Katja Hellsten
Antti Leppänen
Bella Lerch
Okko Ojala
Susan Paloranta

IoT-pot -testipäiväkirja ja raportti

Tietotekniikan
harjoitustehtävä
30. toukokuuta 2025

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Kokkolan yliopistokeskus Chydenius

Tekijät: Katja Hellsten, Antti Leppänen, Bella Lerch, Okko Ojala ja Susan Paloranta
Yhteystiedot: kahellst@jyu.fi, antulepp@jyu.fi, belerch@jyu.fi, oaojala@jyu.fi ja sm-palora@jyu.fi
Puhelinnumero:
Ohjaaja: Tuomo Härmänmaa ja Veli-Matti Tornikoski
Työn nimi: IoT-pot -testipäiväkirja ja raportti
Title in English: IoT-pot -test diary and test report
Työ: Tietotekniikan harjoitustehtävä
Sivumäärä: 26
Tiivistelmä: Tämä on IoT-pot testauspäiväkirja ja testausraportti
Avainsanat: yksikkötestaus, integraatiotestaus, järjestelmätestaus, hyväksymistestaus, IoT-pot
Abstract: This is IoT-pot test diary and test report
Keywords: unit testing, integration testing, system testing, acceptance testing, IoT-pot

Sisällys

1 Johdanto

Tämä testaus raportti on tehty Kokkolan yliopistokeskus Chydeniuksen kurssille TIES4571 IoT-projekti. Projektin aiheeksi valitsimme älyruukun toteuttamista huonekasville ja kurssin aikana nimesimme tuotteen nimellä IoT-pot. Projektin asiakkaana on yliopistokeskuksen erikoissuunnittelija Pentti Impiö, joka on mukana LUMA-toiminnassa. Älyruukkua saatetaan käyttää tulevaisuudessa LUMA-kursseilla.

Raportissa ensin esitellään testauksen merkitystä ja erilaisia testausmenetelmiä yleisellä tasolla, sitten käydään läpi tässä projektissa käytettyjä testejä. Lopuksi pohditaan, miten tuloksia hyödynnettiin projektissa.

2 Testauksen merkitys, erilaisia menetelmiä

Jussi-Pekka Kasurisen kirja ohjelmistotestauksen käsikirja määrittelee, että “ohjelmistotestaus on osa suurempaa kokonaisuutta nimeltään ohjelmistotuotanto. Ohjelmistotuotannon tarkoituksena on löytää ja tunnistaa ohjelmistojen valmistamiseen soveltuvia perusperiaatteita, jotka tiedetään toimiviksi ja joiden avulla pystytään esimerkiksi aikaansaamaan parempaa laatua, tai karsimaan ylimääräisiä kustannuksia“ [1].

Testauksesta löytyy monenlaisia määritelmiä. Wikipedia määrittelee ohjelmistotestauksen olevan tapa tutkia ohjelman virheettömyyttä ja muita laatuominaisuuksia sitä yksinkertaisesti sellaisenaan käyttämällä tai erityisiä testaamistekniikoilla käyttäen. Ohjelmistotestauksen päätavoitteena on havaita ohjelmistossa ilmenevät häiriöt, jotta viat voidaan paljastaa ja korjata [3]. Testauksen voi suorittaa manuaalisesti ja automaatiolla [2].

Jussi-Pekka Kasurisen kirja ohjelmistotestausten käsikirja esittelee V-mallin, jossa kuvataan projektin etenemistä. Rakentamisvaiheessa otetaan selvää vaatimuksista, määritellään, suunnitellaan ja ohjelmoidaan. Itse testausvaihe koostuu yksikkötestauksesta, integraatiotestauksesta, järjestelmätestauksesta ja hyväksymistestauksesta [1].

Yksikkötestaus tarkoittaa testausta, jossa testataan yksittäinen moduuli, funktio tai olio. Yleensä tämä testaus tehdään välittömästi toteutuksen yhteydessä. Testaajana toimii ohjelmoija tai kehittäjä [1]. Yksikkötestauksen hyötynä on, että virheet löydetään jo varhaisessa vaiheessa [2].

Integraatiotestaus tehdään tavanomaisesti yksikkötestauksen jälkeen. Ohjelmistotestauksen käsikirja määrittelee, että “integraatiotestauksen tavoite on, että järjestelmän osat toimivat yhdessä. Integraatiotestauksessa todistetusti toimivaan kokonaisuuteen lisätään aina yksi osa lisää ja tarkastetaan toimiiko kokonaisuus edelleen“ [1].

Testauksen kolmannessa vaiheessa tehdään järjestelmätestaus. Testaus tehdään tällöin kokonaiselle järjestelmälle. Testaus suoritetaan testiympäristössä tai muuten rajoitetuissa olosuhteissa. Testauksen tavoitteena on, että järjestelmä toteuttaa kaikki sille asetut tavoitteet [1].

Viimeisessä vaiheessa eli hyväksymistestauksessa testaus tehdään kohdeympäristössä. Hyväksymistestauksen tavoitteena on osoittaa ohjelman olevan riittävän korkealaatuinen. Ohjelma myös täyttää ne vaatimukset mitkä sille on vaatimusmäärittelyssä todettu. Järjestelmä tarkistetaan virallisesti [1].

Erilaisia muita testausmenetelmiä ovat esimerkiksi käytettävyystestaus, kuormitustestaus, suorituskykytestaus, savutestaus, alfa-ja beta testaus, tutkiva testaaminen, ad hoc -testaus ja mallipohjainen testaus. Näitä voidaan käyttää osana yksikkö-, integraatio-, järjestelmä-, ja hyväksymistestausta [1].

Kotipuutarha-projektissa testauksen tarkoituksena oli testata projektin tuote ja varmistaa sen toimivuus. Lisäksi testauksen tarkoituksena oli myös löytää mahdolliset puutteet ja epäkohdat.

Aloitimme testauksen suoraan integraatiotestauksella, koska aika ja osaaminen ei olisi riittänyt yksikkötestaukseen. Testausta tehtiin jatkuvasti ja testauksessa lisäsimme aina yhden osan lisää testaten toimiiko kokonaisuus edelleen. Projektissa testiympäristöjä oli kaksi: Okolla asiakkaan ruukku ja Susanilla oma ruukku. Konfiguraatio (Device-id ja connection)- ja monitorointisovelluksen (kirjautuminen, ruukun lisääminen, päiväkirjamerkinnät ja kuvien lisääminen) testaukseen osallistuivat kaikki ryhmäläiset.

Integrointitestauksessa edettiin alhaalta ylöspäin (bottom up testing). Alhaalta ylöspäin -testauksessa ensimmäisenä järjestelmään tuodaan kaikista matalimman tason moduulit, jotka kommunikoivat suoraan raudan tai käyttöjärjestelmän kanssa (esimerkiksi verkkoyhteydet ja tietokanta). Tämän jälkeen järjestelmään ryhdytään tuomaan uusia moduuleja, jotka käyttävät aina edellisiä aiemmin integrointitestattuja osia, kunnes kaikki osat on tuotu järjestelmään [1].

Järjestelmätestauksessa testasimme asettaako järjestelmä kaikki sille asetetut tavoitteet. Tavoitteet ovat lähtöisin projektin vaatimusmäärittelyn luvusta 4.1.Toiminnalliset vaatimukset. Hyväksymistestausta emme tehneet, koska testausraporttia tehdessä asiakkaan IoT-pot ei ollut vielä asiakkaalla käytettävissä.

2.1 Testauspäiväkirja

Testausta tehtiin projektin aikana jatkuvasti. Varsinaisen testaussuunnitelman sijaan tähän on koottu testauspäiväkirja eli yhteenveto siitä mitä ja miten testattiin. Testaus suoritettiin aina kun yksi osa valmistui.

Testausta tehtiin seuraavasti:

1. Konfiguraatiosovellus: kirjautuminen ja laitteen lisääminen.
2. Sensoreiden testaus
3. Monitorointisovellus: tunnusten luonti ja ruukun/kasvin luonti
4. Sensoreiden ja monitorointisovelluksen välinen yhteys
5. Monitorointisovellus: history-välilehden testaus
6. Monitorointisovelluksen testaus: muistiinpanojen lisäys
7. Sensoreiden arvojen miettiminen sekä asiakkaan näkökulma
8. Monitorointisovellus: muistiinpanoihin kuva, eri kuvamuodot
9. Pumpun testaus
10. Konfiguraatiosovelluksen notification settings -välilehden testaus
11. Konfiguraatiosovellus, watering-välilehti: set watering amount sekä set soil moisture treshold
12. Ledien lisääminen ja testaus
13. Uusi ominaisuus: watering log
14. Ruukkujen kasaus
15. Järjestelmätestaus

1.Testaus: konfiguraatiosovelluksen connection settings -välilehti. Ensimmäisessä testissä, kun laitteet olivat saapuneet, testattiin saavatko kaikki yhdistettyä oman laitteensa (Arduino nano esp32) konfiguraatiosovelluksen kautta wifi-verkkoon. Tietokantaan lisättiin olemassa olevat Device id:t.

2.Testaus: Sensoreiden testaus. Olimme selvittäneet mihin pinneihin sensorit pitää kytkeä, sekä miten saamme sensorit toimimaan. Jokaiselle sensorille oli luotu oma

tiedosto esimerkiksi water level sensor.h. Sensoritestissä testattiin, siirtyvätkö sensorien arvot main.cpp -tiedostoon ja tulostuvatko arvot Visual Studion Serial monitor -ikkunaan.

3.Testaus: Monitorointisovellusta alettiin testaamaan. Kaikki ryhmäläiset kokeilivat, että pystyvät luomaan tunnukset ja kirjautuivat luoduilla tunnuksilla monitorointisovellukseen. Jokaisen ryhmäläisen on myös pystyttävä luomaan omalla Device id:llä oman ruukun/kasvin sekä poistamaan sen. Testauksessa testattiin myös sitä, ettei toisen ryhmäläisen tai keksityllä Device id:llä pysty lisäämään itselle ruukkua.

4.Testaus: Testissä testattiin sensoreiden ja monitorointisovelluksen välistä yhteyttä. Katsottiin siirtyvätkö arvot MQTT-protokollaa käyttäen monitorointisovellukseen real-time välilehdelle sekä HTTPS-protokollaa käyttäen palvelinskriptille ja sieltä tietokantaan.

5.Testaus: Testissä testattiin monitorointisovelluksen history-välilehden toiminta. Siirtyvätkö sensorien arvot history-välilehdelle ja syntyykö kuvaajaa.

6.Testaus: Monitorointisovelluksen muistiinpanojen lisäys. Testattiin merkinnän lisäys, virheilmoitukset esim. tyhjät kentät ja merkintöjen selaaminen history-välilehdellä.

7.Testaus: Ryhmässä mietimme ovatko sensoreista saatavat arvot sellaisia kuin pitää. Asiakastapaaminen: asiakkaan näkökulma (valoisuus ja LUX-arvo).

8.Testaus: Monitorointisovellus ja muistiinpanoihin kuvien lisääminen. Asiakkaalta oli tullut toive kuvan lisäämisestä muistiinpanoihin. Testattiin päiväkirjamerkintä ilman kuvaa. Päiväkirjamerkintä onnistuu kuvan kanssa ja erilaiset kuvamuodot gif, jpg, jne.

9.Testaus: Vesipumppu ja kytkennät. Vesipumpulle tarvittiin mosfet ja diodi. Mosfet:lla voidaan ohjata vesipumppua (päälle/pois) ja diodi taas suojaa vesipumppua jännitepiikeiltä. Testattiin, että saadaan vesipumppu toimimaan.

10.Testaus: Konfiguraatiosovelluksen watering-välilehden testausta. Nyt vesipump-

pua voitiin testata koodin kautta eli vesipumpun pitää antaa haluttu kastelumäärä sekä kastelua ei pidä syntyä, jos mullankosteus on isompi kuin olemassa oleva mitattu mullankosteus tai jos vesimäärä on liian vähäinen vesisäiliössä.

11.Testaus: Konfiguraatiosovelluksen notification settings -välilehden testaus. Käyttäjä voi valita sähköpostiin tulevat hälytykset (soil moisture below threshold, water tank empty, water overflow) sekä lisätä oman sähköpostiosoitteen, mihin nämä hälytykset tulevat. Testattiin, että saadaan järjestelmä tuottamaan nämä notifikaatiot tekemällä oikeita tilanteita: mullan kosteusanturilla ja kapasitatiivisilla vesisensoreilla.

12.Testaus: Ledien testaus. Ledeille luotiin led.h tiedosto sekä global.h tiedostoon lisättiin ledit globaaleiksi muuttujiksi. Testattiin, että ledit toimivat vaatimusmäärittelyn mukaisesti. Samalla mietittiin ikonit ruukun kanteen kuvaamaan ledien toiminnallisuuksia (koko ryhmän toimesta).

13.Testaus: Watering log. Pidemmän testijakson aikana tunnistimme tarpeen kastelukertojen tallentamiselle tietokantaan. Kun laite oli toiminut itsenäisesti viikon ajan, huomasimme, että käyttäjän on vaikea tietää, milloin laite on kastellut kasvin. Joten päätimme, että järjestelmään lisätään vielä loppuvaiheessa uusi ominaisuus eli kastelukertojen tallentaminen tietokantaan. Kastelusta tallentuu tietokantaan kasteluajankohta (aikaleima) sekä kastelun määrä, jonka käyttäjä on konfigurointisovelluksessa asettanut.

14.Testaus: Ruukun kasaaminen. Kaikkien komponenttien pitää sopia niille suunnitelluille paikoilleen.

15.Testaus: Järjestelmätestaus. Kun järjestelmä on kasassa, on hyvä vielä testata, että kaikki komponentit toimivat halutulla tavalla ja IoT-potin vaatimukset toteutuvat. Testasimme, että vaatimusmäärittelyn kohdan 4.1. toiminnalliset vaatimukset täyttyvät.

3 Projektin eri komponentit ja niiden testaus

Testattavia asioita olivat sensorit, konfiguraatio sovellus, monitorointi sovellus, ledit, vesipumppu ja ruukun rakenne. Näihin sovellettiin integraatio testausta. Viimeiseksi tehtiin järjestelmätestaus.

3.1 Sensorien testaus

Sensoreiden testaus suoritettiin neljässä vaiheessa. Ensimmäiset kolme vaihetta suoritti Okko ja Susan. Neljännessä vaiheessa kaikki ryhmäläiset olivat keskustelemassa asiakkaan kanssa.

Taulukko 3.1: Sensoreiden testaus

Testauksen nimi)	**Testaus**	**Testauksen tulos**
1.vaihe: testataan, että sensorit toimivat	Sensorit testattiin luomalla jokaisesta sensorista oma kooditiedosto. Käytetään sensorien tiedostoissa funktioita ja globaaleja muuttujia. Main.cpp tiedostoon haetaan näistä aliohjelmista sensorien arvot. Testattiin, että sensoreista saatiin arvoja main.cpp tiedostoon	Hyväksytty: sensorit toimivat ja arvoja saadaan siirrettyä main.cpp tiedostoon

2. vaihe: testataan, että arvot saadaan siirrettyä monitorointisovellukseen sekä tietokantaan MQTT- ja HTTPS-protokollia käyttäen		Hyväksytty: sensorit toimivat ja arvot lähtevät MQTT-protokollaa käyttäen monitorointisovellukseen sekä HTTPS-protokollaa käyttäen palvelinskriptille ja sieltä tietokantaan. Monitorointisovelluksessa näkyy, että arvot päivittyvät sekä tietokanta päivittyy.
3.vaihe: tutkitaan vielä antaako sensorit luotettavia arvoja		Hyväksytty: sensoreista välittyy luotettavan oloisia arvoja: Huomautus: LDR-sensorin kanssa ollut ongelmia. Tehdään uusi koodi, joka laskee keskiarvon valoisuusarvosta.
4.vaihe: keskustellaan asiakkaan kanssa		Hyväksytty: asiakas on tyytyväinen monitorointisovelluksessa näkyviin arvoihin. Huomautus: valoisuusanturista keskusteltu asiakkaan kanssa.

Testausvaiheet sujuivat yleisesti ottaen hyvin. Ensimmäisessä testausvaiheessa huomattiin, että LDR-sensorin kanssa oli ongelmia, joiden ratkaisemiseksi kirjoitettiin uusi koodi, joka suorien mittausarvojen julkaisemisen sijaan laskee keskiarvon valoisuusarvosta. Asiakaspalaverissa ryhmä keskusteli tästä ratkaisusta asiakkaan kanssa.

Huomautus tietoturvasta: sensoreiden arvojen lähettämisessä käytetään MQTT- protokollaa. Käytössä on MQTT-broker, joka käyttää porttia 8883 ja se on tarkoitettu MQTT-yhteyksille, jotka käyttävät SSL/TLS-salausta turvallisuuden parantamiseksi. Projektissa kaikki MQTT-brokeriin liittyvät tiedot sijoitettiin omaan include/secrets.h -otsikkotiedostoon, joka on lisätty .gitignore -tiedostoon, jotta salaiset tiedot eivät mene versionhallintaan. Jokainen kehittäjä siis luo omalle koneelleen oman henkilökohtaisen secrets.h -otsikkotiedoston ja lisää sinne salaisuudet, API-avaimet, tietokannan käyttäjätunnukset ja muut arkaluonteiset tiedot.

3.2 Konfiguraatiosovelluksen testaus

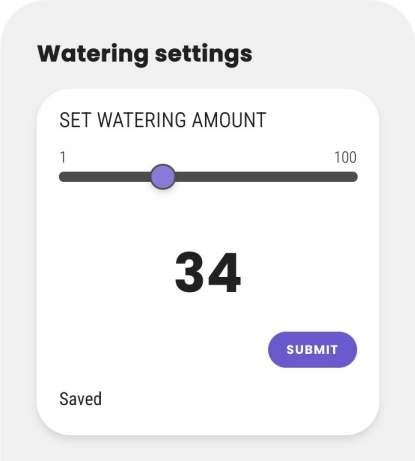
Konfiguraatiosovellus toteutettiin web-sovelluksena ja se koostuu kolmesta näkymästä: connection, watering ja notifications.

Taulukko 3.2: Konfiguraatiosovelluksen testaus

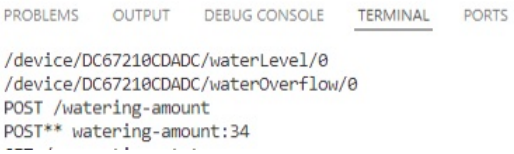
Testauksen nimi)	**Testaus**	**Testauksen tulos**
DEVICE-ID	Sovelluksen tulee näyttää Arduino-laitteen uniikki ID, sovellus hakee tämän Arduino palvelimelta HTTP-GET kutsulla. Seurattiin tietokantaa ja hyväksytään tallennus tietokantaan vain tunnetuilta Device id numeroilta	Hyväksytty: sovellus näyttää Arduino-laitteen uniikin ID:n. Tietokanta: tietokantaan on tallennettu ryhmäläisten käytössä olevat DEVICE ID:t. Tietokantaan ei ilmestynyt tietoa muilta Device id -numeroilta.

Connection	Jokainen ryhmäläinen testasi, että saa lisättyä oman reitittimen SSID:n ja salasanan	Hyväksytty: Connection settings -välilehdellä syttyi vihreä valo ja teksti connected merkiksisiitä, että yhteys on luotu. Jos määrittää väärän SSID:n ja salasanan, yhteys ei onnistu.
Watering	Set Watering amount ja Set Soil Moisture Treshold	Hyväksytty: Voidaan asettaa kastelun määrä sekä maaperän kosteuskynnys. Vesipumppu reagoi (set watering amount) muutokseen. Testauksen apuna: Visual Studioon tulee tieto reaaliaikaisesti, jos set watering amount tai soil moisture sensor arvoja muutettu.
Notifications	Notification setting -välilehdellä voidaan valita kolme erilaista sähköpostiin tulevaa ilmoitusta: soil moisture below threshold, water tank empty ja water overflow	Hyväksytty: Notifications setting -ikkunalla määritetään mitä ilmoituksia halutaan sekä mihin sähköpostiin. Ilmoitukset tulevat sähköpostiin. Huomautus: sähköpostin roskaposti asetukset MQTT-broker ylikuormittui. Okko lisäsi koodin, joka korjasi tilanteen.

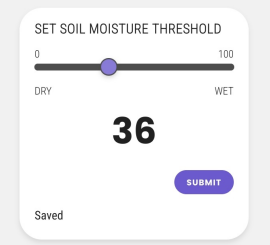
Web-sovellusta testattiin manuaalisesti. Loimme Trelloon testaustapauksia, jotka sisälsivät tiedot testattavasta ominaisuudesta ja ohjeet kuinka suorittaa testaus. Jokainen ryhmäläinen testasi ominaisuutta omalla tietokoneellaan tai mobiililaitteella. Huomiot, bugit ja parannusehdotukset kirjattiin Trello-korteille. Tässä testausvaiheessa huomattiin, että MQTT-broker ylikuormittui, sekä sähköpostiin pitää lisätä roskapostiasetuksiin turvalliset lähettäjät. Näistä enemmän luvussa 4.1. Sensoreiden testausta. Testauksen aikana korjasimme havaitut ongelmat.



Kuva 3.2.1: Watering settings välilehti



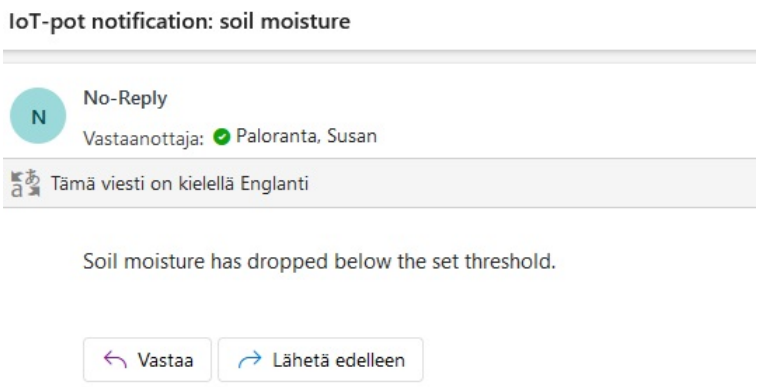
Kuva 3.2.2: Visual Studioon siirtynyt tieto



Kuva 3.2.3: Watering settings -välilehti

```
GET /connection-status
POST /watering-threshold
POST** watering-threshold current:28
POST** watering-threshold new:36
GET /connection-status
```

Kuva 3.2.4: Visual Studioon siirtynyt tieto



Kuva 3.2.5: Sähköpostissa notification viesti

3.3 Monitorointisovelluksen testaus

Monitorointisovelluksen testaus tehtiin kolmessa vaiheessa: testaus 1, testaus 2 ja testaus 3. Testauksen tekivät ryhmäläiset. Tämäkin sovellus toteutettiin web-sovelluksena ja sitä testattiin manuaalisesti klikkailemalla. Samoin kuin konfigurointisovelluksen tapauksessa, tässäkin loimme Trelloon testaustapauksia, jotka sisälsivät tiedot testattavasta ominaisuudesta ja ohjeet kuinka suorittaa testaus. Jokainen ryhmäläinen testasi ominaisuutta omalla tietokoneellaan tai mobiililaitteella. Huomiot, bugit ja parannusehdotukset kirjattiin Trello-korteille.

3.3.1 Monitorointisovelluksen testaus 1

Monitorointisovelluksen 1 testauksessa testattiin tilin luominen, kirjautuminen, laitteen lisääminen poistaminen sekä muiden laitteiden selaaminen.

Taulukko 3.3.1: Monitorointisovelluksen testaus 1

Testauksen nimi)	**Testaus**	**Testauksen tulos**
Tilin luominen	Uuden tilin luominen	Hyväksytty
Sisäänkirjautuminen ilman tilin vahvistamista		Hyväksytty: sisäänkirjautuminen ei onnistunut
Vahvista tili ja kirjaudu	Tuleeko sähköpostiin tilistä vahvistusviesti ja pystyykö vahvistuksen jälkeen kirjautumaan	Hyväksytty
Laitteen lisääminen	Saako laitteen lisättyä ja tuleeko ilmoituksia.	Hyväksytty: vain yksi laite per Device id. Sovellus ei hyväksy Device id:tä joka on jo käytössä toisella asiakkaalla (ilmoitus Device already in use). Sovellus ei anna lisätä kuvitteellisilla nimillä vaan antaa ilmoituksen: No Device found

Laitteen poistaminen	Poiston yhteydessä antaako sovellus varmistuksen	Hyväksytty: näyttää olemassa olevat poistettavat laitteet. Antaa laitteen poiston yhteydessä varmistuksen: Are you sure you want to remove
Muiden laitteiden selaaminen	Jos käyttäjällä on ruukkuja useampi, pystyykö vaihtamaan kasvitietoa ja katselemaan. Pystyykö näkemään muiden asiakkaiden ruukut.	Hyväksytty: sovellus ei näytä kuin yhden asiakkaan laitteet ja laitteet näkyvät vasemmalla allekkain. Muiden asiakkaiden käytössä olevalla Device id:llä ei voi lisätä laitetta

Kuten taulukosta näkee, monitorointisovelluksen ensimmäisen testauksen tuloksiin voitiin olla tyytyväisiä. Testauksessa ei havaittu korjattavia kohtia.

3.3.2 Monitorointisovelluksen testaus 2: päiväkirjamerkinnän lisääminen.

Tämä testausvaihe keskittyi testaamaan päiväkirjamerkintöjen lisäämistä.

Taulukko 3.3.2: Monitorointisovelluksen testaus 2

Testauksen nimi)	**Testaus**	**Testauksen tulos**
Merkinnän lisääminen	Testataan onnistuuko päiväkirjamerkinnän tekeminen	Hyväksytty: sovellus pakottaa käyttäjän lisäämään otsikon ja päiväkirjamerkinnän. Ilmoitus kun merkintä lisätty tietokantaan.

Virheilmoitukset, esim. tyhjät kentät	Testataan antaako monitorointisovellus lisätä merkinnän, jos jompikumpi kentistä on tyhjänä. Tuleeko ilmoitusta	Hyväksytty: sovellus antaa ilmoituksen, jos jompikumpi kentistä puuttuu.
Merkintöjen selaaminen history-sivulla	Testataan merkintöjen selaaminen history-sivulla	Hyväksytty: merkintä tallentuu history-sivulle ja siellä voi selata merkintöjä joko päivä, kuukausi tai vuosi näkymässä. Huomautus: merkintä näkyy väärällä kuukaudella ja kellonajalla

Testauksen tuloksiin oltiin tyytyväisiä. Testauksen aikana huomattiin, että päiväkirjamerkintä tallentuu väärälle kuukaudelle ja kellonajalle. Tämä korjattiin.

3.3.3 Monitorointisovelluksen testaus 3: kuvan lisääminen

Tässä testausvaiheessa testattiin monitorointisovelluksen kuvan lisäämistä. Kuvan lisääminen oli asiakkaan toive.

Taulukko 3.3.3: Monitorointisovelluksen testaus 3

Testauksen nimi)	**Testaus**	**Testauksen tulos**
Päiväkirjamerkintä onnistuu ilman kuvaa	Testattu, että päiväkirjamerkinnän voi tehdä ilman kuvaa	Hyväksytty
Päiväkirjamerkintä onnistuu kuvan kanssa	Testattu, että päiväkirjamerkintä onnistuu kuvan kanssa	Hyväksytty. Huomautus: Ladattu kuva jää kummittelemaan kun klikkaa toista päivää. Korjattu

Testattu erilaisten kuvaformaattien kanssa png, gif jne.	Testattu, että erilaiset kuvamuodot tallentuvat oikein	Hyväksytty: kuvat tallentuvat oikein ja näkyvät sekä skaalautuvat selkeästi ja oikein.

Testauksen tuloksiin oltiin tyytyväisiä. Testauksen aikana huomattiin, että ladattu kuva jää kummittelemaan, kun kalenterissa klikataan toista päivää. Tämä korjattiin.

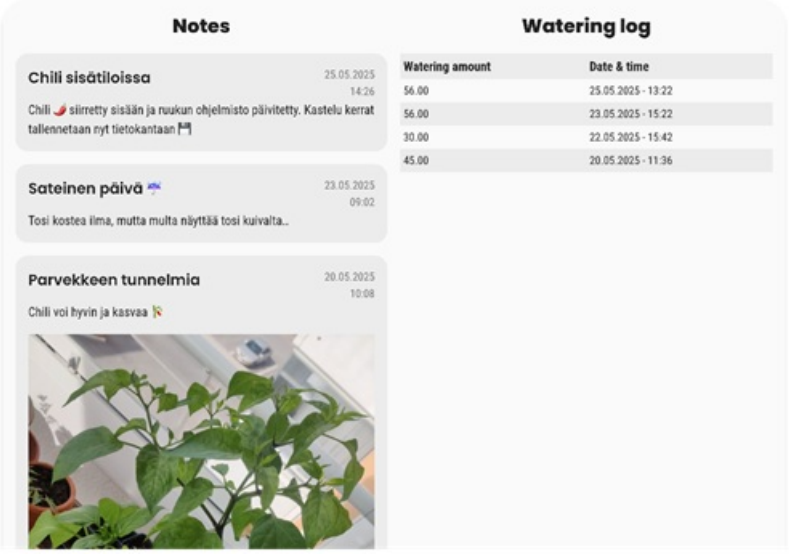
3.4 Vesipumpun toiminnan testaus

Vesipumppu on keskeisessä roolissa IoT-potin toiminnassa ja komponentin käyttö oli jokaiselle ryhmän jäsenelle uusi tuttavuus, joten luonnollisesti tehtiin testauksia.

Taulukko 3.4: Vesipumpun testaus

Testauksen nimi)	**Testaus**	**Testauksen tulos**
1.vaihe: vesipumppu toimintaan	Testataan, että vesipumppu on ehjä ja toimii	Hyväksytty: vesipumppu toimii
2.vaihe: koodi ja konfiguraatiosovellus	Testaus: vesipumppu reagoi koodiin ja kastelun määrää voidaan ohjata watering setting -kohdassa konfiguraatiosovelluksessa	Hyväksytty: koodi toimii halutulla tavalla.
3. vaihe: Uusi ominaisuus: kastelutiedon tallentaminen tietokantaan	Kun kastelu tapahtuu, tieto tallentuu tietokantaan ja monitorointi sovelluksen history välilehdelle watering log kohtaan	Hyväksytty: kastelun yhteydessä tietokantaan tallentuu kastelu tietona päivä ja aika. Tämä tieto näkyy myös monitorointi sovelluksessa

Vesipumppu osoittautui hyvin tehokkaaksi. Pumpun tehokkuuden säätöä varten luotiin ajastus. Uutena ominaisuutena lisättiin kastelutiedon tallentaminen tietokantaan. Kastelutieto on nähtävissä monitorointi sovelluksen history-välilehdellä watering log -kohdassa.



Kuva 3.4.1: Watering settings -välilehti

3.5 Ledin testaus

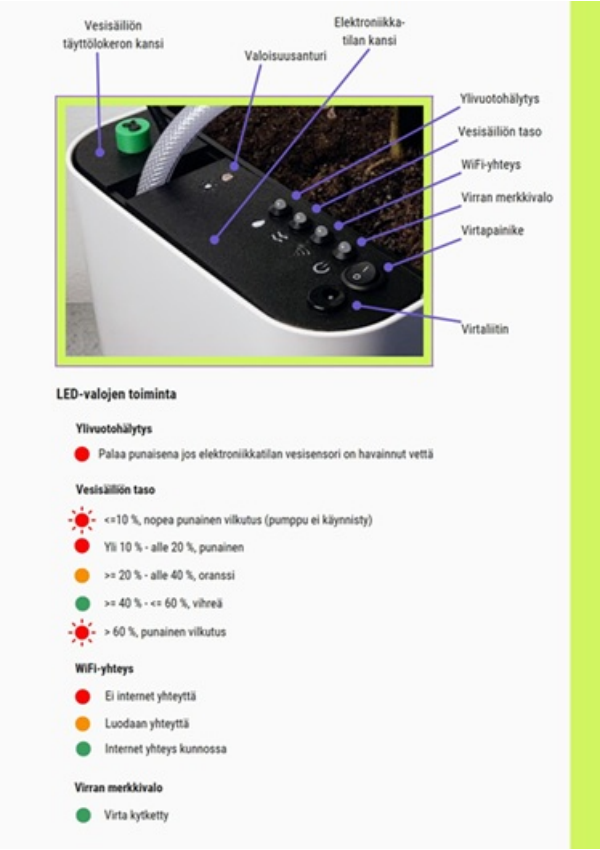
Ledejä on käytössä neljä kappaletta. Ledien käyttötarkoitus on viestittää asiakkaalle seuraavasti (k.kuva):

1. Ledi: virtaliitin
2. Ledi: WiFi-yhteys
3. Ledi: Vesisäiliön taso
4. Ledi: Ylivuotohälytys

Taulukko 3.5: Ledin testaus

Testauksen nimi)	**Testaus**	**Testauksen tulos**
1.Ledi: Virta	Ledi syttyy vihreäksi kun laitteessa on virta	Hyväksytty
2.Ledi: WiFi-yhteys	Ledi palaa punaisena kun yhteyttä ei ole. Keltaisena kun luodaan yhteys. Vihreänä kun internet yhteys on kunnossa	Hyväksytty
3. Ledi: Vesisäiliö	Led vilkuttaa punaisena kun pumppu ei käynnisty. Jos veden taso on yli 10%, mutta alle 20% ledin väri on punainen. Jos veden taso isompi tai yhtäsuuri kuin 20%, tai alle 40%, ledin väri on oranssi. Veden taso on isompi tai yhtäsuuri kuin 40%, tai pienempi tai yhtäsuuri kuin 60% ledin väri on vihreä. Jos vettä on mitattu yli 60%, ledin väri on punainen vilkutus	Hyväksytty
4.Ledi: Ylivuoto	Vilkkuu punaisena, jos sensori havaitsee ylivuodon	Hyväksytty

Ledien testauksessa ei ilmennyt hankaluuksia. Kaikki ledit toimivat halutulla tavalla.



Kuva 3.5.2: Ledien toiminta

3.6 Ruukun rakenteen testaus

Ruukun rakenteen testauksessa haluttiin lähinnä varmistua, että kaikki komponentit mahtuvat suunnitellusti ruukkuun

Taulukko 3.6: Ruukun rakenteen testaus

Testauksen nimi)	**Testaus**	**Testauksen tulos**
Vesipumppu mahtuu ruukkuun	Asennettu vesipumppu ruukun pohjalle	Hyväksytty
Arduino Nano esp32 ja sensorit mahtuvat niille varatuille paikoille	Arduino Nano esp32 ja sensorit kootaan ruukun sisällä olevaan lokeroon	Hyväksytty
Suunnitellut lokerot mahtuvat ulkoruukun sisälle	Asennetaan lokerot ulkoruukun sisälle	Hyväksytty

Testausvaiheessa ei ilmennyt ongelmia ja olimme tyytyväisiä Antin suunnittelemiin ruukun osiin.

3.7 Järjestelmätestaus

Järjestelmätestauksella haluttiin varmistua, että IoT-pot ruukku toteuttaa kaikki sille asetetut tavoitteet ja toimii kokonaisuutena. Testaus tehtiin ajatellen mitä vaatimusmäärittelyssä on kirjattu tuotteesta (4.1. Toiminnalliset vaatimukset).

Taulukko 3.7: Järjestelmätestaus

Testaus (Tuotteen pitää pystyä...)	**Testauksen tulos**
...mittaamaan mullan kosteus ja jos mullan kosteus ei ole riittävällä tasolla, käynnistämään vesipumpun ja antamaan kasville vettä	Hyväksytty: Lisätty ominaisuus vesipumppu ei käynnisty, jos vettä ei ole säiliössä.
...mittaamaan mullan pH-arvo	Hyväksytty: Mullan PH-, kosteus- ja lämpötila-anturi mittaa mullan PH- arvon ja arvo näkyy monitorointi sovelluksen real-time välilehdellä soil-ph kohdassa

...mittaamaan ruukun ympäristöolosuhteet: valon määrä, ilman lämpötila ja kosteus	Hyväksytty: DHT22 ilman lämpötila- ja kosteusanturi ja valoanturi mittaavat ruukun ympäristö olosuhteet, Arvot näkyvät monitorointisovelluksen real-time välilehdellä kohdassa luminosity, air temperature ja air humidity
...antamaan tietoa kasteluveden tasosta ja tarvittaessa antamaan hälytyksen, jos vesisäiliössä ei ole tarpeeksi vettä	Hyväksytty: kasteluveden tasosta saadaan tieto water level -sensorilla. Hälytys saadaan kahdella tavalla: ledin syttyminen ja sähköpostinotifikaatio
...mittaamaan ruukun pohjalla oleva veden taso. Sovellus antaa tietoa kastellaanko kasvia liikaa. Jos ruukun pohjalle jää vettä, annetaan hälytys	Hyväksytty: Kapasitatiivinen sensori mittaa veden pohjalla olevaa vettä ja sovellus antaa hälytyksen: (ledi ja sähköpostinotifikaatio), jos ruukun pohjalla on vettä
...toimia WiFi-yhteyspisteenä ja paikallisena web-palvelimena, joka tarjoaa konfigurointisovelluksen. Konfigurointisovelluksen avulla käyttäjä määrittelee laitteelle Internet-yhteyden tallentamalla yhteyden tarjoavan reitittimen nimen ja salasanan laitteen muistiin.	Hyväksytty: projekti tuotti toimivan konfiguraatio sovelluksen, joka täyttää nämä vaatimukset
Käyttäjä voi lisätä monitorointisovellukseen yhden tai useampia älyruukkuja. Kunkin älyruukun mittaustulokset tallennetaan tietokantaan ja käyttäjä voi seurata mittaustietoja etänä käyttäen monitorointisovellusta tietokoneella tai mobiililaitteella. Laite lähettää mittaustiedot tunnin välein HTTPS-protokollaa käyttäen palvelinpuolen sovellukselle, joka tallentaa mittaustiedot tietokantaan. Näin voidaan toteuttaa mittaustietojen historian selaus.	Hyväksytty: Monitorointisovelluksessa voidaan lisätä yksi tai useampi ruukku (Device-id:n perusteella). Mittaustulokset tallentuvat tietokantaan ja käyttäjä voi etänä seurata mittaustietoja monitorointisovelluksesta.

Asiakas voi tehdä päiväkirjamerkintöjä kasveista monitorointisovellukseen	Hyväksytty: päiväkirjamerkintöjen (teksti ja kuva) tekeminen onnistuu. Merkintöjä voi selailla history-välilehdellä
Laajennettavuus	Hyväksytty: ruukusta syntyi kaksi eri versiota (asiakkaan ja ryhmäläisten oma versio)

Testauksessa kaikki vaatimusmäärittelyn luvun 4.1. vaatimukset havaittiin täytetyiksi. Totesimme lisäksi, että ohjelmoinnissa on lisäominaisuus vesipumpulle eli vesipumppu ei käynnisty, jos vettä ei ole säiliössä.

4 Testausten tulosten hyödyntäminen Kotipuutarha-projektissa

Web-sovelluksia testattiin manuaalisesti klikkailemalla. Loimme Trelloon testaustapauksia, jotka sisältivät tiedot testattavasta ominaisuudesta ja ohjeet kuinka suorittaa testaus. Jokainen ryhmäläinen testasi ominaisuutta omalla tietokoneellaan tai mobiililaitteella. Huomiot, bugit ja parannusehdotukset kirjattiin Trello-korteille, ryhmä kävi kommentit palaverissa läpi ja päätti yhdessä mitkä korjaukset ovat kriittisiä ja mitkä jätettiin jatkokehityksen töiksi. Trello osoittautui käteväksi työkaluksi koordinoida testausta.

4.1 Sensoreiden testaus

Asiakkaan kanssa juteltiin miten hän haluaisi sensoreiden arvojen näkyvän ja onko hänen mielestään arvot monitorointisovelluksessa riittävällä tasolla. Havaitsimme, että valoanturin mittaustulokset vaihtelivat suuresti ja meillä oli aluksi vaikeuksia saada järkeviä mittaustuloksia. Keskustelimme asiasta asiakkaan kanssa, teimme tarvittavat korjaukset ja loppujen lopuksi asiakas oli tyytyväinen valoisuusanturin mittaamaan lux-arvoon ja totesi, että arvo on riittävällä tasolla.

Asiakas teki myös hyvän havainnon mullan pH-arvon esittämisestä, pH-arvo tulee esittää yhden desimaalin tarkkuudella ja arvoavaruus on 0-14. Tämä korjattiin monitorointisovellukseen ja varmistettiin, että tieto tallentuu tietokantaan oikeassa muodossa.

4.2 Järjestelmän lähettämät sähköpostit

Järjestelmä lähettää käyttäjälle useita erilaisia sähköpostiviestejä. Auth0-palvelu lähettää rekisteröityville käyttäjille mm. uuden tilin vahvistusviestin ja salasanan vaihtaminen tapahtuu sähköpostiviestin kautta. Konfiguraatiosovelluksessa käyttäjä voi tilata erilaisia notifikaatioita, jotka backend-sovelluksemme lähettää käyttäen Mailgun-palvelun rajapintoja. Sähköpostien lähettämistä varten luotiin alidomain mail.iot-

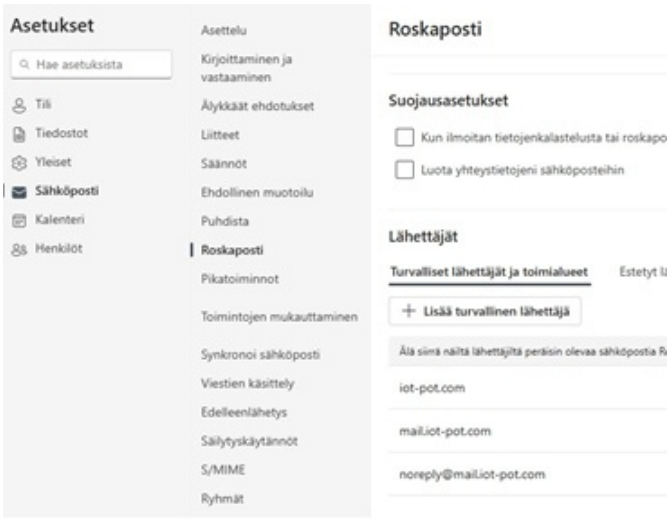
pot.com ja Auth0-palvelu konfiguroitiin käyttämään tätä alidomainia sähköpostien lähettämiseen.

Mailgun-palvelun käyttö yhdessä oman varmennetun verkkotunnuksen ja oikein määriteltyjen DNS-tietueiden kanssa parantaa merkittävästi sähköpostien toimitettavuutta. Tämä tarkoittaa, että sähköpostit päätyvät todennäköisemmin vastaanottajan saapuneet-kansioon sen sijaan, että ne merkittäisiin roskapostiksi tai joutuisivat roskakoriin.

Kun käytössä on oma verkkotunnus, Mailgun mahdollistaa sen vahvistamisen seuraavien DNS-tietueiden avulla: SPF, DKIM ja DMARC. Näiden tietueiden avulla vastaanottavat sähköpostipalvelimet voivat varmistaa, että sähköpostit todella tulevat sallitusta lähteestä, eikä kyseessä ole huijaus tai identiteettivarkaus.

Tämä läpinäkyvyys ja luotettavuus on ratkaisevaa sähköpostien perillemenon kannalta. Ilman näitä varmennuksia monet sähköpostipalvelut, kuten Gmail ja Outlook, saattavat automaattisesti suodattaa viestit roskapostiksi, vaikka sisältö olisi täysin asiallista. Lisäksi oman verkkotunnuksen käyttö antaa viesteille ammattimaisemman vaikutelman ja lisää vastaanottajan luottamusta. Domain iot-pot.com ostettiin Namecheap-palvelun kautta ja DNS-tietueet lisättiin Namecheap-palvelun DNS-hallintanäkymässä.

Erityisen hankalaksi sähköpostipalveluksi osoittautui JYU:n sähköpostipalvelu, opiskelijoiden @JYU osoitteisiin lähetetyt sähköpostit jäivät edelleen roskapostisuodattimeen, vaikka muut sähköpostipalvelut, kuten Gmail hyväksyivät lähetetyt viestit. Ongelman ratkaisu vaati paljon testausta, debuggausta ja Mailgun-palvelun lokien tutkimista. Loppujen lopuksi ongelma ratkaistiin siten, että ryhmäläiset kävivät lisäämässä lähettävän palvelimen osoitteen mail.iot-pot.com Outlook-palvelun asetukseen turvalliset lähettäjät. Myös lähettävä osoite noreply@mail.iot-poit.com lisättiin Outlook-palvelun turvallisiin lähettäjiin ja näin järjestelmän lähettämät viestit saapuivat perille myös JYU:n Outlook-sähköpostiin.



Kuva 4.2.1: Roskapostiasetukset

4.3 MQTT-broker -ongelma

MQTT-brokerina käytettiin HiveMQ-palvelun ilmaista versiota. Ilmainen versio tarjoaa max 100 aktiivista MQTT-yhteyttä laitteiden sekä sovellusten ja brokeriin välille. Huomasimme, että ajan myötä MQTT-broker ylikuormittui, koska laitteiden ja monitorointisovelluksen luomat MQTT-yhteydet palvelimelle eivät katkenneet automaattisesti vaan jäivät kummittelemaan ja tästä syystä 100 yhteyden raja meni täyteen. Tämä korjattiin tutustumalla tarkemmin käytettyihin MQTT-kirjastoihin, kirjastojen dokumentaatiosta löydettiin asetukset, jotka estävät yhteyden päälle jäämisen. Asetukset otettiin käyttöön laitteen ja monitorointisovelluksen koodissa ja broker alustettiin uudestaan HiveMQ-palvelussa, näin ongelma saatiin ratkaistua.

4.4 Monitorointisovellus

Korjattiin ettei kuva jää kummittelemaan väärälle päivälle.

4.4.1 Monitorointisovellus testaus 2

History-näkymässä päiväkirjamerkintä näkyi väärällä kuukaudella ja kellon ajalla. Korjattu.

5 Yhteenveto

Testaukset olivat erittäin tärkeä osa Kotipuutarha-projektia, sillä ne auttoivat projektin etenemistä jokaisessa vaiheessa. Testaamalla osa kerrallaan pystyimme etenemään projektissa ja huomaamaan mahdolliset virheet, viat ja häiriöt.

Testausten ansiosta olemme myös voineet varmistua tuotteen toiminnallisuudesta eli siitä, että tuotteen suunnitellut toiminnot ovat varmistettuja sekä vakaita.

Koemme, että käytetyt testausmenetelmät ja vaiheet ovat olleet sopivan tasoisia ja laajuisia tähän projektiin. Tulokset olivat pääosin heti tyydyttäviä, mikä tarkoittaa, että testauksia tehtiin pääsääntöisesti jo valmiiksi hyvin suunniteltujen ja toteutettujen työvaiheiden jälkeen. Tietenkin välillä löytyi korjattavia asioita, kuten asiaan kuuluukin. IoT-potista on saatu loppujen lopuksi tuote, johon projektiryhmämme on erittäin tyytyväinen.

Lähteet

[1] KASURINEN, J. P. *Ohjelmistotestauksen käsikirja*. DOCENDO, Jyväskylä, 2013.

[2] VALA. Yksikkötestaus: Mitä se on ja miksi se on tärkeää? URL https://www.valagroup.com/fi/blogi/yksikkotestaus/sikkötestaus:Mitäseonjamiksiseontärkeää?-VALA, viitattu 26.5.2025.

[3] WIKIPEDIA. Ohjelmiston testaaminen. URL https://fi.wikipedia.org/wiki/Ohjelmiston_testaaminen, viitattu 26.5.2025.