VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Jesse Koivisto

LÄMPÖTILOJEN SEURANTA GOOGLEN VERKKOSOVELLUKSELLA

Liiketalous
2021

VAASAN AMMATTIKORKEAKOULU
Tietojenkäsittely

TIIVISTELMÄ

Tekijä	Jesse Koivisto
Opinnäytetyön nimi	Lämpötilojen seuranta Googlen verkkosovelluksella
Vuosi	2021
Kieli	suomi
Sivumäärä	39
Ohjaaja	Antti Mäkitalo

Opinnäytetyön aiheena on toteuttaa projekti, jonka avulla voidaan seurata lämpötila-antureiden mittaamia arvoja verkkosovelluksen kautta. Projektissa mitataan lämpötila-arvoja 1-Wire protokollaa sekä NodeMCU-kehitysalustaa käyttäen. Mitatuista lämpötila-arvoista piirretään viivakaavioita Googlen verkkosovellukseen, josta niitä päästään seuraamaan verkkoselaimen kautta.

Opinnäytetyön projektin toteutuksessa käytettiin NodeMCU ESP8266 -kehitysalustaa sekä DS18B20 lämpötila-antureita. NodeMCU ohjelmoitiin Arduino IDE -kehitysympäristön avulla. Ohjelma lukee lämpötila-antureiden antamat arvot, ja lähettää ne Wi-Fi-yhteyden avulla Google Driveen. Google Drivessä verkkosovelluksen kehittämiseen käytetään HTML, CSS sekä Google Apps Script -ohjelmointikieltä. Viivakaaviot piirretään sivulle Chart.js JavaScript-kirjastoa avuksi käyttäen.

Opinnäytetyön alussa käydään läpi projektissa käytettyjä laitteita ja ohjelmia, sekä niiden toimintaperiaatteita. Tämän jälkeen kerrotaan projektin suunnittelusta sekä sen toteutuksen eri vaiheista. Opinnäytetyön lopussa nähdään, miten projektin toteutuksessa onnistuttiin.

Avainsanat GoogleAppsScript, Chart.js, DS18B20, ESP8266

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Tietojenkäsittely

ABSTRACT

Author	Jesse Koivisto
Title	Monitoring temperatures with Google web application
Year	2021
Language	Finnish
Pages	39
Name of Supervisor	Antti Mäkitalo

The purpose of this thesis was to create a project that could be used to monitor temperatures remotely. The temperatures are measured by using 1-Wire protocol and ESP8266 microcontroller. The measured temperatures are used to create multiple line charts in a web application that you can monitor with a web browser.

The measuring of the temperatures for this project was done by using NodeMCU ESP8266 development board and DS18B20 temperature sensors. The development board was programmed to get the values measured by the temperature sensors and send them to Google Drive using a Wi-Fi connection. This program was made with Arduino IDE development environment. The Google web application was developed with HTML, CSS, and Google Apps Script programming language. Chart.js JavaScript library was used to help with data visualization, by creating the line charts for the web application.

At the beginning of the thesis all the hardware and software needed for the project are reviewed. After this the planning of the project and the different stages of its development are described. At the end of the thesis, we see how the final result turned out.

Keywords GoogleAppsScript, Chart.js, DS18B20, ESP8266

SISÄLLYS

KUVALUETTELO

SANASTO

Arduino IDE	Avoimen lähdekoodin ohjelma, jolla ohjelmoidaan Arduino-laitteita.
Argumentti	Aliohjelmalle välitettävä tieto sitä kutsuessa.
Array	Taulukko, joka koostuu peräkkäisistä tallennuspaikoista.
Boards Manager	Arduino IDE -työkalu, jonka avulla asennetaan kehitysalustojen tarvitsemat tiedostot.
CSS	Tyylikieli, jolla määritetään verkkosivujen ulkoasu.
DS18B20	Digitaalinen lämpötila-anturi.
ESP8266	Edullinen Wi-Fi-mikrosiru, jossa on mikrokontrollerin ominaisuudet.
Funktio	Aliohjelma, joka voidaan suorittaa sitä kutsumalla.
Google Sheets	Googlen tarjoama ilmainen verkkopohjainen laskentataulukkosovellus.
GPIO	Signaalinasta, jota voidaan käyttää digitaalisten tulo- tai lähtötoimintojen suorittamiseen.
HTML	Merkintäkieli, jolla luodaan verkkosivujen rakenne.
Kirjasto	Valmiiksi kirjoitettujen koodien kokoelma, joita ohjelma voi käyttää.
Library Manager	Arduino IDE -työkalu, jonka avulla asennetaan kirjastoja.
Mikrokontrolleri	Piiri, joka käsittelee saamansa tiedot ja suorittaa toiminnon tietojen perusteella.

Muuttuja	Muistista varattu paikka, johon voidaan tallentaa ja josta voidaan hakea tietoa.
NodeMCU	ESP8266-kehitysalustan ja siinä toimivan laiteohjelmiston nimi.
Parametri	Aliohjelmassa määritetty sille välitettävä tieto.
Responsiivinen	Verkkosivut toimivat kaikilla eri laitteilla, ja erikokoisissa ikkunoissa.
Wi-Fi	Langaton tekniikka, jota käytetään laitteiden yhdistämiseen internetiin radiosignaaleiden avulla.

1 JOHDANTO

Opinnäytetyön tavoitteena on luoda IoT-projekti, jonka lopputuloksena on verkkosovellus, jota voidaan käyttää lämpötila-antureiden mittaamien arvojen seuraamiseen. IoT tarkoittaa esineiden internetiä. IoT on ollut suuressa kasvussa viime vuosina ja sen kasvu jatkuu edelleen. Suosion kasvuun on vaikuttanut teknologioiden jatkuva kehitys, kuten paremmat ja edullisemmat laitteet sekä luotettava mobiiliyhteys. Nykyään tarjolla on useita hyviä IoT-kehitysalustoja, joiden avulla yksityiset ihmiset pääsevät kehittämään omia IoT-projektejaan edullisesti. IoT:n kasvaessa teknologiat tulevat vain kehittymään entisestään, sekä vielä laajemmin saataville.

Opinnäytetyön idea syntyi, kun sain tietää, että ilma-vesilämpöpumpun lämpötilojen seuraamiseen ei ole saatavilla hyvää sovellusta, mutta sellaiselle olisi tarvetta. Tarvittavan sovelluksen kehitys kuulosti mielenkiintoiselta ja monipuoliselta projektilta. Projektissa pystyisin hyödyntämään ammattikorkeakoulussa opiskeltuja asioita, mutta oppimaan myös paljon uutta.

Projektin tavoitteena on luoda kaksi verkkosovellusta ja yksi ohjelma, jota NodeMCU-kehitysalusta suorittaa. NodeMCU:n ohjelman tulisi mitata lämpötiloja siihen kytkettyjen antureiden avulla, sekä luoda yhteys langattomaan verkkoon. Ensimmäisen verkkosovelluksen tulisi tallentaa mitatut lämpötilatiedot Googlen pilvipalveluun. Toista verkkosovellusta käytetään lämpötilatietojen seuraamiseen, joka tapahtuu sovellukseen piirrettyjen viivakaavioiden avulla.

2 LAITTEET

Tässä pääluvussa käydään läpi kaikki projektissa tarvittavat laitteet ja komponentit. Alussa käsitellään projektin päälaitteita, lämpötila-antureita ja kehitysalustaa. Tekstissä kerrotaan muun muassa näiden laitteiden toimintaperiaatteista ja historiasta. Viimeisessä kappaleessa käydään läpi muita laitteita, kuten kytkennässä käytettyjä komponentteja.

2.1 DS18B20 Lämpötila-anturi

DS18B20 on digitaalinen lämpötila-anturi. Digitaaliset lämpötila-anturit poistavat ylimääräisten komponenttien tarpeen, jolloin järjestelmää ei myöskään tarvitse kalibroida tietyissä vertailulämpötiloissa, kuten esimerkiksi termistoreiden kanssa joudutaan tekemään. A/D-muunninta ei myöskään tarvita, sillä anturi lähettää tasaisesti tarkkaa digitaalista lukemaa. Digitaalisten antureiden tapa toimia ilman muita komponentteja mahdollistaa järjestelmän yksinkertaisen lämpötilanvalvonnan. (ElProCus 2021)

Anturin nimessä kirjaimet DS viittaavat anturin valmistajaan Dallas Semiconductor:iin. Dallas Semiconductor perustettiin vuonna 1987. Se valmisti teknologiatuotteita, kuten esimerkiksi mikro-ohjaimia, lämpöantureita ja tietoliikennetuotteita. Vuonna 2001 Dallas Semiconductorin osti yritys nimeltä Maxim Integrated Products. Tästä huolimatta vanhoissa laitteissa käytetään edelleen Dallas Semiconductorin nimeä tunnistettavuuden vuoksi. (Procure international inc 2021)

DS18B20 Lämpötila-anturi toimii 1-Wire teknologian avulla. 1-Wire perustuu sarjaprotokollaan, jossa yksi päälaite kommunikoi yhden tai useamman orjalaitteen kanssa yhden johdinparin, eli data- ja maajohtimen kautta. Opinnäytetyön projektissa päälaitteena toimii NodeMCU-kehitysalusta ja orjalaitteina DS18B20 lämpötila-anturit. 1-Wire teknologiaan perustuviin laitteisiin saadaan käyttöjännite kahdella eri tavalla, parasiittiteho tilan avulla (Parasite power mode) tai erillisellä virtasyötöllä (Normal mode). Kytkennässä tarvitaan myös 4,7 kΩ ylösvetovastus, joka

kytketään dataväylän ja virransyötön väliin. Väylän ollessa ylätilassa orjalaitteet ottavat virtaa sisäisten kondensaattoreiden lataamiseksi, etteivät ne sammuisi kesken lähetyksen. Erillistä virtasyöttöä käyttäessä tarvitaan yhden johdinparin lisäksi vielä virtajohdin. (Dallas Semiconductor's 1-Wire Protocol 2021)

DS18B20 Lämpötila-anturin toimintalämpötila on -55 ˚C – +125 ˚C. Kaikista tarkimmat lukemat ovat mitattavissa lämpötilan ollessa -10 ˚C ja +85 ˚C välillä. Anturin lähettämä mittaustarkkuus on ohjelmoitavissa 9-, 10-, 11- ja 12-bitin välillä, joka tarkoittaa käytännössä 0,5 °C, 0,25 °C, 0,125 °C tai 0,0625 °C tarkkuutta. Jokaisella DS18B20 lämpötila-anturilla on uniikki 64-bittinen sarjanumero, joka on asetettu antureihin heti tehtaalla. Nämä sarjanumerot mahdollistavat useiden antureiden toiminnan samalla 1-Wire väylällä. Tämän ansiosta yhdellä mikrokontrollerilla on helppoa ohjata laajalle alueelle jaettuja DS18B20-antureita. (Maxim Integrated DS18B20 datasheet 2019)

DS18B20 Lämpötila-anturiin voidaan asettaa hälytysasetukset. Anturiin ohjelmoidaan ylä- ja alaraja. Jos anturin mittaama lämpötila-arvo ei ole asetetun lämpötilarajan sisällä, tapahtuu hälytys, joka viittaa kyseiseen anturiin sen sarjanumerolla. Hälytysominaisuus on erittäin hyvä järjestelmissä, joissa lämpötila mitataan harvoin. Hälytys tapahtuu heti kun lämpötila-anturi havaitsee lämpötilan, joka ei ole asetetun haarukan sisällä, eikä vasta silloin kun mikrokontrolleri käskee antureita lähettämään sille lämpötila-arvoja. Opinnäytetyön projektissa ei käytetä antureiden hälytys ominaisuutta. Lämpötila-anturit ovat asetettu lähettämään lämpötilatiedot mikrokontrollerille niin pienin aikavälein, että hälytys on parempi suorittaa verkkosovelluksessa. (Maxim Integrated DS18B20 datasheet 2019)

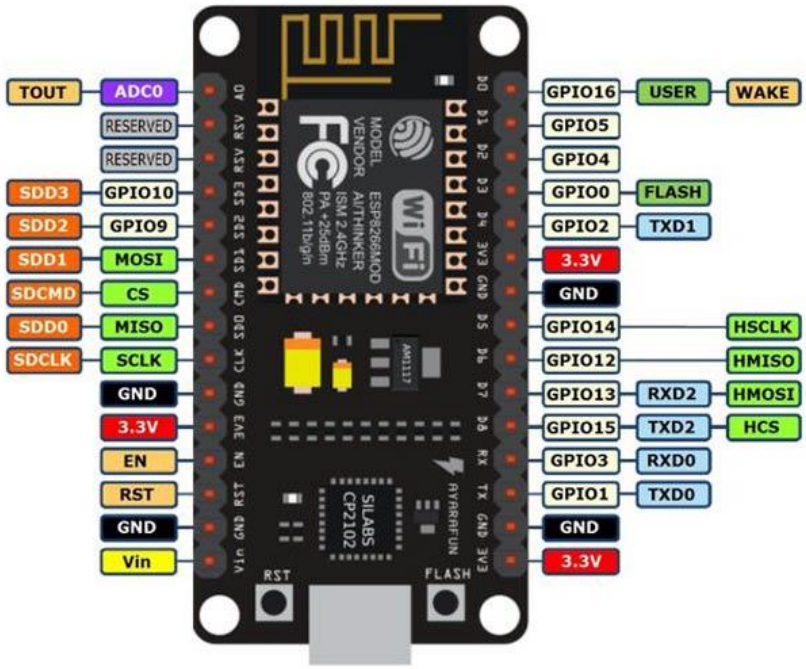
2.2 NodeMCU ESP8266

NodeMCU on laiteohjelmiston ja laitteiston yhdistelmä, joka perustuu ESP8266-12E moduuliin. ESP8266-12E on Espressif Systems -nimisen yrityksen valmistama Wi-Fi yhteyden mahdollistava mikrosiru, jossa on täysin toimiva TCP/IP-pino sekä mikrokontrollerin ominaisuudet. Alun perin termi NodeMCU yhdistettiin vain

ESP8266-kehitysalustoissa toimivaan laiteohjelmistoon, mutta ajan myötä nimestä on tullut synonyymi kehitysalustojen kanssa. Aluksi laitteita pystyi ohjelmoimaan ainoastaan Lua-ohjelmointikielellä. Avoimen lähdekoodin avulla saataville tuli kirjastoja, jotka mahdollistivat laitteiden ohjelmoinnin Arduino IDE -kehitysympäristön avulla. Lua:n suosio laski nopeasti, kun käyttöön saatiin suositumpi Arduino-ohjelmointikieli, sekä ajan myötä myös muita ohjelmointikieliä kuten esimerikiksi JavaScript ja MicroPython. Tämän myötä NodeMCU-kehitysalustojen suosio on noussut suuresti, johtaen niiden edulliseen hintaan tänä päivänä. (TheMakersWorkbench 2019)

ESP8266-sirussa on Tensilica Xtensa® 32-bit LX106 RISC mikroprosessori, joka toimii 80–160 MHz säädettävällä kellotaajuudella sekä tukee RTOS-tekniikkaa (Reaaliaikainen käyttöjärjestelmä). Laitteen 128 KB keskusmuistin lisäksi siinä on 4 MB ulkoista muistia. Ulkoista muistia on riittävästi tallentamaan ohjelmia, dataa, sekä tiedostot nettisivustoa varten. Siru sisältää myös 802.11b/g/n HT40 Wi-Fi -lähetin/vastaanottimen. Wi-Fi yhteyden lisäksi se voi siis luoda myös oman verkon, johon pystytään yhdistämään muita laitteita. (LastMinuteEngineers 2021)

ESP8266:n käyttöjännite on 3 ja 3,6 voltin välillä. Piirilevyssä on regulaattori, joka pitää käyttöjännitteen tasaisesti 3,3 voltissa. Regulaattorin virran kesto on jopa 600 mA, jonka pitäisi olla enemmän kuin tarpeeksi, sillä ESP8266 ottaa noin 80 mA RF-lähetyksen aikana. Regulaattorin lähtö on yhdistetty myös piirilevyssä nimettyyn 3V3-nastaan. Tästä nastasta voidaan syöttää virtaa oheislaitteille. Laitteeseen voidaan syöttää virta kahdella eri tavalla. Se tapahtuu joko MicroB USB-liitännän kautta, tai vaihtoehtoisesti reguloidun 5 V virtalähteen avulla, joka on kytketty VIN-nastaan. Nasta syöttää virran NodeMCU:lle ja kaikille oheislaitteille. NodeMCU:ssa on yhteensä 30 nastaa, joista 17 on GPIO-nastoja (kuva 1). GPIO-nastat ovat ohjelmoitavia digitaalisia nastoja, joilla ei ole ennalta määriteltyjä toimintoja. Niiden yleisiä käyttötarkoituksia ovat esimerkiksi LED-valojen säätäminen, anturien lukeminen ja ulkoisten laitteiden virran ohjaaminen. (LastMinuteEngineers 2021)



Kuva 1. NodeMCU kantajärjestys. (Components101 2020)

2.3 Muut laitteet

Laitteiston kytkennässä tarvitaan riviliitintä, ylösvetovastusta sekä kytkentäjohtoa. Riviliitintä käyttämällä saadaan kytkettyä useita antureita yhteen väylään. Kytkentäjohdon avulla anturit saadaan asetettua haluttuihin paikkoihin. Kytkennässä tarvitaan myös yksi ylösvetovastus, joka varmistaa, että laite pystyy lukemaan antureita. Käyttöjännite laitteelle tuodaan USB-virtalähteestä. Lopuksi NodeMCU asennetaan muoviseen laitekoteloon, suojaamaan sitä kosteudelta ja pölyltä. Saatavilla täytyy myös olla langaton verkko, johon laite yhdistetään. (Microcontrollerslab 2021)

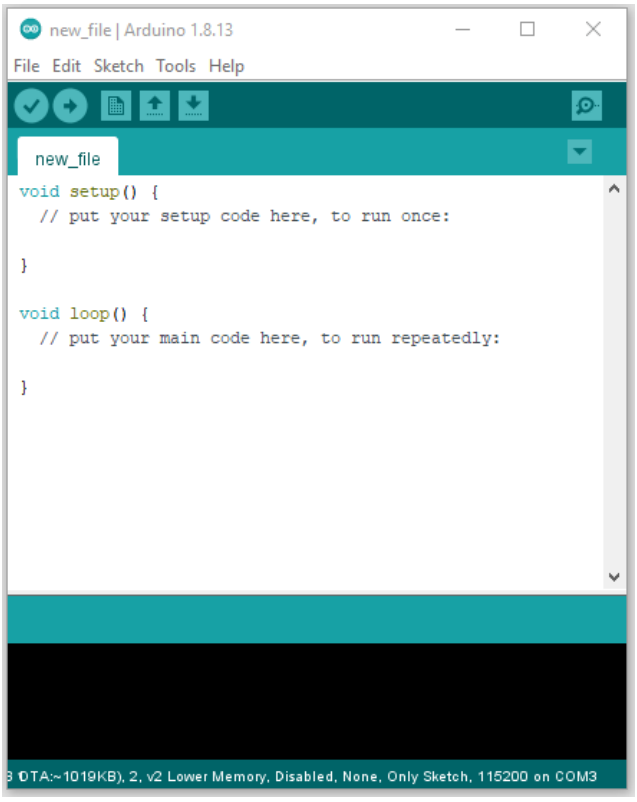
3 OHJELMISTOT

Tässä pääluvussa käydään läpi projektissa käytettyjä ohjelmia ja ohjelmointikieliä. Ensimmäiseksi esitellään ohjelmointiympäristö, jota käytetään NodeMCU-kehitysalustan ohjelmointiin. Tekstissä käsitellään myös Googlen omaa ohjelmointikieltä, sekä projektissa käytettyä JavaScript-kirjastoa.

3.1 Arduino IDE

Arduino IDE on ilmaiseksi käytettävä ohjelmointiympäristö. Se on toteutettu avoimen lähdekoodin avulla. Arduino IDE on suosittu, sillä se toimii kaikilla käytetyimmillä käyttöjärjestelmillä: Windowsilla, Mac OS X:llä ja Linuxilla. Ohjelmointiympäristö on kirjoitettu Java-ohjelmointikielellä ja se perustuu Processing- sekä muihin avoimen lähdekoodin ohjelmistoihin. (Arduino 2021)

Arduino IDE:n avulla muodostetaan yhteys Arduino- tai Genuino-laitteisiin, jolloin laitteiden kanssa pystytään kommunikoimaan ja lataamaan niihin ohjelmia. Ohjelmointiympäristö sisältää tekstieditorin koodin kirjoittamista varten, viestialueen, tekstikonsolin, työkalurivin, jossa on painikkeet yleisille toiminnoille, ja kaikki tarvittavat valikot (kuva 2). Viestialue antaa palautetta ohjelmien lataus- tai tallennusprosessien aikana, ja ilmoittaa mahdollisista virheistä. Konsoli näyttää ohjelmistoympäristön tulostamat viestit, joita ovat esimerkiksi koodissa tapahtuneet virheet. Ikkunan oikeassa alakulmassa nähdään määritetty kortti ja sarjaportti. Työkalupalkin painikkeiden avulla voi tarkistaa ja ladata ohjelmia, luoda, avata ja tallentaa tiedostoja, sekä avata sarjamonitorin. Sarjamonitoria käytetään laitteen kanssa kommunikointiin. Sen avulla voidaan lähettää ja vastaanottaa tekstiä, jota voidaan käyttää avuksi virheenjäljityksessä. Tämän lisäksi sitä voidaan käyttää myös laitteen ohjaamiseen näppäimistön avulla, lähettämällä laitteelle komentoja. (Arduino 2021)



Kuva 2. Arduino IDE käyttöliittymä.

3.2 Google Drive

Google Drive on pilvipalvelu, johon voit tallentaa tiedostojasi ja käyttää niitä missä tahansa, millä tahansa laitteella, jolla on pääsy internettiin. Kaikki, joilla on Google-tili, voivat käyttää Google Driveä ilmaiseksi. Ilmaista tallennustilaa on 15 Gt. Google Driven kaltaisten pilvipalveluiden käyttämisessä on useita etuja, kuten tiedostojen helppo jakaminen ja varmuuskopioiden tallentaminen. Muita pilvipalveluita ovat esimerkiksi DropBox ja Applen iCloud-palvelu, joista Google Drive on kuitenkin suosituin. Suosio perustuu sen sujuvaan toimintaan Googlen tuote- ja palvelukokonaisuuden kanssa. Yksi Google Driven parhaista ominaisuuksista on sen integrointi Googlen pilvipohjaisten sovellusten kanssa, jotka ovat hyvin samankaltaisia kuin Microsoft Office -sovellukset. Useimmat käyttäjät kokevat sovellukset

erittäin hyödyllisiksi. Esimerkkejä näistä sovelluksista ovat muun muassa Google Docs, Google Sheets, Google Slides ja Google Forms. (Nolledo 2020)

3.2.1 Google Apps Script

Google Apps Script on sovelluskehitysalusta, jonka avulla voidaan nopeasti ja helposti tehdä sovelluksia, jotka integroituvat Google Driven työvälineisiin. Koodi kirjoitetaan JavaScriptillä, jonka lisäksi käytössä on sisäänrakennetut kirjastot Googlen eri ohjelmistoja kuten Gmail, Analytics ja Mapsia varten. Mitään ohjelmia ei täydy erikseen asentaa, sillä kaikki voidaan tehdä selaimessa Googlen koodieditorissa, ja tehdyt sovellukset toimivat Googlen palvelimilla. Yleisesti Google Apps Scriptiä käytetään laajentamaan Googlen työvälineiden vakio-ominaisuuksia. Tämä tapahtuu automatisoimalla toistuvasti tehtyjä tai paljon aikaa vieviä tehtäviä. Esimerkkejä tästä ovat muun muassa mukautetut toiminnot ja makrot Google Sheetsiin, automatisoitu sähköpostiviestien lähetys tai verkkosovellus, joka näyttää Google AdSensen ja Analyticsin tietoja samanaikaisesti. (Google 2021)

3.3 Chart.js

Chart.js on yhteisön ylläpitämä avoimen lähdekoodin JavaScript-kirjasto. Sitä käytetään avuksi datan visualisoinnissa piirtämällä erilaisia kaavioita. Chart.js tukee kahdeksaa erilaista kaaviota, joita ovat esimerkiksi viiva-, pylväs- ja ympyräkaavio. Kaikki nämä kaaviot ovat responsiivisia, mikä tekee niiden käytöstä helppoa. Käytännössä tämä tarkoittaa sitä, että kun olet kerran asettanut kaavion asetukset, Chart.js pitää huolen siitä, että kaavio on aina helposti luettavissa. Kaavioista voi tehdä juuri sen näköisiä kuin itse haluaa, sillä melkein jokainen kaavion osa on muokattavissa. Tämän lisäksi kaavioihin voidaan lisätä animaatioita, sekä ladata lisäosia, joiden avulla kaavioista voidaan tehdä vielä käytännöllisempiä. (Ahlin 2017)

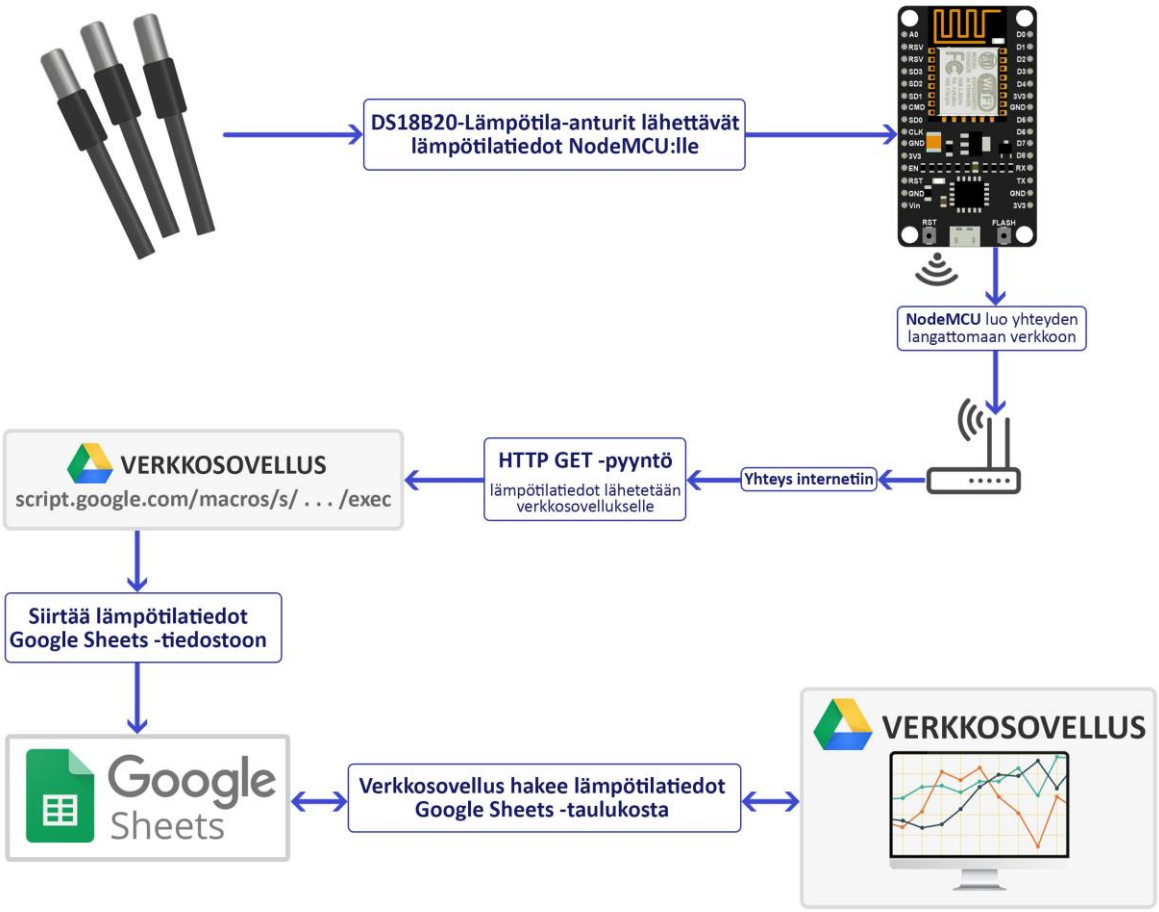
Ennen kuin kaavio voidaan piirtää, HTML-tiedostoon täytyy määrittää alue, johon kaavio tullaan piirtämään. Tämä tapahtuu canvas-elementin avulla. Canvas-elementtiin lisätään myös id (tunnus), johon viitataan kaavion asetuksissa. Näin kaaviot saadaan asetettua haluttuihin paikkoihin, jos niitä on sivulla useampia. Kaavion toteutuksessa ensimmäinen vaihe on antaa canvas-elementin tunnus, jonka avulla kaavio piirretään haluttuun kohtaan. Tämän jälkeen määritetään kaavion tyyppi. Seuraava vaihe on syöttää kaaviolle tiedot, joista kaavio halutaan piirtää. Tiedot täytyy olla tallennettuna array-muuttujiin, että Chart.js voi käyttää niitä. Tiedot, joiden halutaan näkyvän x-akselilla, asetetaan labels-osioon. Tiedot, joista halutaan tehdä kaavioon esimerkiksi viivoja tai pylväitä, asetetaan datasets-osioon. Jos viivakaavioon halutaan lisätä useampia viivoja, täytyy silloin datasets-osioon lisätä jokaista viivaa kohden oma objekti. Objektista voidaan muuttaa muun muassa viivan tiedot, nimen ja värin. Lopuksi määritellään kaavion asetukset. Asetuksista voidaan muuttaa esimerkiksi kaavion värejä, x- ja y-akselin minimi- ja maksimiarvoja sekä animaatioita. Asetuksia ei ole pakko muuttaa, jos on tyytyväinen kaavion oletusulkonäköön. (Ahlin 2017)

4 PROJEKTIN TOIMINTAPERIAATE

Aluksi NodeMCU tallentaa lämpötila-antureiden mittaamat arvot muuttujiin. Tämän jälkeen NodeMCU luo yhteyden langattomaan verkkoon. Kun yhteys on luotu, ohjelma avaa Google Drivessä tehdyn verkkosovelluksen, jonka avulla lämpötilatiedot saadaan siirrettyä muuttujista Google Sheetsiin. NodeMCU toistaa tämän ohjelman kahden minuutin väliajoin.

Google Sheets tallentaa lämpötila-arvot kahden vuorokauden ajalta. Kun uudet lämpötilatiedot siirtyvät Google Sheetsiin ja rivien määrä ylittää 1440, vanhin rivi poistetaan automaattisesti. Näin nähdään kaikki lämpötilatiedot viimeisen 48 tunnin ajalta, ilman että taulukko täyttyy ajan myötä. Google Sheetsiin on myös ohjelmoitu hälytys, joka ilmoittaa sähköpostiviestillä, jos huoneen lämpötila nousee yli 50 °C.

Verkkosovelluksen toteutus tehdään Google Drivessä normaalisti HTML-tiedoston avulla. Lämpötilatiedot haetaan Google Sheets -taulukosta, josta ne tallennetaan moniulotteiseen array-muuttujaan. Tämä array jaetaan moneen pienempään array-muuttujaan, jolloin jokaisen anturin lukemat ovat omassa muuttujassaan. Näistä muuttujista voidaan hakea tiedot Chart.js:än avulla piirrettyihin viivakaavioihin. Ruutu jaetaan kahteen osaan, jonka toisessa osassa nähdään jokaisen anturin mittaamat arvot viimeisen tunnin ajalta. Nämä kaaviot tehdään funktion avulla, sillä kaikissa kaavioissa on samat asetukset. Sivun toisessa osassa taas on suurempi viivakaavio, josta nähdään kaikki arvot viimeisen 24 tunnin ajalta. Projektin toimintaperiaate on visualisoitu kappaleen lopussa sijaitsevaan kuvaan (kuva 3).



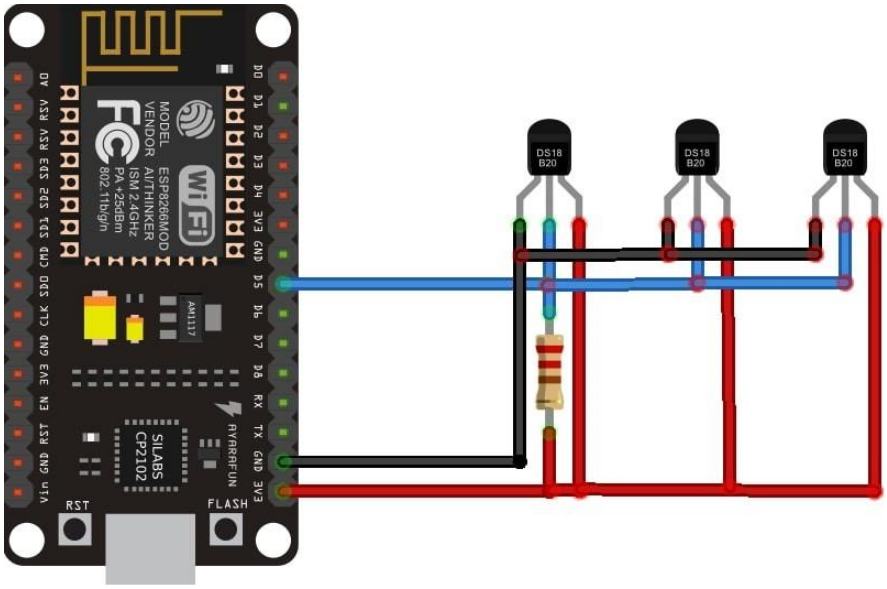
Kuva 3. Kaavio projektin toimintaperiaatteesta.

5 PROJEKTIN TOTEUTUS

Tässä pääluvussa kerrotaan projektin toteutuksesta. Tekstissä käydään läpi kaikki toteutuksen eri vaiheet, ja miten toteutus tapahtui. Projektin eri vaiheet ovat: laitteiden kytkentä, Google Sheets -verkkosovellus, kehitysalustan ohjelmointi ja lopuksi HTML-sivun toteutus. Lisäksi luvussa näytetään esimerkkikuvia työssä kirjoitetusta koodista.

5.1 Kytkentä

Lämpötila-antureita käytetään normal modessa, eli anturit saavat virran NodeMCU:n 3V3-nastasta. Antureissa on kolme johdinta: plusjohdin, miinusjohdin ja datajohdin. Anturin plusjohdin kytketään NodeMCU:n 3V3-nastaan, ja anturin miinusjohdin kytketään gnd-nastaan (kuva 4). Anturin datajohdin voidaan kytkeä mihin tahansa sopivaan GPIO-nastaan. Ohjelmaa tehdessä täytyy muistaa, mihin nastaan datajohdin on kytketty. Työssäni käytetään D2-nastaa. Datajohtimen ja 3V3-nastan väliin asetetaan 4,7 kΩ ylösvetovastus. Vastuksen ja antureiden kytkentä väylään tehdään riviliittimellä.



Kuva 4. NodeMCU ja DS18B20 kytkentäkaavio. (Microcontrollerslab 2021)

5.2 Google Sheets -tietokanta

Ensimmäiseksi tehdään uusi Google Sheets -tiedosto. Taulukko täytyy nimetä, että siihen voidaan viitata myöhemmin koodissa. Seuraava vaihe oli nimetä sarakkeet. Ensimmäiseen sarakkeeseen tulee päivämäärä, toiseen sarakkeeseen kellonaika, jonka jälkeen jokaisen lämpötila-anturin arvolle on oma sarake. Tämän jälkeen tehdään verkkosovellus, jonka avulla lämpötilatiedot tallennetaan kaavioon. Ohjelman kirjoittaminen voidaan aloittaa valitsemalla valikosta: Työkalut -> Ohjelman muokkaustyökalu.

Ohjelma voi olla verkkosovellus, jos se sisältää joko doGet(e)- tai doPost(e)-funktion. Verkkosovellus tehdään doGet(e)-funktion avulla. Tämä tarkoittaa, että kun käyttäjä vierailee sovelluksessa tai ohjelma lähettää sovellukselle HTTP GET-pyynnön, sovellus suorittaa doGet(e)-funktion automaattisesti. E-argumentti edustaa event-parametria, joka voi sisältää tietoja request-parametreista. Esimerkiksi parametrit käyttäjänimi ja ikä voidaan välittää URL-osoitteen kautta alla esitetyn kuvan tavalla (kuva 5). (Google 2020)

```
https://script.google.com/.../exec?username=jsmith&age=21
```

Kuva 5. doGet(e)-funktio esimerkki. (Google 2020)

Koodin alussa tarkistetaan, onko parametreja välitetty. Jos parametreja ei ole lähetetty, ohjelmaa ei suoriteta. Taulukkoon vietävät tiedot tallennetaan ensin array-muuttujaan, josta ne viedään taulukkoon uudelle riville, kun kaikki tiedot on saatu tallennettua. Päivämäärä ja kellonaika tehdään new Date -konstruktorin avulla. Array-muuttujan ensimmäiseen elementtiin tallennetaan päivämäärä, ja toiseen elementtiin kellonaika. Seuraava vaihe on tallentaa vastaanotetut lämpötila-arvot. Tämä tehdään for/in-toistolausekkeen avulla, joka käy järjestyksessä läpi kaikki parametriobjektin arvot. Toistolauseen sisällä käytetään switch-lausekketta, jonka avulla lämpötila-arvot tallennetaan oikeisiin array-elementteihin (kuva 6). Ennen arvojen tallennusta, niistä poistetaan mahdolliset ylimääräiset merkit kuten lainausmerkit. Tämä tapahtuu lyhyen funktion avulla.

```
var rowData = [];
var Curr_Date = new Date();
rowData[0] = Curr_Date; // Pvm sarake A
var Curr_Time = Utilities.formatDate(Curr_Date, "Europe/Helsinki", 'HH:mm');
rowData[1] = Curr_Time; // Kellonaika sarake B

for (var param in e.parameter) {

  var value = removeQuotes(e.parameter[param]);

  switch (param) {
    case 'temperature1':
      rowData[2] = value; // Anturi 1 - sarake C
      break;
    case 'temperature2':
      rowData[3] = value; // Anturi 2 - sarake D
      break;
    case 'temperature3':
      rowData[4] = value; // Anturi 3 - sarake E
      break;
    case 'temperature4':
      rowData[5] = value; // Anturi 4 - sarake F
      break;
    case 'temperature5':
      rowData[6] = value; // Anturi 5 - sarake G
      break;
```

Kuva 6. Tietojen tallennus array-muuttujaan.

Kun kaikki tiedot on saatu tallennettua array-muuttujaan, määritetään Google Sheets -taulukon rivi, johon tiedot kirjoitetaan. Rivi määritetään käyttämällä Apps Scriptin metodeja getLastRow ja getRange. Kun rivi on määritetty, tiedot siirretään setValues-metodin avulla. Samalla tarkistetaan mille riville uudet tiedot tulevat. Jos rivinumero on yli 1440, vanhin rivi poistetaan taulukosta if-lauseketta käyttäen. Näin taulukossa on vain uusimmat lämpötilatiedot 48 tunnin ajalta, eikä se täyty vanhoilla tiedoilla. Kun ohjelma on valmis, se täytyy julkaista verkkosovelluksena. Tämä tapahtuu valitsemalla valikosta: Julkaise -> Ota käyttöön verkkosovelluksena.

5.2.1 Sähköpostihälytys

Lämpötilatietojen array-muuttujasta valitaan elementti, johon on tallennettu huoneen lämpötila, ja se tallennetaan erilliseen muuttujaan. Muuttuja muutetaan liukuluvuksi, jonka jälkeen sitä verrataan if-lausekkeessa. Sähköpostihälytys-funktio suoritetaan, jos muuttujan arvo on yli 50. Funktio ei kuitenkaan jatka hälytysten lähettämistä, jos muuttujan arvo on 50 useamman minuutin ajan. Tämä johtuu siitä, että on mahdollista, että lämpötila-anturiin tulee häiriö, jolloin se voi antaa väärää arvoa niin kauan kunnes häiriö korjaantuu. Häiriön sattuessa ei haluta, että turhia sähköpostihälytyksiä lähetetään kahden minuutin välein. Tämä ratkaistaan sillä, että funktio hakee taulukosta viimeksi mitatun huoneen lämpötilan ja pyöristää sen ylöspäin seuraavaksi kokonaisluvuksi. Pyöristettyä lukua verrataan mitattuun huoneenlämpötilaan. Sähköpostihälytys lähetetään vain, jos uusi lukema on yhtä suuri tai suurempi kuin pyöristetty luku. Tällöin ensimmäinen hälytys lähetetään aina kun huoneen lämpötila ylittää 50 °C, mutta seuraavat hälytykset lähetetään vain, jos lämpötila nousee yli yhden celsiusasteen kahdessa minuutissa.

Sähköpostihälytys-funktioon määritetään yksi parametri, johon tallennetaan lämpötila-anturin mittaama huoneenlämpötila. Vanha lämpötila-arvo saadaan tallennettua muuttujaan käyttämällä Apps Scriptin getRange- ja getValue-metodeja. Tämän jälkeen se pyöristetään ylöspäin käyttämällä Math.ceil-funktiota. Kun luku on saatu pyöristettyä, sitä verrataan mitattuun huoneen lämpötilaan if-lausekkeessa. If-lausekkeen toteutuessa, sähköpostihälytys toteutetaan GmailApp-luokan avulla, joka tarjoaa pääsyn Gmailin ominaisuuksiin. Sähköpostin lähetyksessä käytetään sendEmail-metodia. Metodin argumenteiksi asetetaan vastaanottajan sähköpostiosoite, sähköpostin otsikko ja sähköpostin viesti (kuva 7).

```
function sendEmail(temp){
  var row = 1440;
  var col = 7;

  var oldTemp = SpreadsheetApp.getActiveSheet().getRange(row, col).getValue();     // Viimeksi mitattu huoneen lämpötila
  var oldTempCeil = Math.ceil(oldTemp);                                            // Viimeksi mitattu huoneen lämpötila pyöristetty

  if (temp >= oldTempCeil) {
    GmailApp.sendEmail("sähköposti@gmail.com", "Hälytys - Lämpötila", "Lämpötila: " + temp); // Lähetä sähköpostihälytys
    Logger.log("email sent");
  }
  else {
    Logger.log("returning");                                                       // Lopeta funktio
    return;
  }
}
```

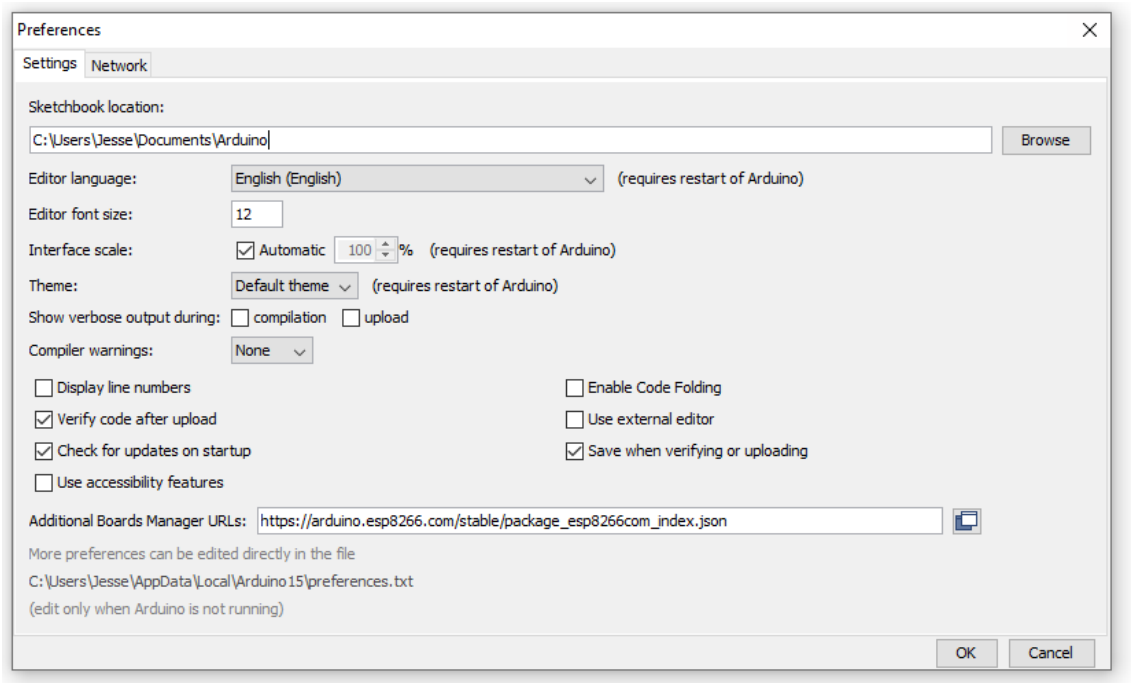
Kuva 7. Sähköpostihälytys-funktio.

5.3 NodeMCU ohjelmointi

Tässä alaluvussa käydään läpi NodeMCU-kehitysalustan ohjelmointi. Luvun alussa kerrotaan Arduino IDE -kehitysympäristön valmistelusta, ennen kuin ohjelmointi voidaan aloittaa. Tekstissä käsitellään myös työssä tarvittavia kirjastoja, sekä ohjelman asetus- ja toistofunktiot.

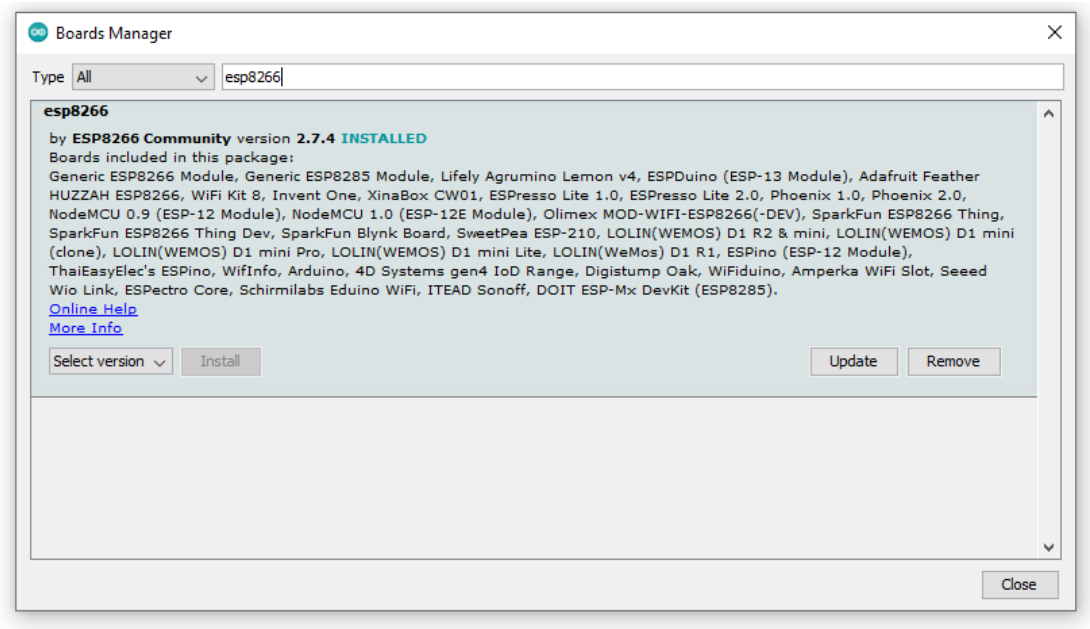
5.3.1 Arduino IDE asetukset ja kirjastojen asennus

Ennen kuin NodeMCU:ta voidaan ruveta ohjelmoimaan, täytyy ladata vaaditut tiedostot, joiden avulla Arduino IDE tunnistaa ohjelmoitavan piirilevyn. Seuraava vaihe on määrittää Arduino IDE:n asetuksiin ESP8266-mikrokontrollerin hallintatyökalun JSON-tiedoston URL-osoite (kuva 8).



Kuva 8. Arduino IDE Preferences-ikkuna.

Tämän jälkeen käytetään Boards Manageria, josta oikeat tiedostot löydetään helposti käyttämällä hakukenttää (kuva 9). Hakukenttään kirjoitetaan ESP8266, jonka jälkeen tiedostot voidaan asentaa. Kun tiedostot on asennettu, tarvitsee enää vain valita asetuksista käytettäväksi laitteeksi NodeMCU. Asetuksista täytyy valita myös tietokoneen COM-portti, johon laite on kytketty. Tämän jälkeen NodeMCU:ta voidaan ruveta ohjelmoimaan.



Kuva 9. Arduino IDE Boards Manager -ikkuna.

Ohjelmassa käytetään DallasTemperature- ja OneWire-kirjastoja lämpötilojen mittaamiseen. Muita kirjastoja, joita ohjelmassa tarvitaan ovat ESP8266WiFi ja WiFiClientSecure. Näiden kirjastojen avulla saadaan luotua yhteys verkkoon ja lähetettyä tiedot Google Sheetsiin. Kirjastot tarjoavat ohjelmalle lisätoimintoja sekä valmiita koodinpätkiä, jotka auttavat ohjelman kirjoittamisessa. Kirjastoja voidaan asentaa kehitysympäristöön library managerin avulla.

5.3.2 Kirjastot ja vakiomuuttujat

Ohjelman alussa määritetään tarvittavat kirjastot. Tämän jälkeen määritetään muuttumattomat arvot. Tähän voidaan käyttää define-komentoa tai const-määritettä. Define-komennon avulla pystytään nimeämään vakioarvo. Tästä on apua esimerkiksi piirilevyn nastojen numeroita nimetessä, joka tekee koodista helpommin luettavaa. Const-määritteen avulla voidaan luoda muuttuja normaalisti, mutta kyseisen muuttujan arvo pysyy aina samana, ja sitä ei voida muuttaa myöhemmin. Näihin vakiomuuttujiin tallennetaan muun muassa langattoman verkon

tiedot, sekä Google Sheets -tiedoston URL-osoite, jota tullaan käyttämään myöhemmin (kuva 10).

```
#include <ESP8266WiFi.h>                 // Wifi kirjastot
#include <WiFiClientSecure.h>

#include <DallasTemperature.h>           // Lämpötila-anturi kirjastot
#include <OneWire.h>

#define ONE_WIRE_BUS 4                   // NodeMCU D2 nasta
#define ON_Board_LED 2                   // NodeMCU LED


const char* ssid = "*******";            // Wi-Fi tiedot
const char* password = "*******";

const char* host = "script.google.com";  // Palvelin
const int httpsPort = 443;               // Portti

const char* GAS_ID = "AKfycbwV7Qp4KaeaNpuzy92NHagMYAQKGoyflyeUq9etjUx-pwuX5pAK"; //Google Sheets verkkosovellus ID
```

Kuva 10. Linkitetyt kirjastot ja vakiomuuttujat.

Että antureiden mittaamia arvoja voitaisiin lukea, täytyy ohjelman alussa käyttää linkitettyjä kirjastoja avuksi. Aluksi tehdään 1-Wire-objekti syöttämällä anturin signaalinastan numero konstruktorille. Tämän 1-Wire-objektin avulla voidaan kommunikoida 1-Wire-laitteiden kanssa. Jotta kommunikointi onnistuisi juuri DS18B20-anturin kanssa, täytyy luoda DallasTemperature-objekti, ja asettaa sen argumentiksi luotu 1-Wire-objekti. Samalla tehdään myös WiFiClientSecure-objekti, jonka avulla Google-verkkosovellukseen lähetetään myöhemmin HTTP GET-pyyntö (kuva 11).

```
OneWire oneWire(ONE_WIRE_BUS);            // 1-wire objekti

DallasTemperature sensors(&oneWire);      // DallasTemperature objekti

WiFiClientSecure client;                  // WiFiClientSecure objekti
```

Kuva 11. 1-Wire-, DallasTemperature- ja WiFiClientSecure-objektien luonti.

5.3.3 Setup-funktio

Ohjelmassa on yleensä aina kaksi pääfunktiota, setup- ja loop-funktiot. Aluksi määritetään setup-funktio (kuva 12). Tämä funktio suoritetaan ensimmäisenä, kun laite käynnistetään, ja se suoritetaan vain yhden kerran. Ensimmäisenä funktiossa suoritetaan Serial.begin-komento. Tämän komennon avulla määritetään tiedonsiirtonopeus, jolla laite kommunikoi sarjamonitorin kanssa. Seuraava funktio joka suoritetaan on sensors.begin. Kyseinen funktio tarkistaa montako anturia 1-Wire väylään on kytketty, ja se täytyy aina suorittaa ennen kuin anturilta voidaan hakea tietoa. Seuraava vaihe on suorittaa WiFiClientSecure-objektin client.setInsecure-funktio, joka poistaa käytöstä sormenjälki varmenteen. Setup-funktion viimeinen vaihe on luoda yhteys langattomaan verkkoon, joka onnistuu käyttämällä ohjelman alussa määritettyjä vakiomuuttujia. While-toistolausetta käyttäen saadaan NodeMCU:n LED-valo vilkkumaan silloin, kun laite yrittää luoda yhteyttä langattomaan verkkoon.

```
void setup() {

  Serial.begin(115200);                  // Sarjamonitori

  sensors.begin();                       // 1-wire väylän anturit

  WiFi.begin(ssid, password);            // Yhdistää langattomaan verkkoon
  Serial.println("");

  pinMode(ON_Board_LED,OUTPUT);          // LED määritetään output-tilaan
  digitalWrite(ON_Board_LED, HIGH);      // Sammuta LED

  Serial.print("Yhdistää verkkoon");

  while (WiFi.status() != WL_CONNECTED) { // LED vilkkuu kunnes yhdistää langattomaan verkkoon
    Serial.print(".");
    digitalWrite(ON_Board_LED, LOW);
    delay(250);
    digitalWrite(ON_Board_LED, HIGH);
    delay(250);
  }
```

Kuva 12. Setup-funktio.

5.3.4 Loop-funktio

Loop-funktio on ohjelma, jota laite toistaa jatkuvasti sen jälkeen, kun setup-funktio on suoritettu. Ohjelma halutaan suorittaa kahden minuutin välein. Tämän takia alussa käytetään delay-funktiota. Delay-funktion argumentiksi asetetaan tauon pituus millisekunteina. Kahden minuutin tauon jälkeen suoritetaan requestTemperatures-funktio, joka lähettää lämpötilan mittaus komennon kaikille antureille, jotka ovat kytketty 1-Wire väylään. Seuraava vaihe on luoda jokaiselle lämpötila-anturille oma muuttuja, johon mitattu arvo tallennetaan. Arvot saadaan tallennettua muuttujiin käyttämällä getTempCByIndex-funktiota (kuva 13). Tämä funktio lukee ja palauttaa yhden anturin mittaaman lämpötilalukeman celsiusasteissa. Funktion argumentiksi asetetaan numero, joka kuvaa halutun anturin paikkaa 1-Wire väylällä. Väylän ensimmäisen anturin numero on nolla.

```
delay(120000);                      // Odota 2 min
sensors.requestTemperatures();      // Mittaa lämpötilat
float temp1 = sensors.getTempCByIndex(0);
float temp2 = sensors.getTempCByIndex(1);
```

Kuva 13. Loop-funktion viive ja lämpötilojen mittaus.

Lämpötilatietojen tallennuksen jälkeen seuraava vaihe on lähettää tiedot Google-verkkosovellukseen Wi-Fi yhteyden avulla. Seuraava osuus koodista suoritetaan aliohjelman (funktion) avulla. Aliohjelmia käytetään yleensä koodinpätkissä, jotka tullaan suorittamaan koodissa useamman kerran. Vaikka käytetty aliohjelma suoritetaan ohjelmassa vain kerran, kunnes loop-funktio suoritetaan uudestaan, se auttaa silti tekemään koodista helpommin luettavaa.

Aliohjelmaa kutsuessa sille annetaan argumenteiksi muuttujat, joihin lämpötilatiedot on tallennettu. Ensimmäiseksi kaikkien lämpötilatietojen muuttujat muutetaan liukuluvuista merkkijonoksi. Tämän jälkeen niiden desimaalipisteet muutetaan pilkuiksi, sillä Google Sheets voi tulkita joitain desimaalipisteellä merkittyjä lukuja päivämääriksi. Seuraavaksi luodaan yhteys Googlen palvelimelle WiFiClientSecure-kirjaston connect-funktion ja ohjelman alussa määritettyjen vakiomuuttujien avulla. Jotta tietojen siirto onnistuisi, täytyy luoda URL-osoite, josta verkkosovellus tunnistaa Google Sheetsiin siirrettävät arvot. URL-osoite luodaan merkkijono-muuttujaan, johon lämpötilatiedot haetaan niiden omista muuttujista. Tämän jälkeen verkkosovellukselle lähetetään HTTP GET-pyyntö WiFiClientSecure-objektin print-funktion avulla (kuva 14). Palvelimen vastaus voidaan lukea readStringUntil-funktion avulla, jolla tarkistetaan onnistuiko tietojen lähetys.

```
client.print(String("GET ") + url + " HTTP/1.1\r\n" +
       "Host: " + host + "\r\n" +
       "User-Agent: ESP8266\r\n" +
       "Connection: close\r\n\r\n");
```

Kuva 14. GET-pyynnön lähetys verkkosovellukseen.

5.4 Google-verkkosovellus

Verkkosovellusta varten tehdään uusi Google Apps Script -tiedosto, sekä HTML-tiedosto. Apps Script -tiedostosta tehdään verkkosovellus käyttämällä doGet(e)-funktiota. Tässä funktiossa suoritetaan createHTMLOutputFromFile-komento. Tämän komennon avulla aina kun verkkosovellus avataan ja doGet(e)-funktio suoritetaan, HTML-tiedosto tulostetaan sivulle. Apps Script -tiedostoon luodaan myös toinen funktio, jonka avulla lämpötilatiedot tullaan hakemaan Google Sheets -taulukosta (kuva 15). Lämpötilatiedot haetaan taulukosta getValues-funktion avulla, joka palauttaa tiedot moniulotteisessa arrayssa.

```
function doGet(e) {
  return HtmlService.createHtmlOutputFromFile('WebAppChart');
}

function getData() {
  var url = "https://docs.google.com/spreadsheets/**********";
  var ss = SpreadsheetApp.openByUrl(url);
  var tempsSheet = ss.getSheetByName("Taulukko1");
  var getLastRow = tempsSheet.getLastRow();

  return tempsSheet.getRange(720, 2, 720, 13).getValues();

}
```

Kuva 15. Google Apps Script -tiedoston funktiot.

HTML-tiedosto tehdään normaalisti, mutta script-elementissä luodaan Chart.js:än avulla viivakaaviot, jotka tulostetaan sivulle body-elementissä. Että Chart.js-kirjastoa voidaan käyttää, sen linkki täytyy asettaa head-elementtiin. Samalla asetetaan myös linkki Chart.js:än lisäosaan, jolla taulukoita pystytään zoomaamaan. Zoomaus-lisäosaa varten tarvitaan myös hammer.js-kirjastoa. Se on avoimen lähdekoodin JavaScript-kirjasto, jonka avulla tunnistetaan kosketusnäytöllä sekä kursorilla tehdyt liikkeet. Myös Hammer.js-kirjasto täytyy linkittää tiedostoon head-elementissä.

Verkkosovelluksen näkymä tehdään käyttämällä flexbox-asettelumoduulia. Flexboxia käyttämällä verkkosovelluksesta saadaan helposti tehtyä responsiivinen. Koko sivu rakennetaan yhden div-elementin sisään, joka on asetettu CSS-asetuksista flexbox-containeriksi. Kyseinen div-elementti toimii säiliönä muille elementeille. Se jaetaan kahteen osaan kahden div-elementin avulla. Vasemmalle puolelle lisätään jokaiselle anturille oma viivakaavio, josta nähdään niiden mittaamat lämpötila-arvot viimeisen tunnin ajalta. Jokainen viivakaavio piirretään omaan div-elementtiinsä. Nämä elementit asetellaan sivulle käyttämällä CSS grid -asettelua (kuva 16). Grid-asetuksen avulla voidaan luoda responsiivisia ruudukkoasetteluita. Oikeanpuoleiseen elementtiin lisätään yksi iso viivakaavio, josta nähdään lämpötilatiedot viimeisen 24 tunnin ajalta. Div-elementit määrittävät vain kaavion paikan sivulla, itse kaavion piirtäminen tapahtuu canvas-elementin avulla. Jokaiselle kaaviolle luodaan oma canvas-elementti. Canvas-elementteihin lisätään myös id-attribuutti, jonka avulla kaaviot piirretään oikeisiin elementteihin.

```
<div class="col-sidebar">

    <div class="grid-container">

        <div class="grid-item">
            <h3>patteri meno:</h3>
            <h4 id="new-data1"> </h4>
            <div><canvas id="hourChart1" width="170px" height="90px"></canvas></div>
        </div>

        <div class="grid-item">
            <h3>lattia meno:</h3>
            <h4 id="new-data2"> </h4>
            <div><canvas id="hourChart2" width="170px" height="90px"></canvas></div>
        </div>
```

Kuva 16. HTML-ruudukko viivakaavioille.

Script-elementissä suoritetaan komento google.script.run.withSuccessHandler. Sen avulla kutsutaan funktiota Apps Script -tiedostosta, joka hakee lämpötilatiedot Google Sheets -taulukosta. Kun funktio on palauttanut lämpötilatiedot HTML-tiedostolle, suoritetaan script-elementtiin tehty funktio, joka piirtää viivakaaviot canvas-elementteihin. Funktion alussa luodaan 13 array-muuttujaa, joihin tallennetaan jokaisen lämpötila-anturin mittaamat arvot, sekä kellonaika, jolloin arvot on mitattu. Moniulotteinen array käydään läpi forEach-metodin avulla, joka siirtää jokaisen anturin arvot omaan muuttujaansa push-metodia käyttämällä (kuva 17). Kun tiedot ovat omissa array-muuttujissaan, niitä pystytään käyttämään Chart.js-kaavioissa.

```
tempData.forEach(function(item, index)
{
  label.push(item[0]);
  data1.push(item[1]);
  data2.push(item[2]);
  data3.push(item[3]);
  data4.push(item[4]);
```

Kuva 17. Tietojen siirto moniulotteisesta arraysta erillisiin array-muuttujiin.

5.4.1 Chart.js viivakaaviot

Ensimmäinen viivakaavio aloitetaan tekemällä sille oma muuttuja. Muuttujassa määritetään canvas-elementti johon kaavio tullaan piirtämään, sekä kerrotaan piirroksen olevan kaksiulotteinen. Tämän jälkeen tehdään uusi kaavio-objekti (kuva 18). Sen argumenteiksi asetetaan aiemmin luotu muuttuja sekä objekti, johon tullaan lisäämään kaikki kaavion tiedot. Ensimmäiseksi objektissa asetetaan kaavion tyyppi, joka tässä tapauksessa on viivakaavio. Tämän jälkeen määritetään datasets- ja options-objektit. Datasets-objektissa ilmoitetaan mitä tietoja kaaviossa tullaan käyttämään. Jos kaaviossa käytetään useampia tietojoukkoja, luodaan array, johon jokaiselle tietojoukolle tehdään oma objekti. Options-objektia käytetään kaavion vakioasetusten muuttamiseen. Sen avulla muutetaan esimerkiksi kaavion ulkonäköä, maksimi- ja minimiarvoja, sekä lisätään kaavioon zoomaus-lisäosa.

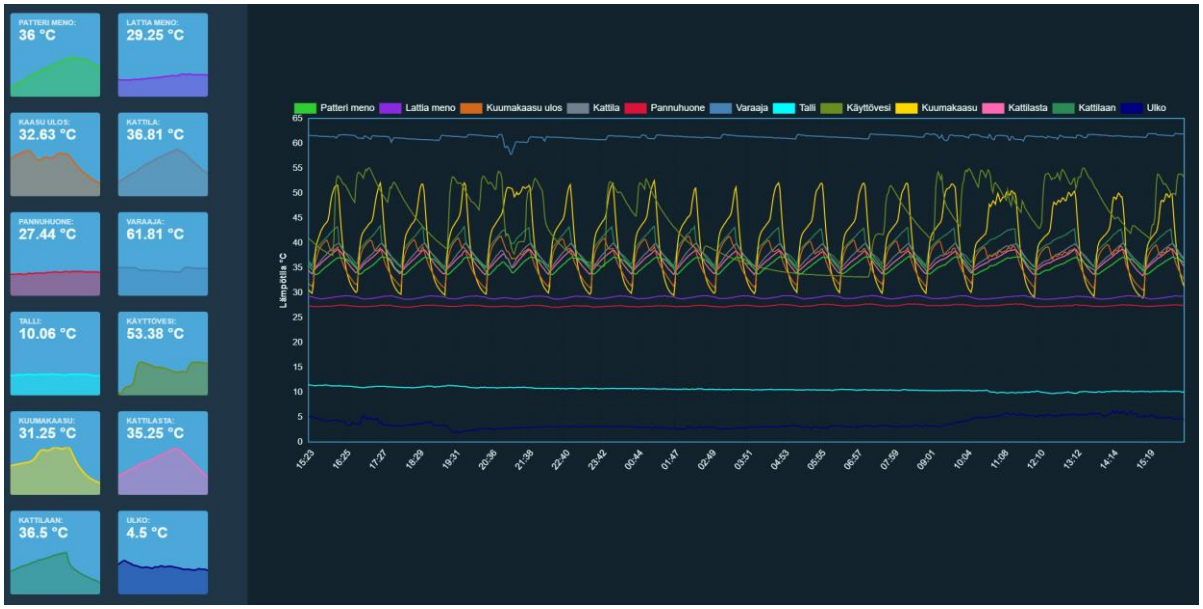
```
var ctx = document.getElementById("lineChart").getContext("2d");
var lineChart = new Chart(ctx, {
type: 'line',
data: {
    labels: label,
    datasets: [
    {
        label: 'Patteri meno',
        fill: false,
        borderColor: '#32CD32',
        backgroundColor: '#32CD32',
        borderWidth: 2,
        pointRadius: 0,
        hoverRadius: 5,
        hitRadius: 6,
        data: data1,
    },
```

Kuva 18. Viivakaavio-objektin luominen.

Seuraava vaihe on luoda pienet kaaviot, jotka näyttävät anturien lämpötilatiedot viimeisen tunnin ajalta omissa viivakaavioissaan. Ensiksi luodaan kopiot kaikista lämpötilatietojen array-muuttujista. Kun tiedot on kopioitu, niistä poistetaan tietoja niin, että jäljelle jää vain viimeisen tunnin ajalta mitatut arvot. Lämpötilatietojen poistaminen tehdään splice-metodilla. Kun viivakaaviossa käytetyt lämpötilatiedot on saatu tehtyä, täytyy viivakaavion värit, nimi sekä canvas-elementti tallentaa muuttujiin. Tämän jälkeen voidaan kutsua funktiota, joka piirtää kaaviot (kuva 19). Funktiota kutsuessa sen argumenteiksi asetetaan juuri luomamme muuttujat. Pienten viivakaavio-objektien luominen on asetettu funktioon, sillä kaikki 12 viivakaaviota ovat samannäköisiä. Funktion parametreiksi on asetettu viivakaaviossa käytettävät tietojoukot, jonka avulla jokaisen anturin tiedoilla piirretään oma viivakaavio. Kun funktio on suoritettu jokaisen anturin tiedoilla, verkkosovellus on valmis (kuva 20).

```
var hourData1 = [...data1];
hourData1.splice(0, 690);
var data1Label = 'Patteri meno';
var data1Color = '#32CD32';
var data1ColorBg = 'rgba(50, 205, 50, 0.4)';
var ctx1 = document.getElementById("hourChart1").getContext("2d");
var hourChart1 = createChart(ctx1, data1Label, hourData1, data1Color, data1ColorBg);
```

Kuva 19. Funktiolle lähetettävät viivakaavion tiedot.



Kuva 20. Valmis verkkosovellus.

6 YHTEENVETO

Opinnäytetyön aiheena oli luoda IoT-projekti, jonka lopputuloksena olisi verkkosovellus, jota voidaan käyttää lämpötila-arvojen seuraamiseen. Työssä onnistuttiin toteuttamaan suunniteltua vastaava kokonaisuus. Uutena kokemuksena minulle tuli laitteiden kytkennät ja niiden ohjelmointi. Kytkennöissä ja laitteen ohjelmoinnissa kuitenkin onnistuttiin, jonka tuloksena antureiden arvoja pystyttiin lähettämään pilvipalveluun langattoman verkon avulla. Työssä onnistuttiin myös kehittämään käyttöliittymä, josta lämpötila-arvoja päästiin seuraamaan verkkoselaimen avulla. Verkkosovelluksen kehitys oli minulle jo entuudestaan tuttua, mutta siitä huolimatta työtä tehdessä vastaan tuli uusia asioita.

Opinnäytetyön projektia tehdessä opin paljon uusia asioita. Opin muun muassa IoT-kehitysalustojen toimintaperiaatteesta, ja niiden loputtomista käyttömahdollisuuksista. Tutustuin työtä tehdessä myös Arduino IDE -kehitysympäristön käyttöön, ja kehitysympäristössä käytettyyn C++-ohjelmointikieleen. Verkkosovelluksia kehittäessä tutuksi tuli myös Googlen oma Apps Script -ohjelmointikieli, sekä sen sisäänrakennetut kirjastot.

Toteutettu ratkaisu on erittäin käytännöllinen, ja sitä pystyy hyödyntämään monessa eri tapauksessa, jossa lämpötilan seuraaminen on olennaista. Hyvä esimerkki tästä voisi olla kylmiön, tai jonkin muun varaston lämpötilan seuraaminen. Projektissa toteutettua ilma-vesilämpöpumpun lämpötilojen seurantaprojektia voitaisiin jatkaa lisäämällä järjestelmään sähköinen säätöventtiili. NodeMCU voitaisiin ohjelmoida säätämään patteriverkoston kiertoveden lämpötilaa säätöventtiilin avulla, mitattujen lämpötilatietojen perusteella.

LÄHTEET

Ahlin, T. 2017. Data visualization with Chart.js: An introduction. Viitattu 4.10.2021. https://tobiasahlin.com/blog/introduction-to-chartjs/

Arduino. 2021. Arduino Software (IDE). Viitattu 30.9.2021. https://www.arduino.cc/en/Guide/Environment

Arduino. 2021. Dallas Semiconductor's 1-Wire Protocol. Viitattu 29.3.2021. https://playground.arduino.cc/Learning/OneWire/

Arduino. 2021. Download the arduino IDE. Viitattu 30.9.2021. https://www.arduino.cc/en/Main/Software

Components101. 2020. NodeMCU ESP8266. Viitattu 22.9.2021. https://components101.com/development-boards/nodemcu-esp8266-pinout-features-and-datasheet

ElProCus. 2021. Temperature Sensors – Types, Working & Operation. Viitattu 29.3.2021. https://www.elprocus.com/temperature-sensors-types-working-operation/

Google. 2020. Web Apps. Viitattu 7.10.2021. https://developers.google.com/apps-script/guides/web

Google. 2021. Overview of Google Apps Script. Viitattu 5.10.2021. https://developers.google.com/apps-script/overview

LastMinuteEngineers. 2021. Insight Into ESP8266 NodeMCU Features & Using It With Arduino IDE. Viitattu 22.9.2021. https://lastminuteengineers.com/esp8266-nodemcu-arduino-tutorial/

Maxim Integrated Products. 2019. DS18B20 datasheet. Viitattu 29.3.2021. https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf

Microcontrollerslab. 2021. DS18B20 with ESP8266 NodeMCU (Single/Multiple): Display Readings on OLED. Viitattu 26.10.2021. https://microcontrollerslab.com/ds18b20-esp8266-nodemcu-single-multiple-display-readings-oled/

Nolledo, M. 2020. What is Google Drive? A guide to navigating Google's file storage service and collaboration tools. Viitattu 6.10.2021. https://www.businessinsider.com/what-is-google-drive-guide?r=US&IR=T

Procure international inc. 2021. Dallas Semiconductor -Maxim Integrated Company Overview. Viitattu 29.3.2021. https://www.procureinc.com/manufacturer/Dallas_Semiconductor_-Maxim_Integrated/

TheMakersWorkbench. 2019. What Is A NodeMCU Anyway? You're About To Find Out! Video 6.3.2019. Viitattu 22.9.2021. https://www.youtube.com/watch?v=IHocU-VqsF0&t=423s&ab_channel=TheMakersWorkbench