport Layer Security. Protokolla, jolla salataan verkon tietoliikenne.
WLAN	Wireless Local Area Network. Langaton lähiverkko.
WPAN	Wireless Personal Area Network. Langaton likiverkko.
3GPP	3rd Generation Partnership Project. Usean standardointijärjestön yhteistyöorganisaatio.

Sisällys

1 Johdanto

Viher- ja hyötykasvit tarjoavat kasvattajalleen monenlaisia etuja. Ne parantavat huoneen sisäilman laatua, lieventävät stressiä ja toimivat sisustuselementteinä. Hyötykasvien käyttö ruoanlaitossa lisää ruokien monipuolisuutta rikastuttamalla makua ja tuoksuja. Kasvien kasvattamisessa tulee huomioida erityisesti veden- ja valonsaanti sekä ilmankosteus ja lämpötila, jotta kasvit kasvavat ja tuottavat satoa. Ulkoilmassa olevien kasvien kastelua vaikeuttavat myös sään vaikutukset kuten kuivuus, sateet ja lämpötilan vaihtelut. Kasvien kastelun helpottamiseksi on kehitetty kastelujärjestelmiä, mutta voisiko kasvien monitorointi- ja kastelujärjestelmän kehittää ja automatisoida esineiden internetin avulla?

Tulevaisuuden työvoimapula ja resurssien, kuten veden väheneminen kuivuuden takia voivat aiheuttaa maataloudelle huomattavia ongelmia. Kasvien kastelu perinteisillä kastelumenetelmillä johtaa usein veden runsaaseen tuhlaamiseen. Kasvien monitorointi- ja kastelujärjestelmä poistaa tarpeen ihmisen välttämättömästä osallistumisesta kastelun suorittamiseen ja mahdollistaa juuri oikean vesimäärän kohdistamisen kasveille, joka vähentää vedenkulutusta.

Tutkielman pääkysymys on "Mitä on otettava huomioon kastelun automatisoinnissa IoT-teknologioiden avulla?". Pääkysymykseen vastataan kahdella alakysymyksellä: "Mitä hyötyjä ja haasteita kastelun automatisointi tuo?" ja "Millaisella IoT-konstruktiolla kasvien kasvattaja voi suorittaa kastelun automatisoinnin ja saada sen tuomat hyödyt?". Tutkielma on konstruktiivinen, jonka alussa tehdään kirjallisuuskatsaus teoriaan. Kirjallisuuskatsauksella selvitetään tutkimuskentän nykytila ja tyypillisen kastelujärjestelmän toteutus. Kirjallisuuskatsauksen lähteet koostuvat tieteellisistä artikkeleista ja julkaisuista. Kirjallisuuskatsaukseen valikoidut tutkimukset ja niiden toteutukset pyrkivät vähentämään maanviljelijöiden manuaalista työtä, tehostamaan kastelua ja vähentämään vedenkulutusta. Tutkielman kehitysvaiheen aikana suunnitellaan ja toteutetaan kasvien monitorointi- ja kastelujärjestelmän prototyyppi. Prototyypin tarkoituksena on mahdollistaa yksittäisen kasvin monitorointi ja kastelun automatisointi. Prototyyppi koostuu mittauslaitteesta, monitorointisovelluksesta ja taustajärjestelmästä. Mittauslaitteen päätehtävänä on tarkkailla maaperän kosteutta ja ohjata vesipumppua, kun maaperän kosteus

laskee asetetun raja-arvon alapuolelle. Mittauslaite pystyy mittaamaan myös kasvin valonsaannin, vesisäiliön vedenmäärän sekä ilman lämpötilan, -kosteuden ja -paineen. Mittausoperaatiosta kerätty data siirretään Google Firebase-palvelun Realtime Database-tietokantaan. Kaikki mittauslaitteelta lähetettävät tiedot salataan AES-lohkosalausmenetelmällä, ja ne puretaan ihmiselle luettavaan muotoon monitorointisovelluksessa.

Tutkielman tuloksena syntyy kastelujärjestelmän prototyyppi, joka mahdollistaa pienimuotoisen kasvien monitoroinnin ja kastelun automatisoinnin. Prototyyppi hyödyntää helposti käyttöönotettavaa Google Firebase-taustajärjestelmää, mikä poistaa tarpeen palvelimen ylläpidolle. Maaperän kosteusmittaukset ja mahdollinen kastelu suoritetaan 12 tunnin välein, kun taas muiden sensorien mittaukset tehdään 29 minuutin välein. Kasvien kasvattaja pystyy tarkastelemaan monitorointisovelluksen avulla mittauslaitteen viimeisimpiä mittauksia sekä lokitietoja mittauslaitteen suorittamista tapahtumista.

Luvussa 2 esitellään suunnittelutieteen määritelmä ja tarkastellaan siihen liittyviä käsitteitä sekä ohjeita sen soveltamiseen. Luvussa esitellään myös millaisia tutkimustuloksia suunnittelutieteen tutkimukset tarjoavat.

Luvussa 3 tutustutaan esineiden internetin määritelmään ja tarkastellaan sen yleistä nelitasoista arkkitehtuuria sekä siihen kuuluvia tasoja. Ensin esitellään, millaisia elektronisia laitteita IoT-mittauslaitteet tarvitsevat mittausten suorittamiseen ja millaisia viestintäteknologioita voidaan hyödyntää kerätyn datan siirtämiseen taustajärjestelmään. Lisäksi esitetään, millaisia nämä pilvessä toimivat taustajärjestelmät voivat olla ja annetaan muutama esimerkki IoT-sovelluksista.

Luvussa 4 tutustutaan tutkimuksiin, joissa käsitellään kastelun automatisointia IoT:n avulla ja tarkastellaan tutkimuksissa ehdotettuja ratkaisuja. Luvussa tarkastellaan kastelun automatisoinnin hyötyjä ja haasteita sekä motivaatiota toteuttaa monitorointi- ja kastelujärjestelmä.

Luvussa 5 esitellään tutkielman aikana toteutetun kasvien monitorointi- ja kastelujärjestelmän prototyypin arkkitehtuuri, toimintamalli, mittauslaitteen laitteistokokoonpano sekä mittauslaitteen ja monitorointisovelluksen kehitetyt ohjelmistot.

Luvussa 6 tarkastellaan tutkielman tuloksia sekä pohditaan, kuinka prototyypin suunnittelu ja toteutus onnistui. Esitetään myös prototyypin jatkokehitysaiheet. Lisäksi arvioidaan tutkielman toteutusta konstruktiivisella tutkimusotteella ja pohditaan muita maatalouteen liittyviä tutkimusaiheita.

Luku 7 muodostaa yhteenvedon, jossa tiivistetään tutkielman sisältö.

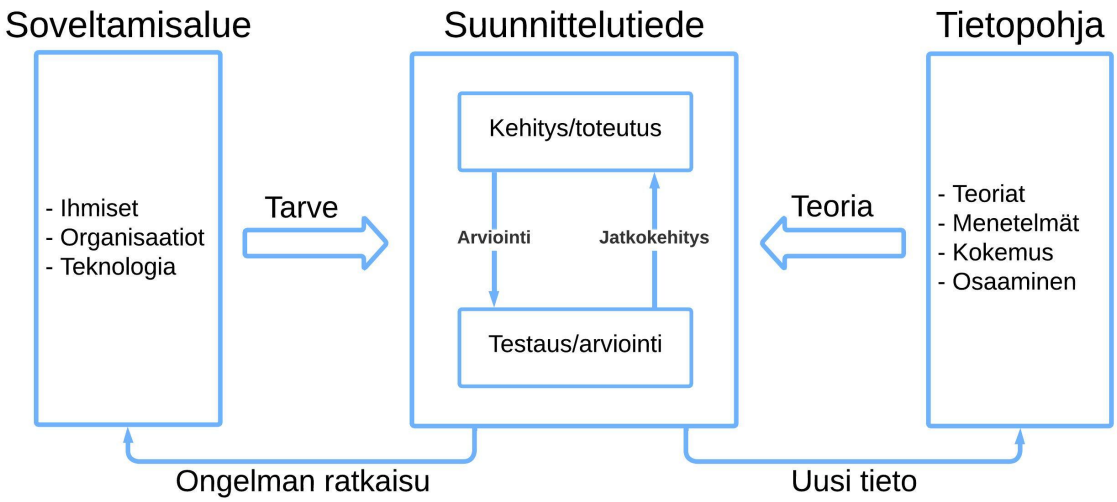
2 Konstruktiivinen tutkimus

Tässä tutkielmassa käytetään konstruktiivista tutkimusotetta suunnittelutieteen viitekehyksessä. Suunnittelutiede on informaatioteknologian tutkimuksissa yleisesti hyödynnetty tutkimusote, joka pyrkii ratkaisemaan ja lisäämään ymmärrystä ongelma-alueesta suunnittelutyön tuloksena syntyvän artefaktin avulla. [13] Nämä artefaktit voivat olla konstruktioita (sanasto ja symbolit), malleja (abstraktiot ja esitykset), menetelmiä (algoritmit ja käytännöt) tai instansseja (toteutetut ja prototyyppijärjestelmät). Konstruktiot määrittelevät peruskäsitteet ja sanaston, jolla ongelmat ja ratkaisut määritellään ja kommunikoidaan [7]. Mallit hyödyntävät konstruktioita esittämään reaalimaailman konteksteja ongelma-alueesta sekä ratkaisuista. Menetelmät määrittelevät ratkaisuissa käytetyt algoritmit ja käytännöt. Instanssit osoittavat, että kaikkia muita edellä mainittuja artefakteja voidaan toteuttaa toimivassa järjestelmässä. Suunnittelutiede on siis pohjimmiltaan ongelmanratkaisu prosessi, jonka tavoitteena on saavuttaa jokin hyöty, jolla ongelman voi ratkaista.

Hevner et al. [13] esittelevät artikkelissaan käsitteellisen kehyksen, jolla informaatioteknologian tutkimusta voidaan ymmmärtää, toteuttaa ja arvioida. Kehys on jaettu kolmeen osa-alueeseen: soveltamisalue, suunnittelutiede ja tietopohja. Soveltamisalue määrittelee ongelma-alueen, jossa tutkimuksessa ratkaistava ongelma tai ilmiö sijaitsee. Ongelma-alue voi koostua ihmisistä, organisaatioista tai niiden olemassa olevista tai suunnitelluista teknologioista. Nämä muuttujat määrittelevät tavoitteet, tehtävät, ongelmat ja mahdollisuudet, joista syntyy tarve suunnittelutieteen soveltamiselle. Suunnittelutieteen osa-alue pyrkii ratkaisemaan tämän tarpeen artefaktien toteutuksen ja arvioinnin avulla. Jos toteutetussa artefaktissa huomataan puutteita arvioinnin aikana, ne kuvataan yleensä tutkimuksen jatkokehitysaiheina. Viimeinen osa-alue eli tietopohja, tarjoaa jo olemassa olevaa tietoa tai kokemusta ongelma-alueesta artefaktin toteutukseen. Valmis artefakti puolestaan tarjoaa uutta tietoa, joka voidaan lisätä tietopohjaan. Käsitteellinen kehys esitellään kuvassa 2.1.

Suunnittelutieteen käsitteellisen kehyksen lisäksi Hevner et al. [13] esittävät artikkelissaan seitsemän ohjeen mallin, jolla suunnittelutiedettä voi soveltaa informaatioteknologian tutkimuksissa. Malli tarjoaa konkreettisia ohjeita, jotka mahdollistavat tutkijoiden ja ammattilaisten ymmärtää ja käsitellä ongelmia, jotka liittyvät

tietojärjestelmien kehittämiseen ja onnistuneeseen toteuttamiseen.



Kuva 2.1: Suunnittelutiede, käsitteellinen kehys. Hevner et al. [13], muokattu.

Ensimmäisen ohjeen mukaan suunnittelutiedettä soveltavan tutkimuksen tulee tuottaa tarkoituksenmukainen artefakti, tutkimuksessa määritellyn ongelman ratkaisemiseksi. Artefaktin tulee olla perusteellisesti kuvattu, jotta sen toteutus ja käyttäminen soveltamisalueella on mahdollista. Toinen ohje määrittelee tutkimuksen tavoitteen. Tavoitteena tulee olla sellaisen tiedon ja ymmärryksen hankkiminen, joka mahdollistaa teknologiaan perustuvien ratkaisujen kehittämisen ja toteuttamisen.

Kolmantena ohjeena on tutkimuksen tuotoksen eli artefaktin arviointi. Artefaktia voidaan arvioida toimivuuden, suorituskyvyn, luotettavuuden, käytettävyyden tai muiden laatua mittaavien ominaisuuksien avulla. Artefaktin toiminnalliset vaatimukset sekä niiden hyväksyntäkriteerit voivat toimia myös yhtenä artefaktin arvioinnin mittarina. Neljäs ohje käsittelee tutkimuksen tieteellistä arvoa. Tutkimukset jotka soveltavat suunnittelutiedettä tutkimusotteena, voivat luoda uutta tieteellistä arvoa artefaktien eli tuotosten kautta. Tuotos voi laajentaa tietopohjaa tai se voi hyödyntää olemassa olevaa tietoa uudella tavalla.

Viides ohje painottaa tarkkuuden ja perusteellisuuden tärkeyttä suunnittelutieteen tutkimuksissa. Tämä voidaan saavuttaa hyödyntämällä olemassa olevaa tietopohjaa sekä tutkimusmenetelmiä kun artefaktia rakennetaan ja arvioidaan. Kuudes ohje korostaa suunnittelutyön aikana tapahtuvaa huolellista perehtymistä aiemmin toteutettuihin ratkaisuihin. Aiempia tutkimuksia ja niiden tuloksia tutkimalla, voi

löytää yhtäläisyyksiä ratkaisujen välillä, joka auttaa oman tutkimusongelman ratkaisun löytämisessä.

Seitsemäs ohje opastaa tutkimuksen tulosten julkaisussa. Tutkimustulokset on julkaistava sellaisessa muodossa, että ammattilaiset voivat hyötyä artefaktin tarjoamista eduista ja tutkijoiden on mahdollista rakentaa kumulatiivista tietopohjaa artefaktin tuloksista jatkotutkimuksia ja arviointeja varten. Nämä ohjeet esitetään taulukossa 2.1.

Taulukko 2.1: Suunnittelutieteen soveltamisen ohjeet tutkimuksissa Hevner et al. [13], muokattu

Ohje	Kuvaus
Ohje 1: Tuotos on artefakti	Tutkimuksen tuloksena syntyy artefakti, kuten konstruktio, malli, menetelmä tai instanssi
Ohje 2: Ongelman relevanssi	Tutkimuksen tavoitteena on ratkaista jokin ongelma teknologiapohjaisella ratkaisulla
Ohje 3: Tuotoksen arviointi	Artefaktin hyödyllisyys, laatu ja tehokkuus on osoitettava testaamalla tuotos huolellisesti
Ohje 4: Tieteellinen arvo	Tutkimuksen on tarjottava selkeää ja todennettavissa olevaa arvoa artefaktin, tietämyksen tai menetelmien muodossa
Ohje 5: Perusteellisuus	Artefaktin rakentamisessa ja arvioinnissa tulee soveltaa tarkkoja sekä perusteellisia menetelmiä
Ohje 6: Suunnittelu osana tiedonetsintää	Artefaktin toteuttamisessa tulee hyödyntää olemassa olevaa tietoa haluttujen tavoitteiden saavuttamiseksi osana ongelman ratkaisua
Ohje 7: Tulosten julkaisu	Tutkimuksen tulokset tulee julkaista siten, että teknologisesti orientoituneet ihmiset ja päätöksentekijät voivat hyödyntää niitä

Baskerville et al. [7] käsittelevät artikkelissaan suunnittelutieteen tutkimustuloksia ja erityisesti sitä, miten löytää tasapaino toteutetun artefaktin ja teorian välillä. Koska suunnittelutieteen tutkimustulokset ovat osana tieteen ja teknologian evolutiivista vuorovaikutusta, niiden julkaisussa tulee huomioida seuraavia asioita. Tutkijan tulee arvioida tieteellisen ja teknologisen tietopohjan kypsyys sekä esittää selkeästi

mitä uutta tietoa tutkimustulokset antavat. Artefakti tulee kuvata selkeästi osoittamalla sen uutuus ja käytännölliset parannukset, jonka jälkeen artefaktia tulee peilata teoriaan. Artefaktin ja teorian vaikutusten tulee olla kumulatiivisia, yhtenäisiä ja erottamattomia.

Tämä tutkielma hyödyntää tässä luvussa esitettyä suunnittelutieteen viitekehystä. Soveltamisalueena toimii kasvien kastelu ja IoT-prototypointi. Tutkielman tavoitteena on ratkaista kasvien monitorointi ja kastelun automatisointi IoT-konstruktiolla. Tietopohja koostuu tieteellisistä artikkeleista ja tutkimuksista, joissa toteutettiin automatisoitu kasvien kastelujärjestelmä IoT:n avulla. Tutkielman kehitystyön tuotoksena syntyy pienimuotoinen kastelujärjestelmän prototyyppi, joka pyrkii ratkaisemaan kustannustehokkaasti yksittäisen kasvin monitoroinnin ja kastelun automatisoinnin. Tutkimuksella pyritään myös ajantasaistamaan olemassa olevaa tietoa IoT-pohjaisista kastelujärjestelmistä, koska teknologiat kehittyvät.

3 Esineiden Internet, IoT

Esineiden internet on laajalti käytetty termi joukolle teknologioita, järjestelmiä ja suunnitteluperiaatteita, jotka liittyvät fyysisessä ympäristössä toimiviin, internetiin liitettyihin laitteisiin [15]. Terminä esineiden internet on suhteellisen uusi, mutta konsepti tietokoneiden ja verkkojen yhdistämisestä laitteiden valvontaan ja ohjaamiseen on ollut olemassa jo vuosikymmeniä [27]. 1970-luvun lopulla kaupallisessa käytössä olleet puhelinlinjojen kautta toimivat järjestelmät sähköverkon etävalvontamittarien lukemiseen sekä 1990-luvulla yleistyneet M2M(machine-to-machine) ratkaisut yritysten sekä teollisuuden laitteiden valvonnassa ja ohjauksessa olivat ensiaskelia kohti esineiden internetiä [27]. IoT ja M2M-viestintää käytetään yhdistämään sensoreita ja muita laitteita ICT-järjestelmiin langallisten ja langattomien verkkojen kautta. Toisin kuin M2M, IoT viittaa myös näiden järjestelmien ja sensoreiden yhdistämiseen laajempaan internetiin.

IoT ei ole uusi internet, se on laajennus olemassa olevaan internetiin joka lisää internetin yleisyyttä integroimalla jokaisen esineen vuorovaikutuksen alaiseksi sulautettujen järjestelmien avulla [15, 38]. Tämä johtaa erittäin hajautettuun laitteiden verkostoon, mahdollistaen kommunikoinnin älykkäiden esineiden kanssa ja niiden välillä, mikä tukee visiota “milloin tahansa, missä tahansa, millä tahansa medialla, minkä tahansa esineen” viestinnästä. [38, 5]. Samanaikaisesti IoT-sovellukset pyrkivät kattamaan laajan valikoiman ihmisten ja elämän tarpeita älykkäistä ympäristöistä (kaupungit, koti, liikenne), terveyden ja elämänlaadun parantamiseen [6].

3.1 IoT-arkkitehtuurimallit

IoT-arkkitehtuureissa on monenlaisia malleja, mutta yksi laajasti hyväksytty malli on nelitasoinen arkkitehtuuri joka koostuu seuraavista tasoista: sensori, yhdyskäytävä- ja verkko, palvelunhallinta sekä sovellustasosta [2]. Kuvassa 3.1 esitetään kyseinen nelitasoinen arkkitehtuuri. Malli koostuu nimensä mukaisesti neljästä erillisestä tasosta, jotka kuvaavat IoT-järjestelmän toiminnan alkaen sensoritasolla tapahtuvasta tiedon keräämisestä päätyen lopulta sovellustasolle, jossa loppukäyttäjä hyödyntää tätä kerättyä tietoa.

Kuva 3.1: Nelitasoinen arkkitehtuuri Agarwal et al. [2], muokattu.

Sensoritaso koostuu nimensä mukaisesti sensoreista, jotka keräävät tietoa ympäristöstään sekä aktuaattoreista, joiden tehtänä on jonkin toiminnon suorittaminen. Sensoreita ja aktuaattoreita tarkastellaan yleisellä tasolla alaluvussa 3.2.

Yhdyskäytävä- ja verkkotaso välittävät tätä sensoreilta kerättyä tietoa ylempään kerrokseen käyttäen alaluvussa 3.3 esiteltyjä langattomia viestintäteknologioita. Tämä taso vastaa datan prosessoinnista ja varastoinnista pilvipalveluun. Lisäksi tason tulee tukea skaalautuvaa, joustavaa ja standardoitua yleisprotokollaa, jotta tiedonsiirto erilaisista laitteista on mahdollista [32].

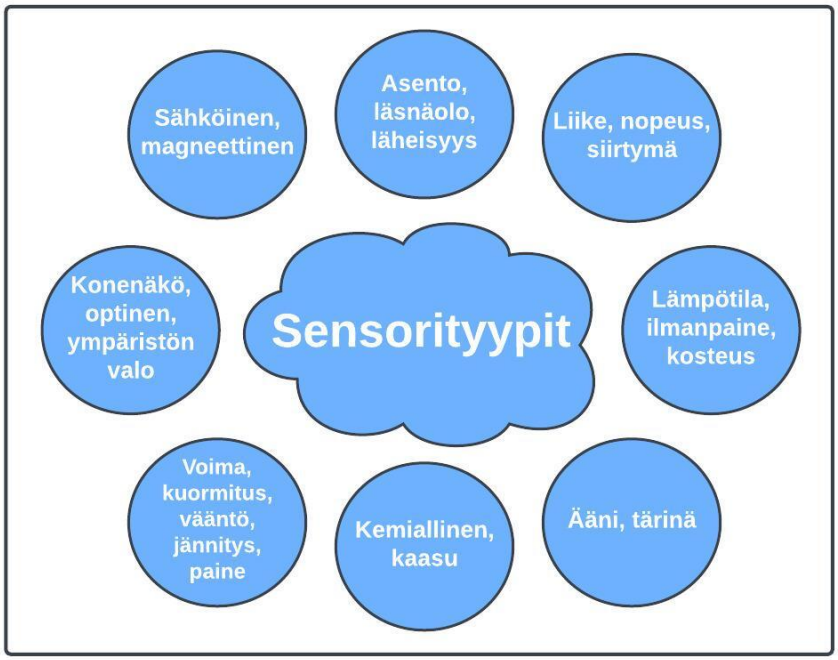
Palvelunhallintatason rooli koostuu vastaanotetun tiedon käsittelystä sekä palveluiden toimittamisesta sovellustasolle. Taso toimittaa palvelut verkon kautta eri tietoliikenneprotokollien avulla [2]. Tietojen turvallisuus ja yksityisyys tulee varmistaa tällä tasolla [32]. Palvelunhallintatasoa tarkastellaan pilvipalvelualustojen näkökulmasta alaluvussa 3.4.

Sovellustaso tarjoaa palvelunhallintatason toimittamat palvelut käyttäjille käyttöliittymän muodossa, kuten potilaan terveyttä mittaavien muuttujien (sydämen syke yms.) jatkuvan seurannan terveydenhuollon ammattilaiselle [2]. Sovelluksia voidaan käyttää myös monilla muilla aloilla, kuten logistiikassa, maataloudessa ja vähittäiskaupassa [32]. Luvussa 3.5 esitellään esimerkkejä IoT-järjestelmistä.

3.2 Sensorit ja aktuaattorit

Tässä alaluvussa esitetään sensorien ja aktuaattorien määritelmät sekä erilaiset sensorityypit. Sehrawat et al. [28] mainitsevat että sensorit ovat tärkeä osa minkä tahansa IoT-sovelluksen automatisointia, sillä ne mittaavat ja käsittelevät kerättyä tietoa fyysisistä asioista, mahdollistaen muutosten havaitsemisen sensorin ympäristössä. Sovelluksen käyttökohde vaikuttaa sensorimallin valintaan, sillä sensorimallit eroavat toisistaan esimerkiksi mittaustarkkuudessa ja mittausvälin suuruudessa. Nämä elektroniset laitteet koostuvat aistivista soluista, jotka pystyvät mittaamaan fyysisiä parametreja, kuten valon vaihtelua valoa tunnistavalla vastuksella, lämpötilan muutoksia termistorilla tai ääniä, liikkeitä sekä muita ympäristön vaihteluita [11].

Aktuaattorit ovat mekaanisia tai elektromekaanisia laitteita, jotka tarjoavat kontrolloituja sekä joskus rajoitettuja liikkeitä tai asentoja. Nämä asennoit aktivoidaan sähköisesti, manuaalisesti tai paineen kuten ilman ja hydraulisten nesteiden avulla [23]. Aktuaattorit voidaan jakaa kahteen ryhmään: mekaanisiin aktuaattoreihin ja toimintoihin. Mekaaniset aktuaattorit voivat olla mm. moottoreita tai vesipumppuja ja toiminnot esim. viestin lähettäminen, laitteen värinän aktivoiminen tai valojen ohjaaminen [11].



Kuva 3.2: Sensorityypit Sehrawat et al. [28], muokattu.

3.3 Langattomat IoT viestintäteknologiat

Tässä alaluvussa käsitellään yleisiä esineiden internetissä käytettäviä langattomia viestintäteknologioita. Alaluku on jaettu kolmeen osaan, matkapuhelinverkkoihin sekä pitkän ja lyhyen kantaman langattomiin teknologioihin.

3.3.1 Matkapuhelinverkot

Viimeisen 20 vuoden aikana maailmanlaajuisessa matkapuhelinviestinnässä on tapahtunut vaikuttavaa kehitystä, kuten siirtyminen äänipainotteisista 1G/2G-palveluista 3G-datapalveluihin vuonna 2001 ja 4G-mobiililaajakaistaan vuonna 2010 [16]. Matkapuhelinverkoissa tapahtuvan tiedonsiirron nopeus on kasvanut 2,4 kilotavusta, 1,2 gigatavuun sekunnissa joka vastaa 500 000-kertaista kasvua vain neljännesvuosisadassa. Matkapuhelinverkko koostuu laajasta radiotaajuuksien peittämästä alueesta joka on jaettu ennaltamääriteltyihin muotoihin ja alueisiin, joita kutsutaan soluiksi. Jokaisella alueella on yksi lähetin-vastaanotin (tukiasema), jonka tehtävänä on yhdistää laite toiseen laitteeseen joka on samassa tai erillisessä solussa. Matkapuhelinverkot tarjoavat pienen energiankulutuksen ja suuren kattavuusalueen [30].

Aiemman sukupolven matkapuhelinverkot (1G, 2G, 3G, 4G) suunniteltiin ja kehitettiin pääosin teleoperaattoreiden toimesta tarkoituksena yhdistää ihmisiä. 5G matkapuhelinverkko on suunniteltu yhdistämään ihmiset sekä esineet, minkä takia sen odotetaan ratkaisevan aiempien sukupolvien teknologioiden asettamia rajoituksia ja lisäävän mahdollisuuksia sekä käyttökohteita IoT:n saralla. [3].

5G

Seuraavan sukupolven 5G-langattoman viestinnän visio perustui erittäin korkeiden tiedonsiirtonopeuksien, matalien viiveiden, tukiasemien kapasiteetin moninkertaisen lisäyksen sekä käyttäjien havaitseman palvelun laadun merkittävään parantamiseen, verrattuna aiemman sukupolven 4G LTE-verkkoihin [3]. 3GPP aloitti 5G standardin määrittelyn vuonna 2015, kehitystyön 2017 ja kaksi vuotta myöhemmin vuonna 2019 ensimmäiset kaupalliset 5G matkapuhelinverkot otettiin käyttöön Etelä-Koreassa ja Yhdysvalloissa [16]. Aiemman sukupolven matkapuhelinverkkoihin verrattuna 5G-matkapuhelinverkot tarjoavat enemmän nopeutta ja kanavia, mikä mahdollistaa useampien laitteiden yhdistämisen verkkoon johtaen suurempaan verkon kapasiteettiin [30]. 5G on ensimmäinen matkapuhelinverkko joka on suun-

niteltu toimimaan millä tahansa taajuudella 400 MHz - 90 GHz välillä ja sen käyttötapaukset levittäytyvät muuallekin kuin kuluttajasovelluksiin. Monet teollisuuden IoT-sovellukset kuten maanviljelyn automaatio sekä terveydenhuollon valvontajärjestelmät on mahdollista toteuttaa 5G matkapuhelinverkon tarjoaman korkean tiedonsiirtonopeuden, matalan viiveen ja luotettavuuden ansiosta [3].

3.3.2 Pitkän kantaman langattomat teknologiat

Esineiden internetin nopean kasvun myötä, matalatehoiset laajaverkot (LPWAN) ovat tulleet suosituiksi. Ne edustavat uutta viestintämallia joka täydentää perinteisiä matkapuhelin- ja lyhyen kantaman langattomia teknologioita pyrkimyksenä ratkaista monipuolisia IoT-sovelluksien vaatimuksia [31, 26]. Nämä matalatehoisissa laajaverkoissa toimivat teknologiat tarjoavat IoT-laitteille kyvyn lähettää pieniä määriä tietoa pitkän kantaman verkoissa. Tämä kyky ottaa huomioon minimaalisen virrankulutuksen, joka johtaa laitteen käyttöiän pitenemiseen [6]. LPWAN-teknologiat on jaettu kahteen kategoriaan, lupavapaisiin ja luvanalaisiin taajuusalueisiin.

LoRaWAN

LoRaWAN on LoRa Alliancen kehittämä lupavapaalla taajuusalueella toimiva avoimen standardin arkkitehtuuri. Se tarjoaa pääsynhallinta mekanismin päätelaittelle mahdollistaen viestinnän yhden tai useamman yhdyskäytävän kanssa [6]. Cycleon kehittämä LoRa on fyysisen kerroksen lupavapaa teknologia joka mahdollistaa 1 GHz taajuusalueella toimivan pitkän kantaman, pienen tiedonsiirtonopeuden ja matalan virrankulutuksen langattoman viestinnän.

LoRaWAN-verkko on tyypillisesti star-of-stars-topologia, jossa LoRaWAN-protokolla määrittää viestintäprotokollan ja verkkojärjestelmän arkkitehtuurin, kun taas LoRa-teknologia mahdollistaa verkon pitkän kantaman linkin [10]. LoRa-sensorit käyttävät single-hop tiedonsiirtoa keskustellessaan yhdyskäytävän kanssa.

LoRaWan-verkot keskittyvät matalan kustannuksen sovelluksiin ja ne voidaan ottaa käyttöön mahdollisimman pienellä infrastruktuurilla ja kapasiteetilla [31]. Sovellusalueisiin kuuluu muun muassa viestintä ajoneuvojen ja infrastruktuuriteknologioiden välillä sekä langatatonta sensoriverkkoa hyödyntävät valvonta- ja hallintajärjestelmät, kuten karjaeläinten juottokaukaloiden vedenpinnan valvontajärjestelmä tai älykkään vesihuollon hallintajärjestelmä [19].

NB-IoT

Kapeaa taajuusaluetta hyödyntävä NB-IoT teknologia kehitettiin 3rd Generation Partnership Project:in (3GPP) toimesta tukemaan matalan suorituskyvyn IoT-sovelluksia [6]. Se on rakennettu luvanalaisella taajuusalueella toimivan LTE-standardin pohjalta, mikä mahdollistaa teknologian integroimisen kyseiseen standardiin. Suurin osa LTE-standardin ominaisuuksista on poistettu käytöstä, jotta NB-IoT teknologia pysyisi mahdollisimman yksinkertaisena. Tällä tavoin laitteiden kustannukset ja virrankulutus pysyy matalina. NB-IoT:n tiedonsiirtonopeus on keskimäärin 50 kbps, käyttöalue noin 15 kilometriä ja verkkotopologiana toimii tähtiverkko.

NB-IoT voidaan ottaa käyttöön päivittämällä olemassa oleva matkapuhelinverkko, mutta sen käyttöalue rajoittuu alueelle jonka matkapuhelinverkko kattaa [31]. NB-IoT on suunnattu sovelluksille, jotka vaativat toiminnassaan korkeaa palvelun laatua ja alhaista tiedonsiirron viivettä kuten terveydenhuollon sovellukset.

3.3.3 Lyhyen kantaman langattomat teknologiat

Lyhyen kantaman langattomat teknologiat, jotka toimivat langattomien likiverkkojen (WPAN) alueella kuten WiFi, Bluetooth ja Zigbee ovat suosittuja viestintäteknologioita IoT-sovelluksissa niiden laajan saatavuuden takia. Useimmissa IoT-laitteissa on integroituna WiFi tai Bluetooth 4.0 moduulit, jotka mahdollistavat laitteen yhdistämisen internetiin [30].

Bluetooth

Bluetooth-viestintäteknologia keksittiin vuonna 1994 ruotsalaisen telekommunikaatioyrityksen Ericsson:in toimesta [22]. Tarkoituksena oli luoda eri laitteiden välille tilapäisiä, lyhyen kantaman langattomia verkkoja. Vuonna 1998 Ericsson, IBM, Intel, Nokia ja Toshiba loivat standardointijärjestö Special Interest Group:in (SIG) joka kehittää ja edistää Bluetooth-teknologiaa. Bluetooth standardin ensimmäisen version julkaisusta on yli 20 vuotta ja nykypäivänä tämä teknologia sekä erityisesti Bluetooth Low Energy (BLE)-teknologia on noussut yhdeksi esineiden internetin peruspilareista [30]. Bluetooth mahdollistaa kahden tai useamman laitteen yhdistämisen toisiinsa tarjoten jatkuvan yhteyden laitteiden välillä. Bluetooth teknologian tiedonsiirto tapahtuu radioaaltoja pitkin 2.4 GHz taajuusalueella usein maksimissaan 100 metrin alueella. Lähetettävän tiedostotyypin muotoa ei ole rajoitettu. Tämä mah-

dollistaa tiedonsiirron datana, kuvina, dokumentteina sekä ääni- tai videotiedostoina [8]. Ennen Bluetooth 4.0 versiota, teknologiaa käyttävän laitteen virrankulutus kasvoi. Tästä syystä niiden hyödyntäminen IoT-sovelluksissa ei ollut suosittua.

Tähän on tullut muutos sen jälkeen kun SIG-standardointijärjestö julkaisi BLE-teknologian. BLE on noussut laajalti käytetyksi teknologiaksi erilaisissa IoT-sovelluksissa kuten älykodeissa, teollisuuden IoT-ratkaisuissa ja älykaupungeissa. Se suunniteltiin erityisesti laitteille, jotka käyttävät matalaa kaistanleveyttä ja toimivat akun tai paristojen varassa [30]. Lisäksi teknologia käyttää vähän energiaa, tukee rajoittamatonta tähtitopologiaa ja pystyy säilyttämään standardin mukaisen 1 - 100 metrin kantaman tiedonsiirron aikana [22].

IEEE 802.11

IEEE 802.11 (WiFi) on yleisimmin käytetty Institute of Electrical and Electronics Engineers:in (IEEE) määrittelemä lyhyellä kantamalla toimiva viestintästandardi [30]. 2000-luvun alussa tapahtuneen julkaisun jälkeen, WiFi:n maailmanlaajuista suosiota on pääosin perusteltu sen suuren joustavuuden, kantaman, tiedonsiirtonopeuden, laitteiden liikkuvuuden, kustannustehokkuuden, toteutuksen yksinkertaisuuden ja markkinapenetraation ansiosta [30, 1].

Vaikka IoT:ssa mahdollisesti hyödynnettäviä 802.11 standardeja löytyy monenlaisiin eri käyttötarpeisiin, erityisesti 802.11ah (WiFi HaLow) on suunniteltu IoT:ta ja sensoriverkkoja varten [34]. WiFi HaLow-standardissa tiedonsiirtoon käytetään alle 1 GHz:n taajuusaluetta, kun yleisesti WiFi-standardien tiedonsiirto tapahtuu 2.4 - 5 GHz taajuusalueella. Standardin käyttämä matala taajuusalue mahdollistaa sensorien ja laitteiden yhdistämisen matalatehoisiin tukiasemiin laajalla jopa yhden kilometrin kokoisella alueella. Tämä ominaisuus tekee IEEE 802.11ah-standardista pitkän kantaman langattoman teknologian.

WiFi Halow-teknologiaa voidaan hyödyntää älykaupungin IoT-sovelluksissa kuten jäte- ja vesihuollossa sekä maanviljelyn saralla, maaperän laadun seurantaan ja kulunvalvontaan [36].

Zigbee

Zigbee on 2000-luvun alussa kehitetty langaton verkkoteknologia, joka on vakiinnuttanut markkinapaikkansa sen ominaisuuksien ja erityisesti sen verkossa toimivien laitteiden pitkän käyttöiän takia [30]. Kun muut langattomat verkkoteknolo-

giat ovat pyrkineet tuomaan markkinoille uusia ominaisuuksia parantaen samalla suorituskykyään, Zigbee on keskittynyt 8-bittisiin mikro-ohjaimiin tehden siitä käytetyimmän viestintäteknologian langattomissa sensoriverkoissa.

ZigBee muistuttaa Bluetooth-teknologiaa mutta se erottuu siitä sen matalan tiedonsiirtokyvyn, siirron luotettavuuden, alhaisen hinnan ja erilaisen verkkotopologian (silmukkaverkko) takia [14]. Koska Zigbee käyttää silmukkaverkkoa se pystyy luomaan laajan luotettavan verkon jossa lähetyksen peittoalue voi olla jopa 100 metriä, jos reitillä ei ole esteitä. Zigbee-teknologian sovelluksia voivat olla IoT-pohjaiset maataloussovellukset kuten avomaaviljely ja toimitusketjun seuranta.

3.3.4 Langattomat IoT viestintäteknologiat yhteenveto

Matkapuhelinverkot ovat tarjonneet perinteisesti yhteyksiä kuluttajien mobiililaitteille ja kannettaville tietokoneille, mutta 5G:n tarjoamien suurten tiedonsiirtonopeuksien, kapasiteetin ja saatavuuden ansiosta matkapuhelinverkoista hyötyy jatkossa myös esimerkiksi teollinen esineiden internet ja kriittinen viestintä [16].

Nykyiset matalatehoiset IoT viestintäteknologiat voidaan jakaa kahteen kategoriaan, langattomat likiverkot (WPAN) ja matalatehoiset laajaverkot (LPWAN) [34]. Markkinanäkökulmasta katsoen lyhyen kantaman langattomat teknologiat kuten ZigBee ja WiFi-viestintä ovat käytetyimpiä langattomien likiverkkojen alueella [6]. Ikpehai et al. [17] mainitsevat, että vaikka lyhyen kantaman teknologiat hallitsevat IoT-yhteyksiä, vuoteen 2025 mennessä matalatehoiset laajaverkko teknologiat toteuttavat 25% teollisuuden IoT-yhteyksistä.

Tässä alaluvussa esitellyistä pitkän kantaman langattomista teknologioista LoRa:lla on etuja akun keston, kapsiteetin ja kustannusten suhteen kun taas luvanalaisella NB-IoT:lla on etuja palvelun laadun, viiveen, luotettavuuden ja kantaman suhteen [31]. De Carvalho Silva et al. [10] totesivat tutkimuksessaan että LoRaWAN osoitti etua muihin LPWAN-teknologioihin pitkän kantaman viestinnän tehonkulutuksen suhteen, noin 3-5 kertaisella hyötysuhteella.

Kun IoT-sovellukselle valitaan viestintäteknologia, toteuttajan tulee pohtia sovelluksen käyttökohdetta sekä suosia muuttujia kuten virrankulutusta, saatavuutta, luotettavuutta sekä laitteiston kustannuksia [30].

3.4 Pilvipalvelualustat

Tässä alaluvussa esitellään pilvipalvelualustojen ominaisuuksia sekä kolme yleisintä IoT-sovelluksissa käytettyä pilvipalvelualustaa yleisellä tasolla ja tarkastellaan niiden tarjoamia palveluita. Pilvipalvelut tarjoavat resursseja käytettäväksi tarpeen mukaan ja ne koostuvat laitteistoista sekä ohjelmistoista, jotka mahdollistavat palveluiden toimittamisen verkossa [29]. Pilvipalvelut viittaavat verkossa oleviin palvelinkeskuksiin, jotka on saatavilla useille käyttäjille. Niiden tarkoituksena on tarjota resursseja taloudellisen hyödyn saavuttamiseksi, mahdollistaen samalla nopean käyttöönoton joustavien resurssien avulla. Nämä palvelut on suunniteltu käsittelemään työkuormia ja tarjoamaan virtuaaliresursseja etänä verkon yli. Pilvipalvelut voidaan jakaa kolmeen kategoriaan: infrastruktuuripalvelut (IaaS), alustapalvelut (PaaS) ja sovelluspalvelut (SaaS). Infrastruktuuripalvelut on julkinen pilvipalveluntarjoaja, joka tarjoaa palveluja maksu per käyttö -periaatteella. Alaluvussa 3.4.1 esiteltävät pilvipalvelualustat ovat kaikki IaaS-pilvipalveluita. Alustapalvelut tarjoaa jaetut työkalut, prosessit ja rajapinnat sovellusten kehityksen, testauksen ja käyttöönoton tukemiseksi. Sovelluspalvelut on puolestaan julkinen pilvipalvelu, joka toimittaa sovelluksia verkon kautta selainta käyttävälle loppukäyttäjälle.

3.4.1 Yleisimmät pilvipalvelualustat

Ucuz et al. [35] toteuttivat kirjallisuuskatsauksen kolmesta suurimmasta pilvipalvelualustan toimittajasta, jotka ovat Amazon, Microsoft sekä Google. Taulukossa 3.1 on esitelty näiden toimittajien pilvipalvelualustojen tarjoamia palveluita. Kuvassa 3.1 esiteltyä arkkitehtuuria voidaan soveltaa AWS, Azure ja Google Cloudiin niiden tarjoamien palvelujen ja ominaisuuksien perusteella. Pilvipalvelualustat tarjoavat käyttöliittymän sekä komentokehotteen laitehallintaan ja ne tukevat MQTT 3.1.1- ja HTTPS-tietoliikenneprotokollia. Lisäksi Azure tukee AMQP 1.0 (WebSocket) -tietoliikenneprotokollaa. Palveluiden integrointi voidaan toteuttaa käyttämällä rajapintoja, ohjelmistokehityspaketteja tai alustojen tarjoamia integraatiopalveluita. AWS:n tukemat ohjelmointikielet ovat Java, JS, C++, Python ja Embedded C, kun taas Azuren tukemat ohjelmointikielet ovat .NET, C, Java, NodeJS, Python ja C#. Google Cloud tukee ohjelmointikieliä kuten C#, Java, NodeJS, GO, PHP, Python ja Ruby. Palvelualustat tukevat tietoturvaominaisuuksia, kuten TLS ja X.509-sertifikaatteja. Tämän lisäksi palvelut tarjoavat omia ratkaisujaan tietoturvan parantamiseksi.

Amazon Web Services (AWS)

Suurinta markkinaosuutta hallitsevan AWS:n tarjoamiin palveluihin kuuluu mm. pilvitallennustila, kattavat kehitystyökalut ja se mahdollistaa viestinnän pilvessä toimivien sovellusten kanssa [35]. Pierleoni et al. [25] laatiman vertailun mukaan AWS hoitaa laitehallinnan IoT Device Management-palvelun avulla. Alusta käyttää sääntöjä vuorovaikutukseen muiden AWS-palveluiden kanssa. Nämä säännöt koostuvat SQL-tyyppisellä syntaksilla kirjoitetusta laukaisimesta ja yhdestä tai useammasta aktivoitavasta toiminnasta. AWS IoT Core mahdollistaa suoran yhteyden Amazon DynamoDB (NoSQL-tietokanta) ja AWS S3 (Simple Storage Service) skaalautuviin tallennustiloihin AWS-pilvessä.

AWS tarjoaa useita palveluita tiedon keräämiseen ja käsittelyyn. Näihin kuuluvat muun muassa Amazon Kinesis Data Stream reaaliaikaiseen suoratoistodatan käsittelyyn, AWS Lambda pilvifunktioiden suorittamiseen, Amazon Simple Notification Service ilmoitusten lähettämiseen tai vastaanottamiseen sekä Amazon Simple Queue Service tiedon tallentamiseen jonossa.

Microsoft Azure

Microsoftin tuote Azure tarjoaa pilvipalveluita, kuten mobiilisovelluspalveluita, tallennustilaa, viestintäalustoja ja virtuaalikoneita [35]. Microsoft Azure for IoT on saatavilla sekä PaaS- että SaaS-ratkaisuna [25]. Molemmissa ratkaisuissa käytetään Azure IoT Hubia yhdyskäytävänä tiedon turvalliseen vastaanottamiseen. Microsoft Azure IoT Hub Device Provisioning Service-palvelua käytetään laitehallintaan. Azure IoT Hub käyttää rajapintojen julkaisuun sääntöjä, jotka on muotoiltu SQL-tyyppisellä syntaksilla. Jotta viestit voidaan reitittää laitteesta näihin rajapintoihin, säännöt sisällytetään viestien otsikoihin ja runkoon.

Tiedon kuumaa varastointia varten Azure tarjoaa Azure Cosmos DB (NoSQL-tietokanta) ja Azure SQL DB (SQL-relaatiotietokanta) tietokantapalvelut. Kylmää varastointia varten ovat Azure Blob Storage (tiedostovarasto) ja Azure Data Lake (hajautettu tietovarasto) palvelut. Lisäksi Azure tarjoaa palvelun nimeltä Time Series Insight, jota käytetään tiedon analytiikkaan, tallennus- ja kokoamispalveluihin. IoT Hub on yhdistetty muihin Azure-palveluihin, kuten web- ja mobiilisovellusten rakentamiseen tarkoitettuun Azure App Service:en, Notifications Hub:in push-ilmoitusten lähettämiseen ja PowerBI:hin hallintapaneelien luomista varten.

Google Cloud

Googlen tuote Google Cloud koostuu laajasta valikoimasta pilvipalveluita [35]. IoT Core Device Manager laitehallintapalvelun tehtävänä on laitteiden rekisteröinti palveluun, kun taas (HTTP/MQTT) tietoliikenneprotokollia käytetään laitteiden yhdistämisessä palveluun ja datan lähettämisessä pilveen [25]. Laitehallintapalveluun kuuluu prosessit laitteiden rekisteröinnille, autentikoinnille ja autorisoinnille. Pilveen saapuvan datan hallitsemiseksi alusta käyttää Google Cloud Data Flow -palvelun tarjoamia julkaisuputkia (pipelines). Julkaisuputket mahdollistavat datan muokkaamisen, kokoamisen, rikastamisen ja siirtämisen muihin palveluihin. Lisäksi jokaiseen julkaistuun tapahtumaan voidaan vaikuttaa erikseen Google Cloud Functions -toiminnoilla, joita voidaan käyttää virheellisen datan suodattamiseen, hälytysten laukaisemiseen tai muiden API-kutsujen tekemiseen. Alusta tarjoaa Cloud Datastoren ja Cloud BigQueryn NoSQL-tietokantoina sekä Cloud BigTablen SQL-rajapinnalla. Cloud Storagea käytetään harvemmin käytetyn datan arkistointiin ja rakenteettoman datan tallentamiseen. IoT Core on integroitu osaksi Googlen suurten tietomäärien ja koneoppimisanalyysien palveluita, kuten Cloud ML, Data Studio ja DataLab.

Taulukko 3.1: AWS, Azure ja Google Cloud palvelut Pierleoni et al. [25], muokattu

	AWS	Azure	Google Cloud
Laitehallinta	IoT Device Management	Device Provisioning Service	IoT Core Device Manager
Vuorovaikutus säännöt	Rules Engine (SQL-tyyliset säännöt)	SQL-tyyliset säännöt	Google Cloud Data Flow, Google Cloud Functions
Tietokannat ja tallennustila	AWS S3, Amazon DynamoDB	Azure Blob Storage, Azure Data Lake, Azure CosmosDB & SQL DB	Cloud Storage, Cloud Datastore, Cloud BigQuery, Cloud BigTable
Integraatio palveluiden välillä	Amazon Kinesis Data Stream, AWS Lambda, AWS SNS, AWS SQS	Azure App Service, Notifications Hub, PowerBI	Cloud ML, Cloud Data Studio, Cloud DataLab

3.5 Esimerkkejä IoT-järjestelmistä

Tässä alaluvussa tutustutaan kolmeen erilaiseen IoT-sovellukseen.

3.5.1 Älykkäät vesijärjestelmät

Sehrawat et al.[28] mainitsevat tutkimuksessaan sensorien ja aktuaattoreiden käyttökohteina mm. älykkäät vedenvalvontajärjestelmät, joiden toiminnan tarkoituksena on säästää vettä. Jokien veden tason vaihteluiden ja haitallisten aineiden seuranta sensoreiden avulla voi mahdollistaa tulvien ennakoimisen ja estää merivesien saastumista. Vedenpaineen mittaaminen vesisäiliöissä ja vesiputkissa auttaa havaitsemaan vuodot ajoissa. Vedenpaineen alenemisesta voidaan lähettää ilmoitus, mikä säästää rahaa ja luonnonvaroja [32]. Älykkäissä vesijärjestelmissä sensorit ovat paikoillaan tai ne virtaavat veden mukana. Sensoreita voidaan käyttää mittaamaan veden laatua, virtausta, nopeutta, lämpötilaa, saastumista ja veden sisältöä. Vesihuollon reaaliaikaisella analyysilla ja hallinnalla voidaan varmistaa, että asukkailla ja liiketoimintaa harjoittavilla yrityksillä on riittävästi vettä käytettävissä.

3.5.2 Älykäs ympäristö ja maapallo

Soumyalatha et al.[32] mukaan saastumisen ja luonnonkatastrofien havaitseminen ja ennustaminen on yksi erittäin tärkeä IoT-sovellus. Ilmansaasteiden minimoimiseksi on mahdollista sijoittaa sensoreita, jotka valvovat tehtaiden ja ajoneuvojen päästöjä keräämällä tietoa hiilimonoksidin ja typpidioksidin pitoisuuksista ilmassa. Luonnonkatastrofit kuten maanjäristykset, maanvyörymät, metsäpalot, tulivuoren purkaukset ja tulvat voidaan ennustaa käyttämällä langattomia sensoreita. Varoitusalueiden määrittäminen, varoitusten välittäminen, haitallisten kaasujen ja maaperän muutosten kuten tärinän, tiheyden ja kosteuden seurannalla on mahdollista tunnistaa vaara ajoissa ja tehdä tarvittavat toimenpiteet ympäristön sekä ihmisten suojaamiseksi [28].

3.5.3 Älykäs maanviljely

Sensoreita käytetään maanviljelyn sovelluksissa laajalti eri tarkoituksiin, kuten maaperän kosteuden mittaamiseen, ilmasto-olosuhteiden valvontaan ja kastelujärjestelmien automatisoimiseen [28, 32]. Sensorien keräämää tietoa voidaan käyttää ilmoittamaan maanviljelijälle kasvien tarvitsemasta hoitomenetelmästä [32]. Järjes-

telmät kuten meteorologinen asemaverkko havaitsee sensoreiden avulla ympäristön sääolosuhteet ja tekee kerätyn tiedon avulla sääennusteita, jotta säässä tapahtuvat muutokset havaitaan ajoissa ja maanviljelijät pystyvät estämään kasveille aiheutuvat mahdolliset vahingot [28]. IoT-teknologiaan pohjautuvia maatalouden hallintatietojärjestelmiä (FMIS) on esitetty viljelijöiden avuksi tehokkaiden päätösten tekemisessä. Järjestelmät hallinnoivat maatiloille asennettujen sensoreiden keräämää tietoa ja ne tarjoavat viljelijöille massadata-analyysin perusteella kerättyjä tietoja maatiloilla käytetyistä kohteista, kuten koneista, siemenistä, torjunta-aineista ja lannoitteista, sekä taloudellisen analyysin tuloksia [20].

Peltojen valvontaa voidaan käyttää maanviljelyn saralla viljely-ympäristöjen hallintaan sadon laadun ja tuoton parantamiseksi. Peltojen valvonta edullisten sensoreiden ja verkkojen avulla on tyypillinen esimerkki IoT:n soveltamisesta maataloudessa. Nämä älykkäät maatalouden peltovalvontajärjestelmät seuraavat maaperän kosteutta ja lämpötilaa. Sensoreilta kerätyt tiedot tallennetaan pilveen tulevaa tietojen analysointia varten, ja niitä voidaan hyödyntää peltojen hallinnassa mahdollistaen vesiresurssien tehokkaan käytön ja säästöt työvoimakustannuksissa.

Rikkakasvien torjuntaan on kehitetty IoT-pohjaisia järjestelmiä, jotka yhdistävät IoT:n, robotiikan ja edistyneet kuvanalyysitekniikat rikkakasvin tunnistamiseen ja torjunta-aineen kohdennettuun käyttöön [33]. Näiden järjestelmien päätoiminta tapahtuu tekoälyyn perustuvalla kuvantunnistusmallilla, joka osaa tunnistaa ja erottaa rikkaruohon viljelykasveista. Rikkakasvien torjunnan automatisoinnin avulla voidaan merkittävästi vähentää tarvetta käyttää torjunta-aineita, mikä taas alentaa terveyteen liittyviä riskejä ja minimoi ympäristövaikutuksia.

Tuotantoeläinten hallintaan ja valvontaan suunnitellut IoT-järjestelmät pyrkivät tehostamaan eläinten terveydenhuoltoa automatisoimalla valvontaprosesseja ja rehun annostelua [33]. Eläinten käyttäytymisen analysoinnin avulla voidaan saavuttaa kattavampi ymmärrys niiden terveydestä ja mahdollistaa taudin havaitseminen varhaisessa vaiheessa, parantaen eläinten hyvinvointia ja tuotantoa.

Kasvihuoneen hallintajärjestelmät hyödyntävät langattomia sensoriverkkoja, jotka mittaavat ympäristömuuttujia kuten kosteutta ja ilman lämpötilaa, mahdollistaen kosteudenhallinnan ja kasvien kastelun automatisoinnin [33]. Lisäksi tuholaistenhallintaan on kehitetty IoT-järjestelmiä, jotka automatisoivat tuholaisten tunnistamisen kasvihuoneissa. Nämä tuholaistenhallintajärjestelmät käyttävät kuvantunnistusmalleja ja RGB-kameroita liimapaperiin pyydystettyjen tuholaisten tunnistamiseksi, mahdollistaen viljelijälle tarvittavien vastatoimenpiteiden suunnittelun.

4 Kastelujärjestelmän automatisointi

Tässä luvussa käydään läpi automatisoidun kastelujärjestelmän motivaatiota ja aiempia tutkimuksia aiheesta. Alaluvussa 4.1 esitellään tutkimuksissa kehitettyjä järjestelmiä sekä mittalaitteita. Alaluvuissa 4.2 ja 4.3 käydään läpi kastelun automatisoinnin hyötyjä ja haittoja. Lisäksi alaluvussa 4.4 tarkastellaan tutkimuksissa käytettyjä sensoreita.

4.1 Aiemmat tutkimukset

Kasvien kastelu voidaan tehdä manuaalisesti tai automaattisesti käyttämällä kastelulaitteita [37]. Kasvien kastelun automatisointia on tutkittu suuressa ja pienessä mittakaavassa peltojen kastelun sekä kasvihuoneen kastelun muodossa. Molemmissa tutkimustapauksissa kasvien kastelun automatisoinnilla on pyritty vähentämään maanviljelijöiden manuaalista työtä ja samalla saavuttamaan tärkeä tavoite, veden säästäminen. Irawan et al. [18] toteuttivat automaattisen chilikasvien kastelujärjestelmän ratkaisuna sprinklerijärjestelmän tehottomuuteen. Tutkijoiden tekemän selvityksen mukaan vesijohtoon liitetyn sprinklerijärjestelmän vedenkulutusta on vaikea seurata ja se voi johtaa liialliseen kasteluun. Kirjoittajien toteuttama järjestelmä mittaa maaperän kosteutta ja lähettää ilmoituksen taustajärjestelmälle, jos mittaustulos laskee asetetun maaperän kosteutta kuvaavan raja-arvon alle. Taustajärjestelmä suorittaa saatujen ilmoitusten perusteella automatisoituja päätöksiä kastelun käynnistämiseen ja lopettamiseen. Tutkijoiden mukaan kehitetty järjestelmä voi helpottaa maanviljelijöitä suorittamaan automatisoitua kastelua.

Myös Anusha et al.[4] tekemä tutkimus tarjoaa kasvien monitorointi- ja kastelujärjestelmän prototyypin joka pyrkii tehostamaan kasvien kastelua. Kirjoittajien toteuttama ratkaisu kerää useiden eri ympäristömuuttujien tietoja ja tekee automatisoituja päätöksiä vesipumpun ohjaamiseen. Järjestelmä pyrkii estämään prototyypin läheisyydessä tapahtuvaa luvatonta liikkumista liiketunnistimella ja aktivoimalla hälyttimen jos liikettä havaitaan. Järjestelmän käyttäjä voi ohjata vesipumppua manuaalisesti tai tarkastella mittauslaitteen keräämiä tietoja web-sovelluksen avulla. Kurniawan et al.[21] esittelevät artikkelissaan järjestelmän kasvihuoneen au-

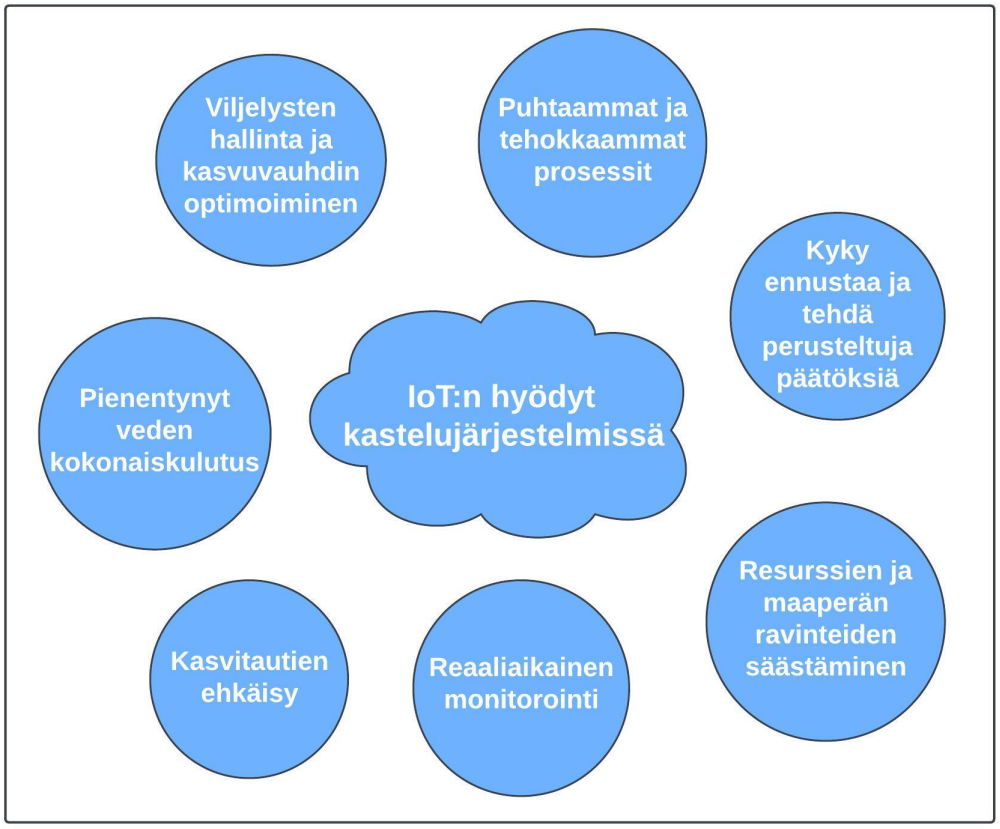
tomatisoimiseen. Toteutettu järjestelmä kerää kasvihuoneen ympäristömuuttujien tietoja joita järjestelmän käyttäjä voi tarkastella reaaliaikaisesti Blynk-sovellukseen tehdyn integraation avulla. Järjestelmä käyttää toiminnassaan sumeaa logiikkaa vesipumpun kastelun keston ohjaamiseen ja LED-nauhan valon voimakkuuden säätämiseen. Tutkijat testasivat ja visualisoivat asettamiaan sumean logiikan sääntöjä Matlab-ohjelmistossa. Tutkijoiden suorittamat testit osoittivat, että heidän asettamansa säännöt vesipumpun kastelun kestolle saivat tarkkuusarvon 98,3 % ja LED-nauhan valon voimakkuuden säätäminen tarkkuusarvon 99,6%. Näin korkeat tarkkuusarvot kertoivat tutkijoille sen, että järjestelmän toiminta oli optimoitu suhteellisen tarkasti. Vastaavasti Waworundeng et al. [37] keskustelevat tutkimuksessaan prototyypistä ja järjestelmästä joka automatisoi kasvien kastelua maaperän kosteustason perusteella. Myös heidän ehdottamansa järjestelmä integroituu Blynk-sovelluksen kanssa, jonka lisäksi he käyttivät ThingSpeak IoT-alustaa analytiikan visualisoimiseen taustajärjestelmässään.

Dahane et al. [9] kehittivät tutkimuksessaan edullisen ja älykkään IoT-ratkaisun kastelutehokkuuden parantamiseksi. Tutkijat mainitsevat että vähäsateiset kesäkuukaudet voivat johtaa veden puutteeseen jonka takia erityisesti maatilat voivat kärsiä taloudellisiakin tappioita. Heidän mukaansa suurin osa aiemmin ehdotetuista kastelujärjestelmistä ei ota sääennusteita (esim. sademäärää) huomioon. Ratkaisu pyrkii ennustamaan ympäristömuuttujien (ilman lämpötila, ilman kosteus, maaperän kosteus) avulla, milloin tehdä päätös aloittaa kastelu ja milloin ei. Tämän pitäisi estää kasteluveden, energian ja liiallisesta kastelusta johtuvaa sadon laadun hukkaa, tapauksissa, kun vesisade seuraa välittömästi kastelua.

4.2 Kastelun automatisoinnin hyödyt

Perinteisissä kastelumenetelmissä suurin osa veden käsittelystä ja kastelutoiminnoista suoritetaan manuaalisesti [24]. Nämä perinteiset menetelmät, joissa ihmisen osallistuminen on välttämätöntä johtaa usein veden runsaaseen tuhlaamiseen. Kasvien kastelu sopivalla vesimäärällä on tärkeää, koska sillä on suora vaikutus kasveihin [37]. Kasvit, jotka saavat oikean määrän vettä, eivät kuivu tai mätäne. Kastelun automatisoiminen vähentää maanviljelijöiden manuaalista työtä ja sen ansiosta kasveille voidaan kohdistaa juuri oikea määrä vettä mikä vähentää veden kokonaiskulutusta. Automaation avulla maanviljelijät voivat reagoida välittömästi kaikkiin merkittäviin muutoksiin säässä ja se mahdollistaa suurienkin peltoalueiden kaste-

lun nopealla aikataululla. Maanviljelijät voivat hyödyntää automatisoitua kastelua myös kasvien lannoitukseen. Dahane et al. [9] mukaan kastelun automatisoiminen parantaa kastelutehoa, kasvien tuottavuutta, laatua, kannattavuutta ja maatalous-tuotannon kestävyyttä. IoT-järjestelmät jotka automatisoivat kasvien kastelun sisältävät jo edellä mainitut hyödyt, mutta niiden käyttöön liittyy myös muita hyötyjä, joita esitellään kuvassa 4.1. Nämä hyödyt johtavat kastelun suorituskyvyn ja kustannustehokkuuden parantumiseen sekä vähäisempään energiankulutukseen [24].



Kuva 4.1: IoT:n hyödyt kastelujärjestelmissä Obaideen et al. [24], muokattu.

4.3 Kastelun automatisoinnin haasteet

IoT-tekniikan lisääntyvä integroiminen maatalouden ja ruoantuotannon kastelujärjestelmiin tarjoaa laajan kokonaisuuden erilaisia käyttömahdollisuuksia mutta myös haasteita [24]. Näissä älykkäissä kastelujärjestelmissä on otettava huomioon monia eri tekijöitä, kuten kustannukset, autonomisuus, siirrettävyys, ylläpito, tehokkuus,

arkkitehtuuri ja luotettavuus, joiden toteuttaminen voi luoda haasteita. Muita IoT-kastelujärjestelmissä esiintyviä haasteita esitellään kuvassa 4.2. Erityisesti maanviljelijöiden IoT-tuntemuksen ja maanviljely sovellusten puute, laitteiden herkkyys haastaville ympäristöolosuhteille sekä heikot viestintäsignaalit ja mahdollisesti rajoittunut luotettavan internet-yhteyden saatavuus kohteissa voi muodostua merkittäväksi esteeksi IoT-kastelujärjestelmän käyttöönotolle. Kattava tietoturvan huomioiminen IoT-kastelujärjestelmän toteutuksessa voi olla haasteellista, sillä järjestelmään voi kohdistua useita uhkia [12]. Näitä uhkia ovat esimerkiksi ohjelmistojen haavoittuvuudet, puutteet yksityisyyden suojassa sekä laitteiden kloonaus ja palvelunestohyökkäykset. Haasteita tarkastellessa on myös otettava huomioon, että maanviljelijöillä on usein rajalliset tulot, mikä tekee älykkäiden kastelujärjestelmien hankkimisesta taloudellisesti haastavaa. Jotta tällaisilla älykkäillä kastelujärjestelmillä olisi mahdollisuus saavuttaa kaupallinen menestys, IoT-laitteiden ja älykkäiden kastelujärjestelmien kokonaiskustannusten tulisi laskea.

Kuva 4.2: IoT:n haasteet kastelujärjestelmissä Obaideen et al. [24], muokattu.

4.4 Tutkimuksissa käytetyt sensorit ja aktuaattorit

Kasvien kasvatuksessa voi hyödyntää monenlaisia sensoreita kasvuympäristön monitorointiin. Sensoreiden avulla maanviljelijä tai kasvien kasvatuksesta kiinnostunut voi saada reaaliaikaista tietoa kasviensa tilasta. Sensorit voivat monitoroida mm. maaperän kosteutta, kasvien valonsaantia sekä ilmanpainetta, -kosteutta ja lämpötilaa. Monitorointi- ja kastelujärjestelmä voi hyödyntää edellä mainittujen ympäristömuuttujien kerättyjä tietoja toiminnassaan, kuten päätöksessä käynnistää vesipumppu, kun maaperän kosteus laskee järjestelmässä asetetun raja-arvon alle.

Valon intensiteettisensori

Useissa tutkimuksissa [21, 4, 9] käytetään valon intensiteettisensoria mittaamaan kasvin valonsaantia. Sensorin mittaustulosten avulla tutkijat pystyivät varmistamaan että kasvit saavat riittävästi valoa kasvaakseen. Kurniawan et al. [21] mainitsevat että valo on kasveille erittäin tärkeää, lähinnä sen roolin vuoksi fysiologisissa toimissa, kuten fotosynteesissä.

Ilmanpaine-, kosteus- ja lämpötila-sensorit

Ilmanpaine-, kosteus- ja lämpötila sensoreilla tutkijat saivat tietoa kasvin ympäristön olosuhteista. Ilmankosteus- ja lämpötila vaikuttavat voimakkaasti kasvien kasvuun. Kasvin kasvu ei ole optimaalista jos ilmankosteus on liian suuri. Kurniawan et al. [21] toteavat, että ihanteellinen ilmankosteus kasvien kasvulle on noin 60-80%.

Maaperän kosteussensori

Maaperän kosteussensori mittaa veden määrän maaperästä. Jokainen osioon 4.1 valittu artikkeli tai julkaisu esitteli järjestelmän, jossa kasteluun liittyvät päätökset tehtiin maaperän kosteussensorin avulla. Kastelupäätökset alkoivat havaitsemalla maaperän kosteustaso sensorin avulla. Waworundeng et al. [37] mukaan vesipitoisuuteen liittyvä maaperän kosteus on kasvin kasvuun vaikuttava tekijä.

Vesipumppu ja solenoidi

Osiossa 4.1 esiteltyjen tutkimusten ratkaisut ohjasivat vesipumppua tai solenoidia, jotka siirsivät kasveille veden vesisäiliöstä tai vesijohdosta putkien avulla.

4.5 Monitorointi- ja kastelujärjestelmän tyypillinen toteutus

Tässä luvussa esitellään kasvien monitorointi- ja kastelujärjestelmän tyypillinen toteutus yleistasolla. Tyypillinen toteutus pohjautuu alaluvussa 4.1 esiteltyjen tutkimusten ratkaisuihin. Alaluvussa 4.5.1 esitellään lyhyesti arkkitehtuurin yleiskuvaus. Alaluvussa 4.5.2 tarkastellaan kasvien monitorointi- ja kastelujärjestelmän toimintamallia. Alaluvussa 4.5.3 esitetään mittauslaitteen laitteistokokoonpano.

4.5.1 Arkkitehtuurin yleiskuvaus

Kasvien monitororointi- ja kastelujärjestelmä koostuu yleisesti mittauslaitteesta sekä taustajärjestelmästä. Mittauslaitteen tehtävänä on kerätä tietoja ympäristömuuttujista ja siirtää ne taustajärjestelmään. Mittauslaitteen korttitietokoneen tulee olla varustettu WiFi- tai radio-moduulilla tiedonsiirtoa varten. Alaluvussa 4.1 esitellyt ratkaisut käyttivät mittauslaitteen ja taustajärjestelmän välillä tapahtuvaan tietoliikenteeseen HTTPS-kyselyitä lukuun ottamatta Dahane et al. [9] toteuttamaa järjestelmää, joka käyttää tietoliikenteeseen SPI-protokollaa. Samaan verkkoon yhdistetty mittauslaite tai aktuaattori, voi vastaanottaa kasteluun liittyviä komentoja taustajärjestelmästä.

Kasvien monitorointi- ja kastelujärjestelmän arkkitehtuuri koostuu yleisesti kolmesta osiosta, jotka ovat mittauslaite, taustajärjestelmä ja päätelaiteympäristö. Tutkimuksissa [18, 4, 37, 21] ehdotetut ratkaisut toimivat samassa langattomassa lähiverkossa. Ratkaisujen käyttämät sensorit ja vesipumppu tai aktuaattori on kytketty mittauslaitteen korttitietokoneeseen. Korttitietokone on yhdistetty WLAN-verkkoon WiFi-moduulin avulla. Päätelaite (puhelin tai tietokone) on yhdistetty samaan WLAN-verkkoon.

Dahane et al. [9] toteuttamassa järjestelmässä on useampi osio. Järjestelmään voi liittää useita eri sensorisolmuja jotka keskustelevat taustajärjestelmän kanssa reunasolmun avulla. Aktuaattori joka ohjaa kastelujärjestelmää toimii ratkaisussa erillisenä osiona. Palvelinympäristö jossa tietokanta sijaitsee, voidaan määrittää toimimaan paikallisesti tai pilvipalvelussa. Palvelinympäristön määrityksestä riippuen, päätelaite voi ottaa yhteyden taustajärjestelmään, joko paikallisen verkon tai internetin avulla. Päätelaitteen käyttäjä voi analysoida järjestelmän keräämiä tietoja ja monitoroida satoa reaaliajassa käyttöliittymän avulla.

4.5.2 Kasvien monitorointi- ja kastelujärjestelmän toimintamalli

Kasvien monitorointi- ja kastelujärjestelmän tarkoituksena on automatisoida kasvien monitorointi- ja kastelu. Tätä varten mittauslaitteen tai sensorisolmun on luotava dataa, jonka pohjalta taustajärjestelmä voi esittää käyttöliittymässä mittaustuloksia, sekä tehdä päätöksiä kastelun aloittamisesta ja keskeyttämisestä eli kastelusekvenssistä. Kastelusekvenssi vaatii tiedon maaperän kosteudesta, sekä rajaarvot jolloin kastelu tulee aloittaa ja lopettaa. Tutkimuksissa [18, 4, 37, 21] toteutetuissa järjestelmissä kastelusekvenssiin liittyvät päätökset toteutettiin seuraamalla vain maaperän kosteutta, kun taas Dahane et al. [9] pyrkivät huomioimaan maaperän kosteuden lisäksi sääennustetiedot, neuroverkkoja hyödyntäen, ennen kastelupäätöksen tekoa. Waworundeng et al. [37] luoma järjestelmä lähettää käyttäjälle ilmoituksen kun kastelusekvenssi alkaa tai loppuu.

Tutkielmassa esitetyistä aiemmista tutkimuksista vain Dahane et al. [9] ja Waworundeng et al. [37] tallensivat kerättyjä mittaustuloksia tietokantaan. Muut järjestelmät näyttivät käyttäjälle vain viimeisimmän mittauslaitteelta saadun arvon. Mittaustulosten tulkinta on tehty käyttäjälle helpoksi. Tieto on muutettu käyttöliittymässä numeerisista arvoista graafiseen muotoon käyttämällä kustomoituja tai markkinoilla olevia analytiikan ratkaisuja.

4.5.3 Mittauslaitteen laitteistokokoonpano

Tutkimuksissa [18, 4, 37, 21] ehdotettujen ratkaisujen mittauslaitteen laitteistokokoonpanossa järjestelmän ytimenä toimii korttitietokone, joka sisältää valmiiksi integroidun WiFi-moduulin tai mahdollisuuden liittää kyseinen moduuli. Dahane et al. [9] toteuttamassa järjestelmässä mittauslaitteena toimii sensorisolmu, joka lähettää kerättyä dataa radio-moduulin avulla reunasolmulle. Reunasolmun tehtävä on prosessoida data oikeaan muotoon, ennen kuin se siirretään taustajärjestelmään. Mittauslaitteeseen tai sensorisolmuun liitettiin tarpeen mukaan sensorit maaperän kosteuden, valonmäärän, ilmanpaineen- -kosteuden ja lämpötilan mittaamista varten. Tutkimusten taustajärjestelmät ohjasivat erilaisia vesipumppuja tai kastelujärjestelmään liitettyjä aktuaattoreita, joilla vesi siirrettiin kasveille. Kurniawan et al. [21] järjestelmä ohjasi vesipumpun lisäksi LED-nauhaa, jonka avulla kirjoittajat saivat tarpeen mukaan lisättyä kasvien valonsaantia ja tuuletinta, joka viilensi kasvihuonetta jos tilan lämpötila nousi liian korkeaksi.

5 Prototyypin suunnittelu ja toteutus

Tässä luvussa esitellään prototyypin toteutus sekä mittauslaitteen ja monitorointisovelluksen ohjelmistot. Alaluvussa 5.1 esitellään prototyypin arkkitehtuuri yleisellä tasolla. Alaluvussa 5.2 esitellään prototyypin toimintamalli. Alaluvussa 5.3 tarkastellaan mittauslaitteen laitteistokoonpanoa. Alaluvussa 5.4 käydään läpi prototyypin suunnittelu ja toteutus ketterän menetelmän näkökulmasta. Alaluvuissa 5.5 ja 5.6 esitellään mittauslaitteen ja monitorointisovelluksen ohjelmistoja.

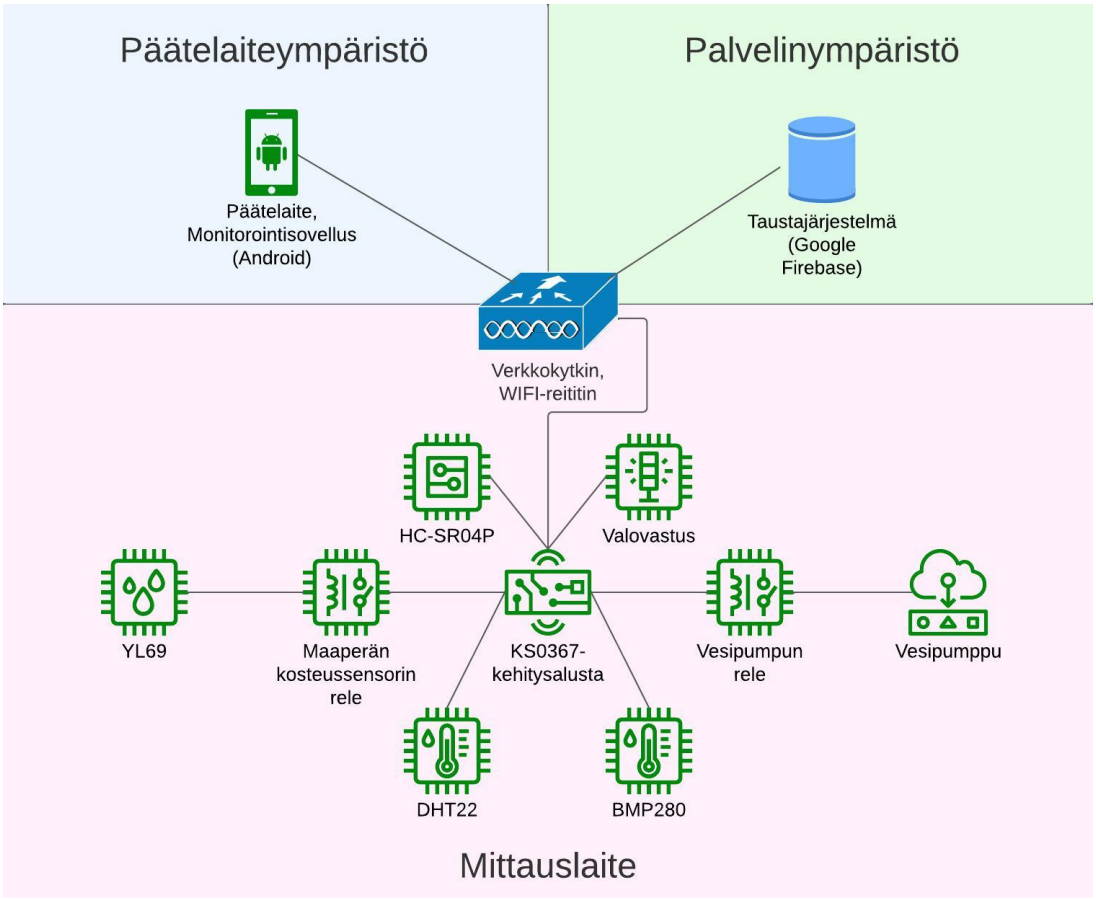
5.1 Arkkitehtuurin yleiskuvaus

Kasvien monitorointi- ja kastelujärjestelmän prototyyppi koostuu mittauslaitteesta, monitorointisovelluksesta sekä taustajärjestelmästä. Nämä esitetään osioittain kuvassa 5.1. Mittauslaite konfiguroidaan otsikkotiedoston avulla toimimaan halutussa langattomassa lähiverkossa ja taustajärjestelmä instanssissa. Sen tehtävänä on kerätä tietoja ympäristömuuttujista sensorien avulla ja siirtää ne taustajärjestelmään salatussa muodossa käyttämällä HTTPS-kutsuja.

Mittauslaite tekee myös päätökset kastelusekvenssin aloittamisesta kun maaperän kosteus on tippunut asetetun raja-arvon alle, ohjaten vesipumppua. Mittauslaitteeseen liitetyt sensorit käyttävät pääasiassa GPIO-pinnejä, mutta maaperän kosteussensori sekä valovastus hyödyntävät analogista pinniä A0 digitaalisen multiplekserin kautta, jotta komponenteista saadaan tarkempi mittausarvo. Mittauslaitteen tarkka kytkentäkaavio on esitetty liitteessä A.

Monitorointisovelluksen avulla loppukäyttäjä pystyy hallinnoimaan käyttäjätilinsä asetuksia, kuten lähiverkon SSID-nimeä. Kun nämä asetukset on määritelty, sovellus näyttää kaikki kyseisessä lähiverkossa toimivat mittauslaitteet. Loppukäyttäjän tulee autorisoida mittauslaite ennen kuin se voi lähettää kerättyä tietoa taustajärjestelmään. Taustajärjestelmään tallennetut tiedot esitetään monitorointisovelluksessa purkamalla ne ihmiselle lukukelpoiseen muotoon.

Taustajärjestelmänä toimii Google Firebase-palvelu. Palvelu tarjoaa käyttäjähallinnan sekä reaaliaikaisen NoSQL-tietokannan, jota käytetään mittauslaitteelta kerättyjen tietojen tallentamiseen.

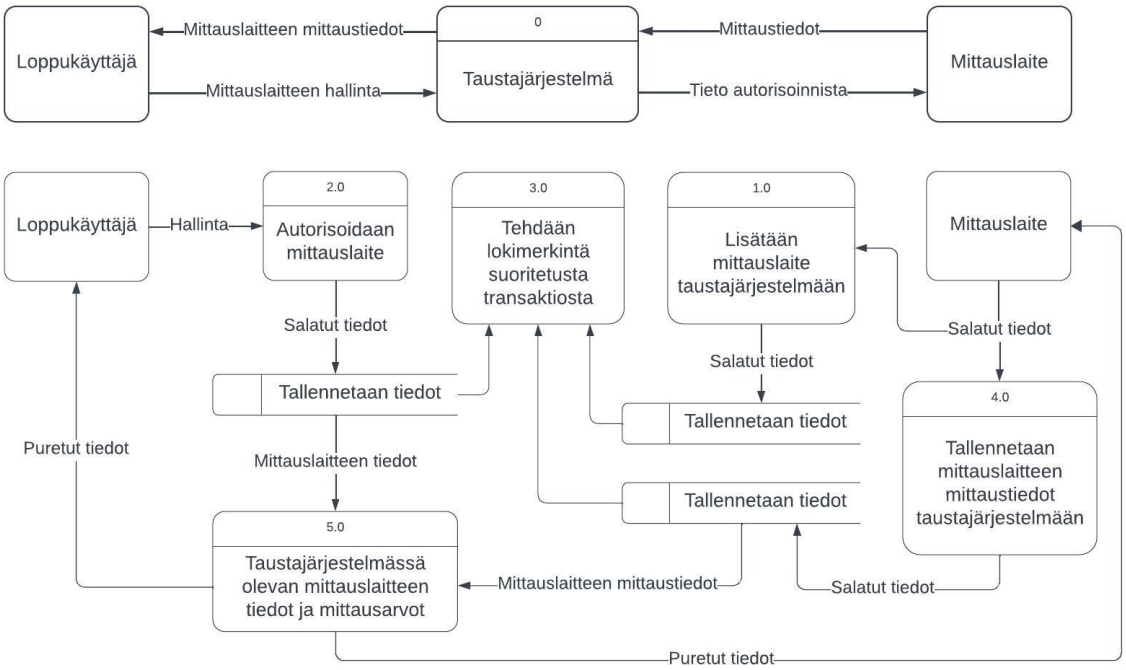


Kuva 5.1: Prototyypin pelkistetty arkkitehtuurimalli.

5.2 Prototyypin toimintamalli

Prototyypin tarkoituksena on automatisoida kasvien monitorointi- ja kastelu. Prototyypin mittauslaitteen avulla voidaan ohjata vesisäiliössä olevaa vesipumppua, joka siirtää vettä kasvin juureen muoviputkea pitkin silloin, kun maaperän kosteus tippuu asetetun raja-arvon alle. Tämän lisäksi mittauslaite suorittaa mittauksia vesisäiliön veden määrästä, kasvin valon saannista sekä ilman lämpötilasta, -paineesta ja -kosteudesta. Edellä mainitut mittaukset tarjoavat loppukäyttäjälle tietoa kasvin kasvutekijöistä, kuten valon- ja vedensaannista sekä lämpötilasta. Numeerisessa muodossa oleva mittaustieto muutetaan mittauslaitteella merkkijonoksi, joka salataan käyttämällä AES-salausalgoritmia ennen sen lähettämistä taustajärjestelmään. Prototyypin päätavoitteena on mahdollistaa kasvin kasvattajalle kastelun automa-

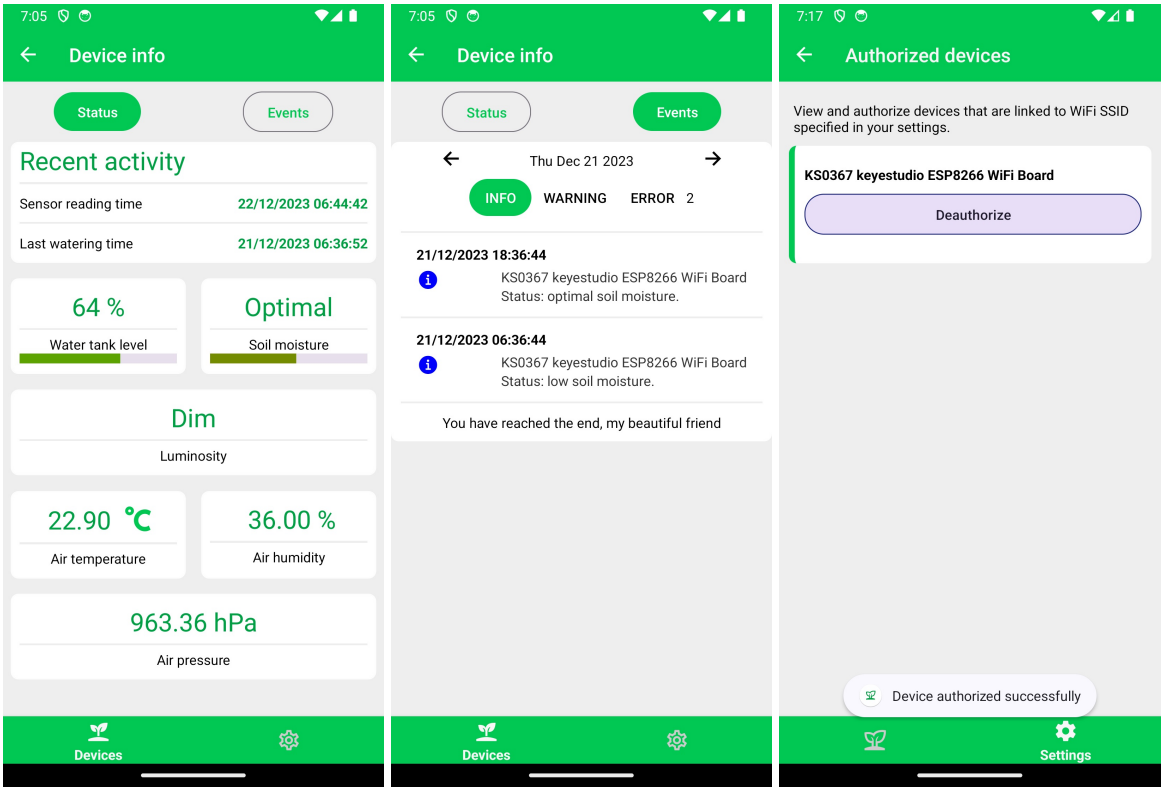
tisointi sekä tarjota reaaliaikaista seurantaa monitorointisovelluksessa kasvin kasvutekijöiden muutoksista. Toisena tavoitteena on luoda monitorointisovellukseen näkymä mittauslaitteen lokitietojen seuraamiseen, jotta loppukäyttäjä voi tarkkailla mittauslaitteen toimintaa. Kuvassa 5.2 esitetään tietovuokaavion avulla, kuinka tieto etenee mittauslaitteelta taustajärjestelmään ja sitä kautta loppukäyttäjälle sekä päinvastoin.



Kuva 5.2: Prototyypin tietovuokaavio.

Tiedon keräämisen edellytyksenä on mittauslaitteen autorisointi, jonka loppukäyttäjä voi suorittaa monitorointisovelluksessa. Kun mittauslaite on konfiguroitu ja aktivoitu, se ilmoittaa tiedon olemassaolostaan taustajärjestelmälle. Loppukäyttäjän tulee syöttää lähiverkon SSID-tunniste asetuksiinsa, jonka jälkeen mittauslaite voidaan autorisoida “autorisointi“-näkymässä. Kun mittauslaite saa vahvistuksen autorisoinnista, tiedon kerääminen voidaan aloittaa. Tiedon kerääminen mittauslaitteella on konfiguroitu siten, että maaperän kosteusarvo mitataan 12 tunnin välein, kun taas muut mittaustiedot kerätään 29 minuutin välein. Mittausten jälkeen tiedot salataan ja lähetetään taustajärjestelmään, jonka jälkeen ne ovat nähtävissä monitorointisovelluksessa. Jos maaperän kosteuden mittaustulos ylittää asetetun rajaarvon, suoritetaan tarkastus viimeisimmästä vesisäiliön vesimäärästä. Tarkastuksen

jälkeen vesipumppu käynnistetään releen avulla 12 sekunniksi edellyttäen, että vesisäiliössä on enemmän vettä kuin asetettu vähimmäismäärän raja-arvo.



Kuva 5.3: Kuvakaappaukset mittauslaitteen tietojen yhteenveto-, loki- ja autorisointinäkymistä monitorointisovelluksessa.

5.3 Mittauslaitteen laitteistokokoonpano

Tässä alaluvussa esitellään toteutetun prototyypin mittauslaitteen laitteistokokoonpanossa käytettävät komponentit. Komponentit valittiin toteuttamaan toimintamallissa määritettyjä tehtäviä. Mittauslaitteen tarkka kytkentäkaavio esitetään liitteessä A.

Järjestelmän ydin

Mittauslaitteen laitteistokokoonpanossa järjestelmän ytimenä toimii KS0367 keyestudio ESP8266 kehitysalusta, joka sisältää ESP8266-järjestelmäpiirin. Kehitysalusta sisältää valmiiksi integroidun WiFi-moduulin, jotta mittauslaite voidaan liittää lan-

gattomaan lähiverkkoon. ESP8266-järjestelmäpiirin suorituskyky on riittävä prototyypin mittauslaitteelle, koska mittauslaite ei suorita sensoreiden mittauksia ja vesipumpun käynnistystä samanaikaisesti. Tämä mahdollistaa tehokkaan järjestelmäpiirin prosessointikyvyn käytön säästäen resursseja.

Maaperän kosteuden mittaus

Maaperän kosteuden mittaamiseen käytetään YL-69 sensoria, joka toimii 3.3-5V jännitealueella. Sensorille ohjataan käyttöjännite 3.3V relemoduulilla. Mittaukset suoritetaan hyödyntämällä YL-69 sensorin analogista ulostuloa, joka tarjoaa tarkemman mittaustuloksen. Analoginen ulostulo tuottaa arvon välillä 0-1024, missä 0 ilmaisee suurta kosteuden määrää ja 1024 pientä. Tätä kerättyä tietoa käytetään mittauslaitteen ohjelmatasolla vesipumpun käynnistämiseen, kun maaperän kosteusarvo on alle 750. Tämä raja-arvo määritettiin vertailemalla muita vastaavia järjestelmiä ja suorittamalla manuaalisia testejä varmistaen, että kasvin maaperän kosteus ei ole liian korkea vesipumpun käynnistyessä.

Lämpötilan ja kosteuden mittaus

Mittauslaite käyttää DHT22-sensoria ilman lämpötilan ja kosteuden mittaamiseen. Tämä digitaalinen sensori toimii 3-5 V jännitealueella ja se tarjoaa tarkat mittaustulokset lämpötila- ja kosteusmittauksille. Sensorin laaja mittausalue tekee siitä sopivan sekä ulko- että sisäkäyttöön. Adafruit Industries tarjoaa ohjelmakirjaston DHT22-sensorille, joka helpottaa sensorin integrointia mittauslaitteeseen.

Ilmanpaineen mittaus

Mittauslaite mittaa ilmanpaineen BMP280-sensorin avulla. Tämä pieni sensori toimii alhaisella 3.3 V jännitteellä ja se mahdollistaa lämpötila- ja ilmanpainemittaukset. Mittauslaite hyödyntää BMP280-sensoria ainoastaan ilmanpaineen mittaamiseen, sillä DHT22-sensori tarjoaa tarkemman mittaustuloksen kuin BMP280. Sensori sisällytettiin mittauslaitteeseen mahdollista tulevaa toiminnallisuutta varten, sillä ilmanpainetietoja voidaan käyttää säätilan muutosten ennustamiseen. Säätilan muutosten ennustaminen voisi mahdollistaa toiminnon, joka estää vesipumpun käynnistämisen, jos ilmanpaine laskee merkittävästi. Toiminto perustuisi siihen, että matalapaineen aikana vesisateen todennäköisyys kasvaa. Adafruit Industries tarjoaa sensorille ohjelmakirjaston, mikä helpottaa sensorin integrointia mittauslaitteeseen.

Vesisäiliön vedenkorkeuden mittaus

Mittauslaite mittaa prototyypin vesisäiliön vedenkorkeuden HC-SR04P sensorilla. Tämä ultraäänietäisyyssensori toimii 3-5V jännitealueella ja se mahdollistaa etäisyyden mittaamisen 2-400 senttimetrin välillä. HC-SR04P ei tarvitse erillistä ohjelmakirjastoa toimiakseen, sillä sitä voi ohjata GPIO-pinneillä.

Kasvin valonsaannin mittaus

Mittauslaite mittaa kasvin valonsaannin yksinkertaisella laajasti saatavilla olevalla valovastuksella. Valovastuksen mittaustieto luetaan mittauslaitteella valitsemalla multiplekseri lukemaan valovastuksen sisään tuleva signaali ja ohjaamalla sen ulostulo KS0367-kehitysalustan analogiseen pinniin. Mittauksen suorittaminen tällä tavoin antaa mittauslaitteelle tarkempia mittaustuloksia kuin pelkästään digitaalista GPIO-pinniä käytettäessä.

Vesipumpun ohjaus

Mittauslaite ohjaa prototyypin yksinkertaista 3.3-5V jännitealueella toimivaa pientä uppopumppua. Uppopumpun käyttöjännitettä ohjataan 3.3V relemoduulin avulla. Koska uppopumppu käyttää vettä voiteluun, uppopumppua ei aktivoida ellei vesisäiliössä ole vettä, jotta sen hajoamisen riski minimoidaan.

5.4 Prototyypin toteutus ketterällä menetelmällä

Tässä alaluvussa käsitellään prototyypin toteutusta ketterän menetelmän avulla. Prototyypin vaatimusmäärittelyt aloitettiin luomalla projekti, joka sisältää vaatimusten ja tehtävien hallinnan, sekä kaksi lähdekoodin etätietovarastoa, jotka sijaitsevat Github-palvelussa. Tämä palvelu toimii mittauslaitteen ja monitorointisovelluksen lähdekoodin versionhallintapaikkana.

Taulukossa 5.1 esitetään prototyypin 11 vaatimusta, jotka koostuvat mittauslaitteen ja taustajärjestelmän halutuista toiminnallisuuksista ja ominaisuuksista. Toiminnallisuudet ja ominaisuudet kattavat mittauslaitteen ympäristömuuttujien ja maaperän kosteuden mittaukset, vesipumpun hallinnan sekä taustajärjestelmän ominaisuudet, kuten sisäänkirjautumisen ja tietoturvallisen tietoliikenteen toteutuksen. Vaatimukset pohjautuvat toiminnallisuuksiin ja ominaisuuksiin, jotka ovat peräisin

vastaavista kastelujärjestelmistä, sekä niihin, jotka kirjoittaja halusi toteuttaa prototyyppiin.

Taulukko 5.1: Prototyypin vaatimukset listattuna taulukkoon.

Nro	Vaatimus
1	Käyttäjänä haluan nähdä viimeisimmän kasteluajankohdan.
2	Taustajärjestelmä saa tiedon vesisäiliön veden määrästä.
3	Mittauslaitteen virransyötön tulee toimia paristoilla.
4	Mittauslaitteen tulee mitata valonsaanti.
5	Mittauslaitteen tulee mitata ilmanpaine.
6	Mittauslaitteen tulee hallita vesipumppua.
7	Mittauslaitteen tulee mitata maaperän kosteutta.
8	Mittauslaitteen tulee mitata ilman lämpötilaa.
9	Mittauslaitteen ja taustajärjestelmän välinen tietoliikenne tulee olla tietoturvallista.
10	Käyttäjänä haluan kirjautua sisään taustajärjestelmään.
11	Käyttäjänä haluan nähdä taustajärjestelmään linkitetyn mittauslaitteen.

5.4.1 Sprint 1

Tavoitteet

Ensimmäisen sprintin tavoitteena on mahdollistaa mittauslaitteen ja monitorointisovelluksen tietoturvallinen yhteys taustajärjestelmän kanssa. Tiedonsiirrossa välitettävät tiedot tulee olla salattuja AES-lohkosalausalgoritmilla ja tiedonsiirron tulee tapahtua HTTPS-protokollalla.

Toteutus

Sprintin toteutus alkoi taustajärjestelmän alustamisella Google Firebase-palvelussa. Kun projekti oli luotu taustajärjestelmään, aloitettiin monitorointisovelluksen alustaminen. Monitorointisovelluksen teknologiaksi valittiin React Native sekä Expo-

kehitysalusta kehitystyön tueksi. Paikallinen kehitysympäristö toimii siten, että Expo tarjoaa React Native -projektin paikallisella kehityspalvelimella, johon saa yhteyden Expo-sovelluksen avulla. Kehitysympäristö edellyttää, että mobiililaite ja tietokone jossa Expo-kehityspalvelin on toimivat samassa langattomassa lähiverkossa.

Mittauslaitteen valmistelu sisälsi KS0367 Keyestudio ESP8266 kehitysalustan kiinnittämisen koekytkentälevyyn sekä lähdekoodin alustamisen. Mittauslaitteen lähdekoodiin lisättiin *ESP8266WiFi.h* kirjasto WiFi-verkkoon liittämista varten sekä *FirebaseESP8266.h* kirjasto yhteyden muodostamista varten Google Firebase taustajärjestelmään. Mittauslaitteen ohjelmointi ja ohjelman kääntäminen tapahtuu Visual Studio Code ja Arduino IDE koodieditoreilla. Tietoliikenteen salaaminen toteutettiin AES-lohkosalausalgoritmilla, muuttamalla lähetettävä tieto heksadesimaalimerkkijonoksi mittalaitteella ja esittäen sen ihmiselle luettavassa muodossa monitorointisovelluksen käyttöliittymässä. Lokitoiminnallisuuden ensimmäinen versio toteutettiin tietojen loki-tallennuksena eri severiteeteillä (INFO, WARNING, ERROR) hieman muunnellussa SYSLOG-muodossa.

Taustajärjestelmään kirjautumisen toteutus sisälsi näkymien ja toiminnallisuuden suunnittelun sekä toteutuksen monitorointisovellukseen. Mittauslaitteen ja taustajärjestelmän “kättely“-toiminnallisuus toteutettiin siten, että mittauslaite lähettää taustajärjestelmään tiedon olemassaolostaan. Kyseinen tieto sisältää mittalaitteen nimen ja langattoman lähiverkon SSID-tunniste. Käyttäjän tulee autorisoida mittauslaite ennen kuin mittauslaite voi lähettää keräämäänsä tietoa taustajärjestelmään. Käyttäjälle annetaan mahdollisuus syöttää asetuksiinsa langattoman lähiverkon SSID-tunniste, jonka jälkeen samassa verkossa olevat mittalaitteet tulevat näkyviin monitorointisovelluksessa mittauslaitteiden “autorisointi“-näkymässä. Autorisoinnin jälkeen “kättely“-toiminnallisuus on suoritettu ja mittauslaite voi lähettää tietoa taustajärjestelmään.

Testaus

Sprintin toteutuksen testaamisen läpivienti tapahtui manuaalisella testauksella. Toiminnallisuudet todettiin valmiiksi, kun mittauslaite sai lähetettyä halutut tiedot taustajärjestelmään ja monitorointisovellus esitti ne käyttöliittymässä.

Arviointi

Sprintin totetuksen aikana ei ilmennyt suurempia ongelmia. Taustajärjestelmän käyt-

töönotto oli yksinkertaista ja projektin luominen Google Firebase-palvelussa tapahtui minuuteissa. Monitorointisovelluksen alustaminen oli myös tuttua, sillä olen käyttänyt valittuja teknologioita työelämän projekteissa. Mittauslaitteen toiminnallisuuksia toteutettaessa oli huomioitava, että toteutetut toiminnallisuudet eivät ole liian raskaita ja että niitä ei suoriteta samanaikaisesti, sillä mittauslaitteella on rajattu prosessointikyky.

5.4.2 Sprint 2

Tavoitteet

Toisen sprintin päätavoitteina oli toteuttaa prototyypin keskeinen toiminnallisuus eli kastelusekvenssi. Toissijaisena tavoitteena keskityttiin jo toteutettujen toiminnallisuuksien parannuksiin ja lähdekoodin siistimiseen.

Toteutus

Monitorointisovelluksen lokinäkymän parannukseen sisältyi sivutustoiminnallisuuden toteuttaminen. Lista lataa käyttäjälle ensin 20 viimeisintä lokimerkintää valitulta päivältä ja kun käyttäjä saavuttaa listan lopun, ladataan lisää lokimerkintöjä valitulta päivältä. Tieto saatiin haettua oikeassa muodossa käyttämällä tietokantakyselyssä Firebase Realtime Database tietokannan orderByChild-, endAt- ja limitToLast-suodattimia. Haettu tieto tuli järjestää käyttöliittymässä käänteiseen järjestykseen ennen sen näyttämistä, jotta viimeisin tietue näkyi listassa ylimpänä. Seuraavien 20 lokimerkinnän hakemiseksi käytettiin endAt-suodatinta, joka sai arvokseen käänteisen listan viimeisimmän lokimerkinnän aikaleiman.

Sprintin aikana toteutettiin myös monitorointisovelluksen sekä mittauslaitteen lähdekoodin siistimistä modulaarisempaan muotoon, jakaen toiminnallisuudet omiin kansioihin ja tiedostoihin. Maaperän kosteussensori YL-69 ja vesipumppu liitettiin mittauslaitteeseen, joka mahdollisti kastelusekvenssin toteuttamisen.

Testaus

Sprintin toteutuksen testaamisen läpivienti tapahtui manuaalisella testauksella. Toiminnallisuudet todettiin valmiiksi, kun mittauslaite sai kerättyä ja lähetettyä halutut tiedot taustajärjestelmään ja monitorointisovellus esitti ne oikein käyttöliittymässä.

Arviointi

Sprintin aikana havaittiin, että monimutkaisten tietokantakutsujen tekeminen Firebase Realtime Database tietokantaan osoittautui haastavaksi, sillä palvelu ei tarjoa kovin kattavia suodattimia tietokantakyselyihin. Tästä huolimatta haluttu lokitoiminnallisuuden parantaminen toteutettiin perehtymällä palvelun dokumentaatioon sen tarjoamista suodattimista.

Vesipumpun liittäminen mittauslaitteeseen toi myös omat haasteensa. Vesipumpun käynnistäminen pudotti kehitysalustan jännitettä niin paljon, että se aiheutti kehitysalustalle Watchdog timer reset-virheen. Jännitteen putoaminen ratkesi luomalla transistoripiiri käyttämällä 2N2222 transistoria, 1N4001 diodia ja 1K ohmin vastusta, joka syöttää herätevirran vesipumppua ohjaavalle relekortille. Tämä kastelusekvenssin ensimmäinen versio syötti käyttöjännitteen vesipumpulle ja relekortille ulkoisesta virtalähteestä. Transistoripiiri sekä ulkoinen virtalähde tullaan korvaamaan erilaisella ratkaisulla tulevissa sprinteissä.

5.4.3 Sprint 3

Tavoitteet

Sprintin päätavoitteina oli liittää BMP280 ja DHT22 sensorit mittauslaitteeseen, ilmanpaineen, lämpötilan ja kosteuden mittaamisen mahdollistamiseksi sekä vesipumpun virransyötön muuttaminen ulkoisesta virtalähteestä kehitysalustalta saatavaan virtaan.

Toteutus

BMP280 sensorin lisääminen kehitysalustaan tapahtui liittämällä sensori digitaalisiin pinneihin D1 ja D2. Tiedonsiirto sensorilta tapahtuu I2C-protokollan avulla, joka aktivoitiin syöttämällä 3.3V jännite sensorin CSB pinniin. DHT22 sensorin liittäminen mittauslaitteeseen oli suoraviivaista. Vesipumppu muutettiin toimimaan ilman ulkoista virtalähdettä. Ratkaisuksi muodostui kehitysalustalta saatavan 5V jännitteen muuttaminen 3.3V jännitteeksi lineaarisella jännitesäätimellä. Jännitteen pudottaminen vähensi vesipumpun vedensyötön voimakkuutta, mutta muuten ratkaisu toimii kuten aiemminkin. Sensorien liittämisen ja niiden mittausoperaatioiden toteutuksen jälkeen, lokitapahtumien tallennus taustajärjestelmään ei toiminut

enää halutulla tavalla. Ongelma johtui tavasta jolla viestien tunnukset luotiin. Aikaisempi toteutus oli muotoa (mac-osoite + : + epoch aikaleima), mutta koska tapahtumat luotiin samanaikaisesti, tapahtumajonossa saattoi olla useita tapahtumia samalla tunnuksella. Tästä syystä tallennuksen yhteydessä tietokannassa jo oleva tunnusta vastaava solmu ylikirjoitettiin.

Korjauksen jälkeen viestien tunnukset ovat muotoa (mac-osoite + : + epoch aikaleima + : + tapahtuman tunnus). Tapahtuman tunnukset ovat merkkijonoja muodossa "0x0, 0x1, 0x2...". Tunnukset on sidottu tiettyihin toimintoihin, jotka tehdään vain kerran pääohjelman ajon aikana. Esimerkiksi "0x0"vastaa mittauslaitteen rekisteröintiä ja "0x1"alustan tietojen lähetystä taustajärjestelmään.

Testaus

Sprintin toteutuksen testaamisen läpivienti tapahtui manuaalisella testauksella. Toiminnallisuudet todettiin valmiiksi, kun mittauslaite sai kerättyä ja lähetettyä halutut tiedot taustajärjestelmään ja monitorointisovellus esitti ne oikein käyttöliittymässä.

Arviointi

BMP280 sensoria liittäessä mittauslaitteeseen huomasin, että kehitysalustan suojamaadoitus (GND) pinnit eivät olleet yhteydessä toisiinsa. Jotta BMP280 sensori löytyi osoitteesta 0x76, se piti maadoittaa samaan suojamaadoitus pinniin kuin muut sensorit ja vesipumppu.

5.4.4 Sprint 4

Tavoitteet

Sprintin päätavoitteena oli liittää viimeiset prototyypissä käytettävät sensorit mittauslaitteeseen. Toissijaisena tavoitteena oli toteuttaa taustajärjestelmään tuki historiikkidatalle ja testata mittauslaitteen virransyöttöä ilman verkkovirtaa.

Toteutus

Sprintin totetus alkoi valovastuksen lisäämisellä mittauslaitteeseen. Valovastuksella mitataan prototyypin ympäristön valonmäärän saantia. Vesisäiliön vedenmäärän

seuraaminen toteutettiin HC-SR04P ultraäänietäisyyssensorilla. Mittauslaitteen keräämät tiedot tallennetaan tietokannan historiikkitauluun. Mittauslaitteen virransyötön testaaminen 9V paristolla ei tuottanut tarpeeksi virtaa mittauslaitteelle.

Testaus

Sprintin toteutuksen testaamisen läpivienti tapahtui manuaalisella testauksella. Toiminnallisuudet todettiin valmiiksi, kun mittauslaite sai kerättyä ja lähetettyä halutut tiedot taustajärjestelmään ja monitorointisovellus esitti ne oikein käyttöliittymässä. Historiikkidatan toteutus todettiin valmiiksi, kun mittauslaite lähetti halutut tiedot taustajärjestelmään ja ne näkyi oikein tietokannassa.

Arviointi

Ultraäänietäisyyssensori tuli sijoittaa riittävän suureen vesisäiliöön, jotta sensori ei saanut häiriöitä heijastuksista, jotka voisivat aiheuttaa virheellisiä mittauksia. Mittauslaitteen virransyötön testaaminen 9V paristolla ja 5V lineaarisella jännitteensäätimellä ei antanut tarpeeksi virtaa mittauslaitteelle. Virransyötön testaaminen olisi pitänyt toteuttaa dedikoidulla paristokotelolla, mutta tutkielman toteuttajalla ei ollut saatavilla kyseistä komponenttia. Tästä johtuen mittauslaitteen virransyöttö toteutetaan usb-verkkovirta-adapterilla.

5.4.5 Sprint 5

Tavoitteet

Prototyypin toteutuksen viimeisessä sprintissä tavoitteina oli käyttöliittymän virtaviivaistaminen, monitorointisovelluksen kääntäminen APK-julkaisupaketiksi sekä maaperän kosteussensorin kytkentöjen ja lähdekoodin parannuksia.

Toteutus

Käyttöliittymän virtaviivaistaminen saavutettiin jakamalla näkymät kahteen päänäkymään: “Devices“ ja “Settings“. “Devices“-näkymästä käyttäjä pääsee tarkastelemaan mittauslaitteen sensorien viimeisimpiä mittaustuloksia ja lokitietoja. “Settings“-näkymään ei tullut muutoksia. Sovelluksen kääntäminen APK-julkaisupaketiksi to-

teutettiin Expo Application Services (EAS) avulla. Sovelluksen kääntäminen aloitetaan komentokehotteessa komennolla "eas build –platform android –profile production". Komento käyttää eas.json määritystiedoston asetuksia. Oletuksena EAS kääntää lähdekoodin .abb tiedostoksi, jota käytetään esim. Google Play -kaupassa. Koska monitorointisovelluksesta ei tehdä julkaisua, lähdekoodin kääntämisen tuotokseksi on määritetty .apk tiedosto. APK-tiedosto mahdollistaa sovelluksen asentamisen älypuhelimeen. Sovelluksen kääntäminen tapahtuu EAS-pilvipalvelussa. Palvelun ilmainen lisenssi säilyttää käännettyä julkaisupakettia pilvessä 30 päivän ajan. Komennon suorittamisen helpottamiseksi package.json-tiedostoon lisättiin "build-skripti. Skriptin voi suorittaa käyttäen joko Yarn tai NPM-työkalua.

Erään verkkosivun maaperän kosteussensoreita käsittelevän julkaisun kommenttikentässä oli maininta, että jatkuva virransyöttö sensorille voi tuhota sensorin elektrolyysin takia. Tästä johtuen maaperän kosteussensorin virransyöttö muutettiin toimimaan releellä, jotta sensorille tulisi lisää käyttöikää. Mittauslaitteen lähdekoodissa tehtiin muutos sensorien lukuoperaatioiden suorittamiseen. Lukuoperaatiot tehdään SensorManager-moduulissa, joka vähensi koodirivien määrää huomattavasti mittauslaitteen pääohjelmaa suorittavassa DeviceManager-moduulissa. Muutos kastelusekvenssin aloittamiseen: kastelusekvenssin päätös tapahtuu aina kosteussensorin ja vesisäiliön arvojen mukaan. Jos maaperä on kuiva mutta vesisäiliössä ei ole tarpeeksi vettä, vesipumppua ei aktivoida. Koska vesipumpun voitelu toimii veden avulla, sen käyttäminen kuivana saattaisi aiheuttaa vaurioita tai rikkoa sen.

Testaus

Sprintin toteutuksen testaamisen läpivienti tapahtui manuaalisella testauksella. Toiminnallisuudet todettiin valmiiksi, kun monitorointisovellus esitti muutokset oikein käyttöliittymässä. Testauksen mittarina toimi myös onnistunut lähdekoodin kääntäminen EAS-pilvipalvelussa ja julkaisupaketin asentaminen älypuhelimeen.

Arviointi

EAS-palvelun käyttö julkaisupaketin luomisessa oli erittäin helppoa. Julkaisupaketin luominen on mahdollista myös paikallisella Expo-asennuksella, mutta tämä vaatii sen, että pakettien versiot ja työkalut ovat yhteensopivia keskenään. Pilvipalvelun käyttö poisti tämän vaatimuksen, minkä ansiosta julkaisupaketin luominen oli helppoa ja vaivatonta.

Kuva 5.4: Mittauslaite toiminnassa. Sprint 5.

5.5 Mittauslaitteen ohjelmisto

Mittauslaitteen ohjelmisto on kehitetty käyttäen Arduino IDE ja Microsoft Visual Studio Code -koodieditoreita. Arduino IDE:tä hyödynnettiin mittauslaitteen lähdekoodissa käytettävien kirjastojen hallintaan ja lähdekoodin kääntämiseen. Microsoft Visual Studio Code:lla toteutettiin mittauslaitteen ohjelmointi ja ohjelmointikielenä käytettiin C++. Mittauslaitteen ohjelmiston tarkoituksena on mahdollistaa sensoreiden käyttäminen mittauksissa, vesipumpun ohjaaminen sekä kerätyn tiedon siirtäminen taustajärjestelmään salatussa muodossa langattoman lähiverkon kautta. Mittauslaitteen lähdekoodi on saatavilla kokonaisuudessaan Github-palvelussa. Etätietovaraston URL-osoite löytyy liitteestä B.

Kirjastot

Mittauslaitteen ohjelmisto käyttää useita kirjastoja eri toiminnoissa, kuten WLAN-tietoliikenteessä, kerättyjen tietojen salaamisessa AES-lohkosalausalgoritmilla, tietokantakyselyiden muodostamisessa Google Firebase taustajärjestelmään ja sensorien mittauksissa. Sensoreiden mittauksissa hyödynnettiin Adafruit:in valmiita ohjelmistokirjastoja kuten *Adafruit BMP280 library* ja *DHT sensor library*. Lokitietojen

aikaleiman muodostuksessa hyödynnettiin kirjastoja *NTPClient*, *WifiUdp* ja *TimeLib*. Kirjastot otettiin käyttöön *#include*-direktiivillä. Edellä mainittujen kirjastojen hyödyntäminen nopeutti kehitysprosessia, sillä ne tarjoavat valmiita funktioita halutttujen toiminnallisuuksien, kuten taustajärjestelmään kirjautumisen toteuttamiseksi. Listauksessa 5.1 esitetään kirjastojen käyttöönotto lähdekoodissa.

```
1 // Kirjasto WiFi-yhteyden muodostamista varten
2 #include <ESP8266WiFi.h>
3 // Kirjasto I2C-protokollan alustamiseen sensorille BMP280
4 #include <Wire.h>
5 // Kirjasto sensoreiden lukua varten
6 #include <Adafruit_Sensor.h>
7 // Kirjasto sensorille BMP280
8 #include <Adafruit_BMP280.h>
9 // Kirjasto sensorille DHT22
10 #include <DHT.h>
```

Listaus 5.1: Ote mittauslaitteen lähdekoodista, kirjastot

Setup-funktio

Mittauslaitteen ohjelmakoodissa Setup-funktio suoritetaan kerran mittauslaitteen käynnistyessä. Funktiossa suoritetaan sarjaliikenteen, sensoreiden käyttämien digitaalisten pinnien ja ohjelmiston koodimoduulien alustaminen. Lisäksi tämä funktio suorittaa mittauslaitteen rekisteröinnin taustajärjestelmään. Listauksessa 5.2 esitetään ote mittauslaitteen Setup-funktiosta.

```
1 ...
2 void DeviceManager::setup() {
3     // Sarjaliikenteen alustaminen
4     Serial.begin(SERIAL_BAUD_RATE);
5     // Alustetaan digitaalisten pinnien tilat
6     pinMode(DIGITAL_CD74HC4051E_CONTROL_PIN_2, OUTPUT);
7     pinMode(DIGITAL_CD74HC4051E_CONTROL_PIN_3, OUTPUT);
8     pinMode(DIGITAL_SOIL_MOISTURE_SENSOR_PIN, OUTPUT);
9     pinMode(DIGITAL_WATER_PUMP_PIN, OUTPUT);
10    // Asetetaan vesipumpun ja maaperän kosteussensorin tilaksi LOW
11    digitalWrite(DIGITAL_WATER_PUMP_PIN, LOW);
12    digitalWrite(DIGITAL_SOIL_MOISTURE_SENSOR_PIN, LOW);
13    // Alustetaan koodimoduulit
```

```
14      initModules();
15      // Asetetaan muuttujiin mittauslaitteen Id ja SSID
16      deviceId = getDeviceId();
17      networkName = getNetworkName();
18      // Tarkastetaan onko mittauslaite rekisteröity taustajärjestelmään
19      if (!isDeviceAuthorized(deviceId)) {
20          registerDeviceForAuthorization(deviceId);
21      } else {
22          Serial.println("Device is authorized.");
23      }
24  }
```

Listaus 5.2: Ote mittauslaitteen lähdekoodista, Setup-funktio

Loop-funktio

Mittauslaitteen käynnistymisen jälkeen ohjelma ajaa jatkuvasti Loop-funktiota. Tässä ohjelman funktiossa suoritetaan sensoreiden mittaukset ja mittauslaitteen päätoiminto eli kastelusekvenssi. Listauksessa 5.3 esitetään ote mittauslaitteen Loop-funktiosta.

```
1  void DeviceManager::loop() {
2      // Laitteen käynnistämisestä kulunut aika millisekunteina
3      unsigned long currentMillis = millis();
4      // Tarkastus sensoreiden mittauksista ja maaperän kosteusarvosta
5      if (sensorReadingsDone && startWateringSequence) {
6          handleWateringSequence(currentMillis);
7      } else {
8          // Tarkastetaan onko aika tehdä sensorien mittaukset.
               Mittaukset suoritetaan 29 minuutin välein.
9          if ((currentMillis - previousSensorMillis >= SENSOR_INTERVAL)
               && isDeviceAuthorized(deviceId)) {
10             // Suoritetaan mittaukset sensoreilta
11             handleSensorReadings(currentMillis);
12         } else {
13             // Lähetetään luodut lokitapahtumat taustajärjestelmään
14             eventModule.loop();
15         }
16         // Tarkastetaan onko aika mitata maaperän kosteus. Mittaus
               suoritetaan 12 tunnin välein.
17         if ((currentMillis - previousSoilMoistureMillis >=
               SOIL_MOISTURE_INTERVAL) && isDeviceAuthorized(deviceId)) {
```

```
18              // Suoritetaan maaperän kosteuden mittaus
19              handleSoilMoistureReading(currentMillis);
20          }
21      }
22      // Tarkastetaan onko vesipumppu käynnissä ja pysäytetään
            vesipumppu kun kastelusekvenssi on ollut aktiivisena 12
            sekuntia
23      if (waterPumpActivated && (
24          currentMillis - waterPumpActivatedMillis >= WATERING_SEQUENCE)
25      ) {
26          handleWaterPumpDeactivation();
27      }
28  }
```

Listaus 5.3: Ote mittauslaitteen lähdekoodista, Loop-funktio

Muut mittauslaitteen funktiot

Muita mittauslaitteen funktioita on lokitapahtumien luonti, kerättyjen tietojen salaaminen AES-lohkosalausalgoritmilla sekä tietojen lähettäminen taustajärjestelmään. Lokitapahtumat antavat käyttäjälle tietoa mittauslaitteen suorittamista tapahtumista ja virhetilanteista, mikä auttaa järjestelmän ylläpidossa. Lisäksi lokitapahtumat mahdollistavat tehokkaan vianetsinnän. Kerättyjen tietojen salaaminen AES-lohkosalausalgoritmilla parantaa mittauslaitteen tietoturvaa. Vaikka mittauslaitteelta lähetettävät tiedot ei sisällä arkaluontoista tietoa kuten henkilötietoja, tietoliikenteeen tietojen salaaminen on hyvä käytäntö. Tietojen lähettäminen taustajärjestelmään mahdollistaa kasvin tilan reaaliaikaisen seurannan monitorointisovelluksessa.

5.6 Monitorointisovelluksen ohjelmisto

Monitorointisovelluksen ohjelmisto on kehitetty Microsoft Visual Studio Code koodieditorilla. Toteutukseen valittiin Facebook:in kehittämä React Native sovelluskehys ja Microsoft:in kehittämä Typescript ohjelmointikieli. Monitorointisovelluksen ohjelmiston tarkoituksena on tarjota käyttäjälle mahdollisuus tarkastella hänen määrittelemässä langattomassa lähiverkossa toimivien useiden eri mittauslaitteiden keräämiä tietoja. Monitorointisovelluksen lähdekoodi on saatavilla kokonaisuudessaan Github-palvelussa. Etätietovaraston URL-osoite löytyy liitteestä B.

Kirjastojen ja komponenttien käyttöönotto

Monitorointisovelluksen ohjelmisto käyttää useita kirjastoja ja komponentteja eri toiminnoissa, kuten taustajärjestelmään rekisteröitymisessä ja kirjautumisessa, mittauslaitteen tietojen haussa sekä lokitietojen esittämisessä. Taustajärjestelmään rekisteröitymisessä ja kirjautumisessa sekä tietokantakyselyiden luomisessa käytettiin *firebase* kirjastoa. Käyttöliittymäkomponentit hyödynsivät suurilta osin *react-native* kirjaston komponentteja, mutta osa monitorointisovelluksessa käytettävistä komponenteista luotiin käyttämällä *react-native-paper* kirjastoa. Kyseinen kirjasto tarjoaa laajan valikoiman valmiita käyttöliittymäkomponentteja. Kirjastot ja komponentit otettiin käyttöön *import*-komennolla. Edellä mainitun sovelluskehyksen, kirjastojen ja komponenttien hyödyntäminen nopeutti kehitysprosessia, sillä React Native mahdollistaa natiivisovelluksien teon ilman erillisiä koodipohjia iOS- ja Android-laitteille. Lisäksi Google Firebase tarjoaa nopean ja helpon tavan prototyyppien kehittämiselle tarjoamalla valmiita palveluita kuten reaaliaikaisen tietokannan. Listauksessa 5.4 esitetään kirjastojen ja komponenttien käyttöönotto lähdekoodissa.

```
1  // Otetaan käyttöön React
2  import React from 'react';
3  // Otetaan käyttöön komponentit View, Text ja StyleSheet react-native
       kirjastosta
4  import { View, Text, StyleSheet } from 'react-native';
5  // Otetaan käyttöön komponentti TextInput
6  import { TextInput } from 'react-native-paper';
7  // Otetaan käyttöön react-hookit useForm ja Controller
8  // lomakkeen validointia varten
9  import { useForm, Controller } from 'react-hook-form';
10 // Otetaan käyttöön EMAIL_REGEX säännöllinen lauseke
11 import { EMAIL_REGEX } from '../../utils/regex';
12 // Otetaan käyttöön AppTitle komponentti
13 import AppTitle from '../common/app-title';
14 // Otetaan käyttöön CustomButton komponentti
15 import CustomButton from '../common/custom-button';
16 // Otetaan käyttöön defaultStyles objekti joka sisältää sovelluksessa
       käytettäviä tyylejä
17 import defaultStyles from '../../assets/themes/default-styles';
```

Listaus 5.4: Ote monitorointisovelluksen lähdekoodista, kirjastojen ja komponenttien käyttöönotto

Käyttäjän rekisteröityminen ja kirjautuminen

Jotta monitorointisovelluksen käyttäjä pystyy tarkastelemaan mittauslaitteiden tietoja, käyttäjän tulee rekisteröityä ja kirjautua palveluun. Käyttäjätilin luominen tapahtuu rekisteröintinäkymässä, jossa käyttäjä voi määrittää käyttäjätilin sähköpostin sekä salasanan. Määritetyt tiedot lähetetään taustajärjestelmään ja jos tiliä ei ole jo olemassa, ponnahdusilmoitus kertoo käyttäjälle tilin onnistuneesta luomisesta. Tämän jälkeen käyttäjä voi navigoida kirjautumisnäkymään, jossa hän syöttää käyttäjätilin sähköpostin sekä salasanan. Onnistunut sisäänkirjautuminen näyttää käyttäjälle ponnahdusilmoituksen,jonka jälkeen käyttäjä viedään näkymään joka listaa mittauslaitteet. Listauksessa 5.5 esitetään ote käyttäjän kirjautumisesta.

```
1  ...
2  // Asetetaan firebase Auth-instanssi vakiomuuttujaan auth
3  const auth = getAuth();
4  // Määritetään tyyppi SignInScreenProps
5  type SignInScreenProps = StackScreenProps<any, 'SignIn'>;
6  // "Kirjaustumisnäkymä"-komponentin määritys
7  const SignInScreen: React.FC<SignInScreenProps> = () => {
8    // Funktio joka hoitaa käyttäjän sisäänkirjautumisen ja antaa tietoa
         rajapinnan palauttamista virheistä ponnahdusilmoituksessa.
9    const handleEmailPasswordSignIn = async (
10     { email, password }: FormData
11   ) => {
12     try {
13       await signInWithEmailAndPassword(auth, email, password);
14       Toast({ message: `Logged in as ${email}` });
15     } catch (apiError) {
16       // Asetetaan apiError virhe muuttujaan tyypillä AuthError
17       const errorObject = apiError as AuthError;
18       // Geneerinen virheviesti jos virhekoodia ei löydy.
19       let errorMessage = 'An error occurred while signing in.';
20       // Tarkastetaan erilaiset rajapinnan virhekoodit ja asetetaan
             oikeanlainen virheviesti virhekoodin mukaisesti.
21       if (errorObject.code === 'auth/invalid-email') {
22         errorMessage = 'Please enter a valid email address.';
23       } else if (errorObject.code === 'auth/user-disabled') {
24         errorMessage = 'This user account has been disabled.';
25       } else if (errorObject.code === 'auth/user-not-found') {
26         errorMessage = 'User not found. Please check the email and
               password.';
```

```
27        } else if (errorObject.code === 'auth/wrong-password') {
28          errorMessage = 'Email not found or invalid password. Please
                try again.';
29        }
30        // Näytetään virheviesti ponnahdusilmoituksessa
31        Toast({ message: errorMessage });
32        // Konsoliloki virheiden debuggausta varten
33        console.error('Email/Password Sign-In Error:', apiError);
34      }
35    };
36    // Kirjautumisnäkymän käyttöliittymäkomponentit
37    return (
38      <View style={styles.container}>
39        <AuthForm buttonText="Sign In" onSubmit={
              handleEmailPasswordSignIn} />
40      </View>
41    );
42  };
43  // Exportoidaan SignInScreen komponentti
44  export default SignInScreen;
```

Listaus 5.5: Ote monitorointisovelluksen lähdekoodista, kirjautumisnäkymä

Mittauslaitteiden haku

Ennen mittauslaitteiden hakua, käyttäjän tulee asettaa käyttäjätilin asetuksissa langattoman lähiverkon nimi *SSID*, jossa mittauslaitteet toimivat. Asetuksissa suoritettavan mittauslaitteiden autorisoinnin jälkeen, käyttäjä voi navigoida “Devices“-näkymään joka listaa asetuksissa määritetyssä lähiverkossa toimivat mittauslaitteet. Mittauslaitteet haetaan tietokannasta kokoelmana ja niiden tiedot esitetään listassa. Listauksessa 5.6 esitetään ote mittauslaitteiden tietojen hakemisesta. Kun käyttäjä haluaa tarkastella yksittäisen mittauslaitteen tietoja, listasta valitun mittauslaitteen *id*-avain asetetaan navigointipolkuun ja käyttäjä siirretään “Device info“-näkymään, jossa oikea mittauslaite suodatetaan mittauslaitteiden kokoelmasta navigointipolussa olevalla avaimella. Kuvassa 5.3 esitetään kuvakaappauksen muodossa mittauslaitteen tietojen yhteenveto-, loki- ja autorisointinäkymät.

```
1  ...
2  // Funktio joka hakee mittauslaitteiden kokoelman tietokannasta
3  const fetchDevicesData = React.useCallback(() => {
4      // Asetetaan loadingDevicesData-boolean arvoksi true
```

```
5     setLoadingDevicesData(true);
6     // Funktio joka suoritetaan onnistuneen hakuoperaation jälkeen
7     const onDataFetched = (data: any) => {
8       // Jos saatu data on tyhjä, asetetaan laitteiden data tyhjäksi
            objektiksi muuten asetetaan arvoksi vastaanotettu data.
9       if (data == null) {
10        setDevicesData({});
11      } else {
12        setDevicesData(data);
13      }
14      // Asetetaan loadingDevicesData-boolean arvoksi false
15      setLoadingDevicesData(false);
16    };
17    // Funktio joka suoritetaan epäonnistuneen hakuoperaation jälkeen
18    const onError = () => {
19      // Asetetaan loadingDevicesData-boolean arvoksi false
20      setLoadingDevicesData(false);
21    };
22    // Asetetaan hakuehdot
23    const queryOptions = {
24      // Järjestetään tulokset ssid:n perusteella
25      orderByChildValue: 'ssid',
26      // Haetaan kokoelmasta solmut joiden ssid vastaa settings?.ssid
27      equalToValue: settings?.ssid,
28    } as QueryOptions;
29    // Kutsutaan tietokantaa hakufunktiolla
30    const unsubscribe = fetchFromDatabase(NODE_PATHS.DEVICES,
          onDataFetched, onError, queryOptions);
31    // Funktio joka suoritetaan kun komponentti menee "unmount"-tilaan
32    return () => {
33      unsubscribe();
34    };
35 }, [settings?.ssid]);
36 ...
37 // useEffect-hook joka kutsuu fetchDevicesData-funktiota kun user-
       muuttuja sekä käyttäjän settings-muuttuja ovat asetettu.
38 React.useEffect(() => {
39    if (user && settings != null) {
40      fetchDevicesData();
41    }
42 }, [user, settings, fetchDevicesData]);
```

Listaus 5.6: Ote monitorointisovelluksen lähdekoodista, mittauslaitteiden haku

6 Tulokset ja pohdinta

Tässä luvussa tarkastellaan tutkielman tuloksia ja pohditaan kuinka prototyypin suunnittelu ja toteutus onnistui. Alaluvuissa 6.1 ja 6.2 arvioidaan mittauslaitteen ja monitorointisovelluksen toteutusta. Toteutusten onnistumista arvioidaan alaluvussa 5.2 esitettyjen toimintojen toimivuuden ja luotettavuuden avulla. Prototyyppiä peilataan myös alalukuihin 4.2 sekä 4.3 ja tarkastellaan ilmenikö toteutuksessa kyseisissä alaluvuissa esiteltyjä hyötyjä tai haasteita. Alaluvussa 6.3 esitellään prototyypin jatkokehitysaiheet. Lisäksi alaluvussa 6.4 arvioidaan tutkielman toteuttamista suunnittelutieteen viitekehyksessä ja pohditaan muita maatalouden automaatioon liittyviä tutkimusaiheita.

6.1 Mittauslaitteen arviointi

Tämän tutkielman pohdintaa kirjoittaessa sprinttien aikana toteutettu mittauslaite on ollut toiminnassa noin viiden kuukauden ajan. Mittauslaite on toteuttanut sensoreiden mittauksia luotettavasti kuluneen ajanjakson aikana. Tiedonsiirto taustajärjestelmään on ollut myös luotettavalla tasolla, sillä taustajärjestelmään lähetetty aikaleima viimeisestä mittaus- ja kasteluajankohdasta on näkynyt käyttöliittymässä oikein. Myös maaperän kosteuden mittaamisesta luodun lokitiedon aikaleima tukee tätä huomiota. Mittauslaitteen päätoiminto eli vesipumpun ohjaaminen kasvien kastelua varten koki pienen takaiskun noin neljän kuukauden käytön jälkeen. Prototyypissä käytetty halpa uppopumppu ruostui käyttökelvottomaksi, mikä esti kastelusekvenssin käynnistymisen kunnes uppopumppu vaihdettiin. Tämä on ainut suuri prototyypin luotettavuutta horjuttava huomio kuluneelta ajankohdalta. Näitä huomioita tarkastellaan tarkemmin seuraavissa kappaleissa.

Mittaukset ja tiedonsiirto

Mittauslaitteen suorittamat mittaukset ovat antaneet johdonmukaisia mittaustuloksia koko sen toiminnassa olon aikana. Mittauslaitteen mittaamia huoneilman lämpötilan arvoja on verrattu toiseen lämpömittariin ja ne ovat olleet noin asteen verran lämpimämpiä. Tämä johtuu todennäköisesti siitä, että sensori joka mittaa lämpöti-

laa on ollut mittauslaitteen kotelon sisällä, jonka takia mittauslaitteesta tuleva hukkalämpö on nostanut kotelon sisälämpötilaa. Ilmankosteuden ja ilmanpaineen mittaustuloksia ei ole verrattu muiden laitteiden mittaustuloksiin. Mittaustuloksia tarkastellessa ilmankosteus on vaihdellut 20-80% välillä ja ilmanpaine 980-1010 hPa:n välillä. Vesisäiliön vedenkorkeutta seuraavan ultraäänietäisyyssensorin kalibroiminen oli myös onnistunut. Mittaukset ovat antaneet oikeanlaisia mittaustuloksia vesisäiliönä toimineesta yhden litran mittakannusta. Sprintissä 4 havaittuja heijastuksia, jotka vääristivät mittaustuloksia ei ole enää ilmennyt. Kasvin valonsaantia mittaava valovastus on myös reagoinut valon lisääntymiseen tai vähenemiseen. Maaperän kosteussensori on toiminut erinomaisesti ja sen antamat mittaustulokset ovat olleet luotettavia. Sensoriin syötetään virta releen avulla vain mittausten yhteydessä, mikä saattaa pidentää sen käyttöikää. Koska kastelupäätös tehdään maaperän kosteussensorin mittaustuloksen perusteella, tämä vastaa luvussa 4.2 esitettyyn kastelunjärjestelmän hyötyyn "Kyky ennustaa ja tehdä perusteltuja päätöksiä".

Prototyypin järjestelmän ytimenä toimiva ESP8266-järjestelmäpiiriin pohjautuva KS0367 keyestudio ESP8266 kehitysalusta on osoittautunut hyväksi valinnaksi. Kehitysalustaan valmiiksi integroidun WiFi-moduulin tai sitä hyödyntävän ohjelmistokirjaston käytössä ei ole havaittu ongelmia. Prototyyppiä on käytetty huoneistossa, jossa langaton lähiverkko on ollut jatkuvasti saatavilla. Tästä johtuen tiedonsiirtoon liittyviä haasteita ei ole ilmennyt. Jos prototyyppiä käytettäisiin ulkotiloissa, tiedonsiirron luotettavuutta tulisi testata kattavasti.

Vesipumpun ohjaus

Kun maaperänkosteussensorilta saatu mittaustulos alittaa mittauslaitteen lähdekoodissa asetetun raja-arvon kastelusekvenssin aloittamiselle, vesipumppu käynnistetään releen avulla. Tämä toiminto on ollut luotettava ohjelmiston tasolla, mutta kuten tämän pääluvun alussa mainittiin, vesipumpun ruostuminen aiheutti vikatilanteen kastelusekvenssissä kunnes uppopumppu vaihdettiin. Prototyypissä käytetty vesipumppu oli halpa 3.3V jännitteellä toimiva pohjaimuinen uppopumppu, jossa ei ollut suodatinta. Rikkinäistä uppopumppua tarkastellessa sen sisällä oleva moottori oli ruostunut. Virransyöttö rikkinäiseen uppopumppuun ei enää käynnistänyt moottoria. Tämä huomio tukee alaluvussa 4.3 esitettyä IoT:n haastetta kastelujärjestelmissä, "Laitteet ovat alttiita haastaville ympäristöolosuhteille". Laitteiden ja komponenttien laadulla on siis merkitystä, jos prototyypille halutaan taata pitkä käyttöikä. Prototyypin käyttö vähentää kasteluun käytetyn veden kokonaiskulutusta noin

1 dl per kastelukerta, mikä tukee luvussa 4.2 esitettyä hyötyä pienentyneestä veden kokonaiskulutuksesta.

6.2 Monitorointisovelluksen arviointi

Monitorointisovellus on osoittautunut testausjakson aikana käytännölliseksi tavaksi tarkastella prototyypin tilaa. Mittauslaitteen mittaustulosten muutokset esitetään reaaliaikaisesti kun tiedot tallentuu taustajärjestelmään, joka täyttää luvussa 4.2 esitetyn kastelun automatisoinnin hyödyn "Reaaliaikainen monitorointi". Aikaleimat tarjoavat käyttäjälle nopeasti tiedon siitä, milloin viimeisimmät mittaukset ja kastelu on suoritettu. Myös kahdesti päivässä tallennettu lokitieto maaperän kosteuden tilasta kertoo, että prototyyppi toimii suunnitellusti ja luotettavasti.

Sovelluskehys

Monitorointisovelluksen toteuttaminen React Nativen avulla toteutui odotetusti hyvin aikaisemmin kertyneen kokemuksen ansiosta. Aiempi tietämys ja taito kyseisestä sovelluskehyksestä antoi vankan perustan projektin etenemiselle ja auttoi ongelmien ratkaisussa. React nativen sekä React Native Paper-kirjaston valmiilla komponenteilla oli helppoa rakentaa käyttöliittymä, joka mahdollisti nopean kehitystyön. Komponenttien kustomointi sekä tyylittelyn muuttaminen vastaamaan sovelluksen tarpeita oli myös suoraviivaista ja selkeää hyvän dokumentaation ansiosta. Myös valmiin sovelluksen kääntäminen APK-julkaisupaketiksi älypuhelimeen asentamista varten onnistui helposti React nativen tukemien työkalujen avulla.

Rajapinnat ja taustajärjestelmä

Mittauslaitteen tietoihin ja käyttäjän asetuksiin liittyvät rajapintakutsut toimivat halutulla tavalla ja luotettavasti. Tätä edesauttaa se että taustajärjestelmä on toteutettu Googlen tarjoamalla Firebase-palvelulla. Palvelu tarjoaa monia valmiita ratkaisuja, jotka säästävät huomattavasti aikaa kehitystyössä. Hyödynnettyjä ratkaisuja olivat mm. pääsynhallinta, tietokanta sekä reaaliaikainen tietojen synkronointi.

Käyttöönotto

Prototyypin käyttöönotto nykyisellä toteutuksella vaatii teknologia-orientoituneen

henkilön työpanosta. Mittauslaitteen kokoaminen, ohjelmiston siirto mittauslaitteelle, monitorointisovelluksen kääntäminen APK-julkaisupaketiksi sekä taustajärjestelmän käyttöönotto voi osoittautua esteeksi prototyypin hyödyntämiselle kasvien kasvattamisessa. Tämä haaste on huomioitu alaluvussa 4.3.

6.3 Jatkokehitys

Prototyypin jatkokehitysaiheet koostuvat olemassaolevien toiminnallisuuksien parannuksista sekä ehdotuksista, joilla prototyypin toimintaa voitaisiin parantaa entisestään. Luvun lopussa tarkastellaan myös muita maatalouteen liittyviä asioita, jotka voitaisiin automatisoida IoT:n avulla.

Mittauslaite

Mittauslaitteen jatkokehitysaiheita ajatellen, olisi hyvä tehdä ensin parannukset nykyiselle ohjelmistolle ja uppopumpulle. Kastelusekvenssin jälkeen tulisi ottaa mittaukset maaperän kosteudesta sekä vesisäiliön vedenmäärästä, jotta monitorointisovelluksessa näytettävä tieto olisi ajantasaista. Uppopumppu tulisi myös korvata luotettavammalla laitteella, sillä sen vaihtaminen neljän kuukauden välein ei ole ihanteellista. Jos prototyyppi toimisi ulkona, kastelusekvenssin kastelupäätöksen tekoa voisi jatkokehittää huomioimaan säätiedot, kuten Dahane et al. [9] ehdottivat tutkimuksessaan. Kastelupäätöksen tekoa voisi viivästyttää jos kolmannen osapuolen palvelusta saatava säätieto lupaa sadetta tai BMP280 sensorista mitattu ilmanpaine tippuu jonkin tietyn asetetun raja-arvon alle.

Jos WiFi-verkon tai Firebase palvelimen tiedot muuttuvat, mittauslaitteen toiminta keskeytyy, koska nykyisessä ohjelmistossa tarvittavat tiedot on tallennettu *config*-otsikkotiedostoon. Tämä olisi mielenkiintoista ratkaista OTA (Over-The-Air) langattoman ohjelmistopäivityksen mahdollistamisella, jota ESP8266-järjestelmäpiiri tukee. Lepotilan tukeminen lisäisi virransäästöä erityisesti silloin, jos mittauslaite toimii paristoilla.

Monitorointisovellus

Tällä hetkellä käyttäjän tulee asettaa asetuksiinsa langattoman lähiverkon nimi, jossa mittauslaite on toiminnassa, ennen kuin kyseisessä lähiverkossa toimivien mittauslaitteiden tiedot voidaan hakea käyttöliittymään. Jos tätä tietoa ei ole asetet-

tu, käyttäjälle tulisi tarjota mahdollisuus syöttää se esimerkiksi dialogin avulla tai siirtämällä käyttäjän näkymä asetuksiin heti kirjautumisen jälkeen. Nykyisessä toteutuksessa tyhjä mittauslaitteiden lista antaa palautetekstin, joka kehottaa tarkistamaan asetukset, mutta edellä mainittu avustettu tietojen pyytäminen voisi lisätä monitorointisovelluksen käytettävyyttä.

Taustajärjestelmä

Firebase toimii oman kokemukseni mukaan hyvin prototyypin tekemiseen. Tästä huolimatta jos prototyyppiä jatkokehitettäisiin, harkitsisin taustajärjestelmän muuttamista täysin kustomoiduksi. Kustomointi tarkoittaisi Google Firebase-palvelun käytön lopettamista taustajärjestelmänä. Taustapalvelimen ohjelmisto voitaisiin toteuttaa esimerkiksi Spring Boot-sovelluskehyksellä, joka voidaan ohjelmoida Java tai Kotlin ohjelmointikielillä. Tietokantaratkaisun vaihtaminen SQL-pohjaiseen tietokantaan nykyisen NoSQL-pohjaisen Firebase Realtime Databasen sijaan mahdollistaisi monimutkaisempien tietokantakutsujen luomisen. Rajapinnat voisi kehittää esimerkiksi Graphql:n avulla, sillä Spring Boot tarjoaa valmiita Graphql-kirjastoja rajapintojen määrittelyyn ja toteuttamiseen. Typescript:illä kehitetty monitorointisovellus saisi Graphql-rajapintojen avulla tarkasti tyypitettyjä olioita jo taustajärjestelmästä asti. Tätä uutta taustajärjestelmää voisi ajaa luvussa 3.4.1 esitettyjen pilvipalveluntarjoajien palveluissa. Kustomoidun taustajärjestelmän käyttöönotto vaatisi laajan refaktorointityön mittauslaitteen ja monitorointisovelluksen lähdekoodissa. Tämä muutos nostaisi myös prototyypin käyttöönoton rajaa entisestään, joten tätä muutosta tulisi harkita perusteellisesti. Jatkokehitystä aloittaessa tulisi myös tutustua ja selvittää voisiko Firebasen tarjoamat Realtime Databasen laajennukset ratkaista tarpeen monimutkaisempien tietokantakutsujen luomiselle.

Prototyypin ylläpidettävyys, skaalautuvuus ja laajennettavuus

Kuten alaluvussa 6.2 ja aiemmassa kappaleessa mainittiin, prototyypin käyttöönotto vaatii erityistä osaamista. Tässä kappaleessa pohditaan, kuinka kastelujärjestelmää voisi kehittää siten, että kuka tahansa voisi ottaa sen käyttöön ja laajentaa sitä muilla mittauslaitteilla. Taustajärjestelmänä käytetty Firebase skaalautuu automaattisesti sovelluksen käytön mukaan, vapauttaen kehittäjän sekä ylläpitäjän skaalautuvuuden hallinnasta. Taustajärjestelmän ilmainen versio sallii 100 eri laitteen samanaikaisen tietokantayhteyden Realtime Database tietokantaan, mutta maksulli-

nen versio sallii jopa 200 000 samanaikaista yhteyttä. Mittauslaitteen kytkentäkaavio on saatavilla Github-palvelussa, mutta kytkentöjen teossa voisi auttaa esimerkiksi ohjeet, jossa kytkennät on kuvattu vaihe vaiheelta. Mittauslaitteen jatkokehitysaiheissa mainittu OTA-päivityksen tuki, voisi ratkaista ohjelmiston lataamisen mittauslaitteelle. Mittauslaitteen ESP8266-järjestelmäpiiri tulisi konfiguroida käyttämään oletuksena tukiasematilaa (AP-tila). Tällöin mittauslaitteen konfiguraation, kuten WiFi-verkon ja Firebase palvelimen vaatimat tiedot voitaisiin päivittää ottamalla yhteys mittauslaitteen tarjoamaan verkkoon. Mittauslaitteen verkossa voisi toimia yksinkertainen verkkosivu, josta löytyisi vaadittavat tekstikentät asetuksien tallentamiselle. Verkkosivulla voisi mahdollistaa myös OTA-päivityksen tarjoamalla käyttäjälle tekstikenttä, joka ottaa vastaan https-osoitteen. Käyttäjä voisi liittää tähän tekstikenttään osoitteen, joka osoittaa Github-palvelussa olevaan ohjelmiston binääritiedostoon. Kun tekstikentän syöte on validoitu, binääritiedosto ladattaisiin ja asennettaisiin mittauslaitteelle, jonka jälkeen mittauslaitteen tulisi olla toimintakunnossa. Monitorointisovellus vaatisi myös muutoksia. Käyttäjälle tulisi mahdollistaa Firebase palvelimen vaatimien tietojen lisääminen jo käyttäjätiliä luodessa, jotta käyttäjä luotaisiin oikeaan palvelin instanssiin. Firebase palvelun käyttöönotto on tehty helpoksi, mutta käyttöönottoon vaadittavat vaiheet voisi dokumentoida ja laittaa saataville Github-palveluun.

6.4 Tutkielman arviointi ja muita tutkimusaiheita

Tässä alaluvussa arvioidaan tutkielman toteutusta konstruktiivisella tutkimusotteella ja tarkastellaan myös muita maatalouteen liittyviä asioita, joita voisi automatisoida.

Konstruktiivinen tutkimusote

Tutkielman toteutus konstruktiivisella tutkimusotteella osoittautui hyväksi tavaksi tutkielman läpivientiin. Soveltamisalueeseen tutustuminen kirjallisuuskatsauksen avulla antoi hyvän tietopohjan tutkielman toteutuksessa tehtävälle kehitystyölle. Tietopohjan avulla tutkielman aikana toteutettavan kastelujärjestelmän vaatimusten määrittely oli helpompaa, sillä soveltamisalue oli tullut tutuksi kirjallisuuskatsauksen aikana. Tutkielman tuotoksena valmistunut kastelujärjestelmä on toiminut luotettavasti, tosin uppopumppu on vaihdettava laadukkaampaan toimilaitteeseen.

Tutkielman kehitystyö suoritettiin hyödyntämällä ketteriä menetelmiä, mutta koska kyseessä oli itsenäisesti opinnäytetyönä toteutettu tutkielma, artefaktin testaus ja arviointi on jäänyt yhden henkilön vastuulle. Näissä vaiheissa olisi hyvä saada palautetta joltakin ulkoiselta taholta tai ryhmältä. Palautetta antavat henkilöt voisivat olla sovellusalueen asiantuntijoita tai henkilöitä, jotka olisivat ottamassa kehitettävää ratkaisua käyttöön. Tutkielma on toistettavissa hyödyntämällä taulukossa 2.1 esitettyjä suunnittelutieteen soveltamisen ohjeita.

Muita maatalouteen liittyviä automatisoitavia asioita

Alaluvussa 3.5 esiteltyjen IoT-sovellusten avulla voidaan automatisoida myös maatalouden harjoittamisesta johtuvien negatiivisten ympäristövaikutusten valvonta.

Älykkäät vesijärjestelmät mahdollistavat maataloudesta aiheutuvien vesistöjä kuormittavien vaikutusten seurannan automatisoinnin. Seuranta toteutettaisiin veden laadun mittauksilla maataloutta harjoittavalla valuma-alueella. Älykkään vesijärjestelmän sensorit asetettaisiin seurannassa olevan valuma-alueen ylä- ja alaosiin. Näiden kahden valuma-alueen osien mittaustuloksia verrattaisiin toisiinsa ja vertailussa käytettävinä muuttujina voisi toimia vedessä olevien ravinteiden määrä, happamuus ja sameus. Vertailulla saataisiin tieto siitä, että kuinka paljon alueen maatalous vaikuttaa vesistöihin laskevan veden laatuun.

Myös maataloudesta aiheutuvien kasvihuonekaasupäästöjen seurannan automatisointi voitaisiin toteuttaa älykkään ympäristön IoT-sovelluksella. Maatilan ympäristöön asennettavat sensorit pystyisivät antamaan reaaliaikaisia kasvihuonekaasupäästöjen mittaustuloksia mm. maaperästä, viljelysmaista tai tuotantoeläimistä. Mittaustuloksissa tarkasteltavia muuttujia voisivat olla hiilidioksidi ja metaani. Vertailukohteena toimisi oletusarvot, jotka perustuisivat muuttujien pitoisuuksiin ilmakehässä. Vertailulla voitaisiin tarkistaa onko maatilalla huomattava vaikutus kasvihuonekaasupäästöihin.

Ympäristövaikutusten valvonnan lisäksi muita maatalouteen liittyviä automatisoitavia asioita voisi olla kasvihuoneen lämmönhallinta. Kurniawan et al. [21] tutkimuksessa esittämää toimintoa kasvihuoneen lämpötilan laskemiselle voisi jatkokehittää siten, että kasvihuoneen ikkunat tai kattoluukut aukaistaan tai suljetaan aktuaattorin avulla lämpötilan noustessa tai laskiessa ylä- tai alaraja-arvon ohi. Tutkijoiden ratkaisussa käyttämää tuuletinta voisi edelleen hyödyntää kasvihuoneen ilmanvaihdon lisäämiseksi kun lämpötila ylittää yläraja-arvon.

7 Yhteenveto

Tämän suunnittelutieteen viitekehystä hyödyntävän tutkielman kehitystyön tuotoksena syntyi kasvien kastelujärjestelmä, joka mahdollistaa kasvien monitoroinnin ja kastelun automatisoinnin. Kastelujärjestelmä koostuu IoT-laitteeksi luokiteltavasta mittauslaitteesta, monitorointisovelluksesta sekä taustajärjestelmästä. Mittauslaitteen keräämä tieto tallennetaan Google Firebase taustajärjestelmään, josta tiedot haetaan ja esitetään monitorointisovelluksessa.

Tutkielman alussa käsiteltiin suunnittelutieteen määritelmää, käsitteellistä kehystä sekä sen soveltamisen ohjeita tutkimuksissa, joiden avulla konstruktiivista tutkimusotetta hyödyntävä tutkimus on mahdollista toteuttaa. Tutkielman teoriaosuudessa esiteltiin esineiden internet käsitteeseen liittyvä laajasti hyväksytty arkkitehtuurimalli, sensorin ja aktuaattorin määritelmät sekä erilaisia langattomia viestintäteknologioita, joita hyödynnetään IoT-järjestelmissä. Lisäksi tarkasteltiin markkinoilla olevia pilvipalvelualustoja sekä niiden käyttömahdollisuuksia ja esiteltiin muutama esimerkki IoT-sovelluksista. Tutkielmassa tutustuttiin kirjallisuuskatsauksen avulla olemassa oleviin IoT-pohjaisiin automatisoituihin kastelujärjestelmiin ja tarkasteltiin monitorointi- ja kastelujärjestelmän tyypillistä toteutusta sekä esitettiin kastelun automatisointiin liittyviä hyötyjä ja haasteita. Tutkielmaan valittujen tutkimusten tulosten tarkastelu osoitti että kasvien monitorointi- ja kastelujärjestelmiä pystyy kehittämään ja automatisoimaan IoT:n avulla kustannustehokkaasti.

Aiemmissa tutkimuksissa toteutetut ratkaisut koostuivat pääosin mittauslaitteesta ja taustajärjestelmästä. Ratkaisujen pelkistetty toimintamalli oli seuraavanlainen: mittauslaite tuotti dataa, jonka pohjalta taustajärjestelmä suoritti kasteluun liittyviä päätöksiä. Osa taustajärjestelmistä toteutti nämä päätökset monimutkaisemmalla logiikalla ottaen huomioon sade-ennusteet tai käyttämällä optimoitua päätöksentekoa. Näiden kastelupäätösten pohjalta taustajärjestelmä lähetti komentoja vesipumpun tai kastelujärjestelmän aktuaattorin ohjaamiseen. Aiemmissa tutkimuksissa toteutettujen ratkaisujen mittauslaitteiden vertailu osoitti, että laitteistokokoonpanot olivat pääpiirteittäin samanlaiset, jos mittauslaitteen ja taustajärjestelmän välillä tapahtuvaan tietoliikenteeseen käytetty teknologia oli sama vertailtavien ratkaisujen välillä. Ratkaisujen taustajärjestelmissä oli enemmän vaihtelevuut-

ta. Osa taustajärjestelmistä oli täysin kustomoituja ja osa käytti markkinoilla olevia valmiita ratkaisuja analytiikan näyttämiseen graafisessa muodossa.

Tutkimuksen kehitystyössä hyödynnettiin ketterää menetelmää, joka on laajalti käytetty menetelmä ohjelmistokehityksessä ja it-projektien läpiviennissä. Kehitystyö alkoi määrittelemällä kehitettävän prototyypin vaatimukset. Vaatimukset pohjautuivat toiminnallisuuksiin ja ominaisuuksiin, jotka oli toteutettu aiemmissa tutkimuksissa tehtyihin kastelujärjestelmiin sekä kirjoittajan haluamiin lisätoimintoihin. Vaatimusten pohjalta valikoitiin komponentit, jotka mahdollistavat vaatimusten toteuttamisen halutulla tavalla. Sprinttien aikana toteutettiin ensin toiminnallisuudet ja ominaisuudet, jotka olivat kastelujärjestelmän toiminnan kannalta merkittäviä. Tästä syystä ensimmäisen sprintin tavoitteena oli mahdollistaa mittauslaitteen ja monitorointisovelluksen tietoturvallinen yhteys taustajärjestelmän kanssa. Tämän toiminnallisuuden valmistuttua oli mahdollista lisätä muut mittauksiin liittyvät toiminnot ja kastelujärjestelmän päätoiminto eli kastelusekvenssi.

Tulokset ja pohdinta luvussa tarkasteltiin tutkielman tuloksia ja pohdittiin kuinka prototyypin suunnittelu ja toteutus onnistui. Mittauslaitteen toteutus onnistui hyvin ohjelmiston kannalta, mutta uppopumppu tulee korvata luotettavampaan toimilaitteeseen. Monitorointisovelluksen toteutus oli onnistunut ja se täyttää sille asetetut vaatimukset mittauslaitteen tilan seurannan mahdollistamiseksi. Lisäksi luvussa esiteltiin kastelujärjestelmän jatkokehitysaiheita. Mittauslaitteen jatkokehityksessä tulisi keskittyä ensin nykyisen ohjelmiston parannuksiin sekä uppopumpun päivittämiseen. Esitettiin myös ehdotus kastelusekvenssin viivästyttämiseen jos säätiedot lupaavat sadetta. Monitorointisovelluksen jatkokehitysaiheena olisi sovelluksen käytettävyyden lisääminen. Taustajärjestelmän jatkokehitysaiheena esiteltiin kustomoidun ratkaisun käyttöönotto, joka mahdollistaisi monimutkaisempien tietokantakyselyiden tekemisen. Tämä kustomoitu taustajärjestelmä lisäisi kynnystä ratkaisun käyttöönottoon, joten sen toteuttamista tulisi harkita perusteellisesti. Tärkeimpänä jatkokehitysaiheena olisi prototyypin ylläpidettävyys ja laajennettavuus. Ehdotettu ratkaisu voisi vähentää käyttöönottoon liittyvää teknologisesti orientoituneen henkilön työpanosta. Luvun lopussa arvioitiin tutkielman toteutusta konstruktiivisella tutkimusotteella ja esiteltiin muita maatalouteen liittyviä automatisoitavia asioita. Suunnittelutieteen viitekehys tarjosi hyvät menetelmät ja työkalut tämän tyylisen konstruktiivisen tutkimuksen toteutukselle. Kehitystyön aikana artefaktin arviointiin ja testaukseen olisi hyvä saada palautetta joltakin ulkoiselta taholta tai ryhmältä.

Lähteet

[1] ADAME, T., CARRASCOSA-ZAMACOIS, M., JA BELLALTA, B. Time-sensitive networking in ieee 802.11 be: On the way to low-latency wifi 7. *Sensors 21*, 15 (2021), 4954.

[2] AGARWAL, P., JA ALAM, M. Open service platforms for iot. *Internet of Things (IoT) Concepts and Applications* (2020), 43–59.

[3] AGIWAL, M., ROY, A., JA SAXENA, N. Next generation 5g wireless networks: A comprehensive survey. *IEEE communications surveys & tutorials 18*, 3 (2016), 1617–1655.

[4] ANUSHA, K., JA MAHADEVASWAMY, U. Automatic iot based plant monitoring and watering system using raspberry pi. *International Journal of Engineering and Manufacturing 8* (2018), 55–67.

[5] ATZORI, L., IERA, A., JA MORABITO, G. The internet of things: A survey. *Computer networks 54(15)* (2010), 2787–2805.

[6] AYOUB, W., SAMHAT, A. E., NOUVEL, F., MROUE, M., JA PRÉVOTET, J.-C. Internet of mobile things: Overview of lorawan, dash7, and nb-iot in lpwans standards and supported mobility. *IEEE Communications Surveys & Tutorials 21*, 2 (2018), 1561–1581.

[7] BASKERVILLE, R., BAIYERE, A., GREGOR, S., HEVNER, A., JA ROSSI, M. Design science research contributions: Finding a balance between artifact and theory. *Journal of the Association for Information Systems 19*, 5 (2018), 3.

[8] COLLOTTA, M., PAU, G., TALTY, T., JA TONGUZ, O. K. Bluetooth 5: A concrete step forward toward the iot. *IEEE Communications Magazine 56*, 7 (2018), 125–131.

[9] DAHANE, A., BENAMEUR, R., JA KECHAR, B. An iot low-cost smart farming for enhancing irrigation efficiency of smallholders farmers. *Wireless Personal Communications 127* (2022), 3173–3210.

[10] DE CARVALHO SILVA, J., RODRIGUES, J. J., ALBERTI, A. M., SOLIC, P., JA AQUINO, A. L. Lorawana low power wan protocol for internet of things: A review and opportunities. Julkaisusarjassa *2017 2nd International multidisciplinary conference on computer and energy science (SpliTech)* (2017), IEEE, 1–6.

[11] GARCÍA, C. G., MEANA-LLORIÁN, D., LOVELLE, J. M. C., ET AL. A review about smart objects, sensors, and actuators. *International Journal of Interactive Multimedia & Artificial Intelligence 4*, 3 (2017).

[12] GARCÍA, L., PARRA, L., JIMENEZ, J. M., LLORET, J., JA LORENZ, P. Iot-based smart irrigation systems: An overview on the recent trends on sensors and iot systems for irrigation in precision agriculture. *Sensors 20*, 4 (2020), 1042.

[13] HEVNER, A. R., MARCH, S. T., PARK, J., JA RAM, S. Design science in information systems research. *Management Information Systems Quarterly 28*, 1 (2008), 6.

[14] HIDAYAT, T., MAHARDIKO, R., JA TIGOR, F. D. S. Method of systematic literature review for internet of things in zigbee smart agriculture. Julkaisusarjassa *2020 8th International Conference on Information and Communication Technology (ICoICT)* (2020), IEEE, 1–4.

[15] HOLLER, J., TSIATSIS, V., MULLIGAN, C., KARNOUSKOS, S., AVESAND, S., JA BOYLE, D. *Internet of things*. Academic Press, 2014.

[16] HOLMA, H., TOSKALA, A., JA NAKAMURA, T. *5G technology: 3GPP new radio*. John Wiley & Sons, 2020.

[17] IKPEHAI, A., ADEBISI, B., RABIE, K. M., ANOH, K., ANDE, R. E., HAMMOUDEH, M., GACANIN, H., JA MBANASO, U. M. Low-power wide area network technologies for internet-of-things: A comparative review. *IEEE Internet of Things Journal 6*, 2 (2018), 2225–2240.

[18] IRAWAN, Y., SABNA, E., AZIM, A. F., WAHYUNI, R., BELARBI, N., JA JOSEPHINE, M. M. Automatic chili plant watering based on internet of things (iot). *Journal of Applied Engineering and Technological Science (JAETS) 3(2)* (2022), 77–83.

[19] KHUTSOANE, O., ISONG, B., JA ABU-MAHFOUZ, A. M. Iot devices and applications based on lora/lorawan. Julkaisusarjassa *IECON 2017-43rd Annual Conference of the IEEE Industrial Electronics Society* (2017), IEEE, 6107–6112.

[20] KIM, W.-S., LEE, W.-S., JA KIM, Y.-J. A review of the applications of the internet of things (iot) for agricultural automation. *Journal of Biosystems Engineering 45* (2020), 385–400.

[21] KURNIAWAN, D., JA WITANTI, A. Prototype of control and monitor system with fuzzy logic method for smart greenhouse. *Indonesian Journal of Information Systems (IJIS) 3* (2021), 116–127.

[22] LONZETTA, A. M., COPE, P., CAMPBELL, J., MOHD, B. J., JA HAYAJNEH, T. Security vulnerabilities in bluetooth technology as used in iot. *Journal of Sensor and Actuator Networks 7*, 3 (2018), 28.

[23] MOUHA, R. A. Internet of things (iot). *Journal of Data Analysis and Information Processing 9*, 2 (2021), 77–101.

[24] OBAIDEEN, K., YOUSEF, B. A., ALMALLAHI, M. N., TAN, Y. C., MAHMOUD, M., JABER, H., JA RAMADAN, M. An overview of smart irrigation systems using iot. *Energy Nexus* (2022), 100124.

[25] PIERLEONI, P., CONCETTI, R., BELLI, A., JA PALMA, L. Amazon, google and microsoft solutions for iot: Architectures and a performance comparison. *IEEE access 8* (2019), 5455–5470.

[26] RAZA, U., KULKARNI, P., JA SOORIYABANDARA, M. Low power wide area networks: An overview. *ieee communications surveys & tutorials 19*, 2 (2017), 855–873.

[27] ROSE, K., ELDRIDGE, S., JA CHAPIN, L. The internet of things: An overview. *The internet society (ISOC) 80* (2015), 1–50.

[28] SEHRAWAT, D., JA GILL, N. S. Smart sensors: Analysis of different types of iot sensors. Julkaisusarjassa *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)* (2019), IEEE, 523–528.

[29] SIKARWAR, R., YADAV, P., JA DUBEY, A. A survey on iot enabled cloud platforms. Julkaisusarjassa *2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT)* (2020), IEEE, 120–124.

[30] SIKIMIĆ, M., AMOVIĆ, M., VUJOVIĆ, V., SUKNOVIĆ, B., JA MANJAK, D. An overview of wireless technologies for iot network. Julkaisusarjassa *2020 19th International Symposium INFOTEH-JAHORINA (INFOTEH)* (2020), IEEE, 1–6.

[31] SINHA, R. S., WEI, Y., JA HWANG, S.-H. A survey on lpwa technology: Lora and nb-iot. *Ict Express 3*, 1 (2017), 14–21.

[32] SOUMYALATHA, S. G. H. Study of iot: understanding iot architecture, applications, issues and challenges. Julkaisusarjassa *1st International Conference on Innovations in Computing & Net-working (ICICN16), CSE, RRCE. International Journal of Advanced Networking & Applications* (2016), vol. 478.

[33] SUBEESH, A., JA MEHTA, C. Automation and digitization of agriculture using artificial intelligence and internet of things. *Artificial Intelligence in Agriculture 5* (2021), 278–291.

[34] TIAN, L., SANTI, S., SEFERAGIĆ, A., LAN, J., JA FAMAEY, J. Wi-fi halow for the internet of things: An up-to-date survey on ieee 802.11 ah research. *Journal of Network and Computer Applications 182* (2021), 103036.

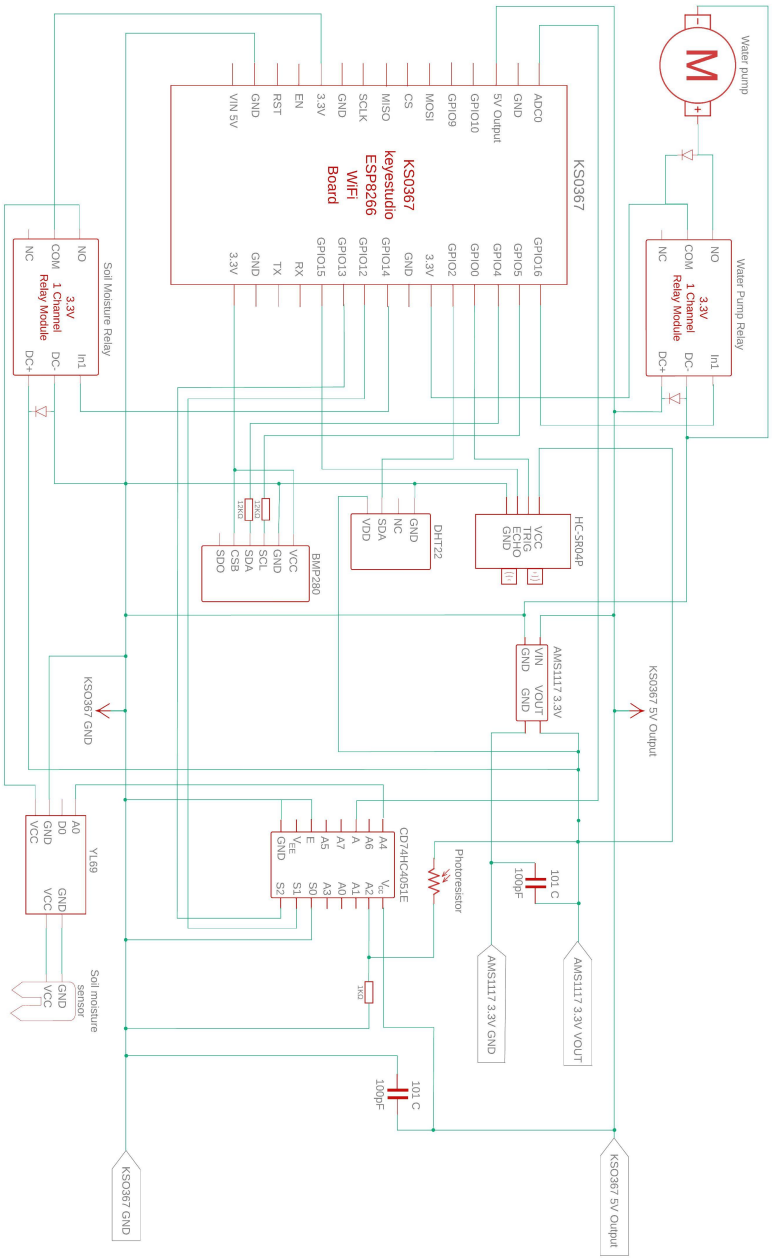
[35] UCUZ, D., ET AL. Comparison of the iot platform vendors, microsoft azure, amazon web services, and google cloud, from users perspectives. Julkaisusarjassa *2020 8th international symposium on digital forensics and security (ISDFS)* (2020), IEEE, 1–4.

[36] VERHOEVEN, R., KEMPINSKI, S., JA MERATNIA, N. Performance evaluation of wi-fi halow, nb-iot and lora for smart city applications. Julkaisusarjassa *Proceedings of the 19th ACM International Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, & Ubiquitous Networks* (2022), 17–24.

[37] WAWORUNDENG, J. M., SUSENO, N. C., JA MANAHA, R. R. Y. Automatic watering system for plants with iot monitoring and notification. *CogITo Smart Journal 4(2)* (2019), 316–326.

[38] XIA, F., YANG, L. T., WANG, L., VINEL, A., ET AL. Internet of things. *International journal of communication systems 25*, 9 (2012), 1101.

A Mittauslaitteen kytkentäkaavio



B Lähdekoodin etätietovarastot

Monitorointisovelluksen lähdekoodi

URL: `https://github.com/nilspert/verdant-sync`

Mittauslaitteen lähdekoodi

URL: `https://github.com/nilspert/verdant-sync-iot`