Arduino-kosteusmittarin hyödyntäminen lukion kemian opetuksessa

Eero Ojala

Kemian opettajakoulutusyksikkö, Kemian osasto,
Matemaattis-luonnontieteellinen tiedekunta, Helsingin yliopisto

Abstrakti: Tässä artikkelissa esitellään Helsingin yliopiston Tutkiva ja eheyttävä kemian opetus -kurssilla, kehitetty Arduino-pohjainen kosteusmittari, jonka käyttöä voidaan hyödyntää projektioppimisen alustana. Suunnittelun lähtökohtana oli toteuttaa etäluettava mittaussensori, jonka avulla oppilaat voivat tehdä mittauksia formaalin opetustilan ulkopuolella. Toteutustavan avulla voidaan parantaa oppilaiden tutkimuksellista osaamista ja hyödyntää opetuksessa erilaisia tieto- ja viestintäteknologisia ratkaisuja. Artikkelissa esiteltävä etäluettava kosteussensori on mahdollista käyttää osana projektioppimista. Projekti voi vaatia syvempää ymmärrystä ohjelmoinnista ja verkkosivustojen rakentamisesta, jolloin opettajan antaman ohjaus muodostuu merkityksellisemmäksi, jotta projekti pystytään toteuttamaan asiallisesti osana laajempaa opintokokonaisuutta. Tietoteknisen osaamisen tarvetta voidaan muokata oppilaiden taitojen pohjalta, vähentämällä ohjelmoinnin osuutta projektissa ja korvaamalla se jollain toisella Arduino-ympäristöön sopivalla komponentilla. Arduino kehitysalustalla voidaan parantaa oppilaiden ymmärrystä erilaisista kemian teknologisista mittausinstrumenteista.

Avainsanat: Arduino, kemian opetus, projektioppiminen

Ota yhteyttä: eero.m.ojala@helsinki

1 Johdanto

Suomalaisessa lukion kemian opetuksessa tieto- ja viestintäteknologian käytön harjoittelu on yksi laaja-alaisen osaamisen alueista, joka on osa monitieteistä osaamista tukevaa kemian opetusta (Opetushallitus, 2019). Nykyteknologia pystyy tarjoamaan opetukseen laajan valikoiman erilaisia ja erihintaisia vaihtoehtoja, joita voidaan hyödyntää erilaisissa opetusympäristöissä (Papadimitropoulos et al., 2021).

Lukion opetussuunnitelman (2019) tavoitteissa opiskelijan tulisi osata toteuttaa kokeellisia tutkimuksia käyttäen kemialle tunnusomaisia tutkimusmenetelmiä ja osata arvioida kemian ja siihen liittyvien teknologioiden tarjoamia ratkaisuja sekä niiden merkitystä yksilön, ympäristön ja yhteiskunnan kannalta. Erilaisten teknisten mittareiden ja sensoreiden avulla opiskelija pääsee tutustumaan teknologioiden käyttöön ja niiden tarjoamiin ratkaisuihin, jos ne pystytään toteuttamaan opiskelijan tietoteknisen ymmärryksen vaatimissa rajoissa.

Lukion kemian opetuksen tulisi kehittää opiskelijan monilukutaitoa ja opetuksen tulisi tukea nykyaikaista ja monitieteistä osaamista hyödyntämällä tieto- ja viestintäteknologiaa (Opetushallitus, 2019). Opiskelijan tutkimisen taitoja tulisi kehittää erilaisten menetelmien harjoittelun kautta. Kemian opetuksessa on vain tietty määrä opetustunteja ja erilaisten taitojen kehittäminen opetuksen aikana saattaa olla opettajalle haastavaa ja vaatia hyviä suunnittelutaitoja.

Nykyisin erilaiset mikrokontrollerit ovat yleistyneet opetuskäytössä, koska ne eivät vaadi käyttäjältään suurta teknologista ymmärrystä. Varsinkin Arduinon käyttö on saavuttanut suurta suosiota erilaisten sensoreiden hyödyntämisessä kemian laboratoriotöiden opetuksessa (Papadimitropoulos et al., 2021). Arduino perustuu avoimeen lähdekoodin käyttöön, joka voidaan ohjelmoida eri tavoin mittaamaan siihen liitettäviä sensoreita ja mittauslaitteita.

Tässä artikkelissa esitellään Helsingin yliopiston Tutkiva ja eheyttävä kemian opetus -kurssilla, kehitetty kosteusmittari, jonka käyttöä voidaan hyödyntää projektioppimisen alustana. Tavoitteena oli käyttää mittausdatan esittämiseen langatonta verkkotekniikkaa, jolloin opiskelijat pystyvät tutkimaan mittaustuloksia etänä, poissa formaalista opetustilasta. Luvussa kaksi käydään lävitse teoreettinen viitekehys, luvussa kolme kehitysprosessin eri vaiheet ja laitteiston toimintaan liittyvä dokumentaatio prototyyppeineen ja testausvaiheineen. Neljännessä luvussa esitellään projektioppimisen ympärille suunniteltu kemian opetuksen konteksti. Viimeinen luku sisältää pohdintaa projektin haasteista ja mahdollisuuksista.

2 Teoreettinen viitekehys

Seuraavassa luvussa käsitellään Arduinon kehitysalustan toimintaa ja tarkastellaan kuinka erilaisia teknologisia mittausinstrumentteja, voidaan liittää osaksi opetusta. Tämän lisäksi teoreettisessa viitekehyksessä esitellään projektioppimisen keskeiset periaatteet, joita hyödynnetään artikkelissa kuvatussa projektissa.

Arduino pohjaisilla mikrokontrollereilla oppilaat voivat suorittaa erilaisia mittauksia ja tiedonkeruuta, liittämällä ja muokkaamalla käyttämäänsä mittalaitetta omien tarpeidensa mukaisesti. Tarvittaessa sensoreita voidaan käyttää etänä ”Internet of Things”-ajattelun mukaisesti (Ga et al., 2024). Suuri osa käytössä olevista ohjelmistoista käyttää avointa lähdekoodia, jonka takia käyttäjällä on mahdollisuus sovittaa aiemmin kehitettyjä ratkaisuja yksilöllisiin tarpeisiinsa (Ambrož et al., 2023). Sensoreista saatua dataa voidaan lähettää USB-kaapelin tai langattoman yhteyden välityksellä kännykälle, tietokoneelle tai tabletille. Data voidaan muokata

digitaaliseen muotoon käyttämällä tarvittavaan ohjelmistoa ja tuottaa se reaaliaikaisesti näytille (Papadimitropoulos et al., 2021). Mikrokontrollerien käyttö opetuksessa on hyvä tapa opettaa tietoteknisten välineiden toimintaa (Mabbott, 2014.)

Haatainen ja Aksela (2021) määrittelevät projektioppimisen olevan oppilaslähtöinen, opettajan ohjaama pedagoginen lähestymistapa, jossa opetus on suunniteltu selkeästi määritellyn projektin ympärille. Sen avulla voidaan yhdistää luonnontieteiden ja matematiikan, eli STEM (Science, Technology, Engineering and Mathematics) -oppiaineiden opetus ja innostaa oppilaita ratkaisemaan todellisen elämän ongelmia. Oppimisen tavoitteet tulisi pohjautua opetussuunnitelmaan ja ryhmissä tapahtuva yhteistyö tulisi olla läsnä projektin kaikissa vaiheissa (Haatainen & Aksela, 2021).

Projektioppimisen avulla oppiminen voidaan muodostaa projektin ympärille (Thomas, 2000) ja opettajille luodaan mahdollisuuksia tarjota opiskelijoille monipuolisia tapoja oppia ongelmanratkaisua ja ryhmätyöskentelyä. Opiskelijat voivat syventää omaa ymmärrystään tutkimalla todellisia ongelmia jotka vastaavat projektissa tavoiteltuun kysymykseen, tuottamalla konkreettisen lopputuloksen (Hall & Miro, 2016).

3 Metodologia

Kolmannessa luvussa esitellään projektin lähtökohdat, suunnitteluprosessi ja siihen liittyvä morfologinen matriisi, sekä laitteiston dokumentaatio, prototyypin rakennus, siihen liittyvät haasteet ja lopullisen laitteiston testaus ja toiminta.

3.1 Lähtökohdat ja suunnittelu

Projektin lähtökohtana oli toteuttaa etäluettava mittaussensori, jonka käyttö voitaisiin liittää ensisijaisesti kemian opetuksen kontekstiin ja sisällyttää siihen STEM-pohjaista opetusta. Valinta kohdistui mullan kosteussensoriin, jonka voisi yhdistää suhteellisen helposti Arduinon langattomaan Wi-Fi lähettimeen. Kokonaisuuden voi liittää myös muihin opetettaviin, kuin STEM-pohjaisiin, oppiaineisiin, jolloin projektin voi toteuttaa eri opettajien kanssa yhteistyönä. Projektissa sovellettiin insinöörimäistä lähestymistapaa, jossa hyödynnetään yleisesti tunnettuja, yksinkertaisia käsitteiden yhdistämisen ja arvioinnin menetelmiä, joiden avulla alkuperäinen idea kehitetään lopulliseksi tuotteeksi (Ambrož et al., 2023.)

Projektin suunnittelussa hyödynnettiin morfologista, eli toiminto- ja toiminnallisuusmatriisia, jota varten projektissa tavoiteltuja erilaisia mittaamiseen liittyviä ominaisuuksia koottiin taulukoihin. Taulukossa 1. on esitetty kehitysalustalta vaadittuja eri ominaisuuksia, mitä projektin toteuttamisessa tulisi huomioida. Jokaiselle komponentille on annettu tietty ominaisuus ja tämän jälkeen on katsottu ominaisuuden vähimmäisvaatimukset ja lisäksi on annettu vaihtoehtoinen tavoiteltavissa olevat vaatimukset, joita olisi haluttu.

Taulukko 1. Projektissa tarvittavia komponentteja ja niiden ominaisuuksia.

Ominaisuus	Vaadittu	Haluttu
Kosteusanturi	Pystyy mittaamaan mullan kosteutta, tunnistaa mittausrajat. Kestää pidempiaikaista jatkuvaa mittausta	Anturi, jossa kosteusmittaus, Wi-Fi ja lämpötilan mittaus.
Wi-Fi-lähetin	Lähettää mittaustuloksia verkkosivustolle.	Lähettää mittaustulokset verkkosivustolle ja lähetin sisältyy kosteusanturiin.
Virtalähde	Sähköjohto, virtalähde.	Paristo tai akku.
Astia + multa	Sopiva, jossa koetta voidaan toteuttaa.	Pieni, mielellään kukkaruukku.
Anturin tiedot.	Verkkosivusto.	Verkkosivu, LCD-näyttö ja ledi, jossa värin vaihtuminen/ eri värisiä ledejä.
Tietojen tallennus.	Paperille tai verkkoon.	Automaattisesti toimiva tiedon keruu, jota voidaan soveltaa käytäntöön. Verkkosivusto, joka toteuttaa tiedonkeruun.

Taulukkoon 2. on kerätty morfologisen matriisin toiminnot ja erilaisia toteutettavia vaihtoehtoja, joiden pohjalta on muodostettu erilaisia konsepteja. Toiminnot on aseteltu omille riveilleen ja niitä on tämän jälkeen tarkasteltu projektin alussa suunniteltujen toteutettavien vaihtoehtojen osalta.

Taulukko 2. Mittarin toiminnot ja tavoitellut toiminnallisuudet, morfologisen matriisin mukaisesti.

Etäluettava hygrometri + lämpötilan mittaus.					
		A	B	C	D
Toiminnot	1. Kosteuden mittaus	Kosteusanturi kytketty Arduino	Manuaalinen mittaus mittarilla.	Mittaus omien havaintojen avulla.	Ei mittausta
	2. Lämmönmittaus	Lämpöanturi, kytketty Arduino	Manuaalinen mittaus	Ei mittausta	
	3. Tietojen tarkastelu	Etänä verkosta, paikallisesti LCD + ledit	Etänä verkosta	Paikallisesti, LCD + ledit	Paikallisesti, ledit
	4. Tiedon tallentaminen	Verkkosivu, tietokanta.	Manuaalisesti	Ei tiedon tallentamista	
	5. Virtalähde	Seinäpistoke	Paristo	Tietokone	
	6. Tarkastelun kohde	Kukkaruukku, multa ja kasvi.	Biojäte.	Multa.	Avoin systeemi

Mahdolliset toteutettavat projektin konseptit on koottu taulukkoon 3., josta voidaan matriisin pohjalta lukea konseptin eri toiminnot ja samalle riville on koostettu myös selite konseptille.

Taulukko 3. Muodostetut konseptit ja niiden toiminnot.

Konsepti	Toiminnot	Selite
Konsepti 1	1A-2A-3A-4A-5B-6A	Suunniteltu etäluettava kosteus- ja lämpöanturi
Konsepti 2	1A-2C-3B-4A-5A-6C	Karsittu etäluettava kosteusanturi
Konsepti 3	1A-2A-3C-4B-5C-6B	Karsittu lähiluettava kosteus- ja lämpöanturi
Konsepti 4	1D-2A-3B-4A-5B-6D	Etäluettava lämpömittari

Toteuttamiskelpoiset konseptit arvioitiin vielä teknisen ja taloudellisen arvon perusteella. Tekniselle arvolle muodostetiin taulukko, jossa eri painotusten perusteella muodostettiin kriteerejä, joilla konsepteja voitiin arvioida. Kriteereiksi muodostui tietojen tarkastelu, eli miten ja missä muodossa sensorin lähettämää dataa voi tutkia. Onko tieto luettavissa etänä, vai ainoastaan paikallisesti, tämä koski myös tietojen tallennusta. Kolmanneksi kriteeriksi muodostui laitteiston kokoamisen vaikeus, paljonko kokoaminen vie aikaa ja miten se saadaan mahdollisesti toimimaan. Lisäksi virranhallinta sai omat kriteerinsä. Taulukosta 4. voi tarkastella eri konseptien teknisiä arvoja ja miten ne ovat sijoittuneet kriteeristön ja painotusten perusteella.

Taulukko 4. Konseptien tekniset arvot ja sijoitukset.

Tekninen arvo					
Kriteerit	Painotus	Ideaali	Karsittu1	Karsittu2	Karsittu3
Tietojen tarkastelu	5	5	4	2	3
Tietojen tallennus	3	3	3	1	3
Laitteiston kokoaminen	5	1	2	3	2
Virranhallinta	2	2	1	2	1
summa abs.	15	43	41	32	36
summa norm.	1	2,87	2,73	2,13	2,40
sijoitus		1	2	4	3

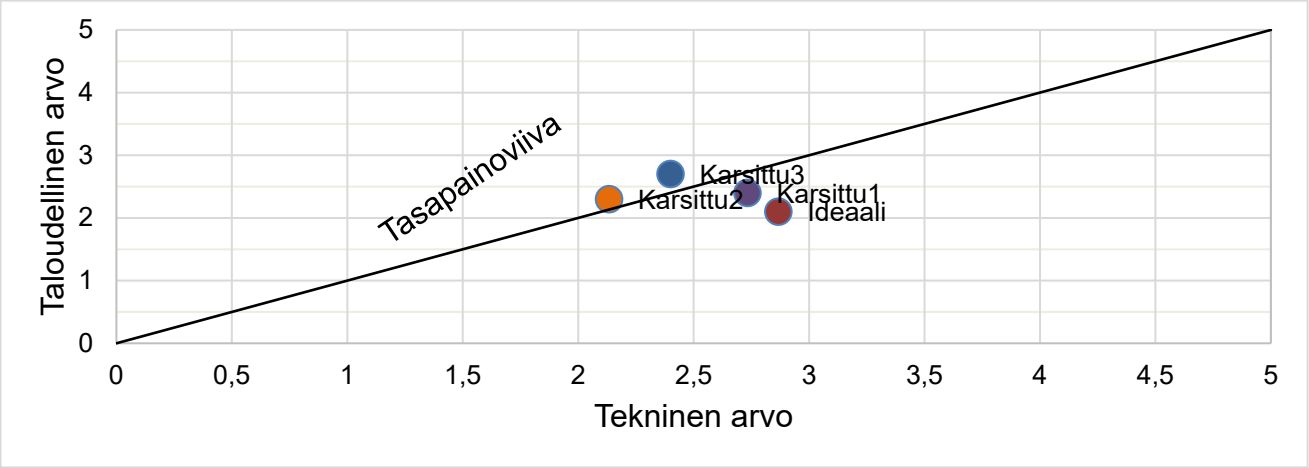
Taloudellinen arvo määräytyi tiedon käytettävyyden, laitteiston hinnan ja arvioitujen ylläpitokustannusten mukaan. Tiedon käytettävyydessä huomioitiin mahdollisten verkkosivustojen käytön maksullisuus ja tiedon tallentamiseen mahdollisesti vaikuttavat taloudelliset seikat. Laitteiston hinta määräytyi

käytettävien sensoreiden ja komponenttien pohjalta. Ylläpitokustannuksiin arvioitiin kuuluvan sähkön käyttö ja mahdollisesti paristojen ja akkujen hinnat, jos mittausta suoritettaisiin pitkäkestoisesti. Taulukossa 5. näkyy taloudelliset ja niiden erilaiset painotukset.

Taulukko 5. Konseptien taloudelliset arvot ja sijoitukset.

Taloudellinen arvo					
Kriteerit	Painotus	Ideaali	Karsittu1	Karsittu2	Karsittu3
Tiedon käytettävyys	4	4	3	2	1
Laitteiston hinta	3	1	2	3	4
Ylläpitokustannukset	1	2	1	1	2
summa abs.	8	21	19	18	18
summa norm.	1	2,63	2,38	2,25	2,25
sijoitus		1	2	3	3

Taloudellisen ja teknologisen arvon pohjalta muodostettiin kuvaaja, jonka pohjalta pystyttiin vertailemaan eri konseptien keskinäistä suhdetta, kuvaaja esitetty Kuvassa 1. Teknistä ja taloudellista arvoa vertaamalla olisi Karsittu 2-projekti ollut kyseisiä arvoja verraten kaikista tasapainoisin ratkaisu. Lähtökohtaisesti haluttiin toteuttaa Ideaalinen projekti, jossa on kaikki halutut ratkaisut muodostettuna ja tätä projektia lähdettiin toteuttamaan. Projektin edetessä, jouduttiin kuitenkin luopumaan ideaalista ratkaisusta ja päädyttiin toteuttamaan Karsittu 1-projekti. Kokonaisuuteen varattu aika oli rajallinen ja Ideaali-projektin havaittiin tarvitsevan enemmän aikaa, kuin oli mahdollista käyttää. Karsittu 2-projekti sisältää ainoastaan etäluettavan kosteusanturin, mutta siinä hyödynnetään verkkosivustoa ja etäluettavuutta.



Kuva 1. Tasapainokuvaaja.

3.2 Laitteiston dokumentaatio

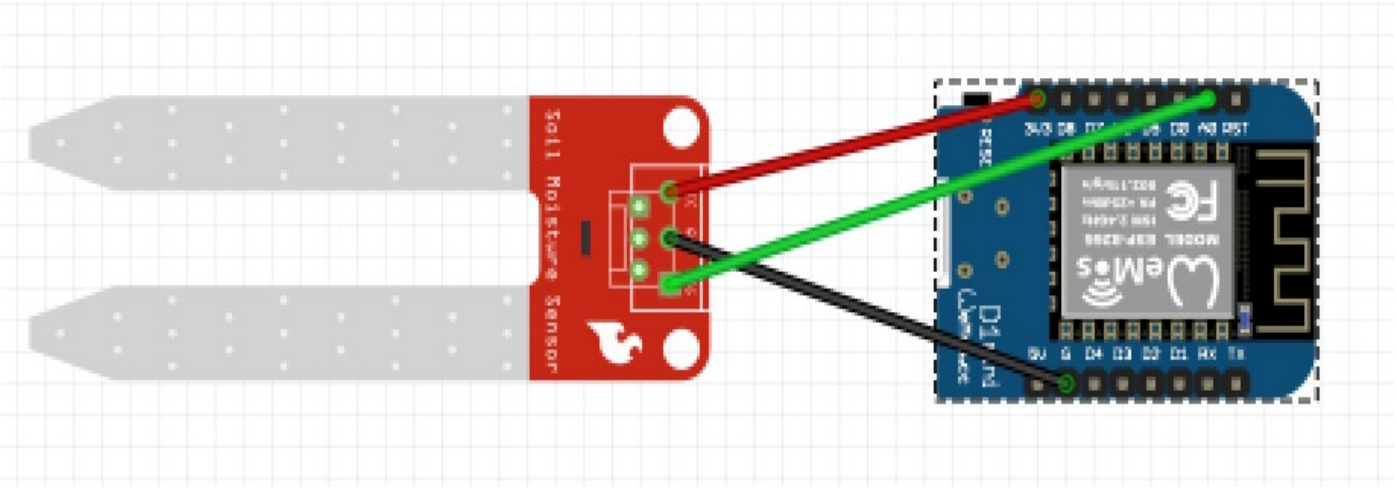
Etäluettava kosteusmittari koottiin Arduino-pohjaiselle järjestelmälle, johon kirjoitettiin ohjelma hallitsemaan tarvittavia teknisiä ratkaisuja. Langaton verkko rakennettiin Wemos D1 mini ESP8266 kehityskortille, johon kytkettiin maaperän kosteusanturi, joka mittaa mullan kosteutta. Arduinon kosteussensorissa on kaksi metallista elektrodia, jotka tulee työntää mitattavaan maa-ainekseen. Sensori mittaa sähkönjohtavuutta, resistanssin ja jännitteen mittaamisen avulla. Mittauksessa muodostuu analogisia arvoja 0–1023 välillä riippuen siitä, kuinka kuivaa tai kosteaa mitattava maa-aines on, mitä pienempi arvo, sitä kosteampaa ainesta. Mittausarvoon vaikuttaa käytetty maa-aines, aineksen pH ja siihen lisätyn nesteen koostumus. Arduinon resistanssin mittaamiseen pohjautuvat kosteussensorit ovat suhteellisen edullisia ja helppokäyttöisiä, mutta luotettavuus ja tarkkuus vaihtelee. Taulukkoon 6. on koottu artikkelin laitteistossa käytettyjä komponentteja ja niiden esimerkkihintoja.

Taulukko 6. Komponenttien esimerkkihintoja.

Komponentti	Komponentin tiedot	Hinta
Wemos D1 mini ESP8266		12,04 €
Maaperän kosteusanturi	LM393	10,00 €
Koekytkentäalusta 400	400 reikää	5,95 €
Hyppylankalajitelma 100	100 kpl	2,95 €

Wemos D1 mini ESP8266 toimii siis täysin itsenäisesti ilman Arduino UNO järjestelmää ja se voidaan ohjelmoida samaan tapaan liittämällä alusta USB-kaapelilla tietokoneeseen, jossa on tarvittava ohjelmisto ohjelman kirjoittamista varten. Ohjelma kirjoitettiin Arduino IDE-ohjelmalle, joka hyödyntää C++-pohjaista ohjelmointikieltä. Kosteusmittausta varten laadittu ohjelma Arduino IDE:lle löytyy liitteestä 1. Kuvassa 2. on esitetty Wemos D1 Minin ja analogisen kosteusanturin kytkennät. Kosteusanturi saa virran Wemos D1 Mini -kehityskortin 3V3-liitännästä (punainen hyppylanka), maadoitetaan GND-liitäntään (musta hyppylanka) ja lähettää analogisen jännitesignaalin A0-liitäntään (vihreä hyppylanka).

Kuva 2. Kytkennät Wemos D1 mini ja kosteusanturi.



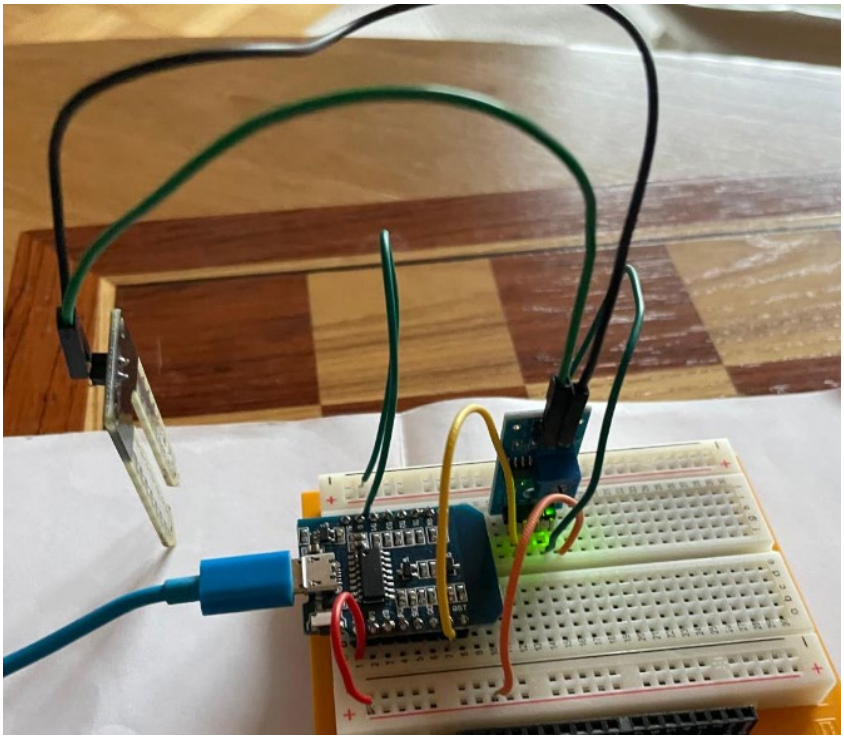
Etänä luettavaa dataa varten luotiin ilmainen verkkosivu, jolle täytyi kirjoittaa omat ohjelmansa, jotta sivustolla näkyisi Wemos D1 minin lähettämä data. Sivusto perustettiin ilmaista verkkosivustoa tarjoavalle palvelulle, joita löytyy internetistä eri käyttöön useita, samalla luotiin tietokanta, johon dataa voi tallentaa. Helppoa sivustolle pääsyä varten, luotiin vielä ilmainen lyhytverkko-osoite. Verkkosivuston käyttöön kirjoitetut ohjelmat löytyvät liitteistä 2.,3. ja 4. Verkkosivuston toiminta perustuu Wemos D1 minin lähettämän tiedon vastaanottamiseen ja sen näyttämiseen halutussa muodossa. Vastaanotettu tieto tallentuu liitteen 2. (data.php) koodin mukaisesti ja muuttaa kosteusarvon prosenteiksi ja määrittää vielä kosteuden tilan, tiedot tallentuvat MySQL-tietokantaan, joista liitteen 3. (get_last_readings.php) koodin mukaan haetaan viimeiset mittaukset ja liitteen 4. (index.php) koodi näyttää arvot verkkosivustolla. Verkkosivusto päivittää itsensä.

Varsinaisen laitteiston kokoamista varten Wemos D1 mini kytkettiin kiinni koekytkentälevyyn, jonka jälkeen siihen liitettiin halutut sensorit ja toiminnot. Kuvassa 3. näkyy koottu Wemos D1 mini ja kosteussensori koottuna kytkentälevylle. Vaadittujen kytkentöjen jälkeen Wemos D1 mini kytkettiin USB-kaapelilla kiinni tietokoneeseen, jolloin järjestelmä saa virtaa ja pystyy muodostamaan mittauksia ja lähettämään dataa.

Arduino-pohjaisia kehitysalustoja rakennettaessa tulee huomioida sähkölaitteiden rakentamiseen liittyvä turvallisuus. Ensi sijassa kytkennät tulee tehdä virrattomassa tilassa, muuten on olemassa sähköiskun vaara ja järjestelmän komponentit voivat vioittua, eivätkä enää toimi. Hyppylankoja ja muita järjestelmään liittyviä komponentteja tulee myös käsitellä varovaisesti, koska liitännöissä ja osissa ei ole erikseen rakennettu suojia, jolloin ne ovat alttiina rikkoutumiselle. Itse kosteussensori kestää multaan lisättyä vettä, mutta veden lisäämisessä tulee myös

noudattaa varovaisuutta. Järjestelmän suojaamista varten on mahdollista 3D-tulostaa sopiva kotelo, johon langattoman lähettimen voi asentaa.

Kuva 3. Kosteussensori ja Wemos D1 Mini-kehitysalusta kiinnitettynä koekytkentälevyyn.



3.3 Prototyyppi

Laitteiston rakentaminen esiteltiin jo luvussa 3.2, mutta tässä luvussa käydään tarkemmin lävitse erilaisia ongelmia, joita rakentamisen aikana tuli vastaan. Prototyyppiä lähdettiin rakentamaan ajatuksella, että kaikki mietityt toiminnot saataisiin lopulliseen versioon toteutettua. Ensimmäisessä vaiheessa ongelmaksi muodostuivat tarvittavien ohjelmistoajureiden asentaminen. Käytetyn tietokoneen mukaan tarvittavat ajurit oli asennettu, ja Wemos D1 Minin tulisi toimia moitteettomasti, mutta Arduino IDE herjasi virhettä COM-portissa, eikä pystynyt syöttämään koodia lähettimelle. Tätä varten etsittiin toimivat ajurit, joiden asentamisen jälkeen ohjelma saatiin lähetettyä Wemos D1 Miniin.

Tarvittava ohjelma kirjoitettiin tekoälyohjelmaa hyödyntäen, mutta toimivaa koodia jouduttiin muokkaamaan useita kertoja, ennen kuin kosteussensorin lähettämä data saatiin asiallisessa muodossa tulostettua. Prototyypin rakentamisen ja testauksen yhteydessä päätettiin luopua LCD-näytön käytöstä, koska sen antama lisäarvo tiedon keräämistä varten oli pienempi, kuin alun perin ajateltiin. Lisäksi koekytkennät ja kirjoitettu ohjelma, rupesi muodostamaan huomattavan rasitteen

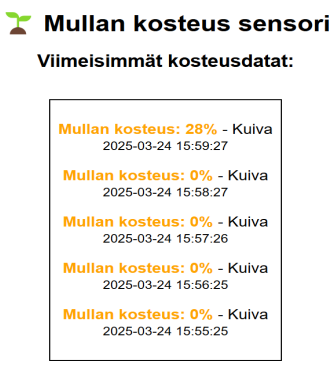
projektin ajalliselle etenemiselle. Pääpaino siirtyi projektin edetessä etänä luettavan datan lähettämiseen ja kosteusanturin toimintaan. Protoamisen aikana konsepti vaihtui siis numero kahteen, jossa projekti toteutettiin karsittuna versiona.

3.4 Laitteiston testaus

Laitteiston testaus suoritettiin useaan kertaan jo prototyypin kokoamisen aikana, koska vaaditun ohjelmiston kirjoittaminen ja testaaminen toteutettiin samanaikaisesti. Lopullista versiota testattiin tarkoituksena tuottaa, kosteussensorin lähettämää dataa, verkkosivulle. Kuvassa 4. on verkkosivustolla näkyvä data. Verkkosivuston esittämän datan päivitysväli voi muokata tarpeen mukaisesti, nyt kuvassa päivitysväli 60 sekuntia, mutta pidempikestoisessa projektissa päivitysaikaa voidaan pidentää. Myös muuta dataa voidaan esittää tarpeen mukaan. Verkkosivustoa ja siinä näytettävää dataa voidaan siis muokata eri tavoin, mutta projektiluontoisessa tapauksessa ajankäytön tehokkuuden merkitys kasvaa ja projekti tulee pyrkiä saamaan päätöksen siihen annetussa aikataulussa. Paikallisesti toimiva kosteusanturi, joka lähettää datan vain Arduinoon on suhteellisen helppo toteuttaa ja koota. Etäluettavan datan toimiminen halutunlaisesti vaatii aikaa ja useita toistoja testivaiheessa.

Testauksen edetessä alkuperäistä Wemos D1 miniin luotua ohjelmaa muokattiin vielä siten, että data näkyisi myös paikallisesti LCD-näytöllä ja mitattavan mullan ollessa kuivaa, syttyisi kytkentälevyllä punainen led-valo. Tämä osuus aiheutti aikaa vievää ongelmanratkontaa, joten LCD-näytön käytöstä päätettiin luopua. LCD-näytön toimivaa käyttöä varten siihen olisi tullut kytkeä painonappi, jolla kerättyä dataa olisi voinut tyydyttävästi tarkastella eri tavoin. Erilaisten toimintojen kirjoittaminen ohjelmaan pidentää koodia ja saattaa aiheuttaa ongelmia kokonaisuuden toimimisessa.

Kuva 4. Verkkosivuston näkymä testausvaiheessa.



4. Projektiesimerkki

Neljännessä luvussa esitellään projektioppimisen periaatteiden pohjalta muodostettu projektiesimerkki ja erilaisia kemian opetukseen liittyviä kontekseja.

Tässä artikkelissa esitetty etäluettava kosteusmittari voi toimia projektioppimiseen liittyvänä todellisena tuotoksena, johon liittyy ongelmalähtöisyys ja ryhmätyöskentely. Opettaja voi toteuttaa projektin yhteistyössä muiden opettajien kanssa, jolloin projektissa voidaan muodostaa todellista työelämää esittävä erilaisten asiantuntijoiden ryhmä. Jokainen projektiin osallistuva opiskelija pääsee toteuttamaan projektia omien vahvuuksiensa kautta. Projektissa voidaan yhdistää esimerkiksi kemian, matematiikan, biologian ja maantiedon oppiaineet ja liittää se ympäristön muutoksen ja kestävän kehityksen kontekstiin. Kosteusmittaus perustuu sähkönjohtavuuteen, jolloin maa-aineksen koostumus ja sen kosteus vaikuttaa tulokseen. Kemian osalta mittausta voitaisiin lähestyä veden kemian kautta, esimerkiksi suolapitoisuuden avulla, jolloin tutkittaisiin suolaveden ja puhtaan veden eroavaisuuksia maaperän kosteuden mittaamisessa ja havainnoitaisiin suolan vaikutusta maaperään.

Oppilaat voisivat tutkia kahden eri ruukun sisältävän mullan kosteutta ja tutkia miten puhdas ja suolainen vesi vaikuttavat mullan ominaisuuksiin. Samaan tutkimukseen voitaisiin liittää myös pH-mittaus ja tutkia kosteuden ja pH:n muutosta vaihtelemalla tutkimuksessa käytetyn veden pH:ta. PH:n mittaus voidaan toteuttaa pH-paperilla ja kosteuden mittaaminen toteutetaan Arduino-pohjaisella kosteussensorilla.

Tässä projektissa tieto- ja viestintävälineiden käytön oppiminen ja hyödyntäminen korostuvat. Kosteussensorin toimintaperiaate ja etäluettavan datan muodostaminen parantavat opiskelijan ymmärrystä erilaisten tietoteknisten välineiden toiminnasta. Mabbott (2014) toteaa artikkelissaan, että elektroniikan toimintaperiaatteiden tuntemus parantaa kemiallisten instrumenttien ymmärtämistä.

Etäluettavan kosteusmittarin käytön voi yhdistää myös biojätteen kosteuden tutkimiseen, jolloin konteksti on yhteydessä energiaan ja energian tuottamiseen. Suomessa käytettävien polttolaitosten jätteeseen päätyy kosteaa biojätettä, jolloin sen polttamiseen tarvitaan enemmän energiaa. Biojätteestä saadaan kierrätyksen kautta valmistettua, myös biokaasua ja kompostoitua seosta, jota käytetään mullan raaka-aineena. Kosteussensorin avulla voidaan tutkia biojätteen kosteuden muuttumista ja tarkastella kosteuden vaikutusta biojätteen massaan.

5. Pohdinta

Arduino-pohjaisen kehitysalustan käyttö kemian opetuksessa antaa mahdollisuuksia hyödyntää projektioppimista osana laajempaa opetuskontekstia. Projektiin voi liittää toisten STEM-aineiden opetusta ja tehdä yhteistyötä muiden aineiden opettajien kanssa. Varsinkin lukio-opetuksessa projektimainen työskentely auttaa oppilaita tekemään yhteistyötä ja jakamaan projektin eri osasia tasapuolisesti kaikkien projektiin osallistuvien henkilöiden kesken. Opettajan rooli projektin ohjaajana auttaa oppilaita ottamaan vastuuta omasta osuudestaan projektissa, jolloin työelämälähtöisyys opetuksessa korostuu.

Artikkelissa esitetty projekti vaatii opiskelijoilta perusymmärrystä ohjelmoinnista ja verkkosivustojen rakentamisesta. Tätä taustaa ei välttämättä kaikilta lukiolaisilta löydy, jolloin eri oppiaineiden opettajien yhteistyö projektin onnistumiselle korostuu. Lukion matematiikassa ohjelmointi kuuluu osaksi Maa11-moduulia ja se painottuu Python-ohjelmointikielen ympärille. Tätä projektia voisi tarjota edistyneemmille opiskelijoille, jotka ovat ohjelmoineet itsenäisesti ja tietotekniset taidot ovat hieman paremmat. Projektissa on tietysti monia eri osasia, jolloin kaikilta opiskelijoilta ei vaadita välttämättä ohjelmointikielen kirjoittamisen osaamista. Tekoälysovellusten avulla on mahdollista tuottaa tässä projektissa tarvittava koodi, joka osaltaan parantaa oppilaiden tvt-taitojen kehitystä.

Kosteussensorin yhteyteen on mahdollista lisätä muitakin sensoreita, kuten lämpötila, pH tai ilmankosteus, jolloin kemian osuutta projektissa voidaan laajentaa tarpeen mukaan. Projektioppimisen avulla oppilaat pääsevät toteuttamaan erityyppistä ongelmanratkontaa ja kokeilemaan tietotekniikan mahdollisuuksia. Yhteistyön merkitys projektin onnistumisessa kasvaa, mitä enemmän vaatimuksia lopullisen tuotoksen ympärille muodostuu. Projektioppimisessa oppimisen arviointi ei perustu ainoastaan lopulliseen tuotokseen, vaan arviointia suoritetaan eri tavoin koko projektin aikana ja lopullisen tuotoksen tulisi vastata projektin alussa esitettyyn kysymykseen tai ongelmaan.

Arduino-pohjaisten sensoreiden avulla kemian opetukseen voidaan suhteellisen helposti ja halvalla lisätä tieto- ja viestintäteknologioiden käyttöä. Tietotekniset vaatimukset voidaan muokata opiskelijoiden osaamistason mukaan ja projektimaisena oppimiskokonaisuutena tavoiteltu oppimistulos voidaan saavuttaa monin eri tavoin.

Lähteet

Ambrož, M., Pernaa, J., Haatainen, O., & Aksela, M. (2023). Promoting STEM Education of Future Chemistry Teachers with an Engineering Approach Involving Single-Board Computers. *Applied Sciences*, *13*(5), Article 5. https://doi.org/10.3390/app13053278

Ga, S.-H., Park, C., Cha, H.-J., & Kim, C.-J. (2024). Science Teachers' Technical Difficulties in Using Physical Computing and the Internet of Things Into School Science Inquiry. *IEEE Transactions on Learning Technologies*, *17*, 1809–1818. IEEE Transactions on Learning Technologies. https://doi.org/10.1109/TLT.2024.3406964

Haatainen, O., & Aksela, M. (2021). Project-based learning in integrated science education: Active teachers' perceptions and practices. *LUMAT: International Journal on Math, Science and Technology Education*, *9*(1), Article 1. https://doi.org/10.31129/LUMAT.9.1.1392

Hall, A., & Miro, D. (2016). A Study of Student Engagement in Project-Based Learning Across Multiple Approaches to STEM Education Programs. *School Science and Mathematics*, *116*(6), 310–319. https://doi.org/10.1111/ssm.12182

Mabbott, G. A. (2014). Teaching Electronics and Laboratory Automation Using Microcontroller Boards. *Journal of Chemical Education*, *91*(9), 1458–1463. https://doi.org/10.1021/ed4006216

Opetushallitus. (2019). *Lukion opetussuunnitelman perusteet 2019*. https://www.oph.fi/sites/default/files/documents/lukion_opetussuunnitelman_perusteet_2019.pdf

Papadimitropoulos, N., Dalacosta, K., & Pavlatou, E. A. (2021). Teaching Chemistry with Arduino Experiments in a Mixed Virtual-Physical Learning Environment. *Journal of Science Education and Technology*, *30*(4), 550–566. https://doi.org/10.1007/s10956-020-09899-5

Thomas, J. W. (2000). *A review of research on project-based learning*. The Autodesk Foundation.

Liite 1. Arduino koodi.

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>

const char* ssid = " ";        // WiFi-verkko
const char* password = " "; // WiFi-salasana

// Soil moisture sensorin pinni
const int moisturePin = A0;  // Käytetään analogista tuloa (A0) sensorille.
void setup() {
  // Aloitetaan sarjaportti ja WiFi-yhteys
  Serial.begin(115200);
  WiFi.begin(ssid, password);

  // Odotetaan, että WiFi-yhteys muodostuu
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }
```

```
  // WiFi-yhteys muodostettu
  Serial.println("Connected to WiFi");
  Serial.print("IP Address: ");
  Serial.println(WiFi.localIP());
}

void loop() {
  // Lue soil moisture sensorin arvo (0-1023)
  int moistureValue = analogRead(moisturePin);

  // Tulostetaan sensorin arvo sarjaporttiin
  Serial.print("Soil Moisture: ");
  Serial.println(moistureValue);

  // Viestin määrittäminen kosteusarvon perusteella
  String message = "";
  if (moistureValue < 500) {
    message = "Todella märkää!";
  } else if (moistureValue >= 500 && moistureValue <= 800) {
    message = "Märkää";
  } else {
    message = "Kuivaa, kastele!";
  }

// URL-enkoodaus viestille (erikoismerkit käsitellään oikein)
  String encodedMessage = urlencode(message);  // Muistetaan määrittää 'encodedMessage'

  // HTTP-pyyntö
  WiFiClient client;
  HTTPClient http;

  String serverPath = "Sivuston url" + String(moistureValue) + "&i=1&message=" + encodedMessage;

  // Aloita HTTP-pyyntö
  http.begin(client, serverPath);

  // Lähetetään GET-pyyntö ja tarkistetaan statuskoodi
  int httpCode = http.GET();

  if (httpCode > 0) {
    // HTTP-vastaus onnistui
    Serial.print("HTTP Response Code: ");
    Serial.println(httpCode);
    String payload = http.getString();
    Serial.println("Server Response:");
```

```
  Serial.println(payload);  // Tulostetaan palvelimen vastaus
 } else {
  // HTTP-pyyntö epäonnistui
  Serial.print("Error on HTTP request: ");
  Serial.println(httpCode);
 }

 // Sulje yhteys
 http.end();

 // Odotetaan hetki ennen seuraavaa pyytämistä
 delay(60000);  // 60 sekuntia ennen seuraavaa pyyntöä
}

// Funktio URL-enkoodaukseen
String urlencode(String str) {
 String encodedString = "";
 char c;
 for (int i = 0; i < str.length(); i++) {
  c = str.charAt(i);
  if (c == ' ') {
   encodedString += "%20";
  } else if (c == '!') {
   encodedString += "%21";
  } else if (c == '"') {
   encodedString += "%22";
  } else if (c == '#') {
   encodedString += "%23";
  } else if (c == '$') {
   encodedString += "%24";
  } else if (c == '%') {
   encodedString += "%25";
  } else if (c == '&') {
   encodedString += "%26";
  } else if (c == '\'') {
   encodedString += "%27";
  } else if (c == '(') {
   encodedString += "%28";
  } else if (c == ')') {
   encodedString += "%29";
  } else if (c == '*') {
   encodedString += "%2A";
  } else if (c == '+') {
   encodedString += "%2B";
  } else if (c == ',') {
   encodedString += "%2C";
```

```
    } else if (c == '-') {
      encodedString += "%2D";
    } else if (c == '.') {
      encodedString += "%2E";
    } else if (c == '/') {
      encodedString += "%2F";
    } else if (c == ':') {
      encodedString += "%3A";
    } else if (c == ';') {
      encodedString += "%3B";
    } else if (c == '<') {
      encodedString += "%3C";
    } else if (c == '=') {
      encodedString += "%3D";
    } else if (c == '>') {
      encodedString += "%3E";
    } else if (c == '?') {
      encodedString += "%3F";
    } else if (c == '@') {
      encodedString += "%40";
    } else if (c == '[') {
      encodedString += "%5B";
    } else if (c == '\\') {
      encodedString += "%5C";
    } else if (c == ']') {
      encodedString += "%5D";
    } else if (c == '^') {
      encodedString += "%5E";
    } else if (c == '_') {
      encodedString += "%5F";
    } else if (c == '`') {
      encodedString += "%60";
    } else if (c == '{') {
      encodedString += "%7B";
    } else if (c == '|') {
      encodedString += "%7C";
    } else if (c == '}') {
      encodedString += "%7D";
    } else if (c == '~') {
      encodedString += "%7E";
    } else {
      encodedString += c;
    }
  }
  return encodedString;
}
```

Liite 2. Verkkosivuston data.php koodi

```
<?php
// Asetetaan aikavyöhyke Suomeen
date_default_timezone_set('Europe/Helsinki');

// Tarkistetaan, että kaikki parametrit tulevat oikein URL-osoitteesta
if (!isset($_GET['moisture']) || !isset($_GET['i']) || !isset($_GET['message'])) {
   echo "<h2 style='color: red; text-align: center;'>❌ Dataa ei saatu oikein.</h2>";
   exit;
}

// Haetaan arvot URL-parametreista
$moisture = intval($_GET['moisture']);  // Kosteusarvo (0-1023)
$sensorId = intval($_GET['i']);         // Sensori ID
$message = htmlspecialchars($_GET['message'], ENT_QUOTES, 'UTF-8');  // Viesti

// Lasketaan kosteusarvo prosentteina (0-100%)
$moisturePercentage = round((1023 - $moisture) / 10.23);

// Määritetään mullan tila kosteuden perusteella
if ($moisturePercentage > 70) {
   $status = "Todella kostea";
} elseif ($moisturePercentage > 40) {
   $status = "Kostea";
} else {
   $status = "Kuiva";
}

// Yhdistetään tietokantaan
$servername = " ";  // Muuta tämä omaksi palvelimeksesi
$username = " ";      // Tietokannan käyttäjätunnus
$password = " ";            // Tietokannan salasana
$dbname = " ";        // Tietokannan nimi

$conn = new mysqli($servername, $username, $password, $dbname);

// Tarkistetaan, onko yhteys onnistunut
if ($conn->connect_error) {
   die("Connection failed: " . $conn->connect_error);
}

// Insert uudet tiedot tietokantaan
$sql = "INSERT INTO readings (moisture, status, timestamp) VALUES (?, ?, ?)";
$stmt = $conn->prepare($sql);
$timestamp = date("Y-m-d H:i:s");  // Aika, jolloin mittaus otettiin
```

```
$stmt->bind_param("iss", $moisturePercentage, $status, $timestamp);

// Suoritetaan kysely ja tarkistetaan, onnistuiko
if ($stmt->execute()) {
   echo "Data tallennettu onnistuneesti!";
} else {
   echo "Virhe tallennuksessa: " . $stmt->error;
}

// Suljetaan yhteys
$stmt->close();
$conn->close();
?>
```

Liite 3. Verkkosivuston get_readings.php koodi

```
<?php
// Yhdistetään tietokantaan
$servername = " ";
$username = " ";
$password = " ";
$dbname = " ";

// Yhdistetään tietokantaan
$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect_error) {
   die("Connection failed: " . $conn->connect_error);
}

// Haetaan viimeisin mittaus tietokannasta
$sql = "SELECT moisture, status, timestamp FROM readings ORDER BY timestamp DESC LIMIT 5";
$result = $conn->query($sql);

$data = [];
if ($result->num_rows > 0) {
   // Jos löytyy mittaustuloksia, haetaan ne
   while($row = $result->fetch_assoc()) {
      $data[] = [
         'moisture' => $row['moisture'],
         'status' => $row['status'],
         'timestamp' => $row['timestamp']
      ];
   }
} else {
   $data[] = [
```

```
        'moisture' => 0,
        'status' => 'Tuntematon',
        'timestamp' => 'Ei saatavilla'
    ];
}

// Palautetaan JSON-muodossa
echo json_encode($data);

// Suljetaan yhteys
$conn->close();
?>
```

Liite 4. Verkkosivuston index.php koodi

```
<!DOCTYPE html>
<html lang="fi">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Kosteusmittari</title>
    <style>
        body { font-family: Arial, sans-serif; text-align: center; padding: 20px; }
        #dataContainer { font-size: 20px; margin-top: 20px; padding: 10px; border: 2px solid black; display: inline-block; }
        .wet { color: blue; font-weight: bold; }
        .moist { color: green; font-weight: bold; }
        .dry { color: orange; font-weight: bold; }
    </style>
    <script>
        function updateData() {
            fetch("get_last_readings.php?t=" + new Date().getTime())  // Varmistetaan, että välimuisti ei aiheuta ongelmia
            .then(response => response.json())  // Haetaan JSON-muotoinen data
            .then(data => {
                let output = "";
                data.forEach(item => {
                    // Luodaan HTML-elementit jokaiselle mittaukselle
                    let statusClass = "dry";
                    if (item.status === "Märkää") {
                        statusClass = "moist";
                    } else if (item.status === "Todella märkää!") {
                        statusClass = "wet";
                    }

                    output += `<p><span class="${statusClass}">Mullan kosteus: ${item.moisture}%</span> - ${item.status} <br><small>${item.timestamp}</small></p>`;
```

```
            });
            document.getElementById("dataContainer").innerHTML = output;
          })
          .catch(error => console.error("Virhe haettaessa dataa:", error));
      }
      setInterval(updateData, 5000); // Päivitetään 5 sekunnin välein
    </script>
</head>
<body>
    <h1>🌱 Mullan kosteus sensori</h1>
    <h2>Viimeisimmät kosteusdatat:</h2>
    <div id="dataContainer">
        <!-- Viimeisimmät tiedot näkyvät täällä -->
    </div>
</body>
</html>
```