SeAMK
SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Antti Hunnakko

Esineiden internet palvelut ja sovellusintegraatio

Opinnäytetyö
Syksy 2022
Insinööri (AMK), Automaatiotekniikka

SEINÄJOEN AMMATTIKORKEAKOULU

Opinnäytetyön tiivistelmä

Tutkinto-ohjelma: Insinööri (AMK), Automaatiotekniikka

Suuntautumisvaihtoehto: Koneautomaatio

Tekijä: Antti Hunnakko

Työn nimi: Esineiden internet palvelut ja sovellusintegraatio

Ohjaaja: Marko Hietamäki

Vuosi: 2022 Sivumäärä: 47 Liitteiden lukumäärä: 0

Opinnäytetyössä tutustuttiin esineiden internettiin, ja siihen liittyviin pilvipalveluihin. Tarkoituksena se, että lukija saa hyvän ymmärryksen siitä, mitä esineiden internet on, ja miten pilvipalveluita käytettään.

Työssä tutustuttiin tarkemmin kahteen pilvipalveluntarjoajaan: Microsoft Azure, sekä Amazon AWS IoT. Näiden pilvipalvelujen käyttöönottoon on työssä tehty kuvitettu ohjeistus, jota seuraamalla pystyy lukija aloittamaan näiden palvelujen käytön, ja samalla tutustumaan palvelun tarjontaan.

Työn lopussa käydään läpi palveluiden hyviä ja huonoja puolia. Molemmista palveluista löytyy hyviä ominaisuuksia, joilla esineiden internet projekteja on helppo toteuttaa. Azuren huonoksi puoleksi paljastuu sellaisten laitteiden yhdistämisen hankaluus, jotka eivät tue ohjelmistonkehityspaketteja. AWS IoT palvelun ilmaisversiosta taas puuttuu täysin datan visualisointi reaaliajassa.

Työn lopussa esitetään myös, miten tätä työtä apuna käyttäen, voitaisiin kehittää parempi esimerkkisovellus, joka pystyttäisiin nopeasti yhdistää pilvipalveluun, ja luoda sinne laitteita ja dataa, joka vastaisi jonkin yrityksen todellisia laitteita, ja näin esittämään yritykselle esineiden internet mahdollisuuksia.

[1] Asiasanat: esineiden internet, Microsoft Azure, AWS IoT, IoT, pilvipalvelu

SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Thesis abstract

Degree programme: Automation Engineering

Specialisation: Machine Automation

Author: Antti Hunnakko

Title of thesis: Internet of things and application integration

Supervisor: Marko Hietamäki

Year: 2022 Number of pages: 47 Number of appendices: 0

In the thesis it was studied what Internet of Things, and its cloud services are. The purpose of the thesis was to give the reader an understanding of these topics, and how to use them.

The study focused more closely on two of the cloud platform providers: Microsoft Azure, and Amazon AWS IoT. During the thesis project a guide on how to start using these platforms was developed. The aim of the guide was to give information on how to start using these platforms, and what kind of services the platforms offer.

At the end the advantages and disadvantages of these services were studied. It was discovered that both services have features that make projects relating to the Internet of Things easy to implement. One fault that Azure had, was the difficulty of connecting devices, which do not support the use of software development kits. It was also discovered that the free version of AWS IoT does not enable the visualization of data in real time.

At the end it was explained also how a better example program could be developed, with the help of the results of the thesis. The example program was planned so that it would be easy to connect it to the could platform, and there would be a simple way to create devices, and simulate data, which would represent the data a company would generate from their physical devices. This would make it possible to show companies the power of the Internet of Things, in a way they can relate to.

[1] Keywords: Internet of Things, Microsoft Azure, AWS IoT, IoT, cloud platform

SISÄLTÖ

Kuvio- ja taulukkoluettelo

Käytetyt termit ja lyhenteet

IoT	Internet of Things, eli Esineiden internet.
Pilvipalvelin	Etänä toimivaa datakeskus.
Pilvipalvelualusta	Pilvipalvelin, johon on rakennettu käytettävyyttä.
Pilvipalvelu	Pilvipalvelualustan sisällä oleva yksittäinen palvelu.

1 JOHDANTO

Opinnäytetyössä tutustutaan teollisuudessa nopeasti kehittyvään esineiden internettiin, ja sen ympärille rakennettuihin palveluihin, joilla esineiden internetiä voidaan toteuttaa teollisuuskäytössä. Työssä käydään läpi, mitä esineiden internetillä tarkoitetaan, ja mitä sillä pystytään rakentamaan. Lisäksi selvitetään, millaisia työkaluja eri pilvipalvelut tarjoavat, ja miten niitä pystytään hyödyntämään esineiden internetin teollisuudessa.

Jotta esineiden internet palvelujen käytön aloitus olisi helpompaa, käydään työssä yksityiskohtaisesti läpi vaiheet palvelujen käytön aloittamiseen. Ohjeita seuraamalla lukija tutustuu palveluun, ja saa peruskäsityksen siitä, miten sitä käytetään. Työssä esineiden internet laitteet on simuloitu esimerkkiohjelmilla. Työssä on käyty läpi myös näiden ohjelmien käyttö tietokoneella, ja tutkittu ohjelmien rakennetta, jotta saadaan ymmärrys siitä, miten ohjelmat toimivat. Ohjeistus on kuvitettu, jotta sitä on helpompi seurata.

Lopuksi käydään läpi palveluiden vertailua, ja tutkitaan, sopiiko toinen palvelu paremmin tietynlaiseen käyttöön, ja onko toisessa palvelussa toteutettu tietyt asiat paremmin kuin toisessa.

2 MIKÄ ON ESINEIDEN INTERNET?

Esineiden internetillä teollisuudessa tarkoitetaan fyysisten laitteiden ja informaatio teknologian yhteenliittymistä (Collin & Saarelainen, 2016, s. 18–20). Tällä pyritään tekemään loppukäyttäjän arjesta helpompaa esittämällä kerättyä dataa selkeässä muodossa. Pystytään esimerkiksi tekemään ennakoivaa huoltoa, jossa kerätystä datasta lasketaan koska huolto tietyille komponenteille tulisi suorittaa.

Teollisuudessa käytössä olevien koneiden ja laitteiden, kuten robottien tai levyntyöstökoneiden, ja niihin liitettyjen linjastojen anturit ovat esineiden internetin fyysinen puoli (Collin & Saarelainen, 2016, s. 18–20). Näistä fyysisistä laitteista pyritään keräämään käyttäjälle hyödyllistä dataa, joka siirretään esineiden internetin digitaaliseen puoleen.

Esineiden internetin digitaalinen puoli koostuu esimerkiksi fyysisistä laitteista kerätystä datasta, pilvipalveluista johon dataa kerätään ja pilvipalveluihin rakennettavista ohjelmistoista, joilla kerättyä dataa pyritään hyödyntämään (Collin & Saarelainen, 2016, s. 18–20).

2.1 Esineiden internet pilvipalvelut

Pilvipalveluilla tarkoitetaan etänä toimivaa datakeskusta, jonka ominaisuuksiin kuuluu datan turvallinen säilyttäminen ja erilaiset sovellukset, joita näillä datakeskuksen servereillä pystytään toteuttamaan (Cloudflare, i.a.).

Useat palveluntarjoajat tarjoavat myös enemmän kohdennettuja esineiden internet tarkoituksiin luotuja pilvipalveluita. Pilvipalveluiden tarjoajat eivät ole kiinnostuneet pilveen syötetyn datan sisällöstä, vaan palveluntarjoajien liiketoiminta perustuu näiden palvelualustojen, ja niiden sisällä olevien alustojen, joiden päälle sovelluksia pystytään rakentamaan, tarjoamiseen loppukäyttäjälle (Martinsuo ym., 2017, s. 23).

Esineiden internettiin kohdentuneet pilvipalvelut voivat tarjota jo valmiiksi pilvessä olevia sovelluksia, joita voi hyödyntää datan esittämisessä ja analysoinnissa. Esimerkiksi kaavioiden luominen saadusta datasta, ja sen esittäminen verkossa, kun palveluun kirjaudutaan.

2.2 Esineiden internet laitteet

Esineiden internettiä pystytään hyödyntämään lähes kaikilla laitteilla, jotka ovat yhteydessä internettiin (Bexell, 2020, s. 10). Tällaisia esineiden internet laitteita voivat olla esimerkiksi uuden malliset hehkulamput, joissa on sisään rakennettua teknologiaa, jolla ne saadaan verkkoon, tai lämpömittari, joka on yhteydessä verkkoon. Tällaisia asioita löytyy esimerkiksi nykyaikaisesta älykodista, ja näissä laitteissa on usein käytössä langaton teknologia, kuten Bluetooth tai Wi-Fi.

Kun esineiden internetiä käytetään teollisuudessa, käytetään siitä usein nimitystä ”Teollinen internet” (Collin & Saarelainen, 2016, s. 30). Tällä tarkoitetaan kuitenkin samaa asiaa kuin esineiden internetillä, se on vain yksilöity enemmän teollisuuteen ja sen laitteistoihin. Teollisuudessa myös laitemäärät nousevat suuremmiksi kuin älykodeissa. Teollisuudessa voi olla tuhansia antureita, kun älykodissa antureita ja ohjattavia laitteita on vain kymmeniä.

Teollisessa internetissä laitteilla tarkoitetaan ohjelmoitavia logiikka yksiköitä ja niihin liitettyjä antureita ja muita laitteita. Teollisuudessa käytettävillä antureilla ei yleisesti itsessään ole ominaisuutta yhdistää internettiin, mutta koska anturit on kytketty ohjelmoitavaan logiikkaan, voidaan näistä laitteista saatavaa tietoa lähettää pilvipalveluun. Teollisuudessa tällaiset laitteet on yleensä johdotettu, jolloin langattoman teknologian käyttö ei ole niin suuressa asemassa.

Teollisuudessa ohjelmoitavan logiikan rinnalla on yleensä valvomo ohjelmisto, jossa logiikalta saatavaa dataa esitetään, ja mistä logiikkaan kytkettyjä laitteita ohjataan (Iot.nxt, 2018). Valvomo ohjelmistoissa on monia mahdollisuuksia datan laskentaan, ja sen esittämiseen. Tällaisilta valvomo ohjelmistoilta kuitenkin puuttuu ominaisuus laajentua yhtä suuresti, kun teollisen internetin pilvipalveluilta, vaikka molempien tarkoituksena on datan keruu, käsittely ja esittäminen käyttäjälle.

Teollisella internetillä, kuten valvomo-ohjelmistollakin, pyritään siirtämään data helpommin saatavilla olevaan paikkaan, eli teollisen internetin tapauksessa pilveen. Tällöin ohjelmoitava logiikka toimii vain viestin välittäjänä laitteiston ja pilvipalvelun välillä, ja datan analysointi jätetään pilveen, jossa siihen päästään helposti käsiksi (Iot.nxt, 2018).

2.3 Laitteiden kommunikointiprotokolla

Kommunikointiprotokollalla tarkoitetaan sitä, miten laitteet puhuvat toisilleen ja siirtävät dataa (Bexell, 2020, s. 49). Esineiden internetissä yksi suosituimmista kommunikaatio protokollista on MQTT (Message Queuing Telemetry Transport). Tämä protokolla on rakennettu TCP/IP, eli internet protokollan päälle ja se on kevyt, jolla tarkoitetaan sitä, että sillä pystytään lähettämään ja vastaanottamaan suuria määriä viestejä ilman verkon hidastumista. Tämä kevyt luonne on toteutettu esimerkiksi siten, että viestien sisältämä datan määrä on pyritty minimoimaan.

MQTT-protokolla toimii julkaise/tilaa-periaatteella (Bexell, 2020, s. 49). Protokollassa on kaksi päätekijää: asiakas ja välittäjä. Asiakkaana järjestelmässä toimivat kaikki laitteet, kuten ohjelmoitavat logiikat, infonäytöt, tai yksittäiset anturit. Kaikki asiakkaat ovat yhteydessä välittäjään, ja julkaisevat sinne tietoa, tai tilaavat sieltä tietoa. Asiakkaat eivät tiedosta toisiaan, vaan tunnistavat ja ovat yhteydessä vain välittäjään. Välittäjänä järjestelmässä toimivat esimerkiksi esineiden internet pilvipalvelut. Välittäjä on yhteydessä kaikkiin asiakkaisiin, eli laitteisiin ja ohjaa asiakkailta saadut viestit eteenpäin toisille asiakkaille, sinne asetettujen sääntöjen mukaan.

Asiakkaiden lähettämät viestit on jaettu aiheisiin (Bexell, 2020, s. 49). Esimerkiksi lämpötilaa mittaava anturi voi julkaista lämpötilalukeman aiheeseen ”tehdas/varasto/lämpötila”. Tämä viesti välitetään välittäjälle, ja samaan välittäjään yhteydessä oleva infonäyttö voi tilata tämän saman aiheen ” tehdas/varasto/lämpötila”, jolloin välittäjä siirtää saamansa datan infonäytölle. Saman aiheen voivat tilata kaikki järjestelmän laitteet. Välittäjän sisällä olevat säännöt voivat perustua esimerkiksi viestin tyyppiin tai sen sisältöön. Tällöin, jos lämpötila anturi lähettää viestin, jossa on tietoa jostain muusta kuin itse lämpötila-arvosta, ja infonäyttö ei tarvitse tätä dataa, ei välittäjä lähetä tätä dataa infonäytölle.

MQTT-protokolla ei itsessään sisällä viestien salausta, ja viestit voidaan salata käyttämällä käyttäjän itse valitsemaansa salaustapaa (Collin & Saarelainen, 2016, s. 187). Koska MQTT-protokolla on rakennettu TCP/IP protokollan päälle, ja koska TCP/IP käyttää vahvaa TLS-salausta, voidaan tätä samaa salausta käyttää myös MQTT-protokollaa käytettäessä.

3 KÄSITELTÄVÄT PALVELUNTARJOAJAT

Opinnäytetyön vertailuun on valittu kaksi esineiden internet pilvipalveluntarjoajaa. Valintaperusteena on tarkasteltu Suomessa toimivien yritysten palveluntarjontaa, ja sieltä on poimittu kaksi yleisimmin tarjottua tai suositeltua palvelualustaa. Yritykset, joita on tarkasteltu ovat Telia, Elisa, Advania sekä Wapice. On myös tarkasteltu ohjelmoitavia logiikoita valmistavia yrityksiä, kuten Beckhoff ja Siemens, ja huomioitu, minkä yrityksien palveluita nämä yritykset suosittelevat esineiden internet palveluiksi.

Molemmilla palveluntarjoajilla esineiden internet palvelut on jaettu palveluihin, kuten esimerkiksi palvelu, jolla laitteita pystytään yhdistämään, palvelu, jolla laitteiden dataa käsitellään, ja palvelu, jolla voidaan rakentaa informaatiota esittävä näyttösivusto, eli visualisoimaan dataa.

Molemmilta palveluntarjoajalta löytyy myös ilmainen kokeiluversio. Kokeiluversiot ovat rajoitettu lähetettyjen ja vastaanotettujen viestien määrään. Viestejä kokeiluversioissa saa yleensä olla muutamia kymmeniä tuhansia kuukaudessa tai muutamia tuhansia päivässä. Vaikka viestien määrä vaikuttaa suurelta, tulee viestimäärä nopeasti vastaan, jos usea laite lähettää viestin esimerkiksi joka viides sekunti. Tällöin jo yksi laite lähettää yli 17 tuhatta viestiä päivässä. Palveluiden maksullisissa versioissa maksut palveluista kertyvät yleensä laitteiden ja lähetettyjen viestien määrästä, datan varastoinnista, ja datan käsittelemiseen hyödynnetyistä palveluista.

3.1 Microsoft Azure

Microsoft Azure on yhdysvaltalaisen Microsoftin vuonna 2010 julkaisema pilvipalvelualusta. Pilvipalvelualustalta löytyy jo yli 200 erilaista palvelua, kuten tietokantajärjestelmät, virtuaalitietokoneet, tietojen säilytys ja tietysti esineiden internet palvelut (Vailshery, 2022).

Microsoft Azure tarjoaa tietysti myös esineiden internettiin kohdenettua pilvipalvelua. Nämä palvelut ovat myös jaettuna erikseen esineiden internet palveluihin, sekä teollisen internetin palveluihin, vaikka molemmilla pystytään toteuttamaan samoja asioita.

Azuren esineiden internet palvelujen päällimmäisenä osana on Azure IoT Central. Palvelussa on kaikki, mitä esineiden internetin käyttöön tarvitaan, helposti yhdessä paikassa. Tässä palvelussa pystytään luomaan ja lisäämään laitteita, ja tarkastelemaan niiden data, ja

ohjaamaan sitä. Kaikki tämä löytyy helposti yhdestä paikasta. Iot Central luo myös automaattisesti Azure IoT Hub palvelun, joka toimii esineiden internetin tiedon välittäjänä (Microsoft, i.a.-a).

Tämän lisäksi löytyy muita palveluita, kuten Azure Maps, jonka kautta pystytään integroimaan karttoja, ja visualisoimaan esimerkiksi laitteiden sijaintia kartalla (Microsoft, i.a.-a).

Kolmas voimakas Azuren tarjoama palvelu on Azure Digital Twin, joka helpottaa tehtaan visualisointia pilvessä ja auttaa tehtaiden tai laitteiden monitoroinnissa (Microsoft, i.a.-a).

Azuren tarjoamat palvelut ovat erittäin esineiden internet keskeisiä, ja Azurelta löytyy useita palveluita, jotka ovat tärkeitä esineiden internet laitteiden hallintaan (Microsoft, i.a.-a). Koska Microsoftin pilvipalvelut ovat keskitettynä samaan paikkaan, pystytään esimerkiksi tietokantajärjestelmä integroimaan projekteihin suoraan Azuren palveluiden kautta. Myös palveluista löytyvä dokumentaatio on hyvää ja yksityiskohtaista, ja eri palveluiden käyttöönotosta löytyy hyviä esimerkkejä Microsoftin omista resursseista.

Microsoft tarjoaa esineiden internet palveluiden käyttäjille ilmaisen kokeiluversion, jossa palvelun maksullisia osioita voi käyttää veloituksetta vuoden (Microsoft, i.a.-b). Ilmaisen käyttäjätilin luomisen yhteydessä Microsoft tarjoaa 200 dollarin arvosta luottoa tilille, joka voidaan käyttää palveluissa. Kun tämä luotto on kulutettu loppuun, alkaa palveluiden käyttö olla maksullista. Tämän kokeiluversion tarkoituksena on antaa käyttäjälle mahdollisuus tutustua palveluun, ja luoda siellä pienen skaalan projekteja tai isojen projektien prototyyppejä.

3.2 Amazon Web Services

Amazon Web Services, eli AWS, on yhdysvaltalaisen Amazonin tarjoama pilvipalvelualusta (Amazon, 2015). Amazonin AWS pilvipalveluihin lisättiin vuonna 2015 AWS IoT, eli Amazonin pilvipalvelualustan oma esineiden internettiin keskittyvä palvelu.

Tähän palveluun kuuluu esimerkiksi AWS IoT Core, jossa pystytään lisäämään laitteita, ja hallinnoimaan viestikulkua laitteiden välillä. AWS IoT Core toimii esineiden internetin viestien välittäjänä (Amazon, i.a.-b).

Toinen esineiden internetin palvelu on AWS IoT Analytics, joka auttaa esineiden internet laitteista saadun datan analysoinnissa, ja palvelu on kehitetty esineiden internet laitteista saatu data mielessä (Amazon, i.a.-b). Palvelussa pystytään analysoimaan dataa, ja poistamaan virheitä, kuten virheellisiä viestejä, laitteelta saatuja vääriä arvoja tai puutteellista dataa. Dataa voidaan myös muokata matemaattisilla kaavoilla tai siihen voidaan liittää ylimääräistä dataa, kuten laitteen tietoja, jotta datan lähde on paremmin selvillä.

Kolmantena esimerkkinä palveluista on AWS IoT Things Graph, jonka tarkoituksena on helpottaa esineiden internet laitteiden visualisoinnissa ja niiden lähettämän datan kulusta (Amazon, i.a.-a). Tämä voi auttaa järjestelmien suunnittelussa suuresti, kun laitteiden ja palveluiden välinen yhteys on visualisoitu, sen sijaan, että laitteet vain näkyvät yhdessä listassa, josta on vaikea nähdä, mihin prosesseihin kyseinen laite on liitetty. Laitteet ja palvelut näkyvät kaaviossa palikoina, ja niiden välinen yhteys näkyy laitteiden välille piirretyllä viivalla.

Esineiden internet palveluihin kuuluu myös AWS IoT SiteWise, jolla pystytään luomaan visualisointeja kerätystä datasta, ja tarkastelemaan sitä reaaliajassa (Amazon, i.a.-b). Tämä ominaisuus on tärkeä esineiden internet laitteissa, koska datan visualisointi tekee siitä helposti luettavaa.

Amazon tarjoaa myös itse valmistamaansa AWS IoT Button laitetta, joka on fyysinen painike, jossa on esineiden internet valmiudet (Amazon, i.a.-b). Tätä nappia voisi käyttää esimerkiksi valopainikkeena kodin automaatiossa.

AWS pitää sisällään myös monia muita esineiden internetissä tarvittavia palveluita, kuten tietokantajärjestelmiä, ja datan käsittelyn ja sen esittämiseen tarkoitettuja järjestelmiä (Amazon, i.a.-b). Koska kaikki AWS-palvelut ovat samassa paikassa, on niiden integrointi helppoa, ja suuretkin projektit onnistuvat käyttämällä vain AWS-palveluita.

Amazon tarjoaa ilmaista kokeiluversiota osasta esineiden internet-palveluistaan (Amazon, i.a.-a). AWS IoT Core ilmaisversiolla pystytään lähettämään ja vastaanottamaan jopa 250 000 viestiä kuukaudessa. Ilmainen kokeilu kuitenkin kestää vain vuoden, jonka jälkeen palvelusta tulee maksullinen.

Ilmaisesta kokeilusta kuitenkin puuttuu aiemmin mainittu AWS IoT SiteWise, joka on keskeinen osa esineiden internet-palveluiden toteuttamista (Amazon, i.a.-a). Datan visualisointiin

palvelussa täytyy siis etsiä toinen järjestelmä. Tähän voidaan käyttää esimerkiksi AWS QuickSight-palvelua, jossa dataa pystytään visualisoimaan. Tämän palvelun kokeiluversio on ilmainen vain kuukauden. Iot Coren sisällä voidaan käyttää myös Jupyter kansiota, jossa datasta voidaan luoda kaavioita, mutta tämä ei ole yhtä hyvä ratkaisu, kun SiteWise tai QuickSight. Jupyter kansioita voidaan käyttää niin kauan, kun Iot Coren ilmainen jakso päättyy.

4 SOVELLUSTEN INTEGROINTI PALVELUIHIN

Tässä osiossa käydään läpi, miten aiemmin esiteltyihin pilvipalveluihin yhdistetään sovellus, miten dataa lähetetään, ja miten sitä voidaan pilvipalvelussa hyödyntää. Lisäksi osiossa käydään läpi vaiheet palvelujen käytön aloittamista varten, jotta uuden käyttäjän on helppo aloittaa ja ymmärtää, mitä käytön aloitus vaatii.

4.1 Laitteiden simulointi esimerkkiohjelmalla

Työssä laitteet on simuloitu tietokoneohjelmistolla. Kukin palveluntarjoaja tarjoaa ohjelmistonkehityspaketteja (Engl. Source Developement Kit) useilla eri ohjelmointi kielillä, kuten C#, C++, Python ja Java. Näitä ohjelmistonkehityspaketteja voidaan käyttää, kun laitteita yhdistetään pilveen. Osa laitteista on mahdollista yhdistää suoraan MQTT-teknologiaa käyttäen, jos laite sitä tukee, ja pilvipalvelusta saadaan tarvittavat tiedot, kuten salasanat ja avaimet laitteen yhdistämiseen. Ohjelmistonkehityspakettia kuitenkin suositellaan käytettäväksi laitteilla, joissa se on mahdollista.

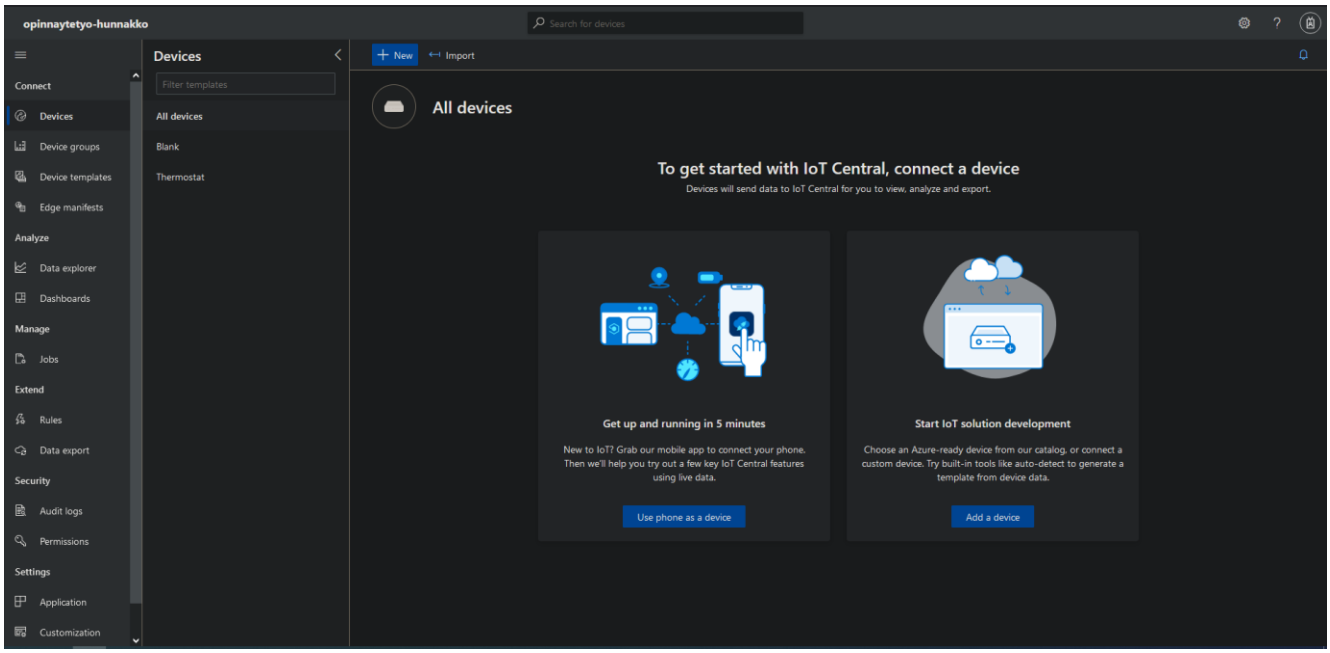
Ohjelmoitavien logiikkayksiköiden valmistajat ovat myös luoneet kirjastoja, joita käyttämällä projektit voidaan yhdistää pilvipalveluun MQTT-protokollaa käyttäen. Esimerkiksi laitevalmistaja Beckhoff tarjoaa kirjastoja pilvipalvelulle, kuten Microsoft Azure tai AWS IoT Core (Beckhoff, i.a.).

4.2 Microsoft Azure sovellusten integrointi

Työssä pilvipalveluun yhdistetään C#-ohjelmointikielellä kirjoitetulla esimerkkiohjelmistolla. Ohjelmisto yhdistää palvelun luotuun laitteeseen, ja sillä pystytään lähettämään sinne dataa. Palvelussa tätä dataa muokataan siten, että palvelussa data on esitetty selvästi. Lopuksi luodaan datan analysointi, sekä infonäyttö, jossa dataa voidaan visualisoida.

4.2.1 Resurssien valmistelu

Ensimmäisenä vaiheena on luoda ilmainen tili Azure IoT Central palveluun. Tämän jälkeen pystytään palveluun kirjautumaan sisälle. Pääsivulta voidaan luoda uusi esineiden internet applikaatio, kohdasta ”My apps”. Applikaation luonnin jälkeen pystytään se avaamaan, ja saavutaan applikaation pääsivulle, joka nähdään kuviossa 1.



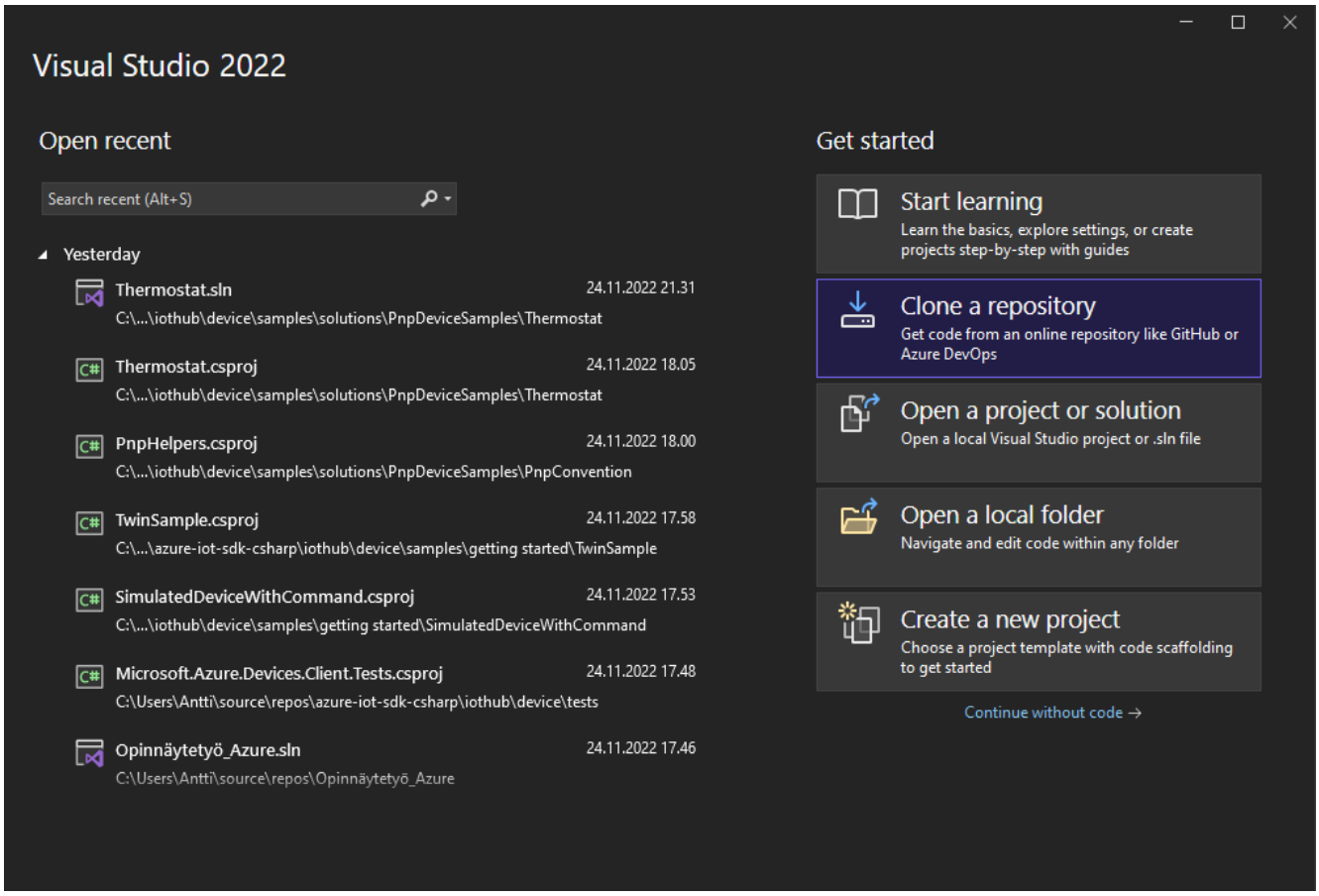
Kuvio 1. IoT Central applikaation päänäkymä

Päänäkymässä vasemmalla reunalla olevista valikoista tärkeitä ovat Devices eli laitteet, Device templates eli laite mallit, Data explorer eli datan analysointi, sekä Dashboard eli kojelauta, jossa pystytään luomaan visualisointia datalle.

Ennen kuin laitetta lisätään, luodaan tietokoneohjelma simuloimaan sitä. Microsoft tarjoaa valmiita mallisovelluksia, joita tarkastelemalla ja muokkaamalla pystytään nopeasti yhdistämään pilvipalveluun. Ohjelmointi tehdään Visual Studio 2022 -ohjelmalla, ja ohjelman suorittamiseksi täytyy .NET Core 6.0 olla asennettuna. Tämä voidaan asentaa Visual Studion kautta. .Net versio voidaan tarkistaa avaamalla komentokehote, ja syöttämällä siihen koodi ”dotnet –info”, joka tulostaa ikkunaan tiedot asennetusta versiosta.

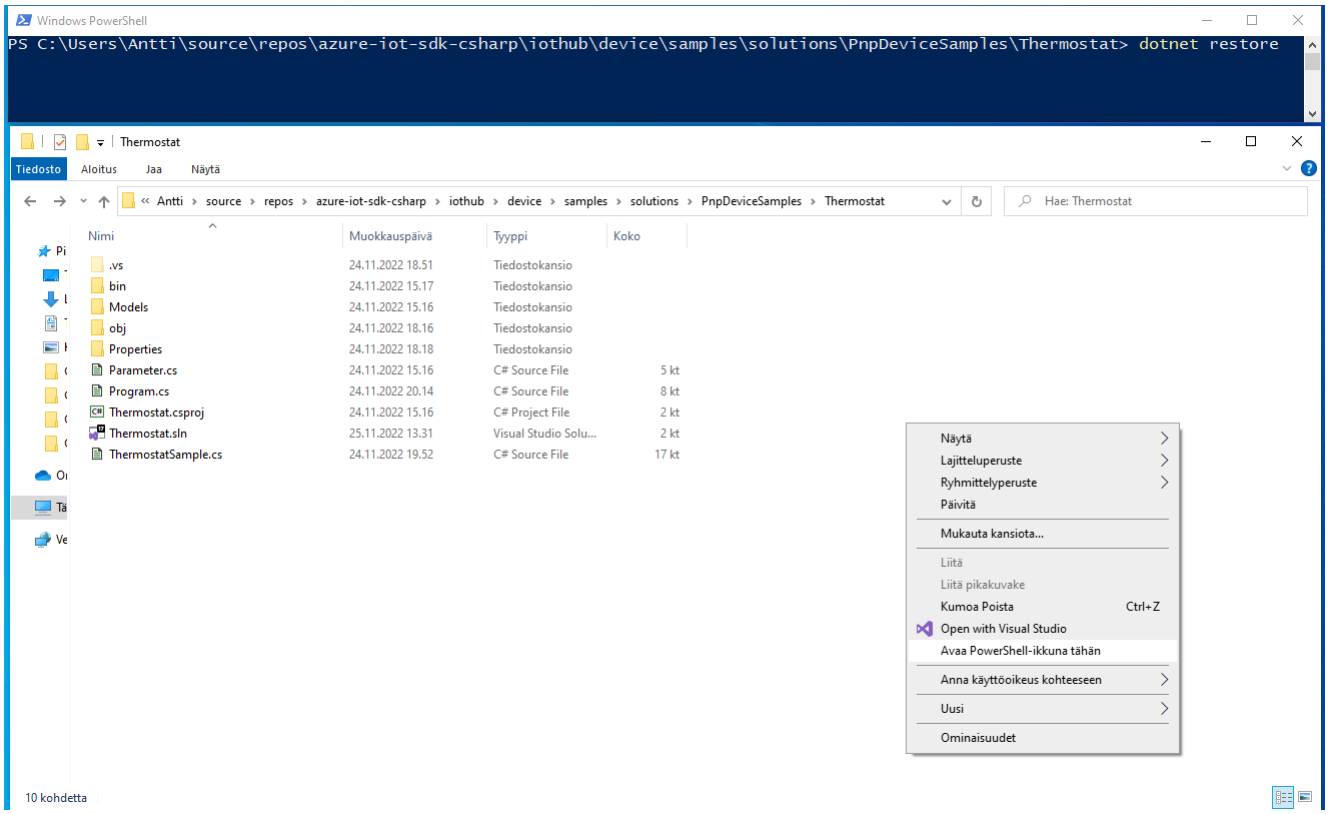
Ensin on ladattava ” Microsoft Azure IoT SDK for C# (.NET)”-ohjelmistonkehityspaketti, jonka mukana tulevat esimerkkiohjelmat, ja tarvittavat kirjastot. Tämä voidaan kopioida, eli ladata omalle tietokoneelle Visual Studion etusivulta tai suoraan GitHub-palvelusta. Ohjelmistonkehityspaketti löytyy osoitteesta: https://github.com/Azure/azure-iot-sdk-csharp.

Visual Studion aloitusnäytöltä, joka on esitetty kuviossa 2, valitaan ”Clone a repository”, ja seuraavassa vaiheessa syötetään linkki, mistä paketti ladataan, ja valitaan, mihin se ladataan. Latauksen sijainti kannattaa huomioida tässä vaiheessa, sillä sitä tullaan tarvitsemaan seuraavassa vaiheessa.



Kuvio 2. Visual studio aloitusnäkymä

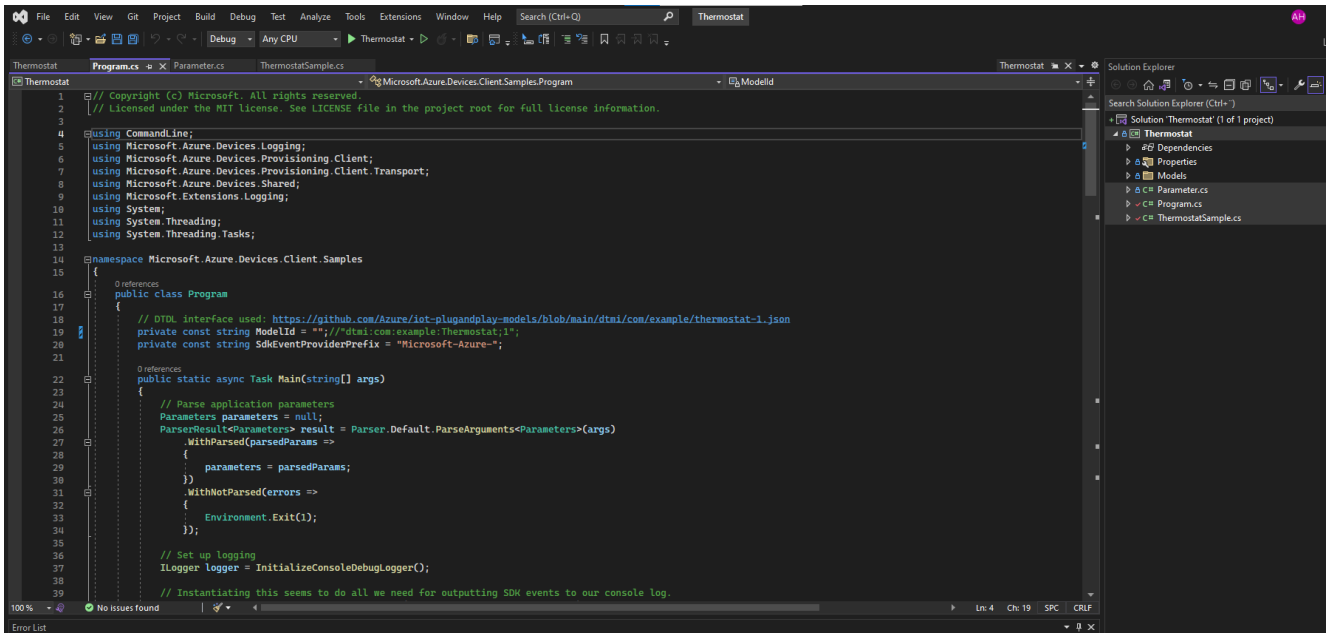
Kun ohjelmistonkehityspaketti on ladattu, aukeaa se Visual Studiossa. Ennen muutoksien tekemistä varmistetaan, että kaikki tarvittavat kirjastot ovat asennettuna, ja voidaan Visual Studio sulkea. Kirjastojen lataamiseksi, siirrytään ohjelmistonkehityspaketin latauskansioon, kuten kuviossa 3, on esitetty. Avataan kansio ”C:\...\azure-iot-sdk-csharp\iothub\device\samples\solutions\PnpDeviceSamples\Thermostat”, ja avataan komentokehote tai PowerShell tähän ikkunaan. Power Shell avataan helposti pitämällä Shift nappia pohjassa, ja klikkaamalla hiiren oikeaa painiketta kansiossa, ja valitsemalla ”Avaa PowerShell-ikkuna tähän”. Tämän jälkeen avautuneeseen ikkunaan kirjoitetaan ”dotnet -restore”, joka lataa kansiossa olevassa ohjelmassa käytettävät kirjastot tietokoneelle. Edellä mainitut toimenpiteet on esitetty kuviossa 3. Tämän jälkeen kansiossa oleva ”Thermostat.csproj” tiedosto voidaan avata Visual Studiolla.



Kuvio 3. Projektin kansio, PowerShell -valikko, ja PowerShell -ikkuna.

4.2.2 Esimerkkiohjelma ja laitteen integrointi

Nyt kun vaaditut kirjastot on ladattu, sekä projekti on avattu, ollaan valmiita testaamaan koodia tai muokkaamaan sitä. Kuten kuviossa 4. on esitetty, Visual Studion oikealla puolella on ”Solution Explorer”, joka näyttää projektin osat. Kyseisen projektin koodi on jaettu kolmeen osioon: Program, Parameter ja ThermostatSample. Nämä osiot riippuvat toisistaan, ja luovat ohjelman kokonaisuuden.

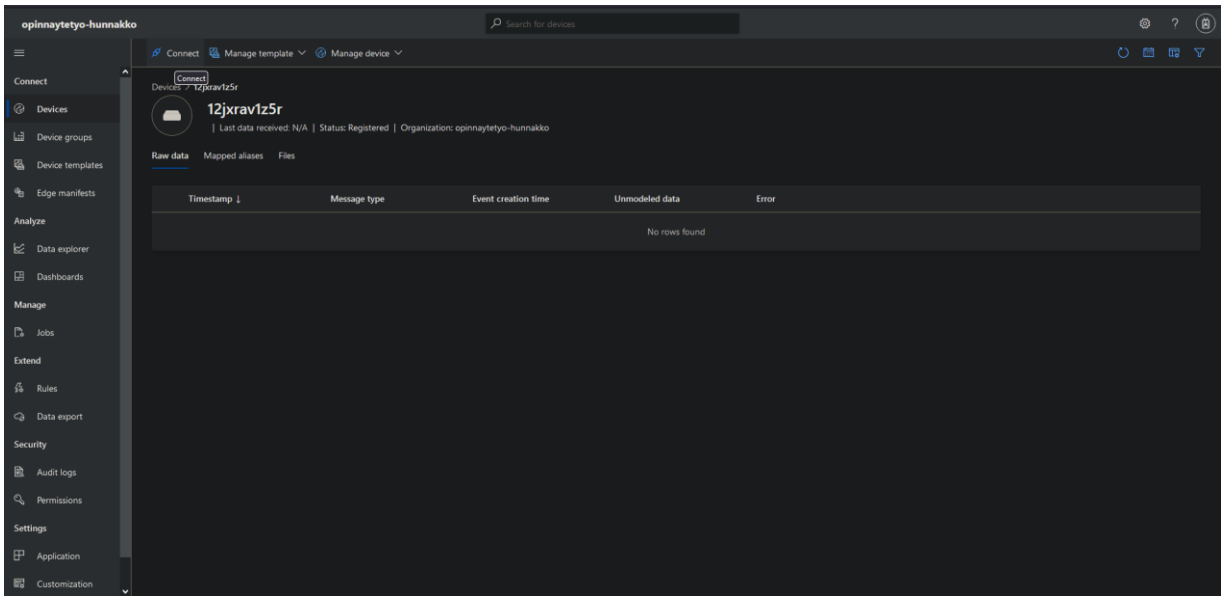


Kuvio 4. Avatun ohjelman pääsivu, ja Solution Explorer.

Program koodissa luodaan ”Logger”, jonka avulla voidaan kirjoittaa lokia ohjelman suorittamista toiminnoista, sekä yhdistetään pilvipalvelimeen. Jotta ohjelma voidaan yhdistää pilvipalvelimeen, tulee ohjelmalle syöttää laitteen ja pilvipalvelimen tiedot. Nämä tiedot kerätään ”Parameter” koodissa.

Laitekohtaiset tiedot luetaan ohjelmassa ympäristömuuttujista. Tämä tehdään siksi, että ympäristömuuttujat tallentuvat koodin ulkopuolelle, eli laitteeseen itseensä. Tämän avulla voidaan samaa koodia käyttää useassa laitteessa, kunhan ympäristömuuttujat ovat asennettuna laitteelle oikein. Ennen ympäristömuuttujien lisäystä, täytyy luoda laite pilvipalveluun.

Laitteen luominen tapahtuu Azure IoT Central applikaation aloitussivun vasemmasta reunasta, painikkeesta ”Device”. Sivun yläreunassa olevaa sinistä ”New” painiketta painamalla voidaan uusi laite lisätä. Laitteelle annetaan nimi sekä ID, ja sen jälkeen se voidaan luoda. Tämän jälkeen laite ilmaantuu listaan, ja sen nimeä napauttamalla pääsemme laitteen asetuksiin.



Kuvio 5. Luodun laitteen aloitussivu. Yläreunassa "Connect" painike

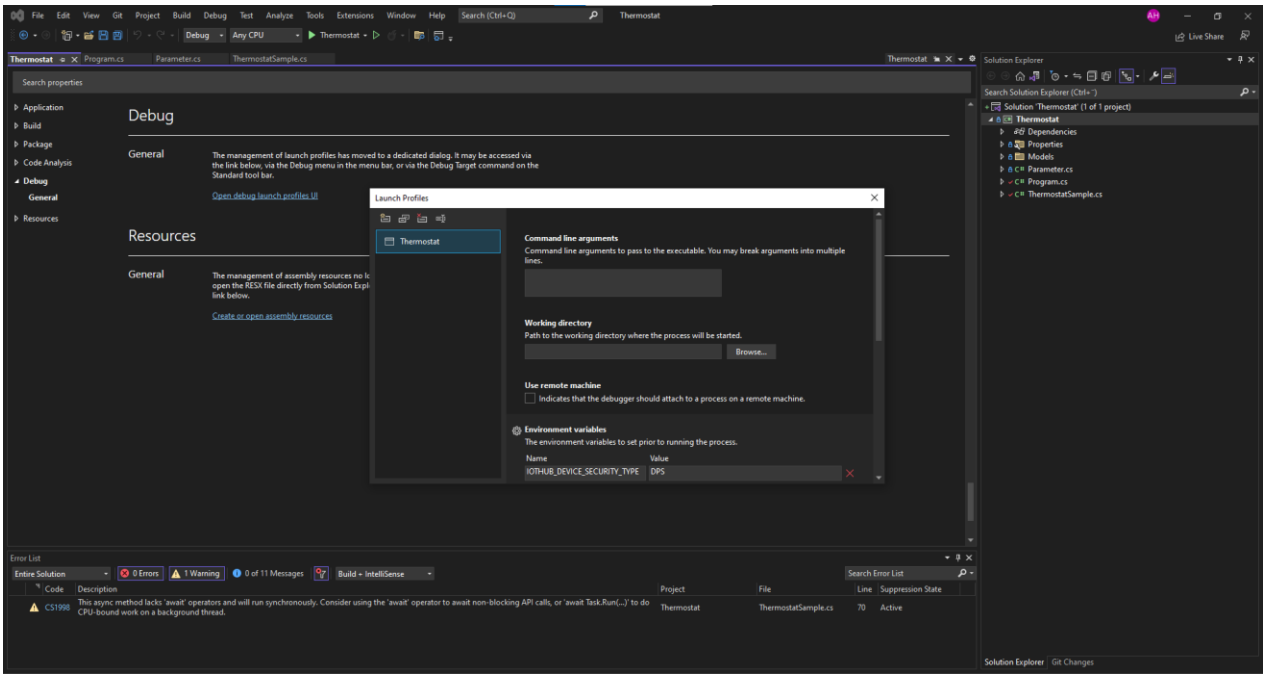
Kuten kuviosta 5. nähdään, sivun yläreunasta löytyy nyt laitteen oma ”Connect”, eli yhdistä painike. Tämä painike avaa sivun, jossa näkyy tämän laitteen yhdistämiseen tarvittavaa tietoa, kuten salausavain. Tämä tieto on sensitiivistä, ja se tulee pitää ulkopuolisilta salassa. Yhdistämiseen tarvitaan tältä sivulta seuraavat tiedot: ID Scope, Device ID sekä Primary Key. Lisäksi tarvitaan osoite, joka ohjaa laitteen rekisteröinti palveluun. Nämä taulukossa 1 esitetyt tiedot syötetään ympäristömuuttujiin.

Taulukko 1. Lisättävät ympäristömuuttujat

Ympäristömuuttujan nimi	**Ympäristömuuttujan arvo**
IOTHUB_DEVICE_SECURITY_TYPE	DPS
IOTHUB_DEVICE_DPS_ENDPOINT	global.azure-devices-provisioning.net
IOTHUB_DEVICE_DPS_ID_SCOPE	Oman laitteesi ID Scope
IOTHUB_DEVICE_DPS_DEVICE_ID	Oman laitteesi Device ID
IOTHUB_DEVICE_DPS_DEVICE_KEY	Oman laitteesi Primary key

Ympäristömuuttujat voidaan lisätä Windows laitteessa avaamalla komentokehote, ja syöttämällä ne yksi kerrallaan tyylillä: ”set Ympäristömuuttujan nimi=Ympäristömuuttujan arvo”, eli esimerkiksi ”set IOTHUB_DEVICE_SECURITY_TYPE=DPS”. Nämä ympäristömuuttujat eivät kuitenkaan toimi, kun koodi käynnistetään Visual Studion sisältä, joten ne lisätään sinne testauksen helpottamiseksi.

Visual Studion Solution Explorerista valitaan ”Thermostat” ja hiiren oikealla painikkeella avatusta valikosta valitaan ”Properties”. Avautuneesta valikosta navigoidaan kohteeseen ”Debug - General - Open debug launch profiles UI - Environment Variables” ja tänne lisätään muuttujat, kuten kuviossa 6. on esitetty.



Kuvio 6. Ympäristömuuttujien lisääminen

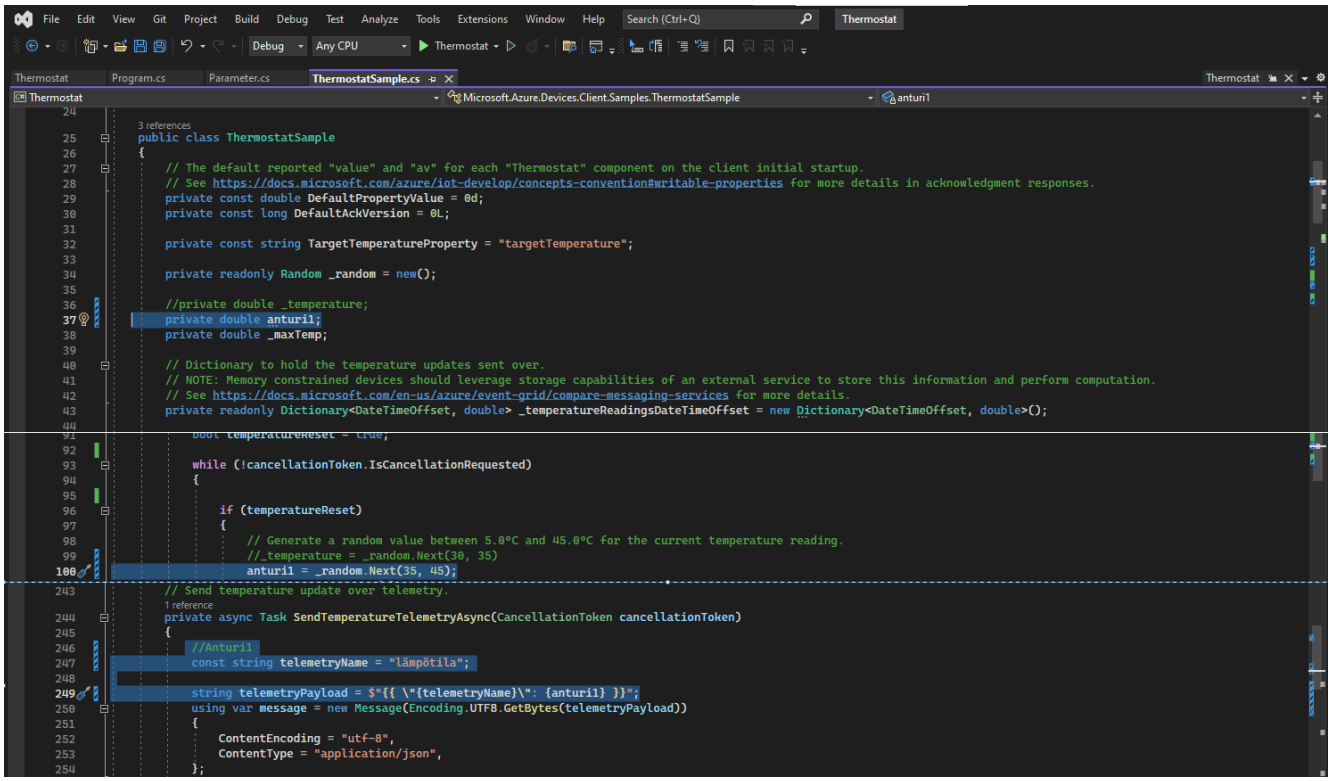
Nyt ohjelma löytää muuttujat ja niihin syötetyt laitteen tiedot, jolla se yhdistää pilvipalveluun. Seuraavaksi voidaan tarkastella ”ThermostatSample” ohjelmaa, jota muokkaamalla voidaan luoda omia laitteita, ja lähettää dataa pilveen.

4.2.3 Esimerkkiohjelman koodi ja laitteen simulointi

Koodia tarkastelemalla nähdään, että ohjelman on yksinkertainen, ja datan lähettämiseen tarvitaan vain muuttuja, jossa laitteelta saatu data on, sekä funktio, jossa data lähetetään eteenpäin. Ohjelmassa data luodaan satunnaisella numerogeneraattorilla. Lähetettävä viesti

kirjoitetaan ensin tekstimuotoon, jonka jälkeen viesti muutetaan biteiksi, ja lähetetään kirjastoon kuuluvalla ”message” komennolla eteenpäin.

Seuraavaksi luodaan lähetettävä data. Luodaan ”Double”-tyypin muuttuja nimeltä ”Anturi1”. Luodaan muuttujaan satunnainen arvo välillä 35–45 käyttämällä ohjelmassa jo luotua ”_random” satunnaislukugeneraattoria. Lopuksi luodaan lähetettävälle arvolle nimi, esimerkiksi ”lampotila” sekä liitetään lähetettävän datan nimi ja arvo toisiinsa, ja lähetetään se pilvipalvelimelle. Lähetetyn viestin tulee olla ”JSON” muodossa. Tässä muodossa viesti tulee lähettää seuraavan näköisenä: ”{”nimi”:”tieto”}”. Nimi kenttään lisätään lähetettävän arvon nimi, kuten ”lampotila”, ja tieto kenttään lisätään lähetettävän arvon tieto, kuten ”45”. Jos lähetetään tietoa useammasta asiasta, erotetaan tiedot toisistaan pilkulla. Koodissa tämä näyttää huomattavan erilaiselta, koska viesti säilötään tekstimuuttujaan. Kuviossa 7. on havainnollistettu koodin muutokset.



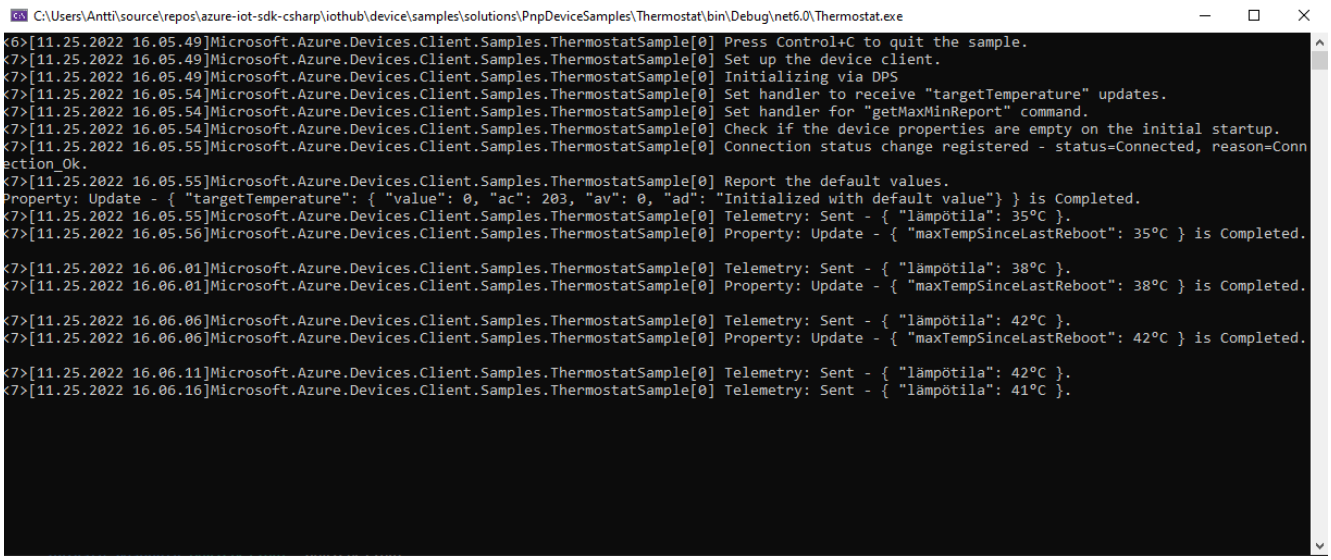
Kuvio 7. ThermostatSample koodin muokkaaminen, oman anturin lisääminen

Myös lokikirjoittajan muuttujat voidaan vaihtaa, jotta komentokehotteessa näkyvä tieto on paikkaansa pitävää. Näiden muokkausten jälkeen on enää yksi muokkaus, joka ohjelmaan tehdään. Azure IoT tukee niin ”Plug and Play” laitteita, jossa laitteen tiedot lisätään automaattisesti pilvipalveluun, ja vastaanotettu data on valmiiksi käsitelty. Tässä työssä kuitenkin

luomme myöhemmin oman laitteen, joten poistetaan tämän automaattinen tietojen lisäys. Tämä osuus löytyy ”Program” koodin riviltä 19, kuten kuviossa 8. on esitetty. Muuttuja nimeltä ”ModelId”, sisältää pilvipalvelusta löytyvän tiedoston sijainnin, joka ladataan, kun pilvipalveluun yhdistetään. Korvataan tämä rivi tyhjällä tiedostonimellä.

Kuvio 8. Vanha "ModelID" kommentoitu, ja uusi "ModelId" luotu.

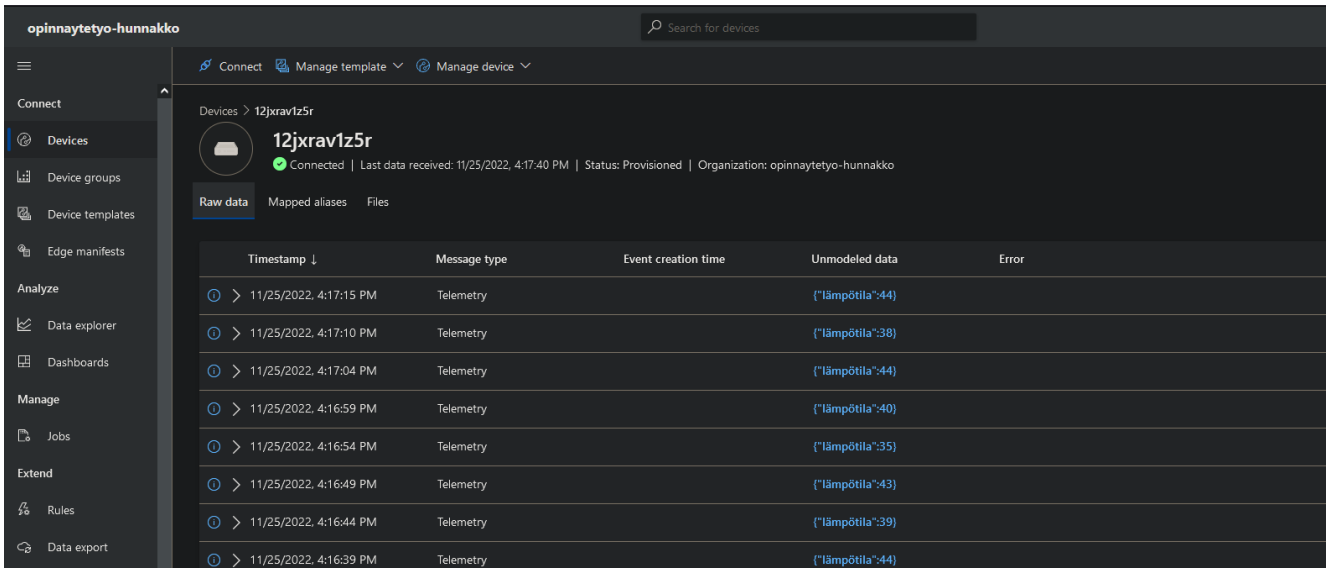
Tämän jälkeen voidaan avata IoT Central-sivusto, ja avata sieltä aiemmin luotu laite. Nyt Visual Studion kautta voidaan käynnistää projekti, painamalla yläreunassa olevaa vihreää play painiketta. Komentokehote avautuu, ja yhteys palvelimeen luodaan. Ohjelma käyttää ympäristömuuttujiin lisättyjä tietoja yhdistämiseen, ja komentokehotteeseen tulostuu tietoja yhdistämisestä, sekä tämän jälkeen rivi riviltä lähetetty data, kuten kuviossa 9. nähdään, mikäli lokikirjoittajan muuttujat on vaihdettu vastaamaan aiemmin luotua anturia.



Kuvio 9. Yhdistetty komentokehote ikkuna.

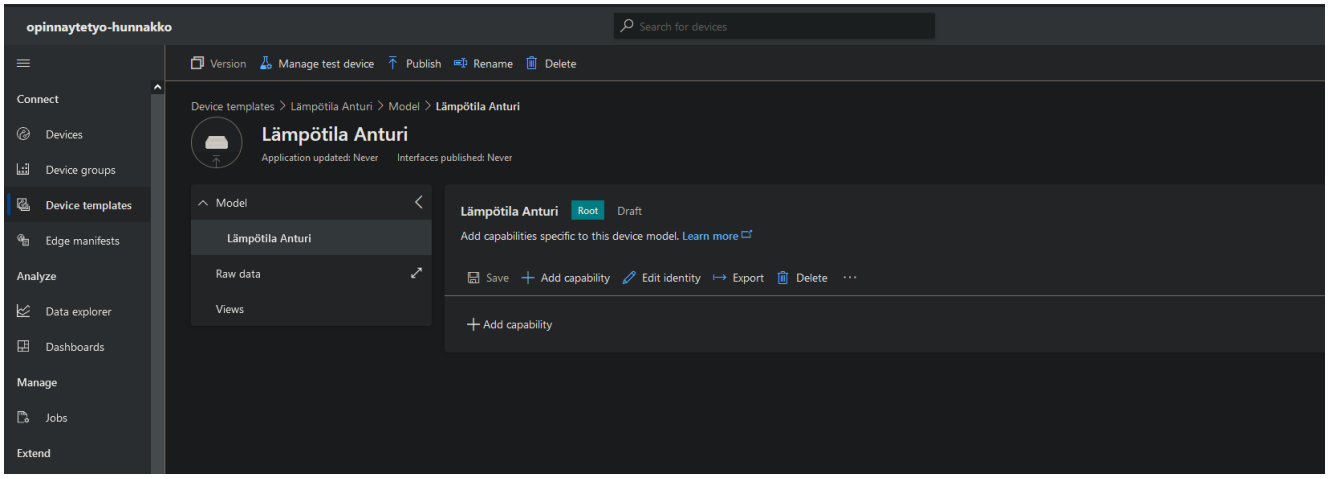
4.2.4 Laitteen datan käsittely pilvipalvelussa

Tämän jälkeen laitteen tila pilvipalvelussa vaihtuu yhdistetty tilaan, ja ”Raw Data” välilehdellä näemmä lähetetyn data riveittäin. Kuten kuviossa 10. on havainnollistettu, kaikki data ilmenee sarakkeen ”unmodeled data” alla, koska datalle ei ole vielä luotu pilvessä tunnistetta. Tämä tunniste tulisi aiemmin muutetun ”ModelId” tiedon kautta. Laitteelle lisätään tunniste sivun vasemmassa reunassa olevan ”Device templates” valikon avulla. Tässä valikossa luodaan laitekohtainen datan hallinta, johon kaikki saman tyypin laitteet voidaan lisätä, jotta kun uusia saman tyypin laitteita lisätään, näkyy niiden data heti oikein. Tämä seuraavaksi luotu laitemalli voitaisiin sijoittaa siis laitteen koodissa aiemmin korvattuun ”ModelId” muuttujaan, jotta se ladattaisiin automaattisesti.



Kuvio 10. Laitteen vastaanottama data näkyy ”Unmodeled data” rivin alla.

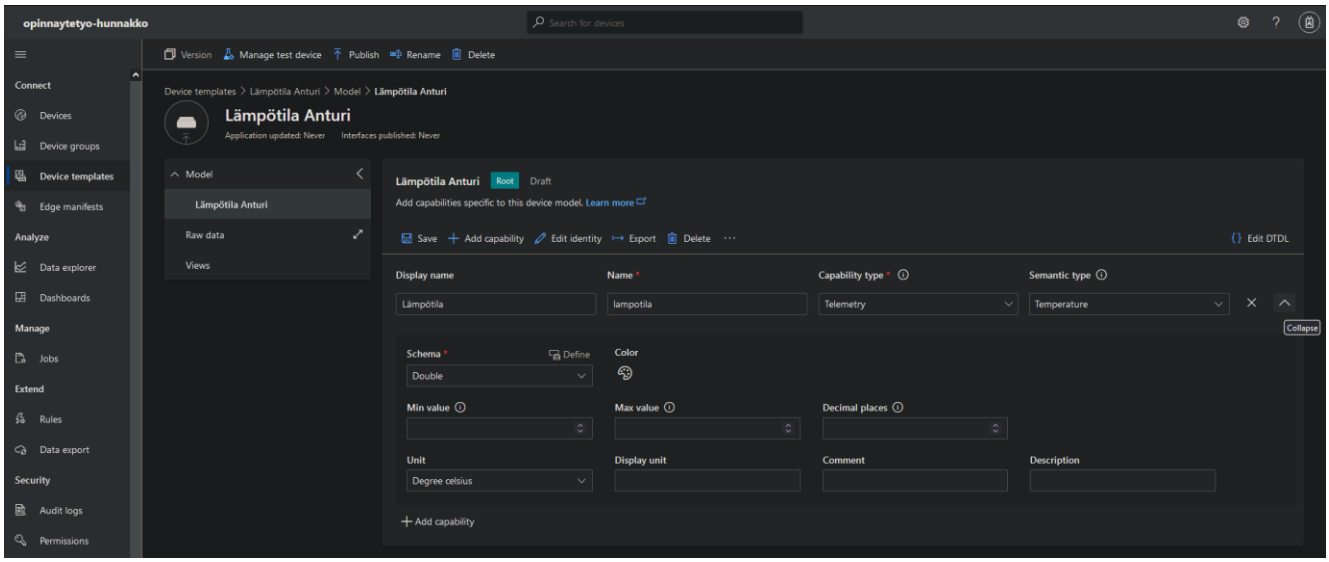
Device teplates sivuston yläreunasta ”New” painiketta painamalla luodaan uusi laitemalli. Painikkeesta avautuvassa valikossa nähdään myös tuetut laitteet, joilla on ”Plug and Play” toimivuus. Valikossa luodaan uusi ”IoT Device”, ja valitaan sivun alareunasta ”Next”. Annetaan laitteelle nimi, ja luodaan se. Tämän jälkeen luodaan uusi malli ”Custom model” painikkeesta, ja datan muokkaus voidaan aloittaa. Esimerkin laitemallin nimeksi annettiin ”Lämpötila Anturi”, jonka muokkaussivu nähdään kuviossa 11.



Kuvio 11. Luotu laitemalli.

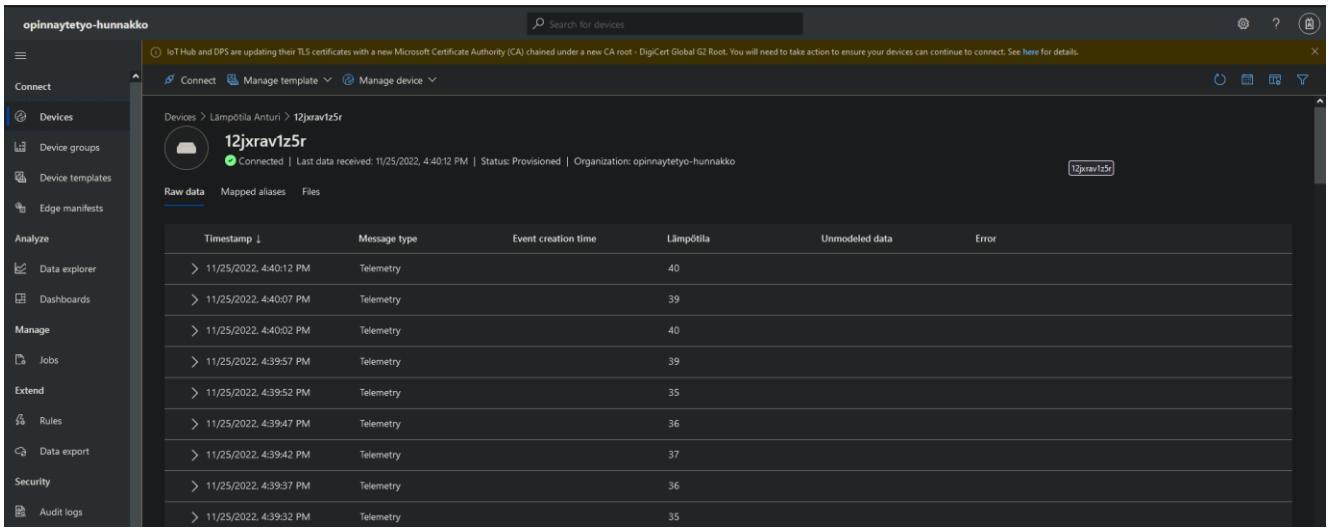
Tämän jälkeen painikkeesta ”Add capability” lisätään ominaisuus. Ominaisuuden näyttö nimeksi voidaan laittaa Lämpötila, ja nimi kenttään lisätään lähetyksen nimi, jota käytetään

viestin lähettämisessä, tässä tapauksessa ”lampotila”. Ominaisuuden tyypiksi tulee ”Telemetry”, ja ”Semantic type” on ”Temperature”. Rivin oikeasta reunassa olevasta nuolesta avataan pudotusvalikko, joka on esitetty kuviossa 12, josta lisätään datan datatyyppi. Esimerkin tapauksessa valitaan Double ja astetta celsius. Tietojen lisäyksen jälkeen valitaan yläriviltä ensin ”Save”, ja sitten ”Publish”, joka on tehtävä aina muutosten jälkeen.



Kuvio 12. Laitemalliin lisätyt ominaisuudet.

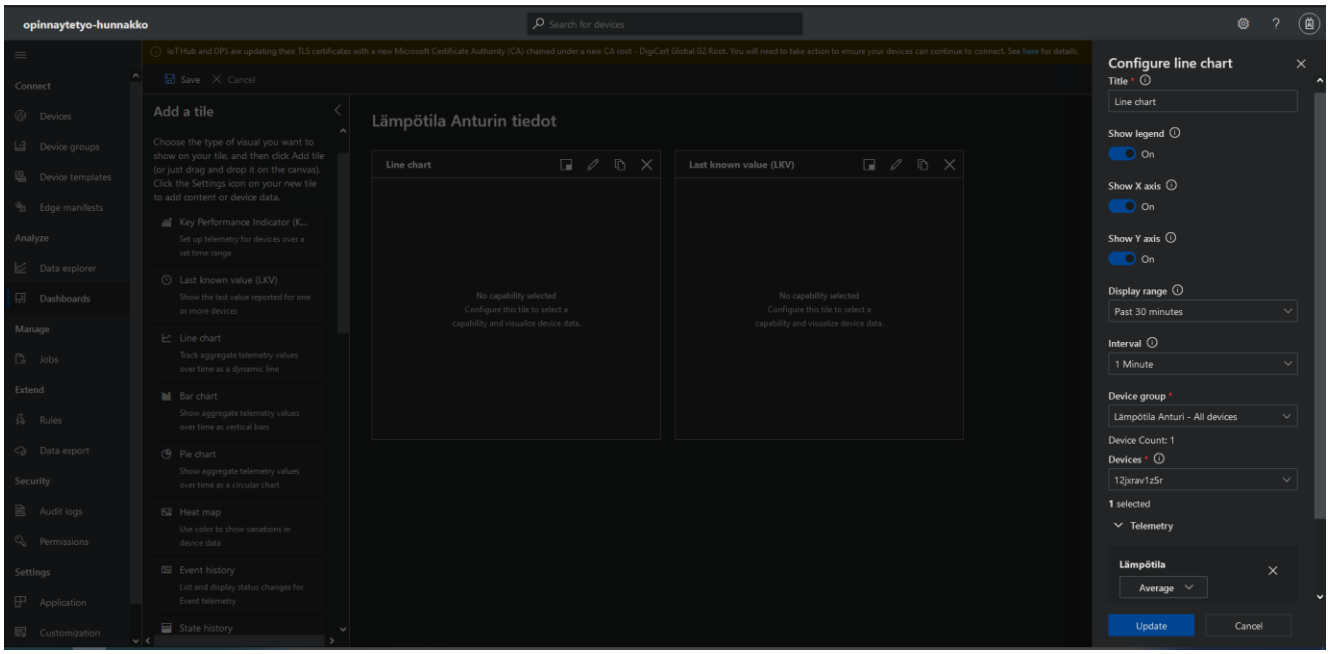
Siirrytään takaisin laitteen valikkoon, ja avataan luotu laite. Sivun yläosasta voidaan nyt linkittää laite luotuun laitemalliin. Avaa valikko ”Manage template”, ja valitse sieltä ”Assign template”, ja valitse äsken luotu laite. Tämän jälkeen päivitetään verkkosivu, ja data alkaa näkyä juuri luodun Lämpötila sarakkeen alla, kuten kuviossa 13. on esitetty.



Kuvio 13. Laitevalikossa näkyvä raakadata oikean rivin alla.

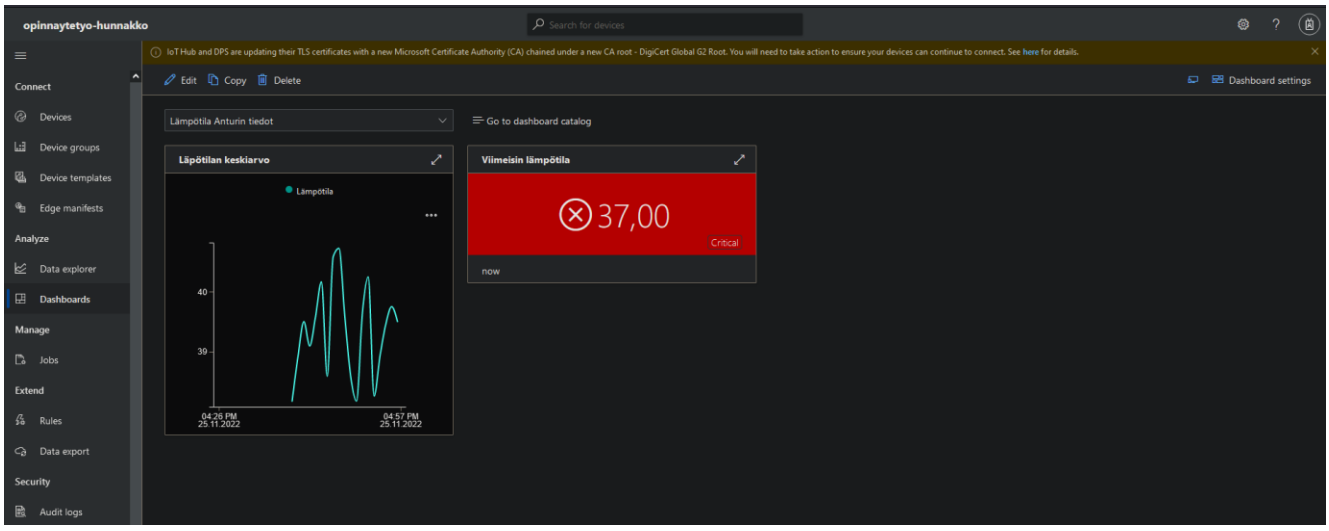
4.2.5 Datan visualisointi

Seuraavaksi siirrytään datan visualisointiin. Tämä tapahtuu vasemmassa reunassa olevan ”Dashboards” sivun kautta. Avataan sivu, ja luodaan uusi infonäkymä. Tämän jälkeen valitaan sivun yläreunasta ”Edit”, jonka jälkeen voidaan eri visualisointeja raahata valikosta näkymään. Valitaan valikosta ”Line chart”, sekä ”Last known value”. Näillä pystytään piirtämään kuvaajan, sekä näyttämään viimeisen tunnetun arvon. Näytölle raahattujen näkymien yläreunassa olevasta kynän kuvasta voidaan valita, mistä data saadaan. Kynän kuvaa painamalla avautuu sivun oikeaan reunaan valikko, kuten kuviossa 14. Valitaan luotu laite, ja lämpötilaarvo, ja hyväksytään valinta. Tehdään sama molemmille näkymille.



Kuvio 14. Dashboard muokkaus valikko.

Tämän jälkeen infonäyttö tallennetaan sivun yläreunasta, ja se alkaa näyttää meille dataa. Muokkaa valikosta voidaan asettaa näkymille myös muuttujia datan perusteella. Esimerkiksi, jos lämpötila anturin arvo on alle tietyn, muuttuu näkymän tausta punaiseksi, kuten kuviossa 15, ja kun lämpötila on asetus arvossa, näkyy se vihreänä.



Kuvio 15. Valmis infonäkymä

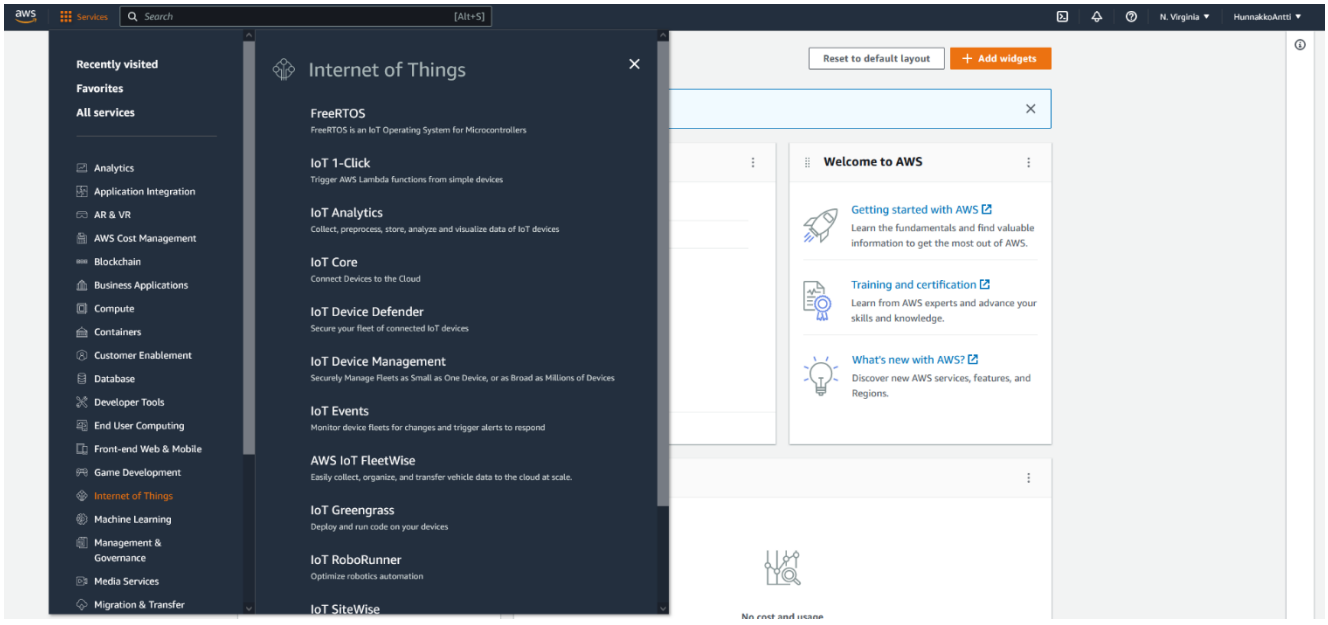
Nyt anturi on integroitu pilvipalveluun, ja sitä voidaan tarkkailla verkosta.

4.3 AWS IoT sovellusten integraatio

Työssä pilvipalveluun yhdistetään C#-ohjelmointikielellä kirjoitettua ohjelmaa ja MQTT kirjasto apuna käyttäen. Ohjelma yhdistetään pilvipalveluun luotuun laitteeseen, ja sillä pystytään lähettämään sinne dataa. Pilvipalvelussa dataa käsitellään ja varastoidaan, sekä luodaan sille visualisointi. Ohjelmointiin käytetään Visual Studio 2022 -ohjelmaa, sekä Windows ja Linux komentokehotetta.

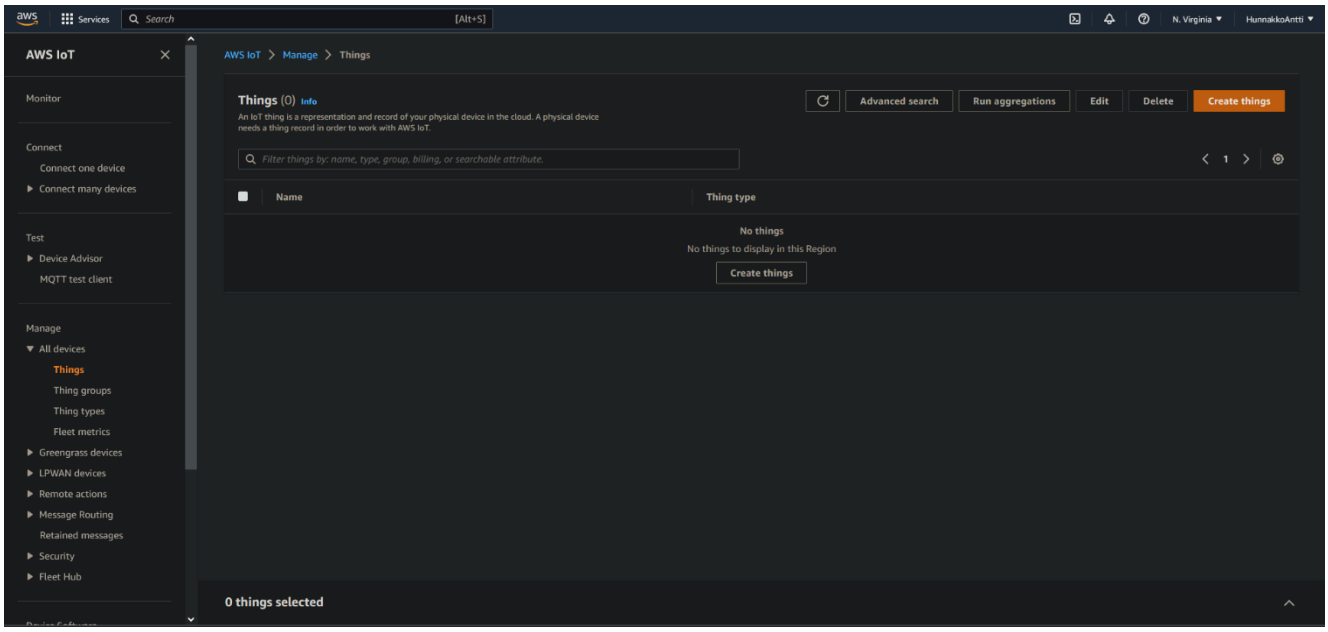
4.3.1 Resurssien valmistelu

Aloitetaan luomalla AWS-tili, ja kirjautumalla sisään. Tämän jälkeen voidaan avata sivun yläreunasta ”Services”- valikko. Valikossa näkyvät kaikki palvelut, ja sieltä voidaan valita Internet of Things, joka näyttää esineiden internettiin liittyvät palvelut, kuten kuviossa 16. on esitetty. Näistä meille ovat tärkeitä IoT Core sekä IoT Analytics. Muita palveluita, joita käytetään esimerkissä ovat IAM sekä QuickSight.



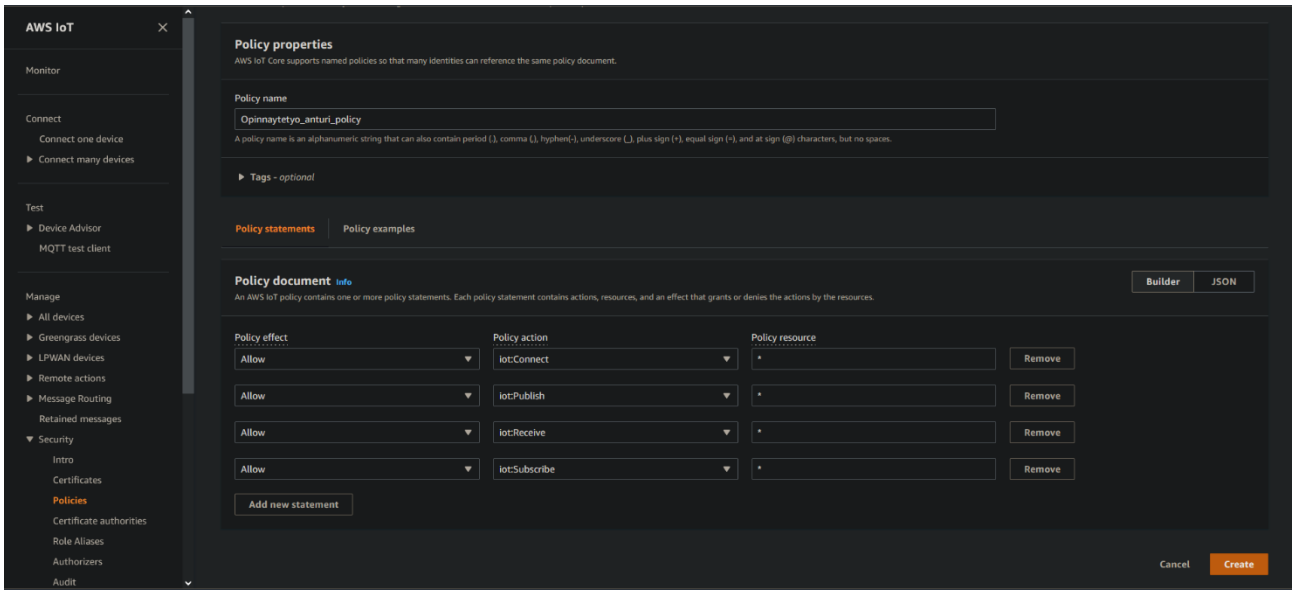
Kuvio 16. AWS palvelut valikko

Valitaan valikosta IoT Core, ja sen pääsivu aukeaa. Seuraavaksi luodaan pilvipalveluun laite, johon ohjelma yhdistetään. Valitaan sivun vasemmasta reunasta ”All Devices” ja tämän jälkeen ”Things”. Avautuu sivusto, josta voidaan lisätä laite, kuten kuviossa 17. on esitetty.



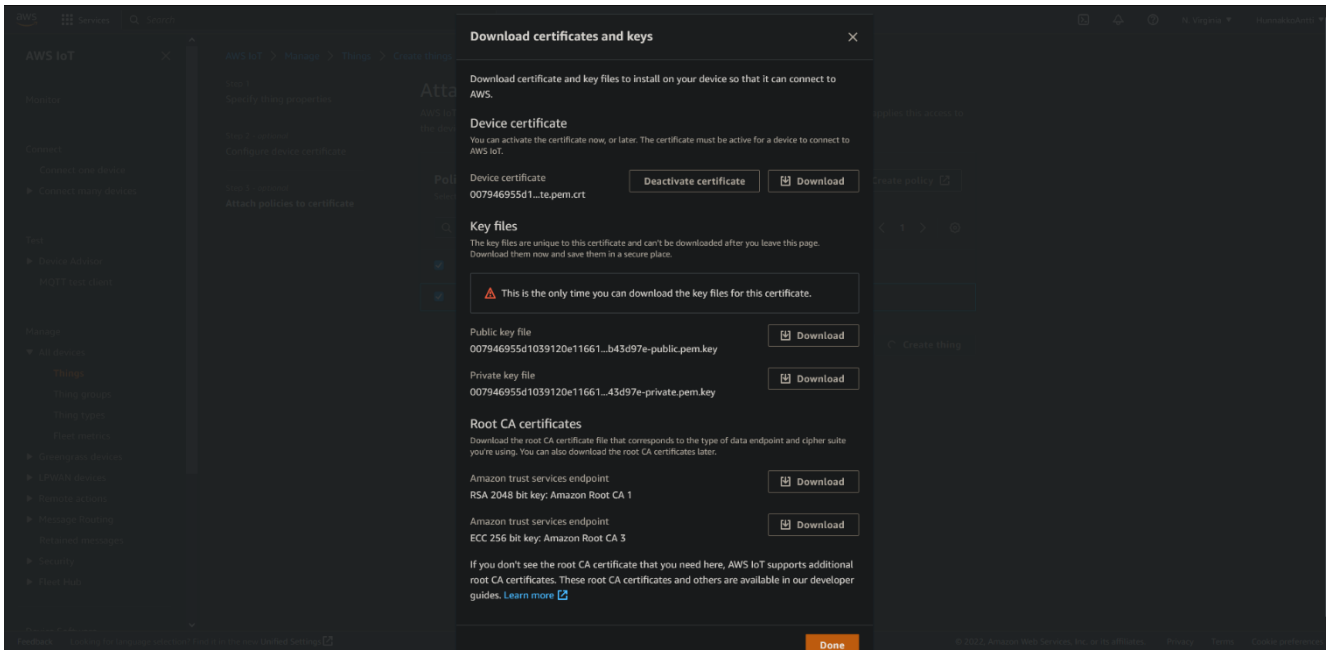
Kuvio 17. AWS IoT laitteiden luomisen valikko

Valitaan sivun yläreunasta painike ”Create things”, josta laitteen luominen aloitetaan. Seuraavissa valikoissa tehdään seuraavat asiat: luodaan yksi laite, annetaan sille sopiva nimi, luodaan automaattisesti sertifikaatit. Tämän jälkeen seuraavassa valikossa luodaan laitteelle käytäntö. Käytännöllä määritellään, mitä laite pystyy tehdä, ja mitä resursseja se saa käyttää. Valitaan ”Create policy”, ja uusi ikkuna aukeaa. Annetaan käytännölle nimi, ja lisätään sivun alaosassa olevasta ”Add new statement” painikkeesta kolme uutta riviä, jotta meillä on yhteensä neljä riviä. Kaikkien rivien ”Policy effect” sarakkeeseen valitaan ”Allow”, sekä kaikkien rivien ”Policy resource” sarakkeeseen kirjoitetaan ”*”. Kuten kuviossa 18 on esitetty, neljään ”Policy action” sarakkeen kenttään valitaan pudotusvalikosta yksi seuraavista: Connect, Publish, Receive ja Subscribe. Tämän jälkeen valitaan ”Create”, ja luodaan tämä käytäntö.



Kuvio 18.Käytännön luominen laitteelle

Palataan verkkoselaimessa välilehteen, jossa laitteen luonti on kesken. Juuri luomamme käytäntö tulisi nyt näkyä ”Policies” listassa. Valitaan tämä käytäntö, ja painetaan ”Create thing” painiketta. Sivustolle aukeaa kuviossa 19. esitetty näkymä, josta laitteen sertifikaatit voidaan ladata. Nämä sertifikaatit lisätään myöhemmin luotavaan laitteeseen, ja näin pilvipalvelu todentaa laitteen aidoksi. Näitä sertifikaatteja ei voi enää myöhemmin ladata, joten jos tämä vaihe keskeytyy, täytyy laitteen luonti aloittaa alusta. Ladataan seuraavat dokumentit: device certificate, public key file, private key file, sekä Amazon Root CA1. Sertifikaatteja tarvitaan myöhemmin laitteen yhdistämiseen.



Kuvio 19. Sertifikaattien lataaminen.

Seuraavaksi avataan Visual Studio 2022 -ohjelman, ja kopioidaan tietokoneelle Amazonin valmistaman esimerkkiohjelman sekä muokkataan ladattuja sertifikaatteja käyttövalmiiksi. Tämä vaihe vaatii sen, että tietokoneessa voidaan käyttää Bash-komentokehotetta, eli Linux käyttöjärjestelmän komentokehotetta. Tämä siksi, että sertifikaattien muuttamiseen tarvitaan OpenSSL-komentoja. Tämä voidaan ottaa käyttöön Windows-käyttöjärjestelmässä, sallimalla kehittäjä tila Windows asetuksista, sekä lataamalla ilmainen ”Ubuntu” Windows sovelluskaupasta.

Kuten kuviossa 2. on esitetty, Visual Studion etusivulta valitaan ”Clone repository”, ja osoitekenttään syötetään GitHub osoite, jossa esimerkkiohjelma sijaitsee, tämä kopioi esimerkkiohjelman tietokoneelle. Esimerkkiohjelma löytyy osoitteesta: https://github.com/aws-samples/iot-dotnet-publisher-consumer.

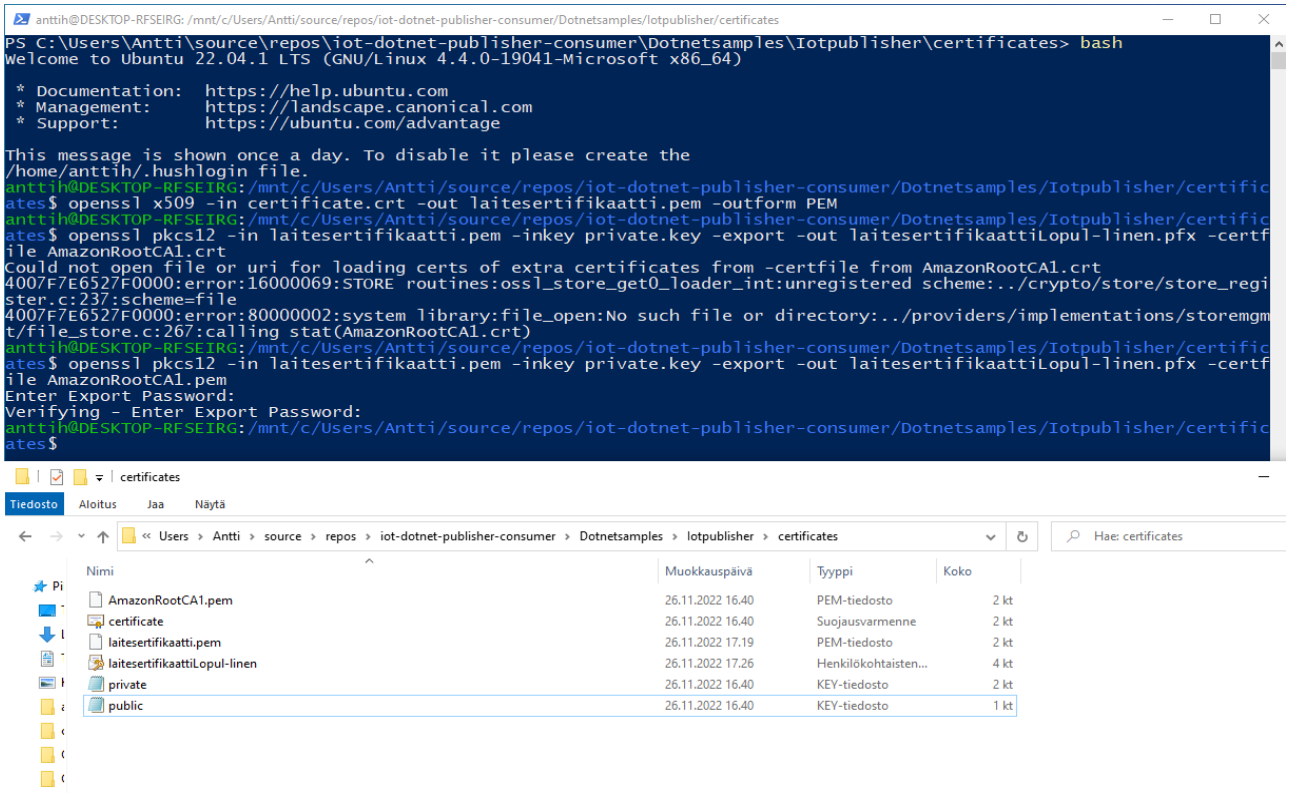
Kiinnitä huomiota, mihin sijaintiin tietokoneella kopioit ohjelman, sillä seuraavaksi siirrytään tähän kansioon. Avataan kansio ” C:\...\repos\iot-dotnet-publisher-consumer\Dotnetsamples\Iotpublisher”, ja luodaan tähän kansioon uusi kansio nimeltä ”certificates”. Avataan kansio, ja lisätään sinne aiemmin ladatut neljä sertifikaattitiedostoa. Lyhennetään tiedostojen nimiä hieman, jotta ne ovat helpommin luettavassa muodossa, mutta ne tulee vielä erottaa toisistaan. Tiedosto, jonka nimessä esiintyy ”certificate.pem”, ei ole oikeasti tiedostomuodossa ”pem”, vaan se täytyy muuttaa tähän muotoon. Tämä tapahtuu Bash-komentokehotteen kautta. Avaa komentokehote kansioon painamalla näppäimistön vaihtonäppäintä pohjassa, ja

klikkaa hiiren oikealla painikkeella kansion pohjasta, ja valitse ”Avaa PowerShell-ikkuna tähän”, kuten kuviossa 3.

Tämän jälkeen Windows komentokehotteeseen syötetään komento ”bash”, jolla Bash-komentokehote avautuu. Tämän jälkeen muutetaan sertifikaatin tiedostomuodosta ”crt”, tiedostomuotoon ”pem”, seuraavalla komennolla: ”openssl x509 -in sertifikaattimuodossa.crt -out sertifikaattimuodossa.pem -outform PEM”

Komento luo kansioon uuden tiedoston, joka on nyt oikeassa muodossa. Tämän jälkeen muutetaan vielä tämä juuri luotu sertifikaatti tiedostomuotoon ”pfx”, seuraavalla komennolla: ”openssl pkcs12 -in sertifikaattimuodossa.pem -inkey private.key -export -out sertifikaattimuodossa.pfx -certfile AmazonRootCA1.pem”

On huomioitava, että yllä mainittuihin komentoihin täytyy sijoittaa kansiossa olevien sertifikaattien nimet, kuten kuviossa 20. on esitetty.



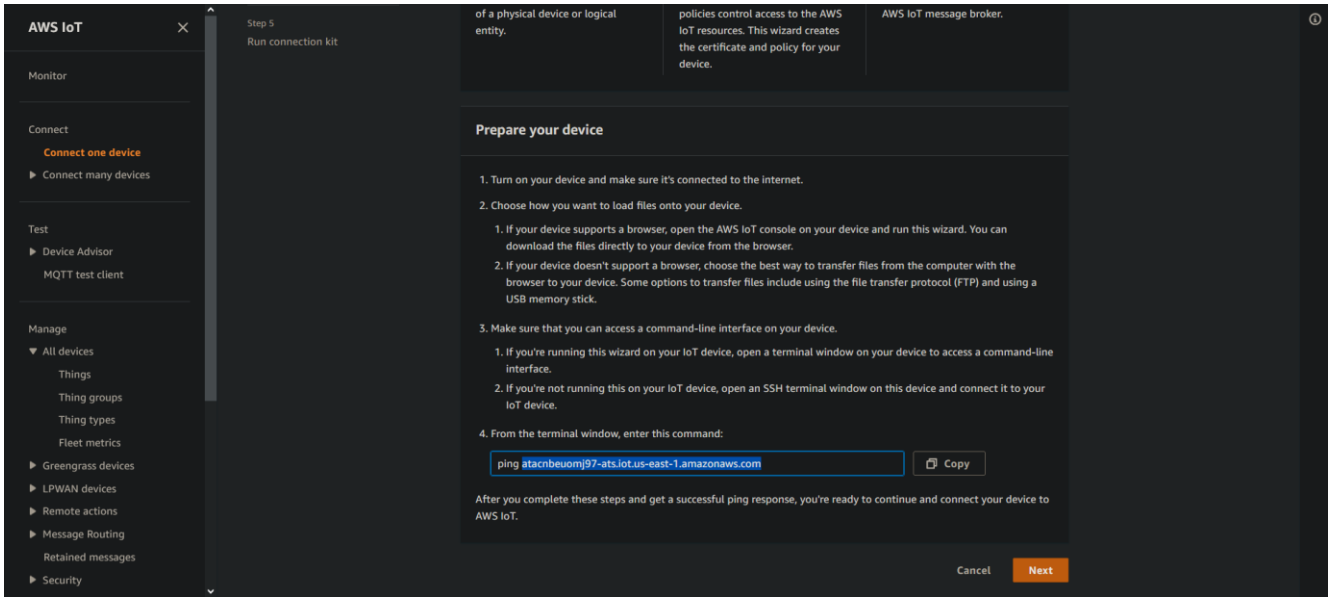
Kuvio 20. Sertifikaattien muokkaus Bash-komentokehotteessa

Tämän jälkeen komentokehotteen ikkuna kysyy käyttäjältä salasanaa, joka lisätään sertifikaattiin. Tätä salasanaa käytetään myöhemmin ohjelmassa, joten se tulee muistaa. Syötä

salasana kahteen kertaan, ja lopulta sertifikaatti on oikeassa muodossa. Huomioitavaa on, että salasanaa kirjoittaessa komentokehotteeseen ei ilmene tekstiä. Kirjoita salasana ja syötä se painamalla näppäimistön ”Enter”, eli syöttönäppäintä.

4.3.2 Esimerkkiohjelman koodi ja laitteen simulointi

Tämän jälkeen siirrytään yksi kansiotaso ylöspäin, kansioon ”iotpublisher”, ja avataan saman niminen Visual Studio-tiedosto. Projekti aukeaa, ja sitä voidaan alkaa tarkastella. Koodiin täytyy lisätä oman pilvipalvelun osoite, sekä luotu sertifikaatin nimi ja salasana, ennen kuin voidaan yhdistää pilvipalveluun. Pilvipalvelun osoite löytyy AWS IoT Coren pääsivulta, valikosta ”Connect one device”. Tämän sivun alareunasta, kuten kuviossa 21. on esitetty, löytyy yhteysosoite, joka sijoitetaan koodissa muuttujaan ”iotEndpoint”.



Kuvio 21. Endpoint muuttujan tiedot verkkosivulla

Tämän jälkeen luodaan koodiin muuttuja anturin tiedolle. Luodaan muuttuja, johon myöhemmin sijoitetaan aikaleima. Luodaan satunnaisnumerogeneraattori. Vaihdetaan ”topic” muuttujan aihe, mihin lähetetyt viestit ilmestyvät pilvessä. Tämän jälkeen muutetaan vielä sertifikaatin nimi vastaamaan oman sertifikaattimme nimeä, ja syötetään sen salasana. Koodiin tehdyt muutokset on havainnollistettu kuviossa 22.

```
namespace Iotpublisher
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            string iotEndpoint = "atacnbeuomj97-ats.iot.us-east-1.amazonaws.com"; //Tähän lisätty oman pilvipalvelimen tiedot
            Console.WriteLine("AWS IoT Dotnet message publisher starting..");

            int brokerPort = 8883;
            double lampotila; //Luotu oma muuttuja, johon syötetään anturin lämpötila arvot
            string time; //Luotu oma muuttuja, johon syötetään aika
            Random _rand = new Random(); //Luotu satunnais generaattori

            string topic = "Opinnaytetyo/lampotila"; //Vaihdettu muuttujaan oma aihealue, jossa lähetetyt viestit näkyvät
            string message;

            var caCert = X509Certificate.CreateFromCertFile(Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "AmazonRootCA1.pem"));
            var clientCert = new X509Certificate2(Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "leitesertifikaattinew.pfx"), "Salasana123"); //Muutettu serfikaatin nimi ja salasana

            var client = new MqttClient(iotEndpoint, brokerPort, true, caCert, clientCert, MqttSslProtocols.TLSv1_2);
```

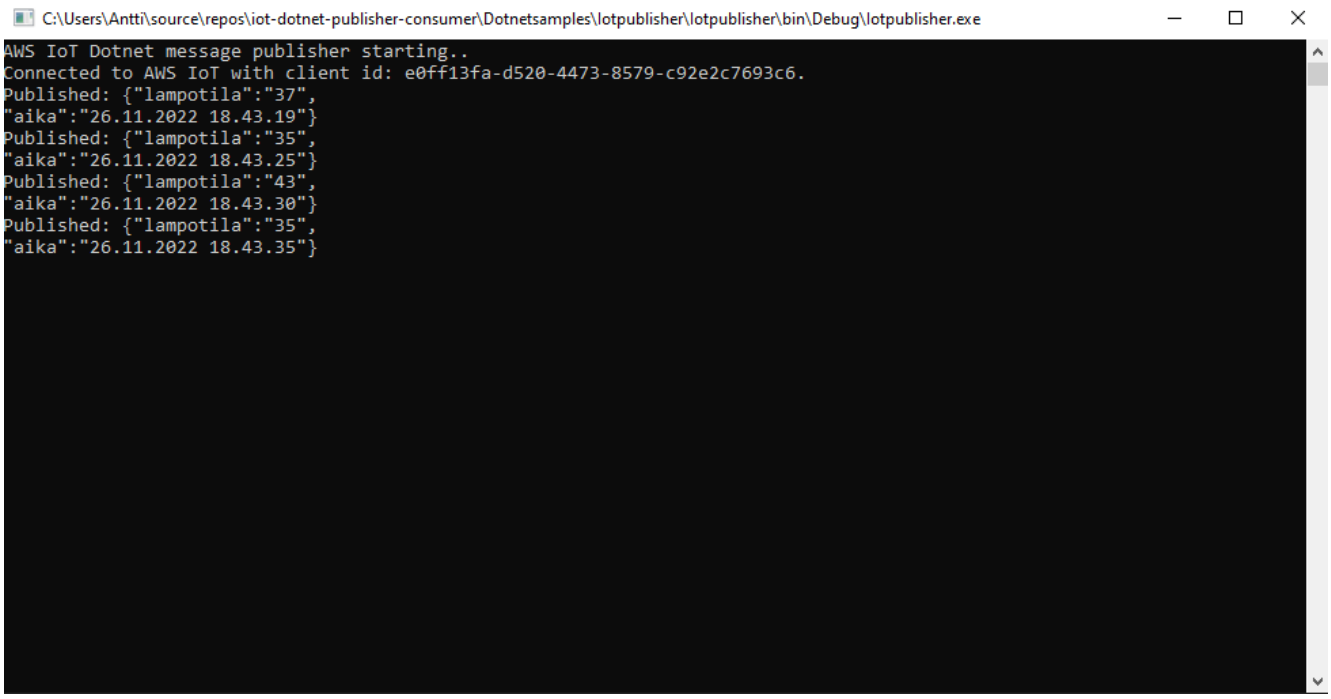
Kuvio 22. Muutettu koodi, ja kommentit.

Tämän jälkeen tarkastellaan koodissa olevan ”while” lauseen alla olevaa koodia, joka nähdään kuviossa 23. Tämän osuuden sisällä tapahtuu viestin lähetys. Lisätään aiemmin luotuun aikamuuttujaan tämänhetkinen aika, jotta saadaan tiedolle aikaleima. Tämän jälkeen luodaan anturin tiedon muuttujaan uusi satunnainen arvo, joka kuvastaa lämpötilaa. Esimerkissä luodaan arvo väliltä 35–45. Lopuksi muokataan ”message” muuttujaa siten, että siihen lisätään sekä aika että lämpötila. Lähetetyn viestin tulee olla ”JSON” muodossa. Tässä muodossa viesti tulee lähettää seuraavan näköisenä: ”{”nimi”:”tieto”}”. Nimi kenttään lisätään lähetettävän arvon nimi, kuten ”lämpötila”, ja tieto kenttään lisätään lähetettävän arvon tieto, kuten ”45”, ja jos lähetetään tietoa useammasta asiasta, erotetaan tiedot toisistaan pilkulla. Koodissa tämä näyttää huomattavan erilaiselta, koska viesti säilötään tekstimuuttujaan. Myöhemmin voidaan kuitenkin tarkistaa, saapuuko viesti perille oikein muotoiltuna. Tämä on tärkeää datan myöhemmän käsittelyn takia.

```
            while (true)
            {
                time = $"{DateTime.Now}"; //Tarkistetaan aika
                lampotila = _rand.Next(35, 45); //Luodaan satunnainen lämpötila-arvo
                message = $"{{\"lampotila\":\"{lampotila}\",\n\"aika\":\"{time}\"}}"; //Lisätään sekä aika, että lämpötila viestimuuttujaan
                client.Publish(topic, Encoding.UTF8.GetBytes($"{message}"));
                Console.WriteLine($"Published: {message}");
                i++;
                Thread.Sleep(5000);
            }
        }
    }
}
```

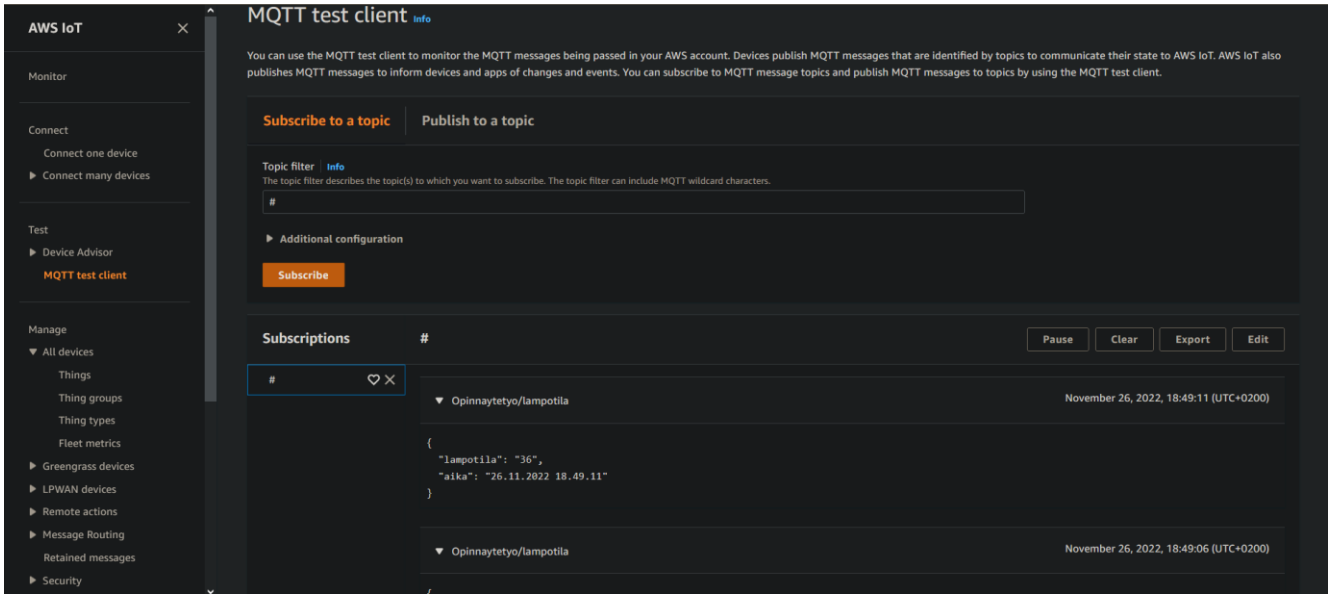
Kuvio 23. Ohjelmalla lähetettävä viesti formatoituna

Tämän jälkeen voidaan testata ohjelmaa ja yhteyttä. Käynnistetään ohjelma Visual Studion yläosasta olevasta vihreästä painikkeesta, ja odotetaan kunnes avautunut komentokehote näyttää, että se lähettää dataa, kuten kuviossa 24. Komentokehotteesta nähdään, missä muodossa viesti lähtee pilvipalveluun, ja voimme tarkistaa, että se vastaa JSON-tyyliä.



Kuvio 24. Ohjelman avaama komentokehoteikkuna, jossa nähdään lähetetyt viestit.

Tämän jälkeen voidaan siirtyä takaisin AWS IoT Core-sivulle, ja valita sivun vasemmasta reunasta ”MQTT test client”. Tältä sivulta voidaan tilata lähetettyjä viestejä, ja tarkastella niitä. ”Topic filter” kenttään voidaan syöttää koodissa käytetty aihe, tai siihen voidaan syöttää ”#”, jolloin tilaataan kaikki palveluun saapuvat viestit kerralla. Tilataan kaikki viestit, ja painetaan ”Subscribe”. Kuten kuviossa 25, sivun alaosassa alkaa näkyä ohjelman pilvipalveluun lähetetyt viestit. Samalla voidaan vielä tarkistaa, että viestit saapuvat oikeassa muodossa.

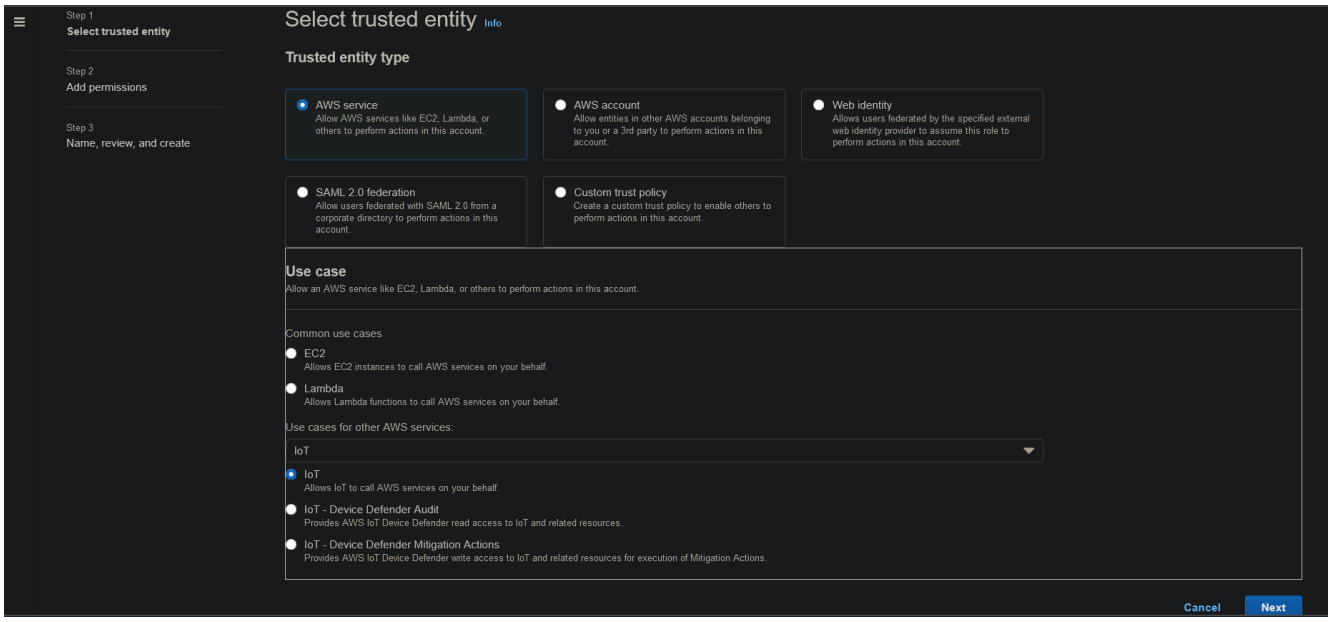


Kuvio 25. MQTT testi ympäristö, jossa lähetetyt viestit näkyvät.

4.3.3 Datan käsittely pilvessä

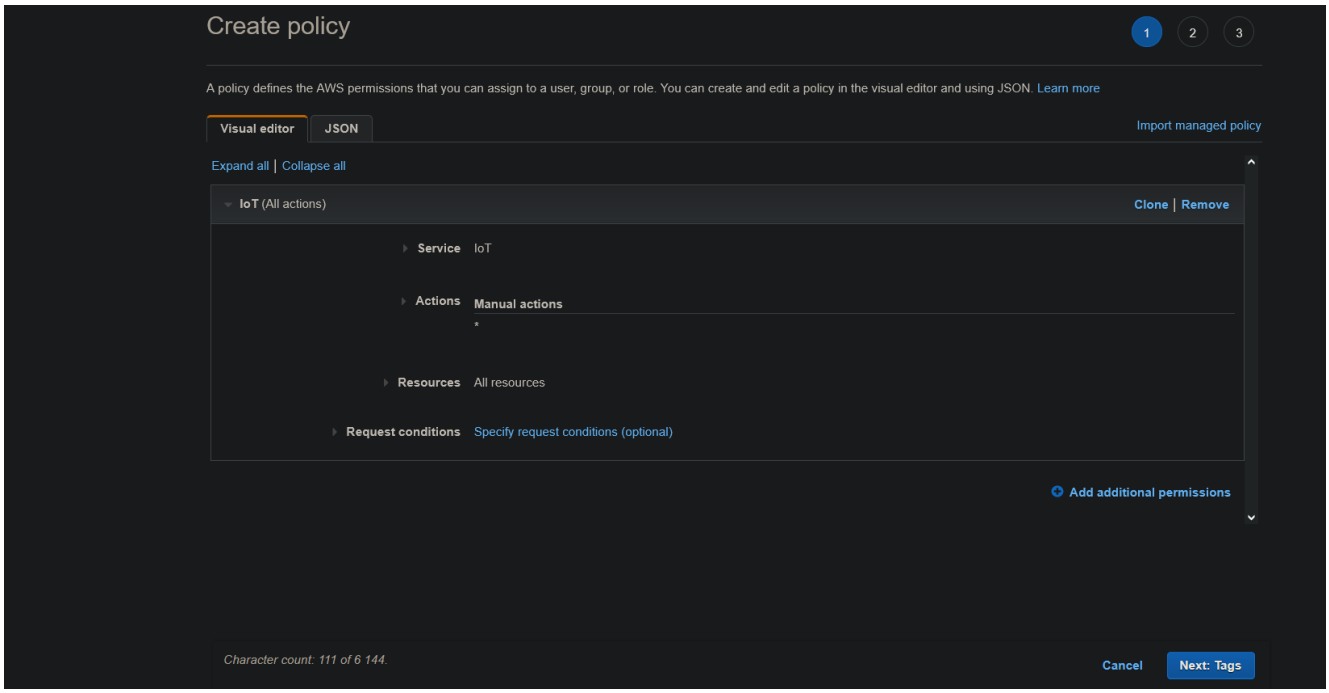
Seuraavaksi luodaan AWS-ympäristöön niin sanottu rooli, joka antaa pilvipalvelimelle luvan lähettää dataa eteenpäin muihin sivulla oleviin palveluihin. Tämän lisäksi luodaan lähetetylle datalle paikka, mihin se varastoidaan ja luodaan sääntö, mihin kirjataan sen lähetys paikka.

Syötetään sivun yläosassa olevaan hakukenttään ”IAM”, ja avataan tämä palvelu. Palvelun pääsivulla vasemmassa reunassa on valikko ”Roles”, josta rooli voidaan luoda. Siirrytään tähän valikkoon, ja valitaan sivun yläosasta ”Create role”. Sivun alaosassa on pudotusvalikko, josta valitaan ”IoT”, ja tämän jälkeen valitaan se uudestaan pudotusvalikon alapuolelle ilmaantuneesta valikosta, kuten kuviossa 26. on esitetty.



Kuvio 26. IAM roolin luonti

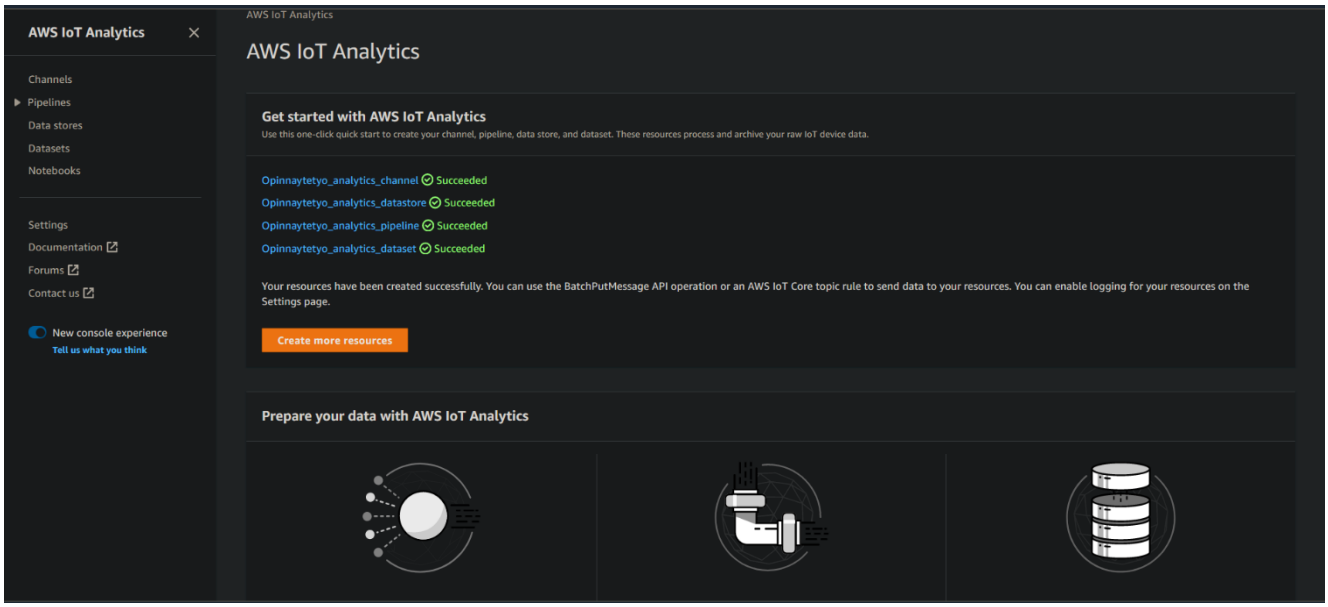
Tämän jälkeen valitaan sivun alaosasta ”Next”, ja seuraavasta valikosta jälleen ”Next”. Lopuksi annetaan nimikenttään roolille nimi, kuten ”IoTAllow”, ja valitaan sivun alaosasta ”Create role”. Roolin luonnin jälkeen siirrytään sivun vasemmassa reunassa olevaan ”Policies” valikkoon, ja luodaan sinne uusi käytäntö sivun yläreunassa olevasta ”Create policy” painikkeesta. Haetaan ensimmäisessä osioissa ”Service” kenttään ”IoT”, ja seuraavassa kohdassa sallitaan kaikki toiminnot, valitsemalla ”All IoT actions”. Viimeisessä osiossa valitaan ”All resources”. Lopuksi tarkistetaan, että tiedot ovat oikein, kuten kuviossa 27. Tämän jälkeen painetaan sivun alareunasta ”Next” painiketta. Painetaan seuraavassa osiossa uudelleen ”Next” painiketta, ja lopuksi annetaan käytännölle nimi, kuten ”IoTPolicy”, ja luodaan se sivun alareunassa olevasta painikkeesta.



Kuvio 27. Create policy valikko, johon arvot syötetty

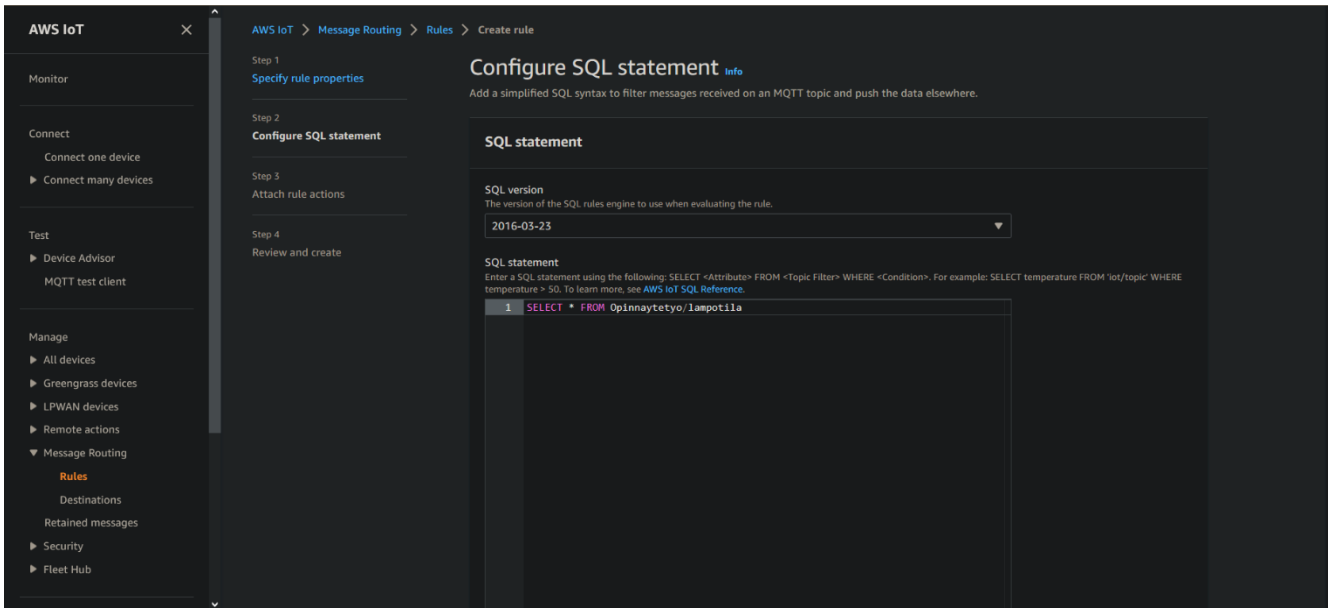
Seuraavaksi lisätään tämä käytäntö vielä aiemmin luotuun rooliin. Valitaan sivun vasemmasta reunasta ”Roles”, avataan aiemmin luotu ”IoTAllow” rooli, ja sen jälkeen sivun yläreunasta ”Add permission” ja ”Attach Policy”. Valitaan valikosta äsken luotu ”IoTPolicy”, ja painetaan sivun alareunasta ”Attach policies”. Nyt rooli on valmiina käytettäväksi.

Seuraavaksi luodaan datan säilytys. Siirrytään sivun yläosassa olevaan hakukenttään, ja haetaan ”IoT Analytics”, ja siirrytään sinne. Tässä palvelussa luodaan seuraavat asiat; kanava, johon dataa syötetään, putkisto, jolla dataa pystytään muokkaamaan sekä datan varastointi. Palvelu luo koko ketjun automaattisesti, kun sivun yläosiossa olevaan kenttään ”Resources prefix”, annetaan nimi, ja valitaan ”Create resources”. Resurssien luonnin jälkeen nähdään, että ne on luotu onnistuneesti, kuten kuviossa 28. on esitetty.



Kuvio 28. IoT Analytics sivu, ja luodut resurssit.

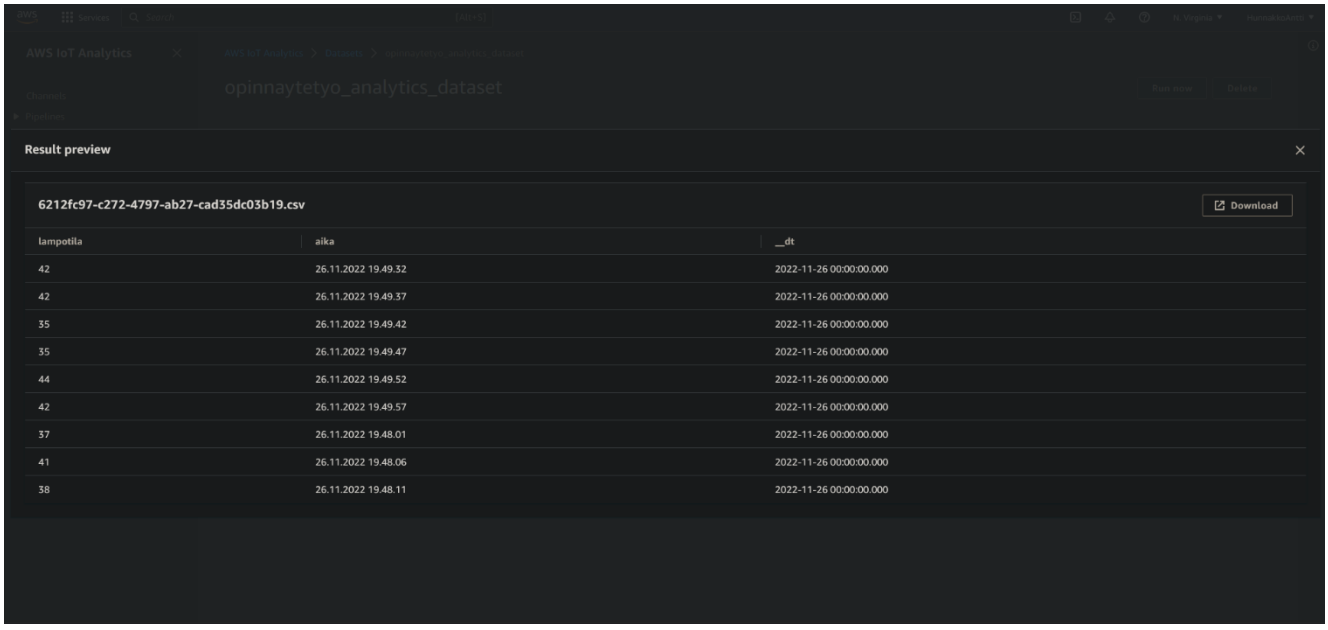
Seuraavaksi siirrytään sivun yläosion hakukenttää käyttäen takaisin ”IoT Core” palveluun. Luodaan sääntö sisään tulevalle datalle, jolla se ohjataan juuri luomiimme ”IoT Analytics” resursseihin. Valitaan sivun vasemmasta reunasta ”Message routing”, ja ”Rules”. Valitaan sivun yläosasta ”Create rule”. Annetaan tälle säännölle nimi, ja valitaan seuraava. Seuraavaksi valitaan, mitä, mistä, ja millä ehdoilla halutaan siirtää resursseihin. Valitaan haettavaksi kaikki tieto lisäämällä kohtaan ”Attribute” *-merkki. Ja syötetään kohtaan ”Topic filter” se aihe, johon ohjelma lähettää dataa. Esimerkissä ”Opinnaytetyo/lampotila”. Komennon lopullinen muoto voidaan nähdä kuviossa 29.



Kuvio 29. Säännön muokkaus, ja datan lähteen valinta

Valitaan sivun alareunasta ”Next”, ja seuraavaksi valitaan, mihin tämän säännön data lähetetään. Valitaan ensin pudotusvalikosta ”IoT Analytics”, jonka jälkeen valitaan kohtaan ”Channel name”, aiemmin luotu kanava, ja kohtaan ”IAM Role”, aiemmin luotu rooli. Tämän jälkeen valitaan ”Create”, ja sääntö on luotu. Tämän jälkeen pilvipalveluun lähetetty data, joka vastaa tätä sääntöä, lähetetään eteenpäin analytiikkaan, jossa se varastoidaan.

Seuraavaksi voidaan siirtyä takaisin ”IoT Analytics” palveluun, ja tarkistaa, että data saapuu perille, ja se tallentuu oikein. Valitaan sivun vasemmasta reunasta ensin ”Datasets”, sitten listasta aiemmin luotu resurssi, tämän jälkeen painetaan sivun yläosasta ”Run now”. Tämä hakee varastosta siellä olevan data, ja luo siitä raportin. Valitse sitten sivun keskeltä välilehti ”Content”, johon äsken luotu raportti on tallentunut. Avaa raportti, ja tarkasta, että data on saapunut säilytykseen oikein, kuten kuviossa 30.



Kuvio 30. Analytiikan datasta luotu raportti.

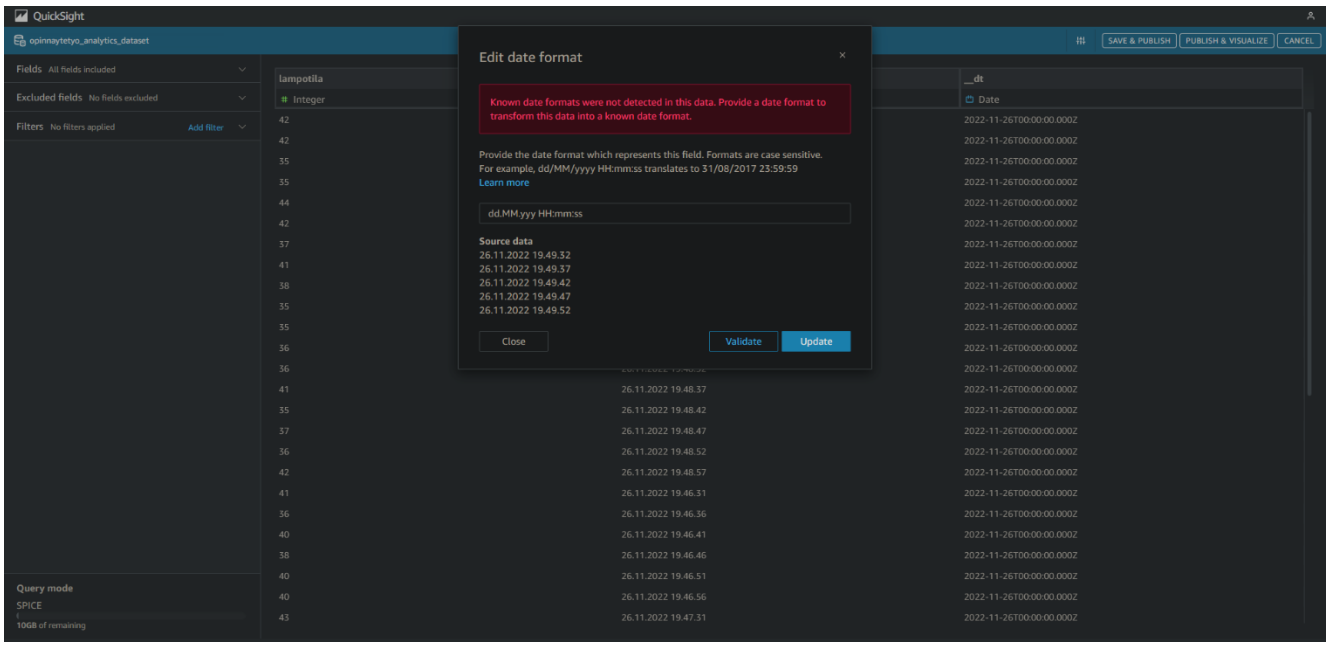
4.3.4 Datan visualisointi

Nyt varastoitu data on oikeassa muodossa. Seuraavaksi luodaan datalle visualisointi, jotta sitä on helpompi tulkita. Siirrytään sivun yläosassa olevaan hakukenttään, ja haetaan palvelua ”QuickSight”, ja siirrytään sinne. QuickSight vaatii erillisen tilauksen, mutta sen ilmaiskokeilua voi käyttää pieniin projekteihin. Maksulliseen palveluun kuuluva ”IoT SiteWise”, on esineiden internet visualisointien valmistamiseen tarkoitettu palvelu, mutta siitä ei ole

ilmaiskokeilua, joten käytämme QuickSight palvelua. QuickSight palvelun varjopuolena on se, että dataa ei voi seurata sieltä reaaliajassa.

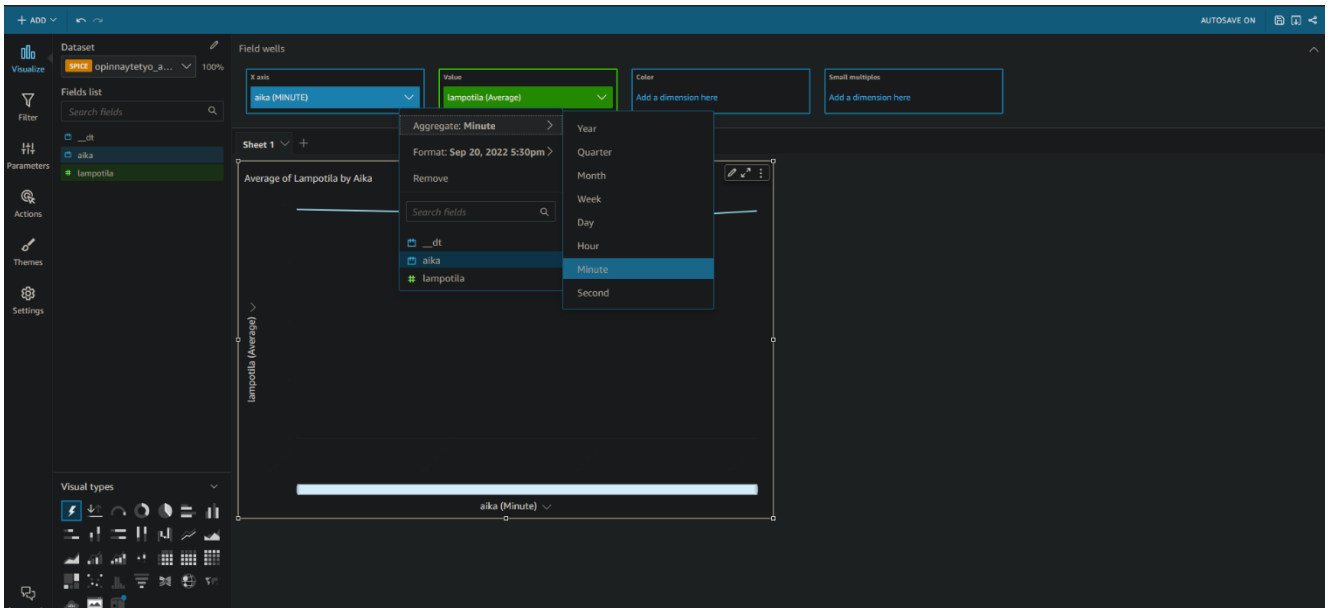
QuickSight palvelussa siirrytään ensin sivun vasemmassa reunassa olevaan ”Analyses” valikkoon, ja valitaan ”New Analysis”. Luodaan uusi datasetti yläosassa olevasta ”New dataset” painikkeesta, valitaan ”IoT Analytics”, ja sieltä aiemmin luotu datasetti, jonka raporttia aiemmin tarkastelimme. Lopuksi valitaan ”Edit/Preview data”, koska aikaleima tulee muuttaa oikeaan muotoon.

Muokkaustilassa valitaan yläreunassa olevan ”aika” kolumnin alla oleva teksti ”string”, ja muutetaan sen muodoksi ”Date”. Sivu ei automaattisesti tunnista aikaleiman muotoa, joten se syötetään käsin kenttään muodossa: ”dd.MM.yyyy HH.mm.ss”, kuten kuviossa 31. on esitetty. Tämän jälkeen palvelu tunnistaa datan päiväykseksi. Painetaan ”Update”, ja ylhäältä ”Publish & visualize” painikkeesta siirrytään datan esittämiseen.



Kuvio 31. Aikaleiman datatyypin muokkaus

Nyt voidaan visualisoinnin ikkunan vasemmasta reunasta valita datasetissä olevaa dataa, joka sijoitetaan automaattisesti valmiina olevaan visualisointiin. Valitaan vasemmalta aika ja lämpötila, jonka jälkeen sivun yläosassa näkyy valitut muuttujat. Muutetaan aika sekunneiksi, ja lämpötila arvo keskiarvoksi, kuten kuviossa 32. Tämän jälkeen visualisoinnin kuvaaja näyttää meille lämpötila-arvon joka sekunti.



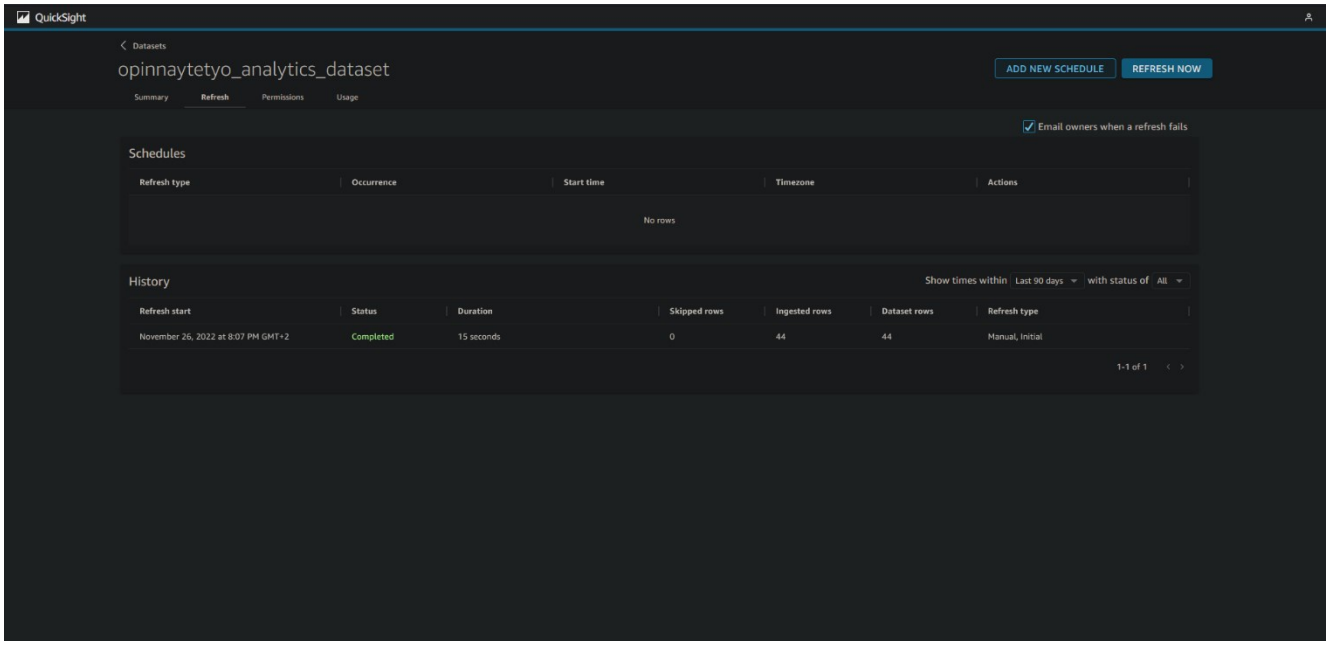
Kuvio 32. Visualisoinnin datan esitystavan muuttaminen

Lopuksi oikeasta yläreunasta valitaan ”Share”, ja ”Publish to dashboard”. Annetaan infonäytölle nimi, ja tämän jälkeen se löytyy pääsivun ”Dashboards” valikosta, jossa luotu näkymä voidaan nähdä ilman muokkausvalikkoja, kuten kuviossa 33.

Kuvio 33. Valmis visualisointi Dashboard valikossa

Koska kuvaajalla esitetty data on haettu aiemmin luodusta datasetistä, ei se ole päivity reaaliajassa. Näkymä kuitenkin voidaan laittaa päivittymään viiveellä, mutta pienin viive on yksi tunti. Painetaan sivun yläreunassa olevaa ”QuickSight” logoa, joka siirtää tekijän takaisin pääsivulle. Tämän jälkeen avataan ”Datasets” valikko, josta valitaan äsken käytetty datasetti.

Sivun yläreunasta avataan ”Refresh” välilehti, ja ylhäältä löytyy painike ”Add new schedule”, kuten kuviossa 34 on esitetty. Tämän kautta voimme lisätä tunneittain päivittyvän datasetin.



Kuvio 34. Datasetin tiedot, ja automaattisen päivityksen lisääminen

Pilvipalveluun on nyt luotu laite, jonka dataa voidaan tarkastella pilvessä. Pilveen saapuva data siirretään säännön mukaan sille luotuun analysointiin, jossa se kerätään yhteen, siirretään varastoon, ja lopulta datasettiin, josta voidaan valmistaa visualisointia.

Palvelusta löytyy myös ”CloudWatch” niminen palvelu, josta voi seurata esimerkiksi, kuinka paljon dataa kukin datan siirtoa käsittelevä sääntö on vastaanottanut tai jos saapuvassa datassa on ollut virheitä, eikä sitä ole osattu tulkita.

5 JOHTOPÄÄTÖKSET JA KEHITTÄMINEN

5.1 Johtopäätökset

Opinnäytetyön tavoitteena oli saa lukijalle ymmärrys esineiden internetistä, sekä antaa ohjeistus esineiden internet pilvipalvelujen käytön aloittamiseen. Molempien palvelujen testaamisen jälkeen saadaan niiden toiminnasta hyvä ymmärrys. Tiedetään, mikä on se prosessi, mitä yksittäinen laite vaatii, jos se halutaan yhdistää palveluun. Tiedetään myös, miten palveluun saapuvaa dataa käsitellään, ja miten sitä siirretään eteenpäin seuraavalle palvelulle, ja on nähty, millaista visualisointia palvelussa voidaan toteuttaa.

Microsoft Azure on keskittänyt esineiden internet palvelut hyvin IoT Central-palveluun. Laitteiden lisääminen ja datan käsittelyn prosessi on tämän työn tekijän mielestä tehty hyvin, ja kun sen on kerran käynyt läpi, on laitteiden, laitemallien ja visualisoinnin rakentaminen nopeaa ja yksinkertaista. Visualisointi on myös sisällytetty ilmaiseen kokeiluversioon, ja siellä näkyvä reaaliajassa oleva data on loistava etu. Haittapuolena IoT-Central palvelussa on se, että se piilottaa käyttäjältä sen automaattisesti luoman IoT Hub-palvelun yhteyspisteen nimen. Jos laitteessa ei pystytä käyttää ohjelmistonkehittämispakettia, täytyy tämä osoite ensin itse etsiä toisella ohjelmistolla, jonka jälkeen sitä voidaan käyttää laitteen yhdistämiseen. Ohjelmistonkehityspaketti on kuitenkin tarjolla monella ohjelmointikielellä, ja sen käyttämä DPS-järjestelmä, joka automaattisesti ohjaa ja varmistaa yhteyden, on helppokäyttöinen.

Amazon on selvästi erotellut palvelut toisistaan. Laitteiden lisääminen, datan käsittely ja visualisointi tapahtuvat kaikki omissa palveluissaan. Näiden palveluiden integrointi toisiinsa on kuitenkin helppoa, ja tapahtuu yleensä pudotusvalikosta oikea palvelu valitsemalla. Palveluihin luodut resurssit saavat aina myös uniikin osoitteen, jolla niihin yhdistäminen on helppoa. Myös roolien lisääminen palveluun on hyvä idea. Rooleilla voidaan helposti säätää, millä palvelulla tai henkilöllä on pääsy tiettyihin asioihin pilvessä. Haittapuolina esineiden internet palveluissa on laitteiden lisäämiseen käytettävät sertifikaatit, joiden ymmärtäminen saattaa olla hankalaa. Periaatteena kuitenkin sertifikaatti on hyvä. Toinen haittapuoli on ilmaisversiossa oleva kömpelö visualisointi. QuickSight-palvelu ei ole esineiden internettiin kohdennettu palvelu, ja vaikka siellä datan esittäminen on helppoa, se ei tapahdu reaaliajassa, mikä on suuri haittapuoli. Maksullinen IoT SiteWise-palvelu kuitenkin korjaa tämän kömpelön visualisoinnin.

5.2 Esimerkkisovellusten kehittäminen

Tämän työn pohjalta, ja esimerkkisovelluksiin tutustumalla voitaisiin rakentaa parempia simulaatio sovelluksia. Tällaisilla sovelluksilla voitaisiin esittää kiinnostuneille yrityksille esineiden internetin mahdollisuutta pilvipalveluissa.

Voidaan esimerkiksi luoda sovellus, jonka tarkoituksena on simuloida pienimuotoista tehdasta ja sen laitteita. Sovellus voitaisiin rakentaa sillä periaatteella, että se on helposti muokattavissa, jotta sovelluksen simuloima data vastaisi sen yrityksen tuotantomallia, jolle sitä esitellään. Tämä tarkoittaisi sitä, että anturien ja laitteiden nimet olisivat helposti muutettavissa ja laitteita voidaan lisätä ja poistaa helposti. Rakennettaisiin sovellukselle hyvä graafinen käyttöliittymä, josta laitteiden luonti tapahtuisi painikkeista.

Valmiilla sovelluksella voitaisiin lisätä pilvipalvelun laitteeseen yhdistämiseen vaaditut tiedot yhdelle sivulle, ja tämän jälkeen aloittaa laitteiden lisääminen omien ryhmiensä alle. Esimerkiksi Levyntyöstökone1, jonka alla tietoa tämän koneen antureista ja laskureista, joissa käsiteltyjä kappaleita tai koneen työtehokkuutta lasketaan.

Näin pystyttäisiin luomaan yrityskohtaisia esimerkkejä, jossa data on heille tutumpaa, ja helposti lähestyttävämpää kuin satunnainen data kuvitteellisista antureista.

LÄHTEET

Amazon. (8.10.2015). *Amazon Web Services Announces AWS IoT.* https://press.aboutamazon.com/2015/10/amazon-web-services-announces-aws-iot

Amazon. (i.a.-a). *Build IoT Solutions for Free on AWS.* Haettu 20.10.2022, https://aws.amazon.com/free/iot/

Amazon. (i.a. -b). *Internet of Things (IoT),* Haettu 20.10.2022. https://docs.aws.amazon.com/whitepapers/latest/aws-overview/internet-of-things-services.html

Beckhoff. (i.a.) *Application Samples.* Haettu 20.11.2022, https://infosys.beckhoff.com/english.php?content=../content/1033/tf6701_tc3_iot_communication_mqtt/3528168587.html&id=

Bexell, O. (2020). *Things you should know about the Internet of things*. Solentro.

Cloudflare. (i.a.). *What is the cloud?* Haettu 15.10.2022, https://www.cloudflare.com/en-gb/learning/cloud/what-is-the-cloud/

Collin, J., & Saarelainen, A. (2016). *Teollinen internet*. Talentum.

IoT.nxt. (18.7.2018). *IoT And SCADA: Is One Going To Replace The Other?* Haettu 25.10.2022, https://www.iotnxt.com/iot-or-scada/

Martinsuo, M., Kärri, T., & Aarikka-Stenroos, L. (2017). *Teollinen internet uudistaa palveluliiketoimintaa ja kunnossapitoa*. Kunnossapitoyhdistys Promaint ry.

Microsoft. (i.a.-a). *Azure Services.* Haettu 18.10.2022, https://azure.microsoft.com/en-us/products/

Microsoft. (i.a.-b). *Build in the cloud with an Azure free account.* Haettu 18.10.2022, *https://azure.microsoft.com/en-us/free/*

Vailshery, L. (15.6.2022). *Microsoft Azure - Statistics & Facts.* Statista https://www.statista.com/topics/8031/microsoft-azure/