HAMK
Hämeen ammatti-
korkeakoulu

Älykotijärjestelmän toteutus ja hallinta virtuaaliympäristössä

Ammattikorkeakoulututkinnon opinnäytetyö

Tieto- ja viestintätekniikka, insinööri (AMK)

Syksy 2025

Tanja Heikkilä

HAMK
Hämeen ammatti-
korkeakoulu

Tiivistelmä

Koulutus	Tieto- ja viestintätekniikka, insinööri (AMK)	
Tekijä	Tanja Heikkilä	Vuosi 2025
Työn nimi	Älykotijärjestelmän toteutus ja hallinta virtuaaliympäristössä	
Ohjaaja	Timo Karppinen	

Opinnäytetyön tavoitteena oli toteuttaa keskitetty Internet of Things (IoT) -laitteiden hallintajärjestelmä, johon on mahdollista kirjautua etänä tietoturvallisesti internetin kautta maksuttomia palveluita hyödyntäen. Nykyaikaiset kodit on varustettu usein eri valmistajien IoT-laitteilla, joiden hallinta tapahtuu useiden eri käyttöliittymien kautta. Tässä työssä toteutettiin kodin IoT-laitteiden keskitetty hallintajärjestelmä HomeAssistant -ohjelmistolla virtualisoituun Docker-ympäristöön vastaamaan kyseiseen tarpeeseen.

Työn tavoitteena oli myös arvioida toteutetun järjestelmän tuotteistettavuutta ja pohtia, millaisia kaupallisia tuotteita sen pohjalta voitaisi kehittää. Järjestelmä olisi tuotteistettavissa esimerkiksi järjestelmän käyttöönotossa avustavaksi sovellukseksi tai IoT-laitteen mukana toimitettavaksi ohjelmistoksi.

Työssä esiteltiin virtualisoinnin ja IoT:n keskeisiä käsitteitä ja niihin liittyvää termistöä. Työssä kuvaillaan järjestelmän kokoonpanossa käytetty laitteisto, ohjelmistot ja konteissa ajettavat palvelut. Järjestelmä toteutettiin laajalti käytettyjen, yhteisöpohjaisten avoimen lähdekoodin projektien – Linuxserver.io:n ja Home Assistantin – virallista dokumentaatiota noudattaen. Järjestelmään liitettiin kaksi IoT-laitetta, joiden avulla integraatioiden ja automaatioiden toimintaa testattiin käytännössä.

Järjestelmä saatiin toteutettua toimivaksi asetettujen tavoitteiden mukaisesti. Tarve jatkokehitykselle kuitenkin jäi sekä järjestelmän että tuotteistamisen kannalta. Tästä huolimatta, järjestelmän toteuttaminen virtuaaliympäristössä tarjosi hyvät olosuhteet testata järjestelmän toimintaa kokonaisuutena, sekä syventää käytännön osaamista konttiteknologian parissa.

Avainsanat	Docker, Home Assistant, Internet of Things, virtuaaliympäristö
Sivut	56 sivua ja liitteitä 7 sivua

HAMK
Hämeen ammatti-
korkeakoulu

Abstract

DP	Information and Communication Technology	
Author	Tanja Heikkilä	Year 2025
Subject	Implementation and Management of a Smart Home System in Virtual Environment	
Supervisor	Timo Karppinen	

The aim of this thesis was to implement a centralized management system for Internet of Things (IoT) devices which can be accessed remotely in a secure way using free services. Modern homes are often equipped with IoT devices from various manufacturers, whose management is made through several different user interfaces. In this thesis, a centralized system for home IoT devices was implemented using Home Assistant software in a virtualized Docker environment to address this need.

The aim of the thesis was also to evaluate the productization of the implemented system and to consider what kinds of commercial products could be developed based on it. For example, the system could be productized as an application assisting the installation of the system or as software bundled with an IoT device.

The thesis introduces the key concepts and terminology related to virtualization and IoT. The thesis describes the hardware, software and containerized services used in the system. The system was implemented by following the documentation of widely used official community-driven open-source projects – Linuxserver.io and Home Assistant. Two IoT devices were connected to the system to test the functionality of integrations and automations in practice.

The system was successfully implemented according to the defined objectives. However, there remains a need for further development, both in the system itself and in productization. Nevertheless, implementing the system in a virtual environment provided good conditions to test the overall functionality of the system, and enhance practical skills in container technology.

Keywords	Docker, Home Assistant, Internet of Things, virtual environment
Pages	56 pages and appendices 7 pages

Sisällys

Kuvat

Liitteet

1 Johdanto

Verkkoon kytketyt IoT-laitteet ovat tulleet jo monissa kodeissa osaksi arkipäivää. Useissa tilanteissa näitä laitteita hallitaan omista valmistajakohtaisista käyttöliittymistä. Tämä luo tarpeen laitteet yhdistävälle, keskitetylle hallintajärjestelmälle, jonka kautta eri valmistajien laitteita voidaan hallita samasta käyttöliittymästä. (RouteThis, 2024)

Opinnäytetyössä toteutettiin kodin IoT-laitteiden älykäs hallintajärjestelmä HomeAssistant -ohjelmistolla virtualisoituun Docker-ympäristöön. Idea toteutukselle lähti kirjoittajan omasta tarpeesta: kotiin oli vuosien saatossa kertynyt useita kotiautomaatioon liittyviä IoT-laitteita, joiden hallintaa haluttiin yhtenäistää lisäämällä ne kaikki samaan järjestelmään. Lisäksi kotiautomaation piiriin ollaan lisäämässä uusia IoT-laitteita, kuten sähköauton latausasema ja aurinkosähköinvertteri, jolloin niiden keskitetyllä ohjaamisella kodin energian kulutusta voitaisiin optimoida tulevaisuudessa.

Opinnäytetyön päätavoitteena oli toteuttaa keskitetty IoT-laitteiden hallintajärjestelmä, johon voidaan kirjautua etänä tietoturvallisesti internetin kautta maksuttomia palveluita ja sovelluksia hyödyntäen. Aihetta internetissä tutkimalla ilmeni, että etäkirjautuminen oli mahdollista toteuttaa helposti Docker-konteilla, mikä oli keskeinen syy järjestelmän toteuttamiseen virtualisoidusti. Valittu toteutustapa toi myös tilaisuuden tutkia virtualisoinnin tuomia mahdollisuuksia kotiautomaation hallinnassa ja tuotteistamisessa.

Toteutettua järjestelmää arvioitiin myös tuotteistettavuuden kannalta. Virtualisointi teki järjestelmästä uudelleenasennettavan, siirrettävän ja laajennettavan jatkossa, mikä tuki tuotteistettavuutta. Opinnäytetyön toteutuksen aikana järjestelmään liittyväksi tuotteeksi muodostui ajatus asennussovelluksesta, joka auttaisi järjestelmän käyttöönotossa vaihe vaiheelta, jotta järjestelmä on mahdollista ottaa käyttöön virheettömästi muissakin käyttöjärjestelmissä.

Tietoturva oli keskeinen näkökulma järjestelmän toteutuksessa. Käytettyjen ohjelmistojen ja konfiguraatioiden tietoturvatekijöitä arvioitiin seuraavilla osa-alueilla. Opinnäytetyössä pohdittiin EU:n markkinoilla digitaalisten tuotteiden myyntiä koskevan kyberkestävyyssäädöksen ehtojen soveltamista järjestelmästä suunniteltuun tuotteeseen.

Lisäksi toteutetun järjestelmän konfiguraatioiden turvallisuutta arvioitiin hyödyntämällä konttiteknologiassa hyväksi havaittuja käytäntöjä ja suosituksia.

2 Virtualisointi

Tässä opinnäytetyössä virtualisoinnilla tarkoitetaan tietojenkäsittelylieteen tekniikkaa, jossa yhden tietokoneen laitteiston päälle luodun ohjelmiston avulla tietokoneen resurssit, kuten muisti, prosessorit ja tallennustila voidaan jakaa useille eri virtuaalikoneille. Jokaiselle virtuaalikoneelle on asennettu oma käyttöjärjestelmänsä, ja ne toimivat isäntäkoneen jakamia resursseja käyttäen itsenäisesti kuten tavalliset tietokoneet. (Susnjara & Smalley, 2025)

Tässä opinnäytetyössä virtualisoinnista puhuttaessa, usein toistuvia keskeisiä termejä ovat mm. isäntäkone (host machine, tai vain host), jolla tarkoitetaan tietokonetta, jossa virtualisointiympäristö toimii. Vieraskone (guest machine tai guest), tarkoittaa virtuaalikoneeseen asennettua käyttöjärjestelmää. Instanssi (instance) tarkoittaa yksittäistä virtuaalikonetta. (TEPA-termipankki, n.d.)

Termillä konttiteknologia (container technology) tarkoitetaan teknologiaa, jossa pienikokoiset suoritettavat ohjelmiston palaset paketoivat sovelluksen koodin ja riippuvuudet, kuten binäärikoodin, kirjastot ja konfiguraatiotiedostot, helposti eri tietokoneympäristöihin siirrettäviksi kokonaisuuksiksi. Docker on yksi konttiteknologiaa hyödyntävä ohjelmisto. (SolarWinds, n.d.)

2.1 Virtualisoinnin historia

Virtualisoinnin voidaan katsoa saaneen alkunsa 1960-luvulla, jolloin virtualisointia alettiin kehittämään ratkaisuksi mahdollistamaan useiden käyttäjien suorittaa yhtäaikaisesti eräajoja keskustietokoneisiin. Tuolloin keskustietokoneen laitteistoresurssit ja ohjelmistot olivat vahvasti sidoksissa toisiinsa ja infrastruktuuri oli muutenkin joustamaton: tietokoneilla voitiin suorittaa vain yhtä ohjelmaa kerallaan, jotta ohjelmistokonflikteilta vältyttiin. Ratkaisuksi ongelmiin yhdysvaltalainen teknologiayritys IBM (International Business Machines Corporation) kehitti virtuaalikoneen, jonka avulla useat käyttäjät pystyivät käyttämään samaa System/360-keskustietokonetta. Näitä virtuaalikonetta hyödyntäviä järjestelmiä olivat IBM CP-40 ja myöhemmin IBM CP-67. (Randal, 2020, s. 6)

1970-luvun alussa IBM jatkoi kehitystyötä julkaisemalla VM/370-virtuaalikoneen, jota kaupattiin nyt virallisesti virtuaalikoneen nimellä. Virtuaalikoneella oli mahdollista käyttää edelleen kehitettyä System/370-keskustietokonetta, ja se sisälsi oman virtuaalimuistilaitteen. Virtualisointijärjestelmän laitteiden liitettävyyttä oli parannettu ohjelmistokomponentilla, jonka avulla tulostaminen ja muiden fyysisten isäntäkoneeseen kytkettyjen laitteiden käyttö oli mahdollista etänä eli virtuaalikoneesta käsin (IBM, 2022).

Tietokoneiden komponentit olivat kalliita 1960- ja 1970-luvuilla, jolloin virtuaalikoneiden kehityksen ja käytön suosiota selitti mahdollisuus käyttää samaa kallista keskustietokonetta usealta eri käyttöjärjestelmäinstanssilta. 1980- ja -90-luvuilla kiinnostus keskustietokoneen virtualisointiin kuitenkin väheni, kun tietokonelaitteistojen hinnat laskivat ja saatavuus kasvoi. Kehitys alkoi suuntautua käyttöjärjestelmiin, minkä myötä henkilökohtaiset työasemat yleistyivät. (Bugnion ym., 2012, s. 2)

Merkittävä vaihe virtualisoinnin historiassa on vuonna 1979 Unix Version 7 -käyttöjärjestelmässä käyttöön otettu *chroot*-järjestelmäkutsu. Sen kehittivät yhdysvaltalainen yritys Bell Labs yhdessä kalifornialaisen yliopiston, Berkeleyn kanssa. Järjestelmäkutsu sisältyi myös 1980-luvun alussa julkaistuun BSD-käyttöjärjestelmään (Berkeley Software Distribution), joka oli yksi Unix-käyttöjärjestelmän variantti.

Chroot-järjestelmäkutsun avulla voidaan muuttaa prosessin ja sen mahdollisten lapsiprosessien juurihakemisto määritettyyn sijaintiin tiedostojärjestelmässä. Tämän seurauksena prosessi pääsee käyttämään vain kyseistä osaa tiedostojärjestelmästä. Juurihakemiston muuttaminen luo eristetyn ympäristön prosessille ja sen lapsiprosesseille, jossa prosessi ei pääse vaikuttamaan koko tiedostojärjestelmään. Näin esim. haitallinen ohjelmakoodi ei pääse vaikuttamaan koko järjestelmään, vaan pysyy eristettynä. *Chroot*-järjestelmäkutsun sisällyttäminen käyttöjärjestelmiin paransi niiden tietoturvaa ja loi pohjan nykyisen kaltaisen konttiteknologian kehitykselle. (TheLinuxCode, 2024)

Vuonna 1999 ohjelmistoyritys VMware julkaisi ensimmäisen x86-tietokonearkkitehtuurille suunnitellun virtualisointiohjelmiston, VMware Workstationin (Randal, s. 12). Virtualisointiohjelmiston asentaminen tietokoneeseen oli suunniteltu siten, että olemassa olevaan laitteistoon tai ohjelmistoon ei tarvinnut tehdä muutoksia. Ohjelmiston toteuttamisessa oli haasteensa, sillä x86-arkkitehtuuri oli monimutkainen ja suunniteltu tukemaan vuosikymmeniäkin vanhojen ohjelmistojen yhteensopivuutta. Arkkitehtuurista puuttui myös sisäänrakennettu tuki virtualisoinnille, minkä vuoksi virtualisointiohjelmistoja lähdettiin tuolloin suunnittelemaan käyttöjärjestelmäkohtaisesti. (Bugnion ym., 2012, s. 5)

Yksi ensimmäisistä hyvin toteutetuista konttiteknologioista oli LXC (Linux Containers), joka oli suunniteltu toimimaan suoraan Linux-ytimessä. Julkaisunsa jälkeen LXC:tä käytettiin suunnittelun pohjana useassa muussakin konttiteknologioissa. Yksi niistä oli vuonna 2013 julkaistu Docker, josta tuli nopeasti suosittu, sillä se erottui muista ohjelmistoista kokonaisvaltaisella konttien hallintaekosysteemillään. Docker kasvattikin merkittävästi konttiteknologian suosiota IT-alalla. (Osnat, 2020).

2.2 Virtuaalikone ja hypervisor

Virtuaalikone ja hypervisor ovat keskeisiä käsitteitä virtualisointitekniikoissa puhuttaessa. Virtualisointi tapahtuu virtuaalikoneen (Virtual Machine, VM) ja hypervisorin, kutsutaan myös nimellä ”Virtual Machine Monitor” eli VMM, avulla. Virtuaalikoneet ovat toisistaan erillään olevia järjestelmiä, jotka käyttävät isäntäkoneen laitteistoresursseja, mutta näyttäytyvät käyttäjälle kuin ne olisivat erillisiä tietokoneita käyttöjärjestelmineen. Virtuaalikoneelle on ominaista, että se on mahdollista siirtää tietokoneelta toiselle, jossa se käyttää uuden tietokoneen resursseja ja sen käyttöjärjestelmä toimii kuten edellisessäkin isäntäkoneessa.

Hypervisor on ohjelmisto, joka mahdollistaa uusien virtuaalikoneiden luomisen ja olemassa olevien virtuaalikoneiden hallinnan erottamalla isäntäkoneen virtuaalikoneille varatut resurssit laitteistosta ja allokoimalla ne edelleen virtuaalikoneiden käyttöön. Toisin sanoen, virtuaalikoneet tunnistavat sijaitsevansa ainoastaan hypervisorissa. (RedHat, 2024)

Hypervisoreita on kahden tyyppisiä. 1-tyypin hypervisor toimii suoraan isäntäkoneen laitteistossa toimien kuten paikallinen käyttöjärjestelmä, josta laitteistoresurssit jaetaan suoraan virtuaalikoneelle. Tämän tyyppinen hypervisor on yleinen datakeskus- ja muissa palvelinympäristöissä. (RedHat, 2024)

2-tyypin hypervisor puolestaan toimii tavanomaisen käyttöjärjestelmän, kuten Windowsin päällä ohjelmistokerroksena tai sovelluksena toimien ns. erottelevana kerroksena virtuaalikoneiden ja isäntäkoneen käyttöjärjestelmän välillä aikatauluttaen virtuaalikoneiden resurssien käytön isäntäkoneen käyttöjärjestelmän resurssien käytön lomaan. Tämän tyyppinen hypervisor soveltuu käytettäväksi työasemalla, jossa halutaan olemassa olevan käyttöjärjestelmän lisäksi työskennellä jollakin toisella käyttöjärjestelmällä. (RedHat, 2024)

2.3 Virtualisointitekniikoita

Täysvirtualisointitekniikassa virtuaalikone luo simuloidun laitteiston, jonka päällä vieraskäyttöjärjestelmää voidaan ajaa sellaisenaan muokkaamattomana. Täysvirtualisointi voidaan jakaa kahteen tyyppiin, ohjelmistoavusteiseen ja laitteistoavusteiseen täysvirtualisointiin. Ohjelmistoavusteisessa täysvirtualisoinnissa laitteisto emuloidaan, eli ohjelmistolla luodaan virtuaalinen laitteisto käyttäen binäärikäännöstekniikkaa. Binäärikäännöstekniikkaa on soimattu sen suorituskykyyn liittyvissä ongelmissa. Joitakin tunnettuja binäärikäännöstekniikkaa käyttäviä ohjelmistoja ovat 32-bittiset VMware Workstation ja VirtualBox, jotka toimivat mm. Windows 11 -käyttöjärjestelmässä. (Cloud_Devops, 2017)

Laitteistoavusteinen täysvirtualisointi nojautuu modernien, eli vuonna 2005 jälkeen valmistettujen x86-arkkitehtuurin prosessorien sisältämään omaan virtualisointitekniikkaan binäärikäännöstekniikan sijaan. Tällaisia ovat mm. Intel VT-x- ja AMD-V-tekniikalla toteutetut prosessorit, joiden virtualisointitekniikka mahdollistaa vieraskäyttöjärjestelmän suorittaa tiettyjä järjestelmäkäskyjä suoraan prosessorissa, mikä osaltaan tekee tästä nopeamman tekniikan verrattuna ohjelmistoavusteiseen täysvirtualisointiin. Tällä virtualisointitekniikalla toteutetut ohjelmistot voidaan edelleen jakaa kahteen alaryhmään: tyypin 1 hypervisoreihin, eli ns. *bare-metal*-toteutuksiin ja tyypin 2, eli ns. isännöityihin hypervisoreihin. (Cloud_Devops, 2017)

Paravirtualisointitekniikassa virtuaalikoneen vieraskäyttöjärjestelmälle ei simuloida laitteistoa, vaan hypervisor on asennettu suoraan tietokoneeseen ja vieraskäyttöjärjestelmä on asennettu hypervisoriin. Tekniikalle erityistä on, että vieraskäyttöjärjestelmän pohjakoodia on muokattu sellaiseksi, että koko järjestelmä on tietoinen virtuaalikoneessa sijaitsevasta vieraskäyttöjärjestelmästä, jonka ansiosta sen on mahdollista suorittaa tietynlaisia järjestelmäkäskyjä isäntäkoneen laitteistossa. (Cloud_Devops, 2017)

Hybridivirtualisointitekniikassa on yhdistetty täysvirtualisoinnin ja paravirtualisoinnin tekniikoita. Sen tarkoitus on ollut yksinkertaistaa paravirtualisointitekniikan monimutkaisuuden aiheuttamaa epätehokkuutta laitteistoajurien avulla varsinkin fyysisen laitteiston käytön ja paljon muistitehoa käyttävien työkuormien osalta. (Cloud_Devops, 2017)

Käyttöjärjestelmätason virtualisointi (operating system level virtualization) tarkoittaa virtualisointitekniikkaa, jossa on mahdollista ajaa yhtä aikaa useita konttipohjaisia

käyttöympäristöjä jakamalla isäntäkäyttöjärjestelmän Linux-ydin niiden kesken. Käyttöjärjestelmätason virtualisointi on suorituskyvyltään tehokas, koska käyttöympäristöjen ohjelmat suoritetaan suoraan isäntäkoneen Linux-ytimessä ilman erillistä käyttöjärjestelmäkerrosta. Docker on yksi tunnetuimmista tätä virtualisointitekniikkaa hyödyntävistä ohjelmistoista. (Cloud_Devops, 2017)

2.4 Virtualisointikonsepteja

Palvelinvirtualisointi on konsepti, jossa fyysisen palvelimen resurssit jaetaan yksittäisiksi, toisistaan eristetyiksi virtuaalipalvelimiksi, joka puolestaan mahdollista usean eri käyttöjärjestelmän ajamisen samassa palvelimessa. Palvelinvirtualisointia käytetään nykypäivänä etenkin pilvipalveluiden yhteydessä, mm. koska se tehostaa palvelinresurssien käyttöä, mahdollistaa virtuaalikoneiden keskitetyn hallinnan ja uusien virtuaalikoneiden pystyttäminen ja käytöstä poistaminen on nopeaa. Kuten virtualisoinnissa yleisestikin, myös palvelinvirtualisoinnissa laitteistokomponentit virtualisoidaan hypervisorin avulla. Palvelinvirtualisoinnissa eniten käytetty tekniikka on laitteistoavusteinen täysvirtualisointi, mutta myös paravirtualisointia, käyttöjärjestelmätason virtualisointia sekä ohjelmistoavusteista täysvirtualisointitekniikkaa käytetään. (Fernandez, 2023)

Datan, eli tiedon virtualisointi on yksinkertaistettuna konsepti, jossa eri tietolähteissä sijaitseva data kerätään ohjelmistorajapintojen, kuten REST API, avulla yhteen virtualisoituun lähteeseen, eli virtualisointikerrokseen, josta se on suoraan käyttäjien, kuten data-analyytikoiden, ohjelmistokehittäjien tai vaikkapa sovellusten käytettävissä helposti. Virtualisointikerros näkyy käyttäjälle yhtenä virtuaalisena tietokantana, joka kuitenkin voi sisältää dataa erilaisista tietokantatyypeistä, relaatiotietokannoista tai ns. rakenteettomista tietokannoista. (Löppönen, 2021)

Muistin virtualisoinnilla tarkoitetaan pääpiirteittäin tekniikkaa, joka optimoi ja nopeuttaa tietokoneen muistin käyttöä luomalla abstraktiokerroksen RAM-muistin ja muistia käyttävän sovelluksen väliin. Abstraktiokerros hallinnoi, optimoi ja allokoi edelleen vain tietyt muistinosat, joita prosessit, ohjelmat ja virtuaalikoneet tarvitsevat. (Jakhar, n.d.)

Tallennusvirtualisoinnilla tarkoitetaan yksinkertaistetusti esitystapaa, jolla tietokone tai palvelin näkee fyysiset tallennusresurssit loogisessa muodossa. Palvelimet voivat esim. nähdä kaksi virtuaalista tallennustilaa, volyymia, jotka voivat olla osoitettuna eri sovellusten

tallennustilaksi, mutta todellisuudessa nämä kaksi volyymia muodostuvat esim. neljästä eri fyysisestä tallennuslaitteesta. (Storage Tutorials, n.d.)

Verkon virtualisoinnissa on periaatteena, että fyysiseen verkkoon on mahdollista luoda useita loogisia, eli toisistaan erillään olevia virtuaaliverkkoja abstraktoimalla fyysiset verkkolaitteet. Virtuaaliverkkojen yksi tärkeimmistä piirteistä on mahdollistaa eri käyttäjäryhmien ja verkkoympäristöjen verkkoliikennöinti turvallisesti keskenään. (GeeksforGeeks, 2024a)

Verkon virtualisointi voidaan toteuttaa useilla eri tavoilla, joita ovat mm. VLAN (Virtual Local Area Network) eli virtuaalilähiverkko, SDN (Software Defined Network) eli ohjelmallisesti toteutettu verkko, VPN (Virtual Private Network) eli virtuaalinen erillisverkko, sekä MPLS-tekniikka (Multi Protocol Label Switching). Edellä mainituista tekniikoista VLAN-tekniikkaa käytetään yleensä lähiverkon sisällä ja vastaavasti SDN-, VPN- ja MPLS-tekniikoita käytetään yleensä yhdistämään lähiverkot turvallisesti internetin yli. (GeeksforGeeks, 2024a)

3 IoT

IoT, eli *Internet of Things* -sanalla viitataan verkkoon, sekä siinä toimiviin fyysisiin laitteisiin, kuten esim. ajoneuvoihin, kodinkoneisiin ja älykelloihin, jotka keräävät ja jakavat dataa. Tällaista dataa on esim. anturien keräämä tieto lämpötilasta, ilmankosteudesta, ilmanlaadusta, energiankulutuksesta. Data voi siis olla mitä tahansa mitattavissa olevaa tietoa laitteen toiminnasta tai sen ympäristöstä. Kerätystä datasta voidaan analysoimalla tunnistaa toistuvia tapahtumia, kehityssuunnan muutoksia ja poikkeavia tapahtumia. Analysoitua dataa voidaan hyödyntää esim. yksittäisen laitteen tai kokonaisen tuotantolaitteiston toiminnan tehostamisessa ja valvonnassa.

IoT-laitteet ovat sulautettuja järjestelmiä, eli ne sisältävät mm. antureita, ohjelmiston, ja ne on mahdollista liittää verkkoon, minkä avulla ne keräävät ja jakavat dataa esim. toiselle IoT-laitteelle. IoT-laitteet voivat olla esim. ns. älykodin yksinkertaisia laitteita, kuten huonetermostaatteja, puettavia asusteita, kuten älykelloja, tai tuotantolaitteiston yksittäisiä osia. (IBM, 2023)

3.1 Historia

Ensimmäiset IoT-teknologiaan rinnastettavat sovellutukset kehitettiin 1980- ja -90-luvuilla, jolloin niiden toiminta oli alkukantaista, mutta ne kuitenkin täyttivät IoT:n määritelmän ainakin jossain määrin. Ensimmäisen IoT-sovellutuksen kerrotaan olleen erään yhdysvaltalaisen yliopiston juoma-automaatti, jonka täyttömäärää ja juomien lämpötilaa pystyttiin seuraamaan varhaisen internetin kautta juoma-automaattiin asennettujen mikrokytkimien ja lämpötila-anturin avulla. (Khvoynitskaya, 2019)

Vuonna 1999 ensimmäisen kerran termiä ”the internet of things” käytti teknologian tutkija nimeltä Kevin Ashton Procter & Gamble -nimiselle organisaatiolle pitämässään puheessa. Siinä hän kuvaili IoT:n olevan teknologinen ratkaisu toimitusketjun hallintaan, jossa useita laitteita yhdistetään toisiinsa RFID-tagien avulla. RFID (Radio Frequency Identification) tarkoittaa teknologiaa, jossa radiotaajuudella toimivan etätunnistuksen avulla on mahdollista seurata, tunnistaa ja yksilöidä tuotteita niiden toimitusketjun aikana. (Khvoynitskaya, 2019; Riffid, n.d.)

2000-luvun aikana yleistyivät nykyisen kaltaiset IoT-laitteet, kuten kuluttajille myytävät internetiin kytketyt kodinkoneet, jotka käyttävät kommunikointiin IP-, eli internet protokollaa. IP-protokolla on myös nykyisen internetin keskeinen käsite, sillä mm. data liikkuu internetissä IP-paketeissa, joiden otsikkotiedoista löytyy mm. paketin lähettäjän ja vastaanottajan IP-osoite, jonka perusteella paketit reitittyvät niiden vastaanottajalle. (Khvoynitskaya, 2019; Hakatemia, n.d.)

2010-luvulta tähän päivään saakka IoT-laitteet ovat yleistyneet koko maailmassa, ja niistä on tullut arkipäiväisiä esineitä kuluttajien keskuudessa, mutta ne ovat myös olennainen osa lähes jokaisella teollisuuden alalla. Suuret, monikansalliset teknologia-alan yritykset ovat keskittyneet kehittämään ja tuotamaan eri tarpeisiin soveltuvia antureita ja laitteita kuten termostaattijärjestelmiä, puettavia älylaseja ja jopa itsestään ajavia autoja. (Khvoynitskaya, 2019)

3.2 Kotiautomaatio ja älykodit

Kotiautomaatiolla tarkoitetaan kodinkoneiden ja kodin olosuhteita valvovien laitteiden hallintaa joko lähiverkon, tai etähallintayhteyden kautta. Sanalla automaatio tarkoitetaan tekniikkaa, menetelmää tai järjestelmää, jonka on tarkoitus vähentää ihmisen tarvetta

puuttua järjestelmään liitettyjen laitteiden suorittamiin toimintoihin. Kotiautomaatiojärjestelmä suunnitellaan tehokkuuden näkökulmasta: optimoimalla laitteiden käyttöä pienennetään kiinteistön sähkönkulutusta. (Gunge & Yalagi, 2016)

Kotiautomaatiojärjestelmä voidaan jakaa osiin seuraavasti:

- Käyttöliittymä, jonka kautta annetaan käskyt hallintajärjestelmälle. Käyttöliittymään päästään kirjautumaan esim. tietokoneella tai älypuhelimella.
- Liitäntätapa, kuten Ethernet-kaapelilla toteutettu yhteys, tai langaton yhteys esim. radioaalloilla, Bluetoothilla tai WLAN- eli langattomalla lähiverkkoyhteydellä toteutettu yhteys.
- Hallintajärjestelmä, ns. keskusyksikkö, joka toimii laitteiston kokoavana kytkentäpisteenä. Se ottaa vastaan käskyjä käyttöliittymältä, joiden perusteella se ohjaa laitteiden toimintaa.
- Sähkölaitteet, eli kotiautomaatiojärjestelmän kanssa yhteensopivat kodin laitteet, joita voidaan ohjata järjestelmään kytkettyinä. (Gunge & Yalagi, 2016)

Älykoti, engl. *smart home*, on asuintila, jossa voidaan hallita kodin laitteita ja toimintoja keskitetyn järjestelmän kautta. Hallinta voidaan suorittaa verkkoon kytketyn laitteen, kuten kannettavan tietokoneen, älypuhelimen tai tablettitietokoneen avulla. (Lake, 2025)

Älykotijärjestelmään kytkettäviä laitteita ovat mm. älykkäät termostaatit, valaistus eli lamput kytkimet, ohjattavat pistorasiat, älylukot, valvontakamerat, kodin murtosuojausjärjestelmät ja ääniohjattavat avustajat, kuten Google Assistant ja Siri. Kytkemällä laitteita älykotijärjestelmään niiden toimintaa voidaan automatisoida esim. aiempaan käyttöön perustuvan ennusteen avulla tai asettamalla laitteiden toiminnalle loogisia sääntöjä, kuten *jos huoneessa havaitaan liikettä, kytke valot päälle ja lisää huoneen lämpötilaa.* Älykotijärjestelmän tarkoitus on parantaa kodin asumismukavuutta automaatioiden ja laitteiden keskitetyn hallinnan avulla. (Lake, 2025)

3.3 Protokollat

IoT-laitteiden liittäminen esim. älykotijärjestelmään tapahtuu hyödyntämällä useita eri protokollia, kuten LAN, Zigbee, WiFi ja MQTT. Yhteysprotokollat määrittelevät mm. miten laitteet ”keskustelevat” keskustelevat keskenään ja miten yhteyden suojaus on toteutettu. Lyhyt kuvaus joistakin älykotijärjestelmissä käytetyistä protokollista:

MQTT – kevyt protokolla mm. sensorien ja sovellusten väliseen tiedon välitykseen. MQTT-protokolla muodostaa yhteyden TCP- ja IP-protokollien avulla. TCP, eli Transmission Control Protocol mahdollistaa viestien saapumisen perille alkuperäisessä muodossaan ja IP, eli Internet Protocol mahdollistaa viestien päätymisen oikeaan osoitteeseen (FiTech101, n.d.). Protokollan toiminta perustuu malliin, jossa *publisher* eli julkaisijakomponentti kerää ja välittää sensorilta saadun tiedon eteenpäin, *subscriber* eli tilaajakomponentti vastaanottaa sensoritiedon ja *broker* eli välittäjäkomponentti välittää tiedot julkaisija- ja tilaajakomponenttien välillä. (Microcontrollers Lab, n.d.)

IoT-toteutuksissa välittäjäkomponentin kautta tapahtuvalle tiedonsiirrolle ominaista on, että julkaisija- ja tilaajakomponenttien ei tarvitse olla samanaikaisesti yhteydessä välittäjäkomponenttiin, eikä niiden tarvitse olla tietoisia toisistaan. Julkaisija- ja välittäjäkomponentit toimivat toisistaan riippumatta, eli niiden toiminta ei keskeydy tiedon lähetyksen tai vastaanottamisen aikana. Julkaisija- ja tilaajakomponentteja voidaan lisätä ja poistaa järjestelmästä vapaasti. (HiveMQ Team, 2023)

Bluetooth ja BLE eli Bluetooth Low Energy – langattomia tiedonsiirtoprotokollia mm. IoT-laitteiden tiedonsiirtoon lyhyen kantaman sisällä. Molemmat protokollat toimivat samalla 2,4 GHz:n taajuudella. BLE-protokolla käyttää pienikulutuksista tiedonsiirtotekniikkaa, joten se soveltuu käytettäväksi laitteissa, kuten älykelloissa, joissa akun halutaan kestävän pitkään. Bluetooth-protokolla soveltuu paremmin sovelluksiin, jossa tarvitaan suuria tiedonsiirtonopeuksia reaaliajassa.

Zigbee 3.0 - langaton tiedonsiirtoprotokolla IoT-laitteiden väliseen kommunikointiin saumattomasti eri verkkojen välillä. Protokollan ominaisuuksia ovat mm. matala virrankulutus, matala tiedonsiirtonopeus ja 10–100 m:n toimintasäde. Protokolla on myös luotettava, joten sitä käytetään myös teollisuuden IoT-laitteissa.

Wi-Fi – tiedonsiirtotekniikka, joka toimii langattomassa lähiverkossa. Tekniikka on laajasti käytössä mm. nykyaikaisissa, ns. kodin langattomissa reitittimissä ja se mahdollistaa nopean tiedonsiirron suurille datamäärille. WiFi-tekniikka perustuu IEEE 802.11-standardiin. (Microcontrollers Lab, n.d.)

IoT:n protokollat poikkeavat toimintavarmuuden kannalta teollisuusautomaation protokollista. IoT-protokollissa tieto lähetetään tyypillisesti vain tilan muutoksesta, kuten valaistuskytkimen painalluksesta. Teollisuusautomaation protokollille ominaista on lähes reaaliaikainen tiedonsiirto, jossa kaikki anturitiedot, kytkintiedot, asetusarvotiedot ja

ohjaustiedot lähetetään jatkuvasti tyypillisesti vähintään kerran kymmenessä millisekunnissa. (Delta Wye Electric, n.d.)

3.4 IoT-laitteiden tietoturva

IoT-järjestelmän turvallisuus perustuu toimintaan, jossa IoT-laitteita ja verkkoa, johon IoT-laitteet on liitetty, suojataan kyberuhkilta. Kyberuhkat ovat mm. internetistä kodin verkkoon kytkettyihin laitteisiin kohdistuvia hyökkäyksiä, kuten haittaohjelmia tai kiristyshyökkäyksiä, joissa kohteeksi joutunutta käyttäjää kiristetään maksamaan lunnaita esim. tiedostoja salaamalla (F-Secure, 2022). IoT-laitteet on suunniteltu usein mahdollisimman kevyiksi ja helppopääsyisiksi, eikä niihin yleensä ole mahdollista asentaa erillistä tietoturvasovellusta. On myös mahdollista, että hankittu IoT-laite sisältää ohjelmistohaavoittuvuuksia tai jopa esiasennettuja haitallisia komponentteja, jotka puolestaan voivat muodostaa riskin, kun laite kytketään kodin lähiverkkoon. (Fortinet, n.d.)

Muita perinteisesti IoT-laitteisiin liittyviä hyökkäyksille altistavia tekijöitä ovat mm. helposti arvattavien tai oletussalasanojen käyttäminen, laitteiden liikennöinti verkossa tapahtuu turvattomia protokollia tai kanavia käyttäen ja usein salaamattomasti, sekä laitteiden ohjelmistojen haavoittuvuudet yhdistettynä puuttuviin ohjelmistopäivityksiin. (Fortinet, n.d.)

Euroopassa ja Suomessakin myytäville digitaalisen elementin sisältäville laitteille, kuten IoT-laitteille, ja ohjelmistoille ollaan vuosien 2024–2027 välillä ottamassa käyttöön tuotteen valmistajaa velvoittava kyberkestävyyssäädös, engl. Cyber Resilience Act (CRA). Säädöksen tavoitteena on vähentää Euroopan Unionin markkinoille saatettujen tuotteiden sisältämiä haavoittuvuuksia, ja sen vaatimusten toteutuminen vaaditan tulevaisuudessa osana tuotteiden CE-merkintää. Säädöksen turvallisuusvaatimusten täyttyminen tulee jatkossa olemaan tuotteen markkinoille pääsyn edellytys EU:ssa. Tuotteisiin kohdistuvia vaatimuksia ovat mm. ohjelmistojen turvalliset oletusasetukset, automaattiset turvallisuuspäivitykset, luvattomalta pääsyltä estäminen, datan minimointi ja luottamuksellinen säilyttäminen sekä keskeisten toimintojen turvaaminen. (Traficom, 2025)

Opinnäytetyössä ideoidussa tuotteessa, kyberkestävyyssäädöksen piiriin kuuluvia järjestelmän osia ovat ainakin muokatut tiedostot, kuten Docker-konttien konfiguraatiotiedostot, skriptit, eli asennussovelluksen taustalla suorittamat komennot, sekä sovellukseen liittyvät tietoturvaan, päivityksiin ym. liittyvät ohjeistukset. Järjestelmän ohjelmistot on toteutettu käyttäen virallisia Docker-levykuvia, joten ohjelmiston osat ovat

jäljitettävissä ja avoimesti saatavilla tunnetuissa lähteissä. Kyberkestävyyssäädöksen piiriin kuuluvat myös digitaalisia toiminnallisuuksia sisältävät osat, eli säädös koskee myös mahdollisia tuotepaketin osana kaupattavia kotiautomaatiolaitteita.

4 Järjestelmän tuotteistaminen

Opinnäytetyössä toteutetun järjestelmän vaativin osuus liittyi valmisteleviin toimenpiteisiin ja konfiguraatiotiedostoihin tehtyihin määrityksiin, jotka tehtiin ennen konttien käynnistämistä. Muitakin tapoja tämän kaltaisen järjestelmän toteuttamiseen on saatavilla, mutta tässä projektissa konfiguraatiot ja käyttöönoton testaukset suoritettiin WSL-komentoriviltä. Tällä tavalla toteutettuna toimivan kokoonpanon konfiguraatioita tehdessä myös selkeytyi, mitkä muuttujien arvot voidaan määrittää ennalta, ja mitkä puolestaan vaativat käyttäjän syötettä tai muita toimenpiteitä asennustilanteessa.

Komentorivillä työskenneltäessä vaaditaan huolellisuutta ja jonkin verran osaamista, joten suunniteltu tuote voisi olla asennusohjelma helpottamaan ja johdattelemaan käyttäjää järjestelmän käyttöönoton vaiheissa. Tuotteen hyötyjä olisivat ainakin ajan säästö, koska alun konfiguroinnit sujuvat nopeammin, sekä työskentelyn tehostuminen, koska näppäilyvirheiden mahdollisuus pienenee.

Opinnäytetyössä toteutetun järjestelmän pohjalta suunniteltu tuote voisi olla etähallittavan Home Assistantin käyttöönotossa avustava asennusohjelma. Asennusohjelma voisi toimia yksinkertaisimmillaan WSL-komentorivillä, tai graafisena versiona, josta näkisi mm. asennuksen vaiheet ja edistymisen.

Sovellus voisi johdatella asennuksen kulkua seuraavasti:

- Graafisen käyttöliittymän sisältävä sovellus aloittaa asennuksen tarkistamalla löytyykö järjestelmästä WSL-ohjelmisto ja oikea versio.
- Vaihtoehtoisesti WSL-komentorivipohjaisessa asennussovelluksessa aloitetaan tarkistamalla ja asentamalla tai päivittämällä Dockeriin liittyvät ohjelmistot.
- Käyttäjällä on mahdollisuus valita pelkästään paikallinen asennus, jolloin vain Home Assistant -kontti otetaan käyttöön.
- Sovellus kopioi kansiorakenteen ja tiedostot joltakin julkiselta versionhallinnan alustalta.

- Sovellus pyytää käyttäjää syöttämään DuckDNS-sivustolla luodun tunnisteen ja toimialueen osoitteen ym. tiedot tallennettavaksi ympäristömuuttujatiedostoon ja opastaa tarvittaessa niiden luomisessa.
- Käyttäjällä on mahdollisuus aktivoida autentikointi etäkirjautumissivulle, jolloin sovellus ottaa Basic Auth-ominaisuuden käyttöön.
- Käyttäjällä on mahdollisuus valita Home Assistantiin liittyviä ominaisuuksia käyttöön jo asennusvaiheessa.
- Sovellus viimeistelee käyttöönoton käynnistämällä kontit Dockerissa ja tarkistaa esim. konttien lokitiedoista, että käyttöönotto onnistui.
- Sovellus opastaa käyttäjää Home Assistant -tiliin liittyvissä toiminnoissa, kuten tilin luomisessa, sisäänkirjautumisessa ja monivaiheisen tunnistautumisen käyttöönotossa.

Edellä kuvailtu asennussovellus on yksi käytännön tapa työssä toteutetun järjestelmän soveltamiseen tuotteeksi. Toinen idea tuotteistamiseen olisi suunnitella valmis tuote, kuten IoT-laite, jonka mukana toimitettaisiin asetukset sisältävä asennuspaketti. Laite olisi helppo käyttöönottaa erilaisissa ympäristöissä käyttäjän tarpeen mukaan. Jatkoajatuksena tähän, aiemmin kuvattu asennussovellus voisi sisältyä IoT-laitteen mukana toimitettavaksi, jolloin käyttäjä saisi käyttöönotettua omaan tarpeeseensa soveltuvan järjestelmän laitteen asennusta ja automaatioita myöten.

5 Työssä käytetty laitteisto ja ohjelmistot

Suosituin alusta Home Assistant -käyttöjärjestelmän asennukseen on Raspberry Pi, Arduino tai jokin muu ns. yhden piirikortin tietokone. Tällaiset tietokoneet ovat kompaktin kokoisia, mahtuvat aikuisen kokoiselle kämmenelle, mutta tarpeeksi tehokkaita suorittamaan käyttöjärjestelmää vuorokauden ympäri. Suosittuja alustoja ovat myös mini-PC:t eli pienikokoiset tietokoneet, johon HA-käyttöjärjestelmä voidaan asentaa joko tietokoneen ainoaksi käyttöjärjestelmäksi tai tietokoneen käyttöjärjestelmään toimimaan, eli Windowsin päälle virtuaaliympäristöön. (Davidson, 2025)

Järjestelmän asentaminen työasemaan, tai varsinkaan kannettavaan tietokoneeseen ei ole varsinaisesti optimaalista, mm. koska Docker-konttien palvelut sammuvat, kun itse tietokonekin suljetaan esim. isäntäkäyttöjärjestelmän ohjelmistopäivitysten ajaksi. Isäntäkoneen resurssit on jaettu Windowsin ja WSL:ssä ajettavien konttien kesken, mikä voi aiheuttaa koko järjestelmän suorituskyvyn laskua. Mahdolliset USB-portteihin kytketyt IoT-laitteiden lähettimet, eivät myöskään ole laitteiden Bluetooth-kantaman sisällä, jos

kannettava tietokone otetaan reissuun mukaan. Projektissa käytetty kokoonpano tulee toimimaan testiympäristönä, jonka avulla järjestelmän toimivuutta ja siirrettävyyttä muihin tietokoneisiin voidaan arvioida.

Projektissa käytetyt palvelut asennetaan Docker-kontteihin Windows 11 -työasemaan, mutta samankaltainen järjestelmä voitaisiin toteuttaa myös palvelimeen tai palvelinkäytössä olevaan tietokoneeseen, jossa on asennettuna jokin käyttöjärjestelmä. Toteutettu järjestelmä soveltuu myös tämän kaltaiseen ohjelmistokokonaisuuden testaamiseen tilanteessa, jossa ei haluta poistaa alkuperäistä käyttöjärjestelmää kokonaan isäntäkoneelta.

Windows 11 -työasema, jolla projekti toteutettiin, oli liitettynä kodin lähiverkkoon, samoin Home Assistantiin liitetyt IoT-laitteet. Docker-konttien palvelut toimivat isäntäverkosta eristetyssä Docker-verkossa.

5.1 IKEA Trådfri -gateway-tukiasema

Ikea Trådfri -tukiasema toimii kodin lähiverkossa laitteet, kuten Ikean älylamput, valokatkaisijat ja liiketunnistimet, kokoavana yksikkönä. Tukiasema liitetään lähiverkkoon ethernet-kaapelilla, virransyöttö liitetään miniUSB-johdolla ja muuntajalla pistorasiasta. Laitteet liitetään tukiasemaan, ja ne käyttävät Zigbee-protokollaa.

Tukiasemassa on kannen alla portit ethernet-kaapelille ja miniUSB:lle, kolme LED-indikaattorivaloa ja painike laitteen paritusta varten.

Kuva 1. IKEA Trådfri -tukiasema malli E1526.

Tukiaseman malli on E1526. Mallille on ominaista, että ainakaan lamppuja ei voi liittää tukiasemaan suoraan, vaan yksi tai useampi lamppu liitetään ensin valokatkaisijaan, joka puolestaan voidaan liittää tukiasemaan.

Tukiasemaan voidaan liittää suoraan Ikeassa myytäviä uusiakin valaisimia, niissä mukana tulevan yhdistävän laitteen avulla. Tällainen on mm. LED-nauhavalaisin, jolle luodaan automaatio myöhemmin projektin aikana. Tukiasema on otettu käyttöön aikaisemmin ja laitteet on liitetty tukiasemaan Ikea Home Smart -sovelluksen avulla, joten niiden liittämistä ei käydä tarkemmin läpi.

5.2 Shelly Plus Smoke -paloilmaisin

Shelly Plus Smoke malli SNSN-0031Z on kämmenelle mahtuva, kotitalouksiin suunniteltu, älykäs paloilmaisin. Se toimii yhdellä CR123A-tyypin paristolla ja se on varustettu yhdellä laitteen päällä sijaitsevalla painikkeella ja LED-indikaattorivalolla. Paloilmaisin toimii Bluetooth- ja WiFi-yhteydellä, ja tukee myös MQTT-protokollaa. Paloilmaisimen asetuksia on mahdollista säätää verkkosivun kautta, kun sen muodostamaan lähiverkkoon ollaan yhteydessä. (Shelly, n.d.)

Paloilmaisin asennetaan kattoon mukana tulevan kiinnikkeen avulla, ja se ulkomuodoltaan pelkistetty, kuten kuvassa 1 nähdään.

Kuva 2. Shelly Plus Smoke -paloilmaisin.

5.3 WSL2 ja Linux

WSL2 on Windowsin ohjelmisto, joka mahdollistaa Linux-käyttöjärjestelmäympäristön ajamisen Windows-työasemalla luomalla Linux-ytimen kevyeen virtuaalikoneeseen. Linux-jakelun eli käyttöjärjestelmän variaation voi ladata ja asentaa mm. Microsoft Storesta. WSL2:n on mahdollista asentaa useita eri Linux-jakeluita, niistä kukin asentuu virtuaalikoneessa omaan konttiinsa, jolloin jokaisella jakeluista on myös omat toisistaan erilliset tiedostojärjestelmät.

Linuxin tiedostojärjestelmää käytetään Bash-shellistä, eli Bash-komentoriviltä. Bash-komentorivillä voidaan käyttää mm. Bash-työkaluja, -skriptejä sekä Linux-komentorivisovelluksia, kuten tekstieditorit Nano ja Vim. Sovelluksia voi myös käyttää Linuxin ja Windowsin kanssa ristiin tiedostojärjestelmien pysyessä silti erillään, esim. Bash-komentoriviltä on mahdollista avata Linux-tiedostojärjestelmässä sijaitseva koodia sisältävä tiedosto Windowsin Visual Studio Codessa. (Microsoft, 2023b)

Tässä opinnäytetyöprojektissa käytettiin Linux-jakelun versiota Ubuntu 22.04.5 LTS, mutta uudempikin versio Ubuntusta tai jokin muu jakelu toimisi järjestelmän toteuttamiseen yhtä hyvin. Olennaista kuitenkin on, että pyritään käyttämään ohjelmistojen viimeisimpiä tuettuja versioita, jotta vältytään ainakin vanhentuneiden ohjelmistojen aiheuttamilta yhteensopivuus- ja tietoturvaongelmilta (Easy2Patch, 2024).

5.4 Docker

Docker on ohjelmistoalusta, jolla voidaan mm. pakata sovellus ja sen riippuvuudet eli sovelluksen tarvitsemat asetukset yhdeksi paketiksi, jota kutsutaan Docker-kontiksi. Dockerin pääasiallinen tarkoitus onkin, että näitä konteissa sijaitsevia sovelluksia on mahdollista ajaa missä tahansa käyttöjärjestelmässä, johon Docker-ohjelmisto on asennettu. *Docker*-sanalla viitataan itseasiassa Dockerin taustapalveluun, eli Docker daemoniin, jonka kautta konttien hallinta, eli mm. luominen, ajaminen, sammuttaminen ja poistaminen tapahtuvat. (GeeksforGeeks, 2025b)

Docker Desktop – ohjelmisto Linux-, Mac- ja Windows-käyttöjärjestelmille, jonka avulla kontissa suoritettavia sovelluksia ja mikropalveluita voidaan rakentaa, ajaa ja jakaa muille käyttäjille eri pilvialustojen kautta. Ohjelmisto sisältää graafisen käyttöliittymän, joka helpottaa konttien, sovellusten, imagejen sekä konttien käyttöjärjestelmän hallintaa; Windows-käyttöjärjestelmää käytettäessä, WSL 2 avulla luotavien Linux-konttien lisäksi on mahdollista luoda Windows-kontteja Hyper-V-virtualisointia käyttäen. (Dockerdocs, n.d.-b)

Dockerfile-tiedosto – Domain Specific Language eli DSL-kielellä kirjoitettua koodia sisältävä tekstitiedosto, joka sisältää prosessin, eli ohjeet, jonka avulla Docker daemon luo Docker imagen. Dockerfilen sisältämät ohjeet ovat itseasiassa sarja komentoja, jotka DSL-kielelle ominaisesti suoritetaan siinä järjestyksessä, kuin ne on tiedostoon kirjoitettu. Yleisiä Dockerfile-tiedostossa määritettäviä asioita ovat pohjalla toimivan levykuvan osoittaminen, tiedostosijainnin luominen kontissa suoritettavalle sovellukselle, paikallisten ja / tai verkossa sijaitsevien tiedostojen kopiointi kontin tiedostosijaintiin, riippuvuuksien asentaminen, porttien määrittäminen kontille sekä kontin käynnistyksen yhteydessä suoritettavien sovellusten määritys. (GeeksforGeeks, 2024b)

Docker image – levykuva, joka sisältää Docker-kontin luomiseen tarvittavat ohjelmistot ja riippuvuudet, joita kontissa suoritettava sovellus tarvitsee toimiakseen. Luotuun Docker imageen ei varsinaisesti voi tehdä muutoksia, vaan muutokset tehdään Dockerfileen, jonka

avulla luodaan uusi Docker image. Docker image on alustariippumaton, eli sen avulla voidaan luoda sovellusta suorittava Docker-kontti missä tahansa käyttöjärjestelmässä, johon Docker vain on asennettu. Docker image voidaan ladata Docker Hub -palveluun, josta se voidaan asettaa edelleen ladattavaksi muille käyttäjille. (GeeksforGeeks, 2025c)

Docker-kontti eli Docker container – Docker imagen suorittamisesta muodostuva virtuaaliympäristö, jossa lopullista sovellusta ja palveluita suoritetaan. Docker-konttia ei voida jakaa Docker imagen tavoin, mutta luotuun konttiin ja siinä ajettaviin palveluihin on kuitenkin mahdollista tehdä muutoksia muokkaamalla Compose- tai Dockerfile-tiedostoa. (GeeksforGeeks, 2025c)

Docker Compose – työkalu, joka helpottaa usealla Docker-kontilla toteutetun järjestelmän luomisessa ja käynnistämisessä. Konttien luomisessa käytetään apuna Docker Compose -tiedostoa, joka sisältää sovellusten määrittelyt mm. palveluiden, verkkojen, volyymien ja ympäristömuuttujien osalta. Docker Compose -tiedosto on kirjoitettu ihmisille helppolukuisella Yet Another Markup Language eli YAML-kielellä, jota käytetään yleisesti sovellusten konfiguraatiotiedostoissa ja ohjelmointikielten väliseen tiedonsiirtoon. (GeeksforGeeks, 2025a, 2025d)

Docker Volume – Dockerin kontille luoma ja hallinnoima tallennustila, joka säilyy, vaikka kontti sammutetaan tai poistetaan. Kun volume luodaan, se tallentuu isäntäkoneen tiedostojärjestelmään. Kun isäntäkoneelle luotu volume mountataan eli kiinnitetään konttiin, siitä tulee tallennushakemisto, jota kontti käyttää. Volumen määrittely soveltuu tilanteisiin, joissa sitä käyttää vain yksi tai useampi kontti. Mikäli volumea pitäisi voida käyttää myös isäntäkoneelta, ”bind mount” tähän tarkoitukseen paremmin soveltuva tapa luoda säilyvä tallennustila. (Dockerdocs, n.d.-g)

Docker Network – Dockerin luoma verkko, jossa sitä käyttävät kontit voivat kommunikoida keskenään ja isäntäkoneen kautta internetiin päin. Docker muodostaa siltaavan oletusverkon konteille, jos ne esim. käynnistetään käyttäen yhtä Docker Compose -tiedostoa. (Dockerdocs, n.d.-e)

5.5 Konteissa ajettavat palvelut

Projektissa toteutettava järjestelmä muodostuu itse Home Assistant -ohjelmistosta, johon kodin älylaitteita voidaan liittää eri tekniikoiden avulla, sekä kolmannen osapuolen sovelluksista muodostuvan etäyhteyden käyttöliittymään.

Suoraviivainen tapa toteuttaa etäkirjautuminen internetin yli Home Assistant -instanssiin olisi ottaa käyttöön kuukausimaksullinen tili Home Assistant Cloud -palveluun. Palvelu luo Home Assistant -instanssille verkko-osoitteen ja sertifikaatin, jotka mahdollistavat instanssin turvallisen etäkäytön. (Nabu Casa, n.d.)

Internetin yli tapahtuva etäkirjautuminen Home Assistant -järjestelmään toteutetaan tässä projektissa kahdella Docker-kontilla. Linuxserver/duckdns-nimisellä levykuvalla luodaan kontti, jossa toimii DuckDNS -palvelu. Linuxserver/swag-nimisellä levykuvalla luodaan kontti, jossa otetaan käyttöön Nginx -web-palvelin, SSL-varmenteita varten Certbot-ohjelma sekä Fail2ban-tunkeutumisenestojärjestelmä. (LinuxServer.io, 2025-b)

Lisäsuojaksi Home Assistant -tilille kirjautumiseen otetaan käyttöön monivaiheinen tunnistautuminen Google Authenticator -todentajasovelluksella. Sovellus ladataan älypuhelimeen sovelluskaupasta, ja siihen kirjaudutaan jo aiemmin luodun Google-tilin kautta.

5.5.1 Home Assistant

Projektissa käytetään Home Assistantin virallista Docker imagea, joka on Home Assistantin Container-versio. Se on kevyt versio HA:sta, sillä siitä puuttuu mm. suora tuki lisäosien asentamiseen, sekä ohjelmistopäivitystoiminto, koska konttipohjaiset sovellukset päivitetään pääasiassa Dockerista käsin hakemalla uusin versio Docker imagesta. (Home Assistant, n.d.-c)

Home Assistant on älykotijärjestelmä, jossa on monipuoliset mahdollisuudet kodin IoT-laitteiden hallintaan ja se on laajasti muokattavissa käyttäjän tarpeisiin soveltuvaksi. Järjestelmä sisältää tuen laajalle joukolle nykypäivänä markkinoilla oleville kodin IoT-laitteita. Järjestelmä on avoimeen lähdekoodiin perustuva, ja sitä kehitetään aktiivisesti kehittäjien ja yhteisön toimesta, joten tuki uusille laitteille päivittyy nopeasti. Järjestelmään liitettävien eri laitteiden dokumentaatio on laaja ja vapaasti saatavilla internetissä. Järjestelmään voidaankin liittää tuhansia eri valmistajien palveluita ja laitteita, ja niille voidaan asettaa automaatioita lähes rajattomasti. (Diener, 2024)

5.5.2 DuckDNS

DuckDNS:n palvelua ajava kontti otetaan käyttöön linuxserver/duckdns Docker imagella, jota ylläpitää laaja LinuxServer-yhteisö.

Verkossa toimiva DuckDNS-palvelin toimii dynaamisena nimipalvelimena (DNS), mikä tarkoittaa, että palvelu liittää siellä luodun aliosoitteen jonkin muun verkon julkiseen osoitteeseen, kuten tässä tapauksessa kodin verkon julkiseen IP-osoitteeseen. Palvelu ei kuitenkaan automaattisesti tarkista kodin verkon julkista IP-osoitetta, joten jos IP-osoite sattuisi vaihtumaan, etäkirjautumismahdollisuus lakkaisi toimimasta, koska palvelimella olisi väärä IP-osoite.

Verkossa toimiva DuckDNS-palvelin saa tiedon muuttuneesta IP-osoitteesta kontissa toimivan DuckDNS-palvelun avulla. Kontin palvelu selvittää tietyin väliajoin kodin verkon julkisen IP-osoitteen, ja lähettää tiedon vaihtuneesta IP-osoitteesta verkossa toimivalle palvelimelle. (LinuxServer.io, 2025-a)

5.5.3 SWAG

Secure Web Application Gateway (SWAG) -palvelu käynnistetään `linuxserver/swag` Docker imagesta, jota ylläpitää laaja LinuxServer -yhteisö. Imagesta käynnistettävässä `swag`-kontissa ajetaan useita palveluita, joiden avulla muodostetaan turvallinen etäkirjautuminen Home Assistantiin.

Näitä palveluita ovat mm. Nginx-verkkopalvelin, joka toimii käänteisenä välityspalvelimena, eli tässä tapauksessa se ohjaa osoitteeseen ”https://oppariha.duckdns.org” tulevan liikenteen kontin sisäiseen palveluun – eli Home Assistantiin. Certbot-palvelu luo ja uusii Nginx-palvelimen SSL-varmenteen, jota tarvitaan HTTPS-suojattuun liikennöintiin. (LinuxServer.io, 2025-b)

HTTPS, eli Hyper Text Transfer Protocol Secure käyttää tässä tapauksessa SSL-varmennetta muodostaakseen suojatun yhteyden nettiselaimen ja palvelimen välille, jolloin mm. verkkosivulle syötetyt tiedot kulkevat internetissä salattuina. SSL-varmenne (Secure Sockets Layer) sisältää mm. SSL-suojattuun HTTPS-liikennöintiin tarvittavan yksityisen avaimen (private key). Liikennöintiin tarvittava julkinen avain (public key) sijaitsee verkkopalvelimella. (Comodo, n.d.)

Fail2ban-palvelu suojaa verkkopalvelinta ja siinä toimivaa Home Assistant-instanssia mm. estämällä automaattisesti IP-osoitteita, joista yritetään tehdä Home Assistant-tiliin toistuvasti epäonnistuvia kirjautumisyrityksiä. (LinuxServer.io, 2025b)

6 Työn toteutus

Projektissa käytetyt ohjelmat, sovellukset ja tavat eivät ole ainoa keino toteuttaa kuvatun kaltainen järjestelmä, vaan ne ovat pikemminkin tieto- ja viestintätekniikan opintojen aikana hyväksi havaittuja, helppokäyttöisiä ja yleisesti käytössä olevia tapoja. Esimerkiksi projektissa hyödynnetyt WSL 2 -terminaali, Bash-shell eli Bash-komentotulkki ja Bash-kieli ovat suosittuja työkaluja, joiden soveltamiseksi erilaisissa projekteissa löytyy runsaasti käytännön ohjeita internetistä.

Projektin toteutus alkaa ohjelmistojen asennuksesta Windows 11 -työasemaan. Asennettavia ohjelmia ovat WSL 2 ja Docker Desktop. Asennusten jälkeen tehdään valmistelevat toimenpiteet DuckDNS-palveluun, jonka jälkeen luodaan projektille tarvittava kansiorakenne.

6.1 Ohjelmistojen asennukset ja valmistelut

WSL voidaan asentaa työasemaan esim. Windows PowerShellin kautta komennolla `wsl --install` (Microsoft, 2024b). Oletusjakeluna asentuu Ubuntu, joka sopii tämän projektin toteuttamiseen hyvin. Asennuksen yhteydessä tulee valita Linux-käyttöjärjestelmän käyttäjätunnus ja salasana, jotka toimivat samalla järjestelmänvalvojan tunnuksina järjestelmään. Tämän projektin toteutuksessa tarvitaan järjestelmänvalvojan tunnuksia, jotta mm. `sudo` -alkuiset komennot voidaan suorittaa. (Microsoft, 2023)

Jotta Docker voidaan asentaa työasemaan, tulee WSL:n olla versio 2, joka on saatavilla ainakin Windows 11:een ja tiettyihin versioihin Windows 10:ä. Käyttöjärjestelmän tuki yleisesti WSL 2:lle voidaan varmistaa tarkistamalla WSL:n versio ja asettamalla WSL versio 2 oletuksena käyttöön esim. Windows PowerShell -terminaalissa. (Microsoft, 2024-a)

Docker Desktop -ohjelmiston asennuksessa seurattiin Dockerin dokumentaatiosivustolta löytyviä asennusohjeita. Asennusohjeissa lähdettiin liikkeelle asennustiedoston suorittamisesta ja niissä kerrottiin mm. mihin sijaintiin ohjelmiston tiedostot tallentuvat.

Asennusohjelma johdatteli asennusta eteenpäin ja asennus suoritettiin dokumentaation ohjeistusta seuraamalla. Asennuksen aikana valittiin WSL 2 Hyper-V:n sijaan. Asennusohjelman päätyttyä Docker käynnistettiin ensimmäistä kertaa, ja ohjelma otettiin käyttöön hyväksymällä palvelusopimuksen ehdot. Docker Desktopin käyttö on ilmaista yksityishenkilöille ja pienille yrityksille. (Dockerdocs, n.d.-d)

Asennuksen onnistumisen voi tarkistaa esim. WSL 2-terminaalissa komennolla `docker –version`, joka on tässä projektissa 28.0.1. Toiminnan voi vielä testata komennolla `docker run hello-world`, joka asennuksen toimiessa oikein palauttaa komentoriville viestin otsikolla ”Hello from Docker!”. (Kumar, 2025)

Jotta `duckdns`- ja `swag`-kontit saadaan myöhemmin toimimaan `homeassistant`-kontin kanssa, tulee ensin sivustolla ”www.duckdns.org” joko luoda tili tai kirjautua suoraan palveluun esim. GitHub -tunnuksia käyttäen. Sisäänkirjautumisen jälkeen, etusivulla kohtaan ”domains” annettiin haluttu *sub domain*, eli alitoimialueen nimi, ja painettiin ”add domain”. Nimeämisessä on tiettyjä ehtoja, kuten että nimen tulee olla vapaa, eli se ei voi olla jo jonkun toisen käytössä kyseisessä palvelussa. Toimialueen loppuosa päättyy aina ”.duckdns.org”. Riville syötetty alitoimialueen nimi hyväksyttiin, ja se tuli näkyviin etusivulle kohtaan ”domains” – ”domain”. Tiedostossa *docker-compose.yml* tarvittava tunniste eli `access_token` näkyi etusivulla kohdassa ”DuckDNS token”. (taichikuji, n.d.)

Projektin kansiorakenne on varsin yksinkertainen: kotihakemistoon luotiin projektikansio nimeltä *projekti*, johon sijoitettiin konttien imagejen mukaan nimetyt kansiot *duckdns*, *homeassistant* ja *swag*. Kansioon *homeassistant* luotiin valmiiksi kansio nimeltä *config* ja kansioon *swag* luotiin kansio nimeltä *dns-conf*.

Projektikansion sisältö WSL-komentoriville tulostettuna `tree -d` -komennolla:

```
.
├── duckdns
├── homeassistant
│   └── config
└── swag
    └── dns-conf
```

6.2 Projektin konfiguraatiotiedostot

Projektissa luodaan tiedostolla *docker-compose.yml* yhtä aikaa kolme Docker-konttia, jotka rakennetaan käyttäen `homeassistant`, `duckdns` ja `swag` -imageja. Käyttöönoton vaiheissa on noudatettu `linuxserver/swag` -levykuvan dokumentaatiota, josta löytyvät myös raportissa esiintyvien tiedostojen sisällöt, WSL-komennot sekä Nginx-välityspalvelimen toimintaan liittyvät asetukset. (LinuxServer.io, 2025-b)

Tiedosto ***docker-compose.yml*** sijaitsee projektikansion polussa *~/projekti/docker-compose.yml*, ja sen koko sisältö on esitetty Liitteessä 1.

Tiedostossa määritetään seuraavat asetukset, joita Docker käyttää dokumentaation mukaan konttien muodostamiseen ja käynnistykseen (Dockerdocs, n.d.-f):

- `services:` viittaa muodostettaviin palveluihin, eli kontteihin, jotka on määritetty rivin alatasoilla nimillä `homeassistant`, `duckdns` ja `swag`.
- `image:` kertoo, mistä levykuva haetaan ja mitä versiota siitä käytetään, kun kontti käynnistetään.
  `container_name:` määrittää konteille lyhytnimet, joilla niihin voidaan mm. viitata tiedostossa myöhemmin.
- `network_mode: host` tarkoittaa, että kyseinen `homeassistant`-kontti käyttää suoraan isäntäkoneen verkkoyhteyksiä.
- `environment:` alla muuttujat viittaavat kyseisen kontin muodostamisessa käytettäviin *.env* -tiedostossa sijaitseviin ympäristömuuttujiin.
- `volumes:` määrittää kontille oman tiedostosijainti isäntäkoneessa, jotta tiedot säilyvät, vaikka kontti poistettaisi.
- `networks:` viittaa tiedoston lopussa luotavaan aliverkkoon, jonka avulla projektin kontit tulevat olemaan yhteydessä toisiinsa.
- `restart: unless-stopped` määrittää kontti käynnistymään aina uudelleen, mikäli se syystä tai toisesta, kuten isäntäkoneen uudelleenkäynnistämisen tai kontin virhetilanteen vuoksi pääsisi sammumaan.

Seuraavat määrittelyt tehtiin `swag`-kontille, jotta sen palvelut toimivat oikein. Kohdan `cap_add: - NET_ADMIN` määrittelyä tarvitaan, jotta Fail2ban-tunkeutumisenestopalvelu toimii oikein. Määrittelyn avulla kontti saa oikeuden muokata Linuxin `iptables`-taulukoita,

joiden avulla järjestelmä havaitsee esim. *brute force* -kirjautumisyrityksiä ja estää IP-osoitteet, joista niitä yritetään tehdä.

Kohdassa `ports:` määritetään isäntäkoneen portteihin saapuvien yhteyksien ohjaus kontin portteihin. Ilman kyseistä määritystä yhteydet eivät ohjautuisi kontin palveluihin. Portti 443 eli HTTPS-portti mahdollistaa liikennöinnin palvelun avulla luotavan SSL/TLS-salatun yhteyden kautta, ja portti 80 eli HTTP-portti mahdollistaa käytännössä yhteyden uudelleenohjauksen ”http://” -alkuisesta osoitteesta ”https://” -alkuiseen osoitteeseen. (LinuxServer.io, 2020)

Tiedosto ***.env*** sijaitsee projektikansion polussa *~/projekti/.env*, ja sen koko sisältö on esitetty Liitteessä 2. Tiedostoon tallennetaan avain – arvo -pareiksi arkaluonteiset tiedot, joita ei haluta nähtäville suoraan esim. koodiin tai muuhun yleistä tietoa sisältävään tiedostoon. Arkaluonteista tietoa ovat mm. ohjelmistorajapintojen avaimet, tietokantoihin ja ympäristöihin liittyvät asetukset ja salasanat. Hyvä käytäntö on, että näitä tietoja voidaan muokata keskitetysti yhdellä tiedostolla. (Mudadla, 2023)

Tiedostossa *.env* määritetään seuraavat ympäristömuuttujat, jotka liitetään tiedoston *docker-compose.yml* `environment:` -kohtaan:

- Kaikkiin kolmeen palveluun on määritetty muuttuja `TZ=Europe/Helsinki`, jolla varmistetaan, että palvelut konttien välillä toimivat samalla aikavyöhykkeellä.
- Palveluihin `duckdns` ja `swag` määritetään `PUID=1000` ja `PGID=1000`, joilla varmistetaan, että kontilla on oikeus käyttää isäntäkoneessa sijaitsevaa projektikansiota. Arvo ”1000” on sama kuin Linux-järjestelmässä ensimmäisenä luodulle käyttäjälle oletuksena muodostuvat UID, eli käyttäjätunnus ja GID, eli ryhmätunnus.
- Muuttuja `DUCKDNS_SUBDOMAIN` on DuckDNS-sivuston palvelussa rekisteröity alitoimialueen alkuosa, eli tässä ”oppariha” ja `DUCKDNS_TOKEN` vastaavasti kohta, johon sijoitettiin alitoimialueen DuckDNS-tunniste.
- Kohtaan `DOMAIN_NAME` annetaan pääverkkotunnus, jolle kontin palvelu hakee SSL-varmenteen, eli tässä tapauksessa ”oppariha.duckdns.org”. Alempana tiedostossa, kohta `SUBDOMAINS` jätetään tyhjäksi, koska varmenne haetaan ainoastaan edellä mainitulle pääverkkotunnukselle.
- Kohtaan `EMAIL` lisättiin sähköpostiosoite, johon menevät mm. ilmoitukset sertifikaattien vanhenemisesta.
- Kohtaan `VALIDATION` määritettiin varmennustavaksi ”dns”, ja kohtaan `DNSPLUGIN` arvoksi määritettiin ”duckdns”. DNS-varmennetta käytettäessä kirjautumistiedot, eli

DuckDNS-tunniste tulee tallentaa erilliseen *.ini* -tiedostoon, josta kerrotaan tarkemmin seuraavassa alaluvussa. (LinuxServer.io, 2020)

Koska tiedosto sisältää tokenin, eli tunnistetiedon, joka halutaan pitää salattuna, muokataan tiedoston käyttöoikeuksia WSL:ssä komennolla `chmod 600 ~/projekti/.env` .
Komennon avulla tiedoston omistaja saa luku- ja kirjoitusoikeuden kyseiseen tiedostoon ja rajataan oikeuksia niin, että muut prosessit tai käyttäjät eivät voi nähdä tai käyttää tiedoston sisältöä WSL:ssä. (Simard, n.d.)

Tiedosto ***duckdns.ini*** sijaitsee projektikansion polussa *~/projekti/dns-conf/duckdns.ini*. Tiedosto sisältää DuckDNS-palvelun tunnisteen, jota certbot-palvelu käyttää SSL-varmenteen hakemiseen. (LinuxServer.io, 2020)

Tiedosto sisältää yhden rivin tekstiä, jonka arvokohta on tässä anonymisoitu:

`dns_duckdns_token=abc1234yourtoken` .

Kontin palvelu tarvitsee lisäksi lukuoikeuden WSL-tiedostoon. Koska tiedosto sisältää tunnistetiedon, kontille annetaan kyseiseen tiedostoon luku- ja kirjoitusoikeus vain tiedoston omistajalle ja rajataan muita oikeuksia komennolla `chmod 600 ~/projekti/swag/dns-conf/duckdns.ini` . (LinuxServer.io, 2020)

Projektin konttien luomisen yhteydessä polkuun *~/projekti/swag* muodostui myös kansio nimeltä *nginx*, joka sisältää kansiot *proxy-confs* ja *site-confs*. Kansioon *site-confs* luodaan tiedosto nimeltä ***homeassistant.subdomain.conf****.* Tiedosto sijaitsee polussa *~/projekti/nginx/homeassistant.subdomain.conf*, ja sen koko sisältö on esitetty Liitteessä 3.

Tiedoston pohjana käytettiin kansiosta *proxy-confs* tiedostoa *homeassistant.subdomain.conf.sample*, josta poistettiin käyttämättömät rivit selkeyden vuoksi. Tiedostoon tehtiin HTTPS-liikenteelle tarkempia määrityksiä, jotka myös korvaavat osan sample-tiedoston oletusmäärityksistä, kuten kohta `server_name`, joka viittaa projektin käytössä olevaan välityspalvelimen osoitteeseen ja kohta `proxy_pass`, joka viittaa Docker-verkossa sijaitsevaan palveluun johon liikenne ohjataan. (LinuxServer.io, 2025-b)

Tiedosto ***default.conf*** sijaitsee polussa *~/projekti/nginx/default.conf*, ja sen koko sisältö on esitetty Liitteessä 4. Tiedosto toimii Nginx-palvelimen yleiskonfiguraationa ja sisältää mm. oletusporttien määritykset ja HTTPS-ohjaukset.

Tiedoston *default.conf* porttiin 443 liittyvät määritykset siirrettiin kansioon *homeassistant.subdomain.conf*. Porttiin 443 ja toimialueeseen – eli `oppariha.duckdns.org` – liittyvät määritykset sijoitettiin samaan tiedostoon `auth_basic` -määritysten kanssa, jotta Basic Auth-palvelu suojaa juuri oikeaa verkkosivua.

Tiedosto ***configuration.yaml*** sijaitsee polussa *~/projekti/homeassistant/configuration.yaml*, ja sen koko sisältö on esitetty Liitteessä 5. Tiedostossa määritellään Home Assistant -sovelluksen konfiguraatiot, kuten käyttöliittymän teemat, liitetyt laitteet ja automaatiot.

Oletusarvoisesti `homeassistant`-kontti estää yhteysyritykset *proxyilta*, eli välityspalvelimilta. Pääsy sallittiin tiedostossa *configuration.yaml* lisäämällä siihen seuraavat rivit :

```
http:
  use_x_forwarded_for: true
  trusted_proxies:
    - 172.30.0.0/24 .
```

Rivin `use_x_forwarded_for: true` avulla Home Assistant saa tietoonsa tulevan yhteyden IP-osoitteen HTTP-otsikkotiedosta. Em. riviä käytettäessä on aina määritettävä yksittäinen osoite tai verkko, josta yhteys yritetään muodostaa. Kohdassa `trusted_proxies:` sallitaan yhteys vain projektin verkosta 172.30.0.0/24, johon `swag`-kontti kuuluu. (Home Assistant, n.d.-b)

6.3 Konttien käynnistäminen

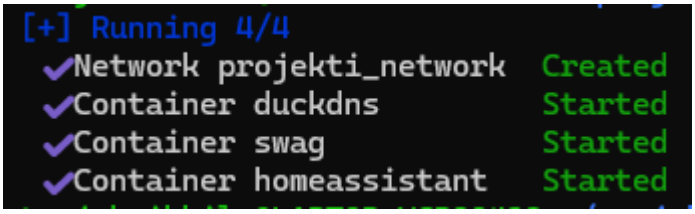
Kontit luodaan ja käynnistetään edellä mainittujen tiedostojen avulla WSL:n komentorivillä komennolla

```
docker-compose up -d .
```

Kun em. komento annetaan, Docker aloittaa levykuvien lataamisen Docker hubista ja GitHub Container Registrysta. Latautumisen edistymistä voi seurata WSL-komentoriviltä.

Docker loi projektille oletusverkon ja käynnisti kontit onnistuneesti, mistä kertoo tila ”started” kontin nimen jälkeen, kuten kuvasta 3 nähdään.

Kuva 3. WSL-komentorivin tuloste komennon ”docker-compose up -d” jälkeen.



Kunkin kontin onnistunut käynnistyminen varmistettiin vielä konttien lokitiedoista. Lokitiedot tulostettiin WSL-komentoriville seuraavilla komennoilla:

```
docker logs homeassistant

docker logs duckdns

docker logs swag .
```

Lokitiedoista ei löytynyt virheilmoituksia ja lueteltujen palveluiden tila oli ”started”. Swag-kontin lokitiedoista löytyivät seuraavat rivit, joista voitiin päätellä, että palvelu on vastaanottanut ja tallentanut SSL-varmanteen sekä luonut avaimen onnistuneesti:

```
Successfully received certificate.

Certificate is saved at:
/config/etc/letsencrypt/live/oppariha.duckdns.org/fullchain.pem

Key is saved at:
/config/etc/letsencrypt/live/oppariha.duckdns.org/privkey.pem .
```

Voidaan siis olettaa, että konfiguraatiotiedostot sisälsivät tarvittavat muuttujat palveluiden käynnistämiseksi ja toisaalta tiedostot eivät sisältäneet kirjoitusvirheitä.

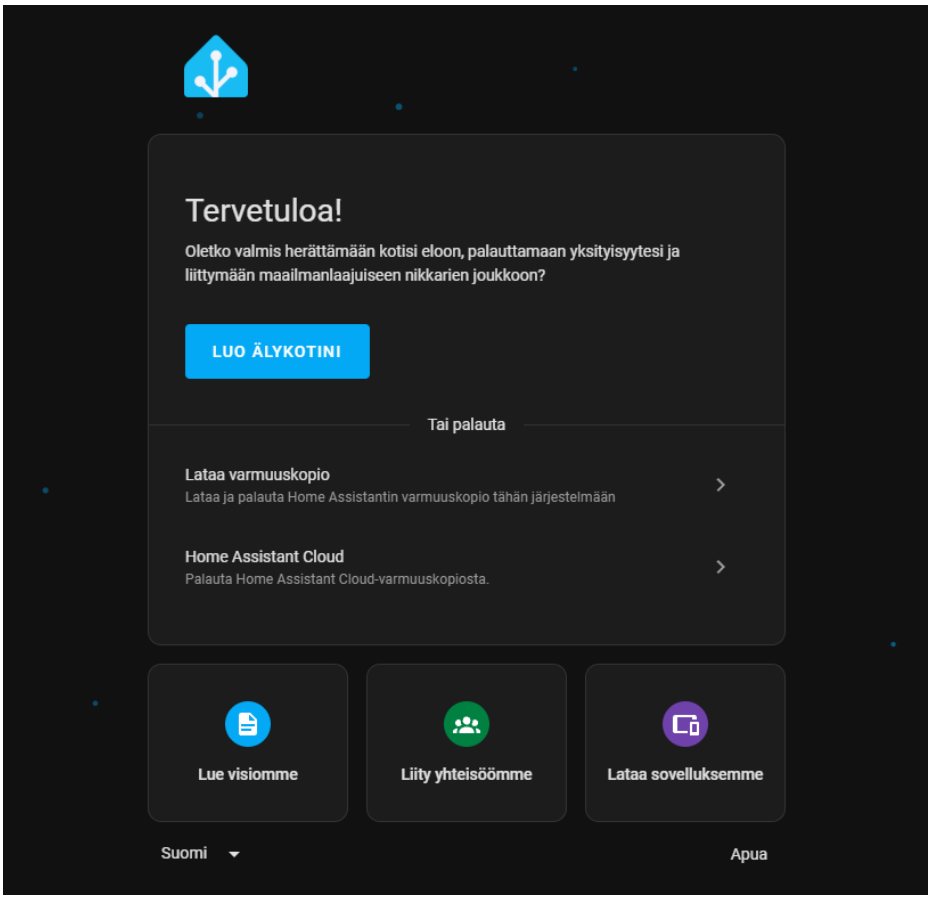
6.4 Home Assistantin käyttäjätilin luominen

Kun kontit palveluineen olivat käynnissä, Home Assistantiin voitiin kirjautua ensimmäistä kertaa. Käyttäjätilin luomisessa ja muissa käyttöönoton vaiheissa sovellettiin Home Assistantin dokumentaatiota. (Home Assistant, n.d.-e)

Jokaiselle luodulle Home Assistant -käyttäjätilille on hyvä ottaa käyttöön salasanatunnistautumisen lisäksi MFA, eli monivaiheinen tunnistautuminen parantamaan Home Assistant -instanssin tietoturvaa. Home Assistantin pääkäyttäjätilin luomisen jälkeen tilille otettiin käyttöön monivaiheinen tunnistautuminen matkapuhelimeen ladattavan Google Authenticator -sovelluksen avulla. (Home Assistant, n.d.-d)

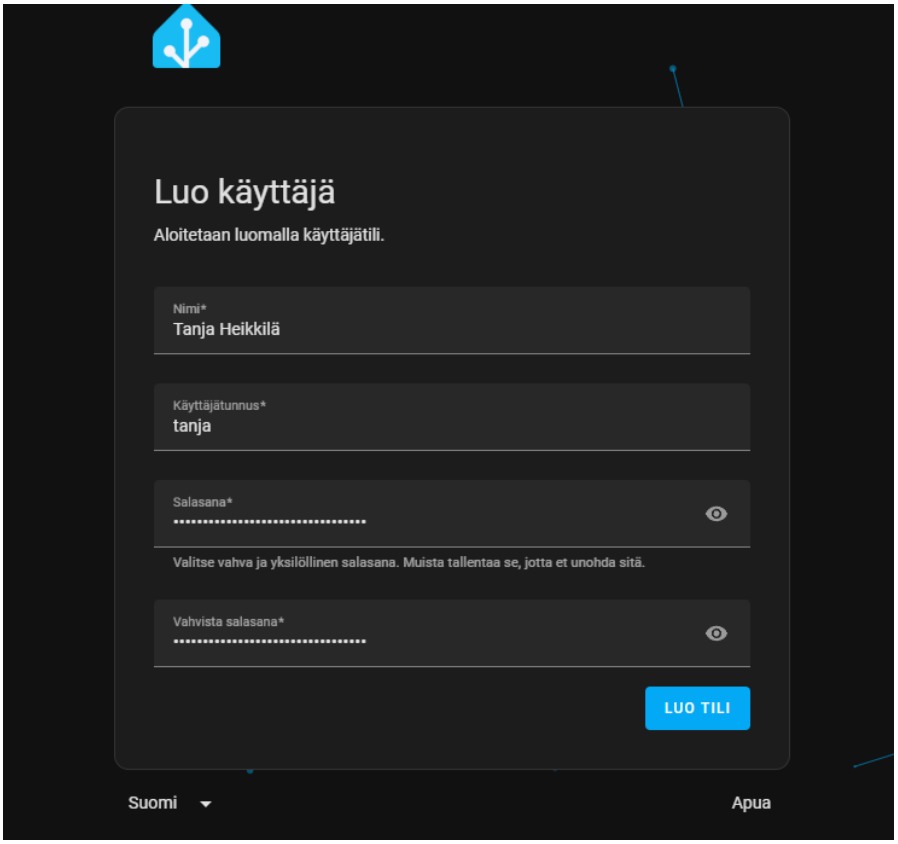
Home Assistantiin kirjaudutaan paikallisesti menemällä Internet-selaimella osoitteeseen `http://localhost:8123`, joka ohjautuu Home Assistantin ns. onboarding- eli käyttöönoton aloitussivulle. Home Assistant -käyttäjätilin luominen tehtiin aloitussivulla kohdassa ”luo älykotini”, joka näkyy kuvassa 4.

Kuva 4. Home Assistantin aloitussivu.



Käyttäjätilille annettiin käyttäjän nimi ja salasana, käyttäjätunnus muodostui automaattisesti nimen perusteella. Tilin luominen viimeisteltiin painamalla kohdasta ”luo tili”, joka näkyy kuvan 5 oikeassa alakulmassa.

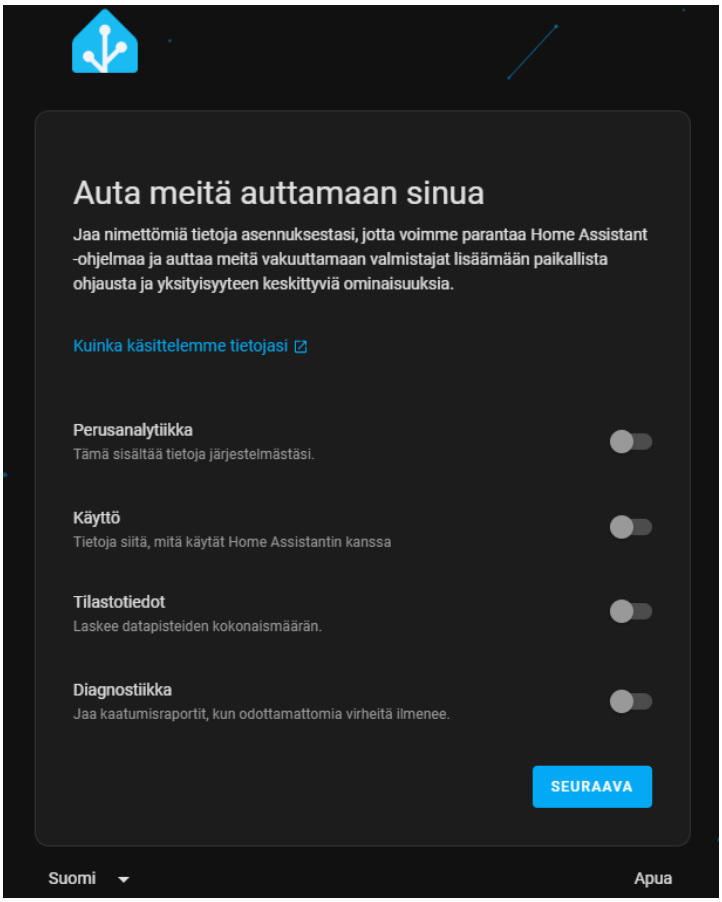
Kuva 5. Home Assistantin käyttäjätilin luominen.



Seuraavaksi asetettiin kodin sijainti, jonka perusteella määräytyy joitakin yleisiä asetuksia Home Assistantissa, kuten aikavyöhyke ja käytössä olevat mittayksiköt. Sijainnin avulla määrittyy myös ns. ”home zone”, eli kotia ympäröivä alue, jonka avulla voidaan myöhemmin luoda määrityksiä kodin älylaitteille. (Home Assistant, n.d.-e)

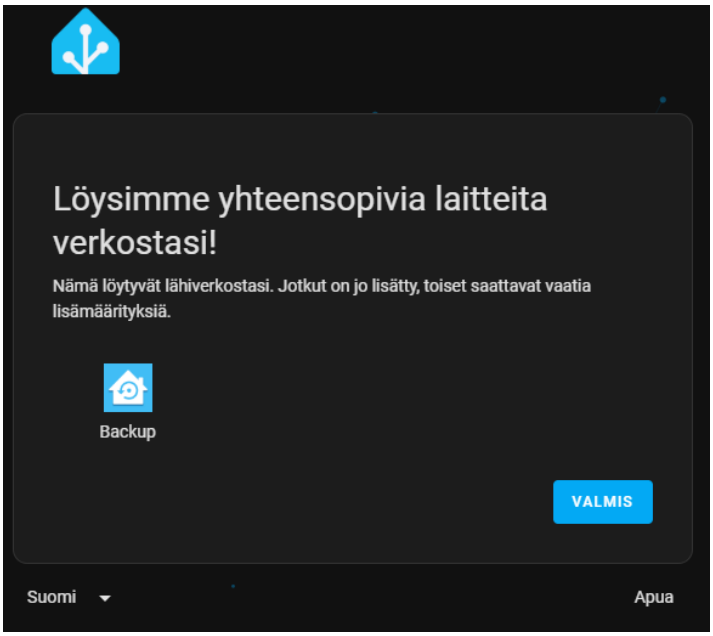
Seuraavassa vaiheessa oli mahdollista jakaa tilikohtaisia analytiikka- ja diagnostiikkatietoja, kuten ilmoituksia sovelluksen kaatumisista Home Assistantin kehittäjille. Kohdat jätettiin oletusasentoon, jolloin tietoja ei jaeta kehittäjille, kuten kuvassa 6 nähdään.

Kuva 6. Analytiikka- ym. tietojen jakaminen.



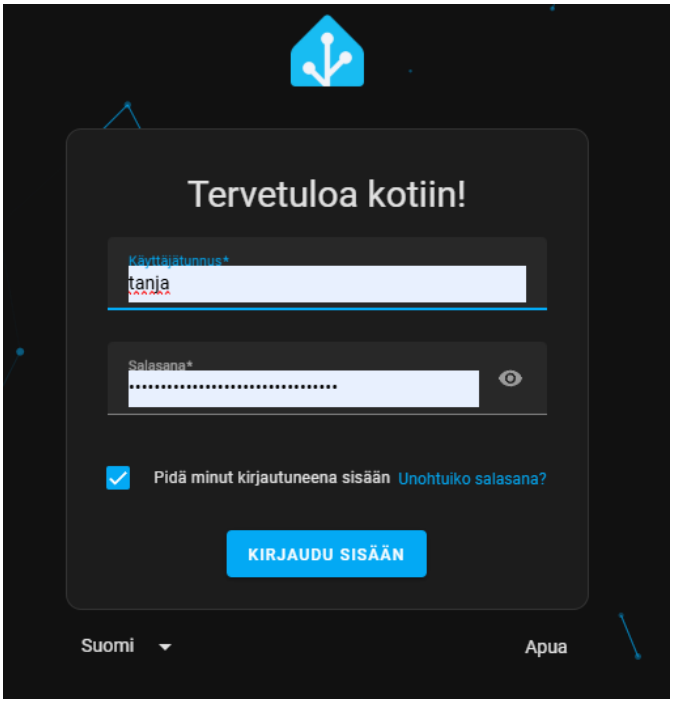
Käyttäjätilin luominen viimeisteltiin kohdasta ”valmis”, joka näkyy kuvan 7 oikeassa alakulmassa.

Kuva 7. Käyttäjätilin luomisen viimeistelysivusta.



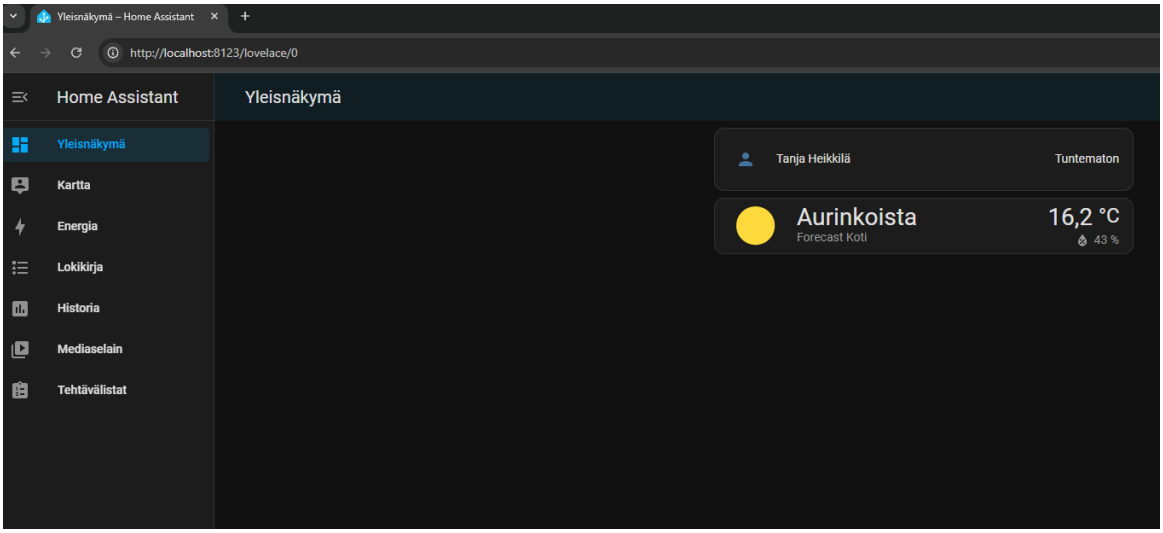
Selaimessa näkyi tässä vaiheessa virheilmoitus koodilla ERR_FAILED . Selaimen virhesivulta voitiin kuitenkin edetä kuvassa 8 näkyvälle Home Assistantin sisäänkirjautumissivulle menemällä selaimella jälleen osoitteeseen `http://localhost:8123/` .

Kuva 8. Home Assistant -tilin kirjautumissivu.



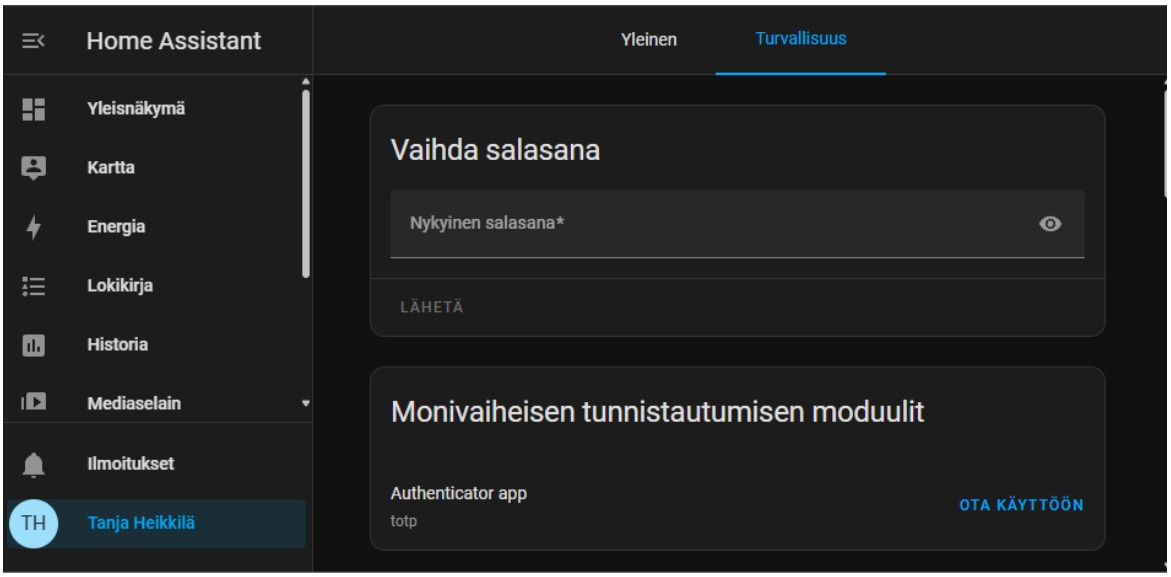
Home Assistantiin kirjauduttiin sisään juuri luoduilla tunnuksilla, jolloin päästiin *dashboardiin*, eli yleisnäkymäsivulle, joka nähdään kuvassa 9. Yleisnäkymäsivulla tulee näkymään myöhemmin järjestelmään liitettyihin laitteisiin liittyviä kuvakkeita, joiden avulla voidaan mm. hallita laitteita ja niiden automaatioita.

Kuva 9. Osa Home Assistantin yleisnäkymäsivusta.



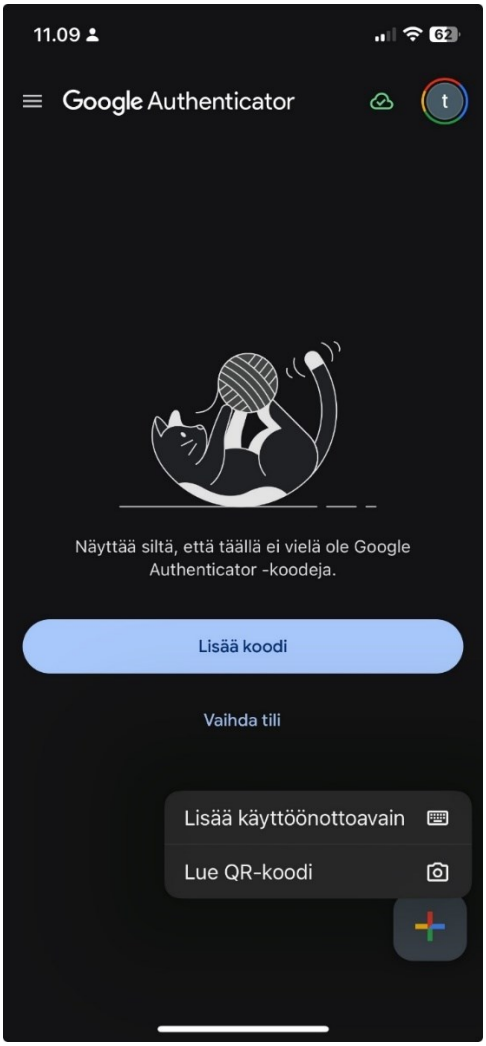
Monivaiheinen tunnistautuminen eli MFA (Multi Factor Authentication) otettiin käyttöön luodulle pääkäyttäjätilille. Asetus tehtiin käyttäjäprofiilin asetuksissa ”Turvallisuus”-välilehdellä kohdassa ”Monivaiheisen tunnistautumisen moduulit – ota käyttöön”, joka näkyy kuvan 10 oikeassa alakulmassa.

Kuva 10. Home Assistantin hallintasivu, "turvallisuus"-välilehti.



Matkapuhelimella avattiin Google Authenticator -sovellus ja lisättiin Home Assistant -tili valitsemalla ”Lue QR-koodi”, joka näkyy kuvan 11 oikeassa alakulmassa.

Kuva 11. Google Authenticator -sovelluksen etusivu älypuhelimessa.

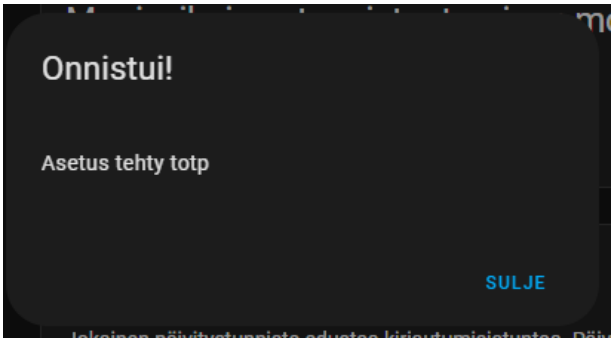


Kuvassa 12 näkyvän ponnahdusikkunan näkyvä QR-koodi skannattiin matkapuhelimen Google Authenticator -sovelluksella. Skannauksen jälkeen Google Authenticator -sovelluksessa näkyvä kuusinumeroinen koodi kirjoitettiin kuvan 12 alareunassa näkyvään kohtaan ”code*” ja valittiin ”lähetä”.

Kuva 12. Monivaiheisen tunnistautumisen liittäminen Home Assistant -käyttäjätiliin.

Onnistuneesta MFA:n käyttöönotosta kertoo kuvassa 13 näkyvä ponnahdusikkuna.

Kuva 13. Monivaiheisen tunnistautuminen lisätty Home Assistant -tiliin onnistuneesti.



6.5 Reitittimen porttiohjaukset

Instanssiin etäkirjautuminen osoitteessa ”https://oppariha.duckdns.org/” ei vielä toiminut, vaan selain antoi seuraavanlaisen virhesanoman `ERR_SSL_VERSION_OR_CIPHER_MISMATCH` .

Isäntäkoneen ja kontin porttien välinen liikennöinti määriteltiin jo aiemmin Docker Compose -tiedostossa seuraavasti: isäntäkoneen portti 80 on ohjattu Docker-kontin porttiin 80 ja vastaavasti isäntäkoneen portti 443 kontin porttiin 443. Jotta ulkoapäin tuleva HTTPS-yhteys voisi toimia, reitittimen asetuksissa tehdään porttiohjaukseen sääntö, joka sallii internetistä päin tietokoneen porttiin 443 tulevan yhteyden. Portin 80 ohjaus ei ole järjestelmän toiminnan kannalta välttämätöntä, mutta porttiohjaus määritetään tässä projektissa, jotta ohjaus http-alkuisesta osoitteesta ”https”-alkuiseen osoitteeseen toimisi käytännössä. Ts. kun selaimella mennään osoitteeseen ”**http**://oppariha.duckdns.org”, Nginx-verkkopalvelin uudelleenohjaa yhteyden osoitteeseen ”**https**://oppariha.duckdns.org”. (LinuxServer.io, 2025-c)

Kuvassa 14 näkyvät reitittimen asetuksiin tehdyt *port forwarding rules*, eli portinohjaussäännöt.

Kuva 14. Reitittimen hallintasivulla tehdyt portinohjaussäännöt.

#	Status	Service Name	Originating IP	WAN Interface	Server IP Address	Start Port	End Port	Translation Start Port	Translation End Port	Protocol	Modify
1		HA Docker	N/A	Auto Detect	[redacted]	443	443	443	443	TCP	
2		HA Docker	N/A	Auto Detect	[redacted]	80	80	80	80	TCP	

6.6 Basic Auth -lisäosan käyttöönotto

Nginx-verkkopalvelimen lisäsuojausta varten otetaan käyttöön `swag`-kontissa palvelu Basic Auth, joka rajoittaa Home Assistantin kirjautumissivulle pääsyä selaimeen ponnahtavalla käyttäjätunnuksen ja salasanan kyselyikkunalla. Basic Auth -palvelun käyttäjätunnus ja salasana tallennetaan tiedostoon *.htpasswd*, ja ne tulevat olemaan eri tunnukset kuin Home Assistant -instanssin käyttäjätunnus ja salasana. Basic Auth -palvelun käyttäminen edellyttää, että palvelulla suojattuun sivustoon kirjaudutaan ainoastaan HTTPS-yhteyttä käyttäen, eli tunnukset lähetetään verkon yli TLS-salattuina. `Swag`-kontti konfiguroitiin

aikaisemmin käyttämään HTTPS-yhteyttä, joten Basic Auth -palvelun ja Home Assistant -instanssin käyttäjätunnus ja salasana siirtyvät verkon yli salattuina. (McKay, 2020)

Basic Auth -palvelu otetaan käyttöön komennolla `docker exec -it swag htpasswd -c /config/nginx/.htpasswd kayttajatunnus1` , josta tunnus on anonymisoitu tässä raportissa.

Dockerin `exec` -komentojen avulla voidaan antaa suoritettavia komentoja WSL-komentoriviltä suoraan käynnissä olevan kontin komentotulkille. Komennon antamisen jälkeen WSL-komentoriville kirjoitettiin salasana komennossa luodulle käyttäjätunnukselle, kuten kuvassa 15 nähdään. Kuvassa näkyville riveille on kirjoitettu täsmäävät salasanat, mutta WSL-komentoriville ominaisesti merkkejä ei näytetä. (Dockerdocs, n.d.-a)

Kuva 15. Salasanan määrittäminen htpasswd-komennolla WSL-komentorivillä.

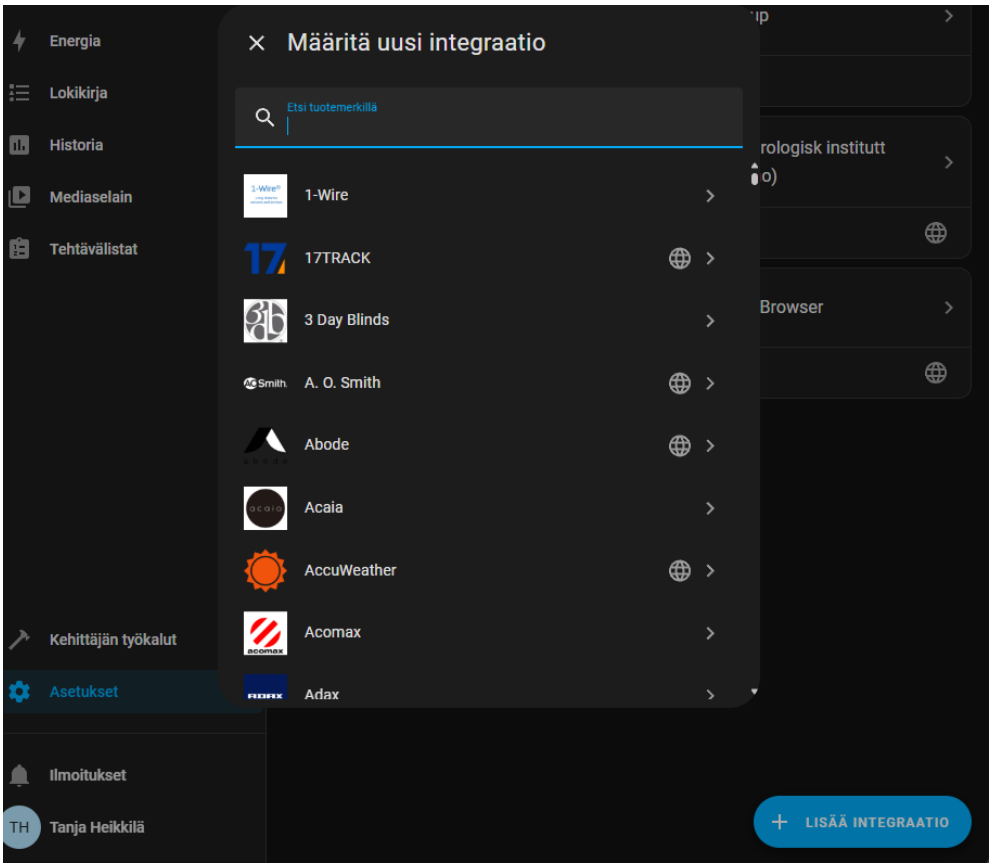
Kohta `swag` viittaa kontin nimeen, jossa komento suoritetaan. Kohdasta `htpasswd` eteenpäin komento käytännössä pyytää luomaan salasanan, jonka jälkeen se luo kontin polkuun uuden .htpasswd -tiedoston ja tallentaa sinne komennossa annetun käyttäjätunnuksen ja komentorivillä juuri luodun salasanan. (The Geek Diary, n.d.)

6.7 Laitteiden lisääminen järjestelmään

Järjestelmään liitettiin kaksi eri liitäntätapaa ja -protokollaa käyttävää kodin IoT-laitetta. Trådfri-tukiaseman lisääminen Home Assistantiin aloitettiin etsimällä laitteen saama IP-osoite kodin lähiverkosta. Tukiaseman MAC-osoite, eli laitteen yksilöivä tunnuskoodi löytyi laitteen pohjasta. IP-osoitetta etsittiin aluksi Windowsin komentokehotteessa MAC-osoitteen perusteella komennolla `arp -a` , joka listaa joitakin verkkoon liitettyjä laitteita näyttäen niiden MAC- ja IP-osoitteet, mutta ei kuitenkaan Trådfri-tukiasemaa. Tukiaseman IP-osoite saatiin lopulta selville kotireitittimen hallintasivulta.

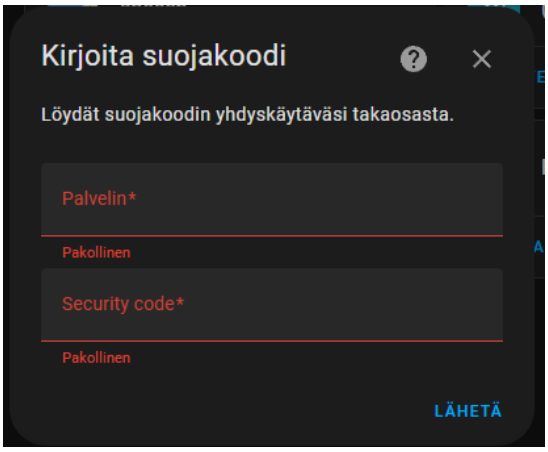
Laitteen lisäämistä jatkettiin Home Assistantissa kirjautumalla ensin sisään, jonka jälkeen Asetukset -valikossa edettiin Laitteet & Palvelut -valikkoon. Välilehdellä Integraatiot painettiin kohdasta ”+ lisää integraatio”, joka näkyy kuvan 16 oikeassa alakulmassa.

Kuva 16. Uusien integraatioiden määrittäminen Home Assistantiin.



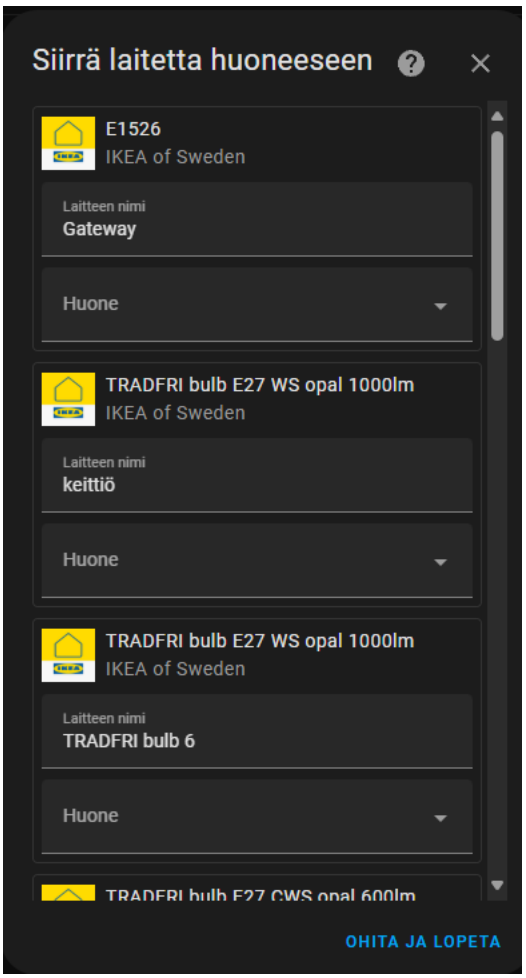
Kuvan 16 yläreunassa näkyvään hakukenttään kirjoitettiin ”IKEA”, jolloin valittavaksi tuli yksi kuvake IKEA:n logolla. Valitsemalla kuvake edettiin kolme eri IKEA:n tuotetta käsittävään listaan, joista valittiin vaihtoehto ”IKEA Trådfri”. Seuraavaksi, kuvassa 17 näkyvän ponnahdusikkunan kenttiin syötettiin laitteen IP-osoite ja laitteen pohjasta ”Security Code” -kohdasta löytyvä koodi, ja valittiin ”lähetä”.

Kuva 17. Trådfri-tukiaseman integraation lisääminen.



Seuraavaksi avautui kuvassa 18 näkyvä ikkuna, jossa kehotetaan siirtämään listassa näkyviä laitteita huoneisiin automaatioita varten.

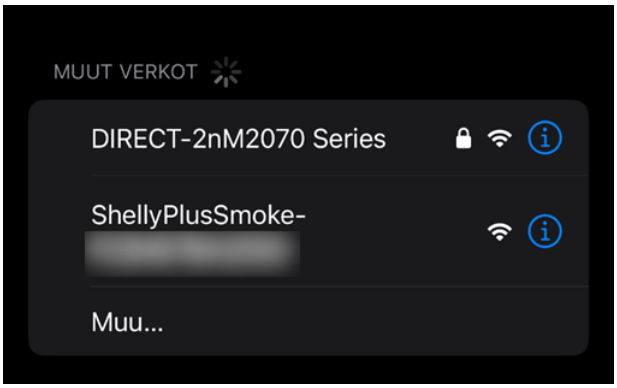
Kuva 18. Trådfri-tukiasemaan liitetyt laitteet Home Assistantissa.



Trådfri-tukiaseman lisääminen viimeisteltiin valitsemalla ”ohita ja lopeta”, joka näkyy kuvan 18 alareunassa. Tukiasemaan liitetyt laitteet ilmestyivät näkymään Home Assistantin yleisnäkymäsivulle. Listatut laitteet ovat jo aiemmin tukiasemaan kytkettyjä kodin laitteita, kuten valaisimen polttimoita.

Shelly Plus Smoke -paloilmaisimen liittäminen aloitettiin asettamalla se huoltotilaan painamalla sen päällä olevaa painiketta kolme kertaa, jolloin paloilmaisin muodosti kahden minuutin ajaksi langattoman lähiverkon. Shellyn lähiverkko oli löydettävissä matkapuhelimen WiFi-asetusten valikossa, joka näkyy kuvassa 19 nimellä ”ShellyPlusSmoke”.

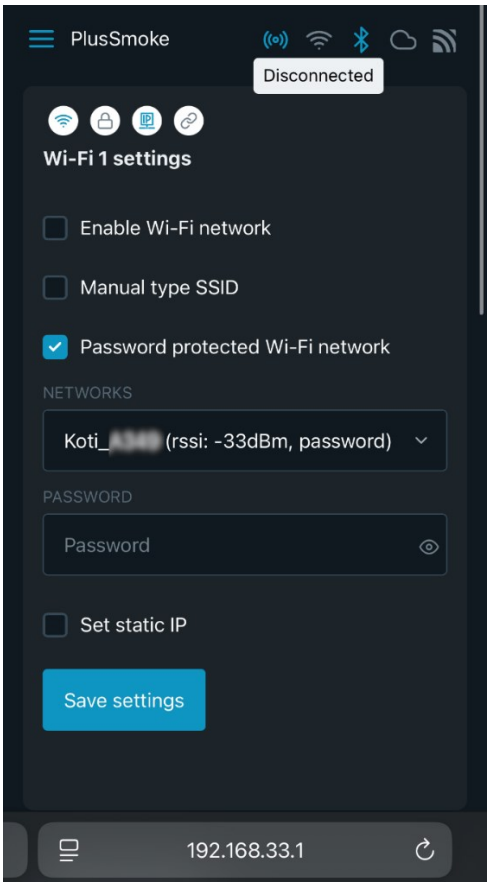
Kuva 19. Älypuhelimen WiFi-asetukset valikko - saatavilla olevat verkot.



Shellyn lähiverkkoon liityttiin älypuhelimella, jonka jälkeen selaimella mentiin paloilmaisimen hallintasivulle osoitteeseen ”http://192.168.33.1”.

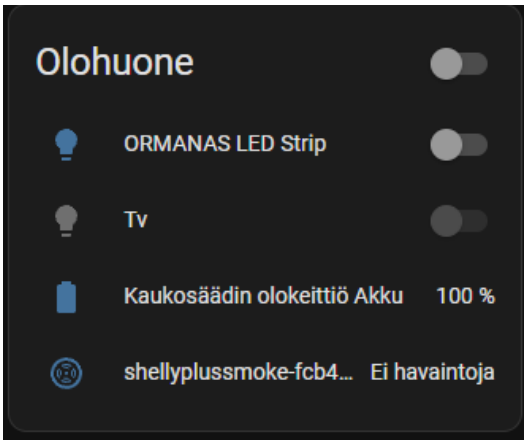
Hallintasivulla paloilmaisin liitettiin kodin lähiverkkoon, eli samaan verkkoon, johon Home Assistant on liitetty. Jotta paloilmaisin saadaan lisättyä Home Assistantiin, hallintasivulla määritettiin paloilmaisimielle oma IP-osoite ja *gateway*- eli oletusyhdyskäytävän osoite avaamalla valikko kohdasta ”set static IP”, näkyvillä kuvassa 20. Paloilmaisin asetettiin liittymään kodin lähiverkkoon valitsemalla aktiiviseksi kohta ”enable WiFi network”. Asetukset tallennettiin valitsemalla lopuksi ”save settings”.

Kuva 20. Shelly Plus Smoke -paloilmaisimen hallintasivu.



Tehtyjen asetusten jälkeen, paloilmaisin liitettiin vielä Home Assistantiin, samalla tavalla kuin IKEA Trådfri -tukiasemankin integraatio lisättiin. Paloilmaisin lisättiin huoneeseen ”Olohuone”, ja se ilmestyi Home Assistantin yleisnäkymään, kuten kuvassa 21 nähdään.

Kuva 21. Home Assistantin yleisnäkymä, laitteet huoneessa "Olohuone".



6.8 Varmuuskopiointi

Suoraviivainen tapa huolehtia Home Assistantin sisäisistä asetuksista, kuten liitetyistä laitteista ja automaatioista, on käyttää Home Assistantin automaattista varmuuskopiointitoimintoa.

Home Assistantissa valittiin Asetukset – Järjestelmä – Varmuuskopiot – Määritä Varmuuskopio. Avautui ponnahdusikkuna, jossa kehotetiin kopioimaan ja tallentamaan salausavain turvalliseen paikkaan, joten salausavain ja muut tiedot tallennettiin salasanojen hallintasovellukseen.

Seuraavassa ponnahdusikkunassa määritettiin automaattiset varmuuskopiot suositelluilla asetuksilla, eli järjestelmästä otetaan varmuuskopio päivittäin ja niistä säilytetään kolme viimeisintä, ja sitä vanhemmat poistetaan automaattisesti. (Home Assistant, n.d.-a)

Varmuuskopio näkyy nyt Home Assistantin valikossa Asetukset – Järjestelmä – Varmuuskopiot – Varmuuskopioni. Tiedosto tallentuu polkuun *../homeassistant/config/backups/* ja sen voi myös tarkistaa WSL:ssä komennolla `ls ~/projekti/homeassistant/config/backups/` , jolloin nähdään listattuna yksi varmuuskopio:
`Automatic_backup_2025.5.1_2025-05-21_15.15_XXXXXXXX.tar`

Jos järjestelmän käyttöä jatketaan tulevaisuudessa, olisi syytä toteuttaa automatisoitu varmuuskopiointi koko projektin tiedostoista. Järjestelmän asetustiedostoista ja Home Assistantille mountatuista volumeista olisi mahdollista ottaa varmuuskopio eri tavoilla.

Yksi tapa olisi kopioida projektin pääkansion tiedostot kerran päivässä, säilyttää kopioita kolmen päivän ajan, ja poistaa sitä vanhemmat kopiot. Käytäntö voitaisi toteuttaa tapahtumaan automatisoidusti WSL-komentoriviltä esim. skriptin avulla. Ratkaisu voisi olla toimiva, mutta järjestelmän kasvaessa myös tiedostojen koko saattaa kasvaa kohtuuttoman suureksi, jolloin tallennustilaa kuluu paljon. (Valencia, 2011)

Toinen tapa koko projektin tiedostojen varmuuskopiointiin olisi toteuttaa se inkrementaalisella varmuuskopioinnilla (incremental backup), jossa tallennetaan vain edelliseen varmuuskopion jälkeen tapahtuneet muutokset tiedostoissa sekä lisätyt ja

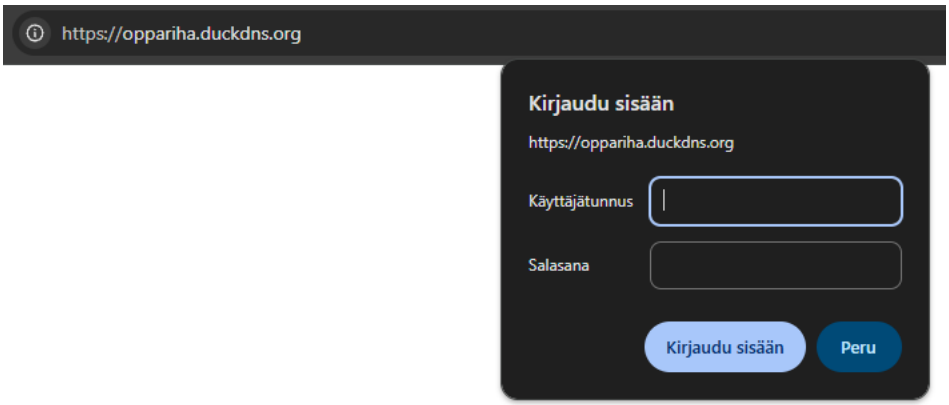
poistetut kansiot. Eräs tapa toteuttaa kuvatun kaltainen automatisoitu inkrementaalinen varmuuskopiointi tapahtumaan säännöllisin väliajoin voitaisi toteuttaa hyödyntämällä *cron* ja *rtab* -työkaluja. (Valencia, 2011)

6.9 Järjestelmän osien testaus

Tässä luvussa kuvaillaan järjestelmän toimintojen testausta käytännössä. Testausten perusteella arvioitiin Home Assistantin Basic Auth -lisäosan ja Home Assistant -tilin monivaiheisen tunnistautumisen toimivuutta. Home Assistantin toimintaa testattiin luomalla kahdelle laitteelle automaatiot ja arvioimalla niiden toimintaa käytännössä. Konttien tietoturvaa arvioitiin mm. analysoimalla konfiguraatiotiedostoja ja tekemällä niihin parannuksia joillakin osa-alueella.

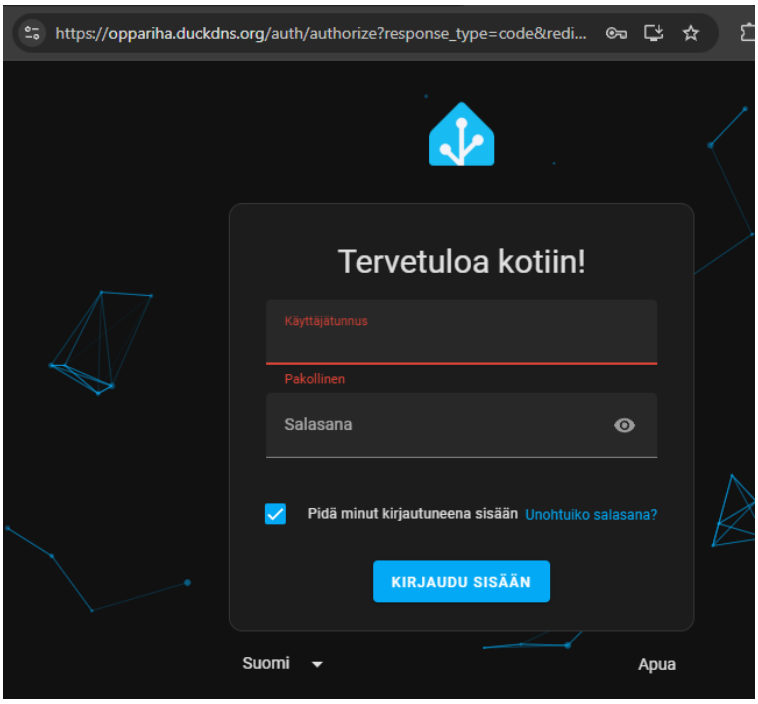
Etäyhteyden kautta tämän projektin Home Assistant -instanssiin kirjaudutaan nettiselaimella osoitteessa ”https://oppariha.duckdns.org/”. Ennen Home Assistantin kirjautumissivun latautumista selaimen osoiterivin alapuolelle avautuu ponnahdusikkuna, jossa kysytään Basic Auth -käyttäjätunnusta ja salasanaa, kuten kuvasta 22 nähdään.

Kuva 22. Basic Auth -tunnuksia kysyvä ponnahdusikkuna selaimessa.



Basic Auth -käyttäjätunnus ja salasana syötettiin kenttiin ja valittiin ”kirjaudu sisään”, selain siirtyi Home Assistantin kirjautumissivulle, joka nähdään kuvassa 23. Selaimen osoiteriville ilmestyi pitkä osoite, eli sivustoon on siirrytty onnistuneen tunnistautumisen kautta. Instanssin kirjautumissivua suojaava Basic Auth -lisäosa todettiin toimivaksi ja oikein konfiguroiduksi.

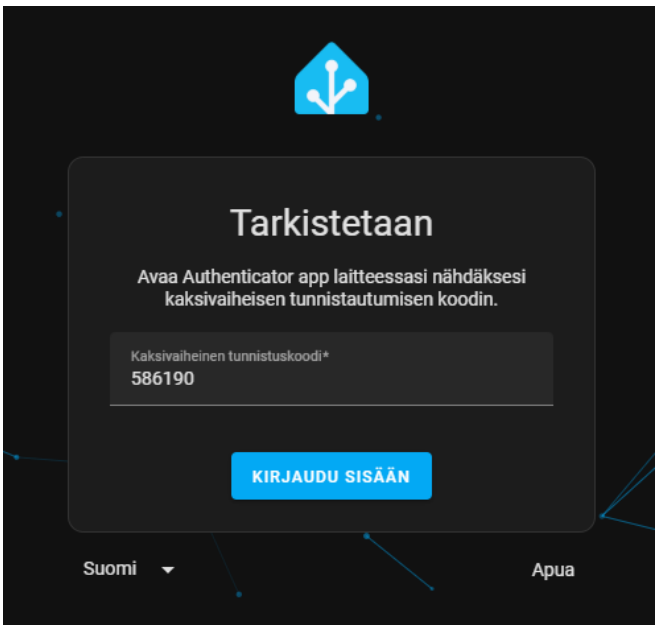
Kuva 23. Home Assistantin etäkirjautumissivu selaimessa.



Home Assistantin tunnukset syötettiin kenttiin ja painettiin kohdasta ”kirjaudu sisään”, selain siirtyi Home Assistantin etusivulle eli yleisnäkymään. Osoiterivillä näkyi instanssin verkko-osoite. Docker-konteilla toteutettu etähallinta todettiin näin ollen toimivaksi.

Monivaiheinen tunnistautuminen lisättiin pääkäyttäjätiliin. Kun Home Assistant -tiliin kirjaudutaan uudelleen tilin käyttäjätunnuksella ja salasanalla, avautuu kuvassa 24 näkyvä ikkuna, johon kirjoitetaan Google Authenticator -sovelluksessa Home Assistant -tilin kohdalla näkyvä koodi.

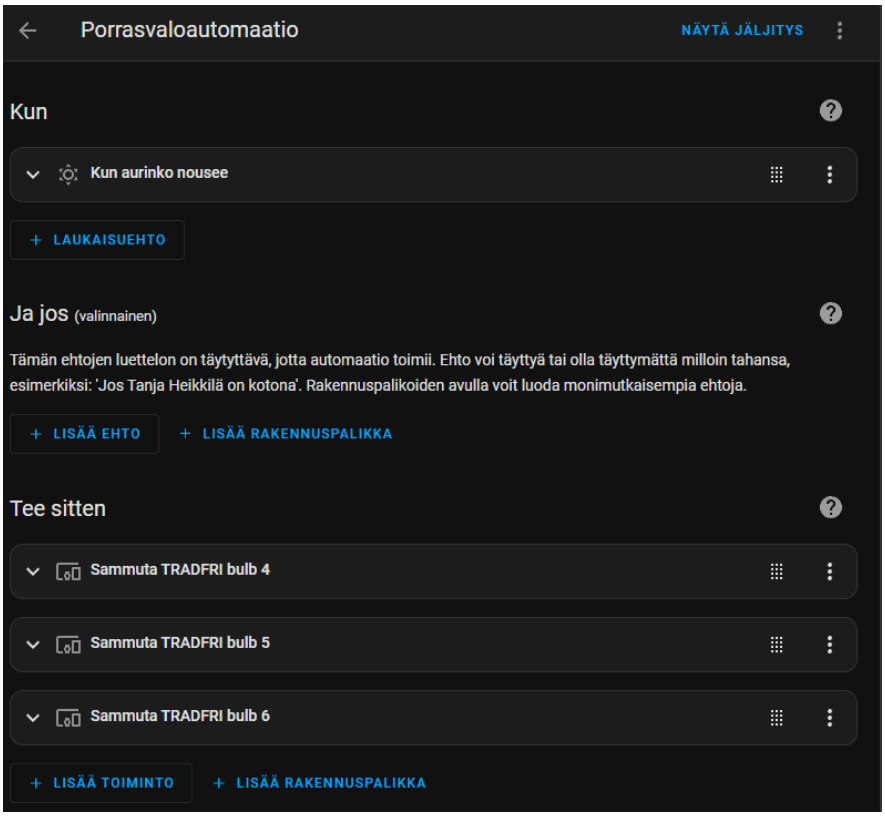
Kuva 24. Monivaiheisen tunnistautumisen koodin tarkistus Home Assistant -tiliin kirjauduttaessa.



Home Assistantin yleisnäkymäsivulle päästiin kirjautumaan, eli monivaiheinen tunnistautuminen otettiin käyttöön onnistuneesti.

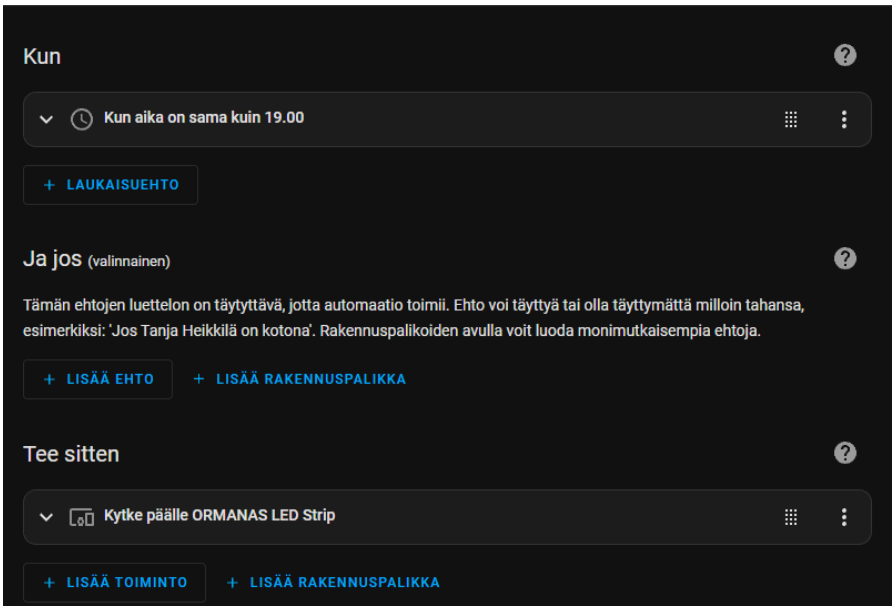
Järjestelmän toimivuutta testattiin luomalla kahdelle järjestelmään liitetylle laitteelle yksinkertaiset automaatiot. Kodin portaikossa sijaitsevalle kolmilamppuiselle valaisimelle luotiin valoisuuteen perustuva automaatio, jonka on tarkoitus sammuttaa valaisin päivän valoisaksi ajaksi. Home Assistantin automaatiot luodaan ja otetaan käyttöön valikossa Asetukset – Automaatiot & tilanteet – Lisää Automaatio – Luo Uusi Automaatio. Laukaisuehdoksi valittiin valikosta ”kun aurinko nousee” ja toiminnoksi ”sammuta Trådfri bulb”, kuten kuvassa 29 nähdään.

Kuva 25. Automaation luominen portaikon valaisimelle Home Assistantissa.



Olohuoneessa sijaitsevalle LED-valonauhalle asetettiin kaksiosainen aikaperusteinen automaatio. Valonauhan käynnistävän automaation asetukset näkyvät kuvassa 30.

Kuva 26. Automaation luominen TV:n LED-nauhalle Home Assistantissa.



Automaatioiden teko asetusvalikon kautta toimi intuitiivisesti, joten ohjeita ei tarvittu näin yksinkertaisten automaatioiden luomiseen. Automaatioita on mahdollista luoda myös suoraan YAML-tiedostoa käyttäen, jolla voidaan toteuttaa monimutkaisiakin automaatioita esim. kopioimalla muiden käyttäjien luoma tiedosto, joka muokataan omille laitteille ja tarpeisiin soveltuvaksi.

Molemmat automaatiot ohjasivat valaisimia odotetusti. Valaisimia pystyi myös kytkemään ilman viivettä tai muita ongelmia päälle ja sammuksiin Home Assistantin kautta suoraan virtakytkinpainikkeista. Voitiin siis todeta, että laitteiden hallinta järjestelmän kautta toimii oikein.

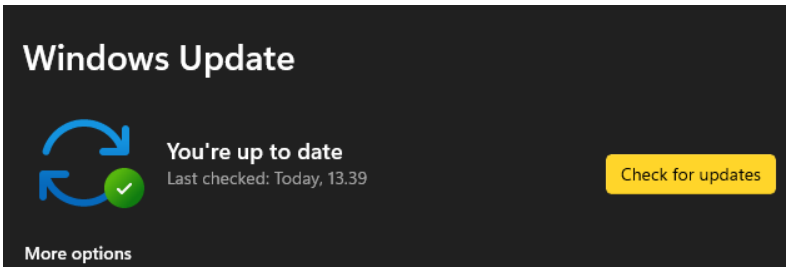
Konttien tietoturvatestauksissa sovellettiin OWASP (Open Worldwide Application Security Project) -nimisen tietoturvaan keskittyvän järjestön listausta tunnetuista ja hyvien käytänteiden mukaisista tavoista analysoida ja parantaa Docker-konttien tietoturvaa. Listauksen kohdissa käydään läpi järjestelmän eri osa-alueita, jotka ovat alttiita tietoturvahyökkäyksille tietyissä tilanteissa, sekä ohjeistetaan määrittämän konfiguraatiot siten, että riskit hyökkäyksille poistettaisi tai ainakin minimoitaisiin. (OWASP, n.d.)

Seuraavissa kappaleissa käydään listaus läpi soveltaen järjestelmään kohtia, joita oli mahdollista senhetkisellä valmiusasteella, ja pohdittiin kohtia, joita voitaisi soveltaa järjestelmän ja tuotteen kehityksen myöhemmissä vaiheissa. Testaukset suoritettiin WSL-komentorivillä samassa isäntäkoneen virtuaaliympäristössä, jossa testattavat kontitkin olivat käynnissä.

Huolehdi ohjelmistopäivityksistä – tämä kohta on tietoturvan kannalta kriittinen vaatimus, ja se koskettaa koko järjestelmää. Päivitettäviä kohteita tässä tapauksessa ovat isäntäjärjestelmä eli Windows 11, Docker Desktop -ohjelmisto Windowsissa, Ubuntu jakelu WSL:ssä ja Docker imaget eli Docker Compose -tiedostossa määritetyt levykuvat.

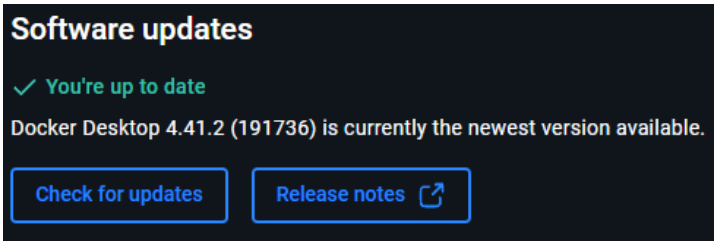
Windowsin asennettavissa olevat käyttöjärjestelmä- ja tietoturvapäivitykset tarkistettiin seuraavasti. Windows käyttöjärjestelmä valikosta Settings – Windows Update painettiin kohdasta ”Check for Updates”, jonka jälkeen järjestelmä tarkisti päivitysten ajantasaisuuden. Päivitykset olivat ajan tasalla, kuten kuvassa 25 nähdään.

Kuva 27. Windows Updaten järjestelmäpäivitykset ajan tasalla.



Docker Desktop -ohjelmisto päivitettiin Docker Desktop -sovelluksen valikosta Settings – Software Updates – Download Updates – Update and Restart. Sovellus latasi ja asensi ohjelmistopäivityksen. Uudelleenkäynnistymisen jälkeen Dockerin taustapalveluiden käynnistyttyä samasta sovelluksen valikosta tarkistettiin päivityksen tila, joka oli nyt ajan tasalla, kuten kuvassa 26 nähdään.

Kuva 28. Docker Desktop -sovelluksen ohjelmistopäivitykset ajan tasalla.

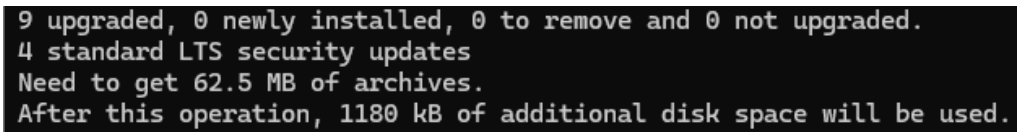


Kontit ajettiin alas Docker Desktopin ohjelmistopäivityksen ajaksi komennolla `docker compose down`. Ohjelmistopäivityksen jälkeen kontit käynnistettiin uudelleen komennolla `docker compose up -d`.

Virtuaaliympäristö WSL ja Ubuntu -käyttöjärjestelmä päivitettiin komennoilla `sudo apt update`, joka listasi uusimmat saatavilla olevat päivityspaketit. Päivityspaketit asennettiin järjestelmän ohjelmiin komennolla `sudo apt upgrade -y` . Päivityksen aikana asentui yhdeksän uutta ohjelmistoversiota, joista neljä ohjelmistojen pitkäaikaisia

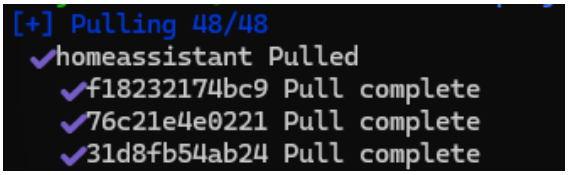
tietoturvapäivityksiä. Kuvassa 27 nähdään myös asennuspakettien vaatiman tilan tarve ja lopullinen lisääntynyt levytilan tarve.

Kuva 29. Komennon ”sudo apt upgrade -y” osittainen tuloste WSL-komentorivillä.



Kaikkien kolmen kontin Docker imaget päivitettiin kansiossa *~/projekti* seuraavasti. Uusimmat versiot tiedostossa *docker-compose.yml* määritetyistä levykuvista ladattiin komennolla `docker-compose pull`. Komennon avulla ladattiin 48 kohdetta, kuten kuvassa 28 nähdään. (LinuxServer.io, 2025-b)

Kuva 30. Komennon ”docker-compose pull” osittainen tuloste WSL-komentorivillä.



Konttien käynnistys uusista levykuvista tehtiin komennolla `docker compose up -d`. Komennon antamisen jälkeen konttien käynnistymisprosessin nähtiin etenevän komentorivillä tilasta ”Recreating”, edeten tilaan ”Recreated”. Lopulta kontit jäivät tilaan ”Started” eli ne käynnistyivät normaalisti. (LinuxServer.io, 2025-b)

Aseta konttien ajonaikaiset käyttöoikeudet käyttäjätasoiseksi – kohta tarkoittaa, että konteilla ei tulisi olla määritettynä ajonaikaisia root- eli pääkäyttäjätasoisia oikeuksia prosessien suorittamiseen, koska ne mahdollistavat hyökkäykset isäntäjärjestelmää kohtaan.

Konttien oikeudet tarkistettiin komennoilla `docker exec -it duckdns id`, `docker exec -it swag id` ja `docker exec -it homeassistant id`. Tulosteista voitiin päätellä, että kontit käyttävät root-oikeuksia prosessien suorittamiseen kontin sisällä.

Kontit tarvitsevat root-oikeuksia konttien palveluiden toiminnan kannalta välttämättömiin prosesseihin, kuten kirjoittaakseen konteille mountattuihin isäntäjärjestelmän tiedostoihin, verkkoyhteyksien toimimiseen ja tiettyjen prosessien hallintaan. Käytännössä Linuxserver -

imagejen avulla luoduissa konteissa *docker-compose.yml* -tiedostossa määritetyt muuttujat `environment: - PUID=1000` ja `PGID=1000` asettaa kontin suorittamaan käynnistyksen jälkeiset prosessit alennetulla käyttöoikeustasolla. Edellä mainitulla tavalla pyritään suojamaan kontteja ulkopuolelta tulevilta uhkia, kuten käyttöoikeuksien korotukseen perustuvia hyökkäyksiä vastaan. (LinuxServer.io, 2023)

Käynnistyksen yhteydessä Docker-konteille muodostuu automaattisesti suojausmekanismeja, kuten *namespaces* ja *control groups*. *Namespaces* on mekanismi, joka käytännössä eristää kontit niin toisistaan kuin isäntäjärjestelmästäkin rajoittamalla prosessien toiminnan konttien sisälle. *Control groups* -mekanismi puolestaan varmistaa, että kontti saa käyttöönsä tarvitsemansa määrän isäntäjärjestelmän resursseja, kuten muistia ja prosessoritehoa, toimiakseen. Mekanismi myös rajoittaa konttien resurssien ylikäyttöä, jolla vältetään resurssien – etenkin muistin – loppumisesta aiheutuva koko järjestelmän kaatuminen. (Dockerdocs, n.d.-c)

Rajoita kontin oikeuksia (*capabilities*) Linux-ytimen toimintoihin – OWASPin listauksen mukaan konttien koventamiseksi suositellaan `--cap-drop all` -asetuksen käyttöä *docker-compose.yml* -tiedostossa. Käytännössä asetus poistaisi kontilta kokonaan oikeudet Linux-ytimen toimintoihin, ja `--cap-add` -asetuksella kontille voitaisi sallia yksittäisiä oikeuksia tarvittaessa.

Vaikka edellä kuvailtu kovennustapa lisäisi konttien suojausta merkittävästi, sen toteuttamisesta luovuttiin sen monimutkaisuuden takia: `--cap-drop` ja `--cap-add` -asetusten käyttö vaatisi joko ennestään perehtyneisyyttä aiheeseen tai tiedonhakua Linux-ytimen *capabilites* -oikeuksista, sekä toimivan yhdistelmän etsimistä yrityksen ja erehdyksen kautta.

Toinen OWASPin ohjeistuksessa mainittu tapa koventaa kontti, olisi välttää `--privileged` -lipun käyttöä kontin käynnistyksen yhteydessä, koska käytännössä se antaa kontille root-oikeudet isäntäkoneen virtuaaliseen Linux-ytimeen. Home Assistantin dokumentissa kyseinen määritys kuitenkin on tehty *docker-compose.yml* -esimerkkitiedostossa, sillä se yksinkertaistaa isäntäkoneeseen esim. USB:llä tai Bluetooth-yhteydellä kytkettyjen laitteiden liitettävyyttä järjestelmään. (Home Assistant, n.d.-c).

Estä mahdollisuus kontin sisäiseen käyttöoikeuksien korotukseen – tämä kohta tarkoittaa, että kontteja tulisi aina ajaa ilman mahdollisuutta kontin sisällä tapahtuvaan käyttöoikeuksien korotukseen. Tällä estetään jo aiemmin mainitun käyttöoikeuksien

korotukseen perustuvaa järjestelmään tunkeutumista. Home Assistantin kontti toimii tällä hetkellä root-tasolla, ja koska se ei tue muita käyttäjätasoja, sille ei ollut syytä asettaa kyseistä määritystä.

Konttien sisäinen käyttöoikeuksien korottaminen estettiin lisäämällä *docker-compose.yml* -tiedoston `duckdns` ja `swag` -palveluiden alle seuraavat rivit:

```
security_opt:
    - no-new-privileges
```

Määritysten toimintaa kokeiltiin ottamalla uudet määritykset sisältävä *docker-compose.yml* -tiedosto käyttöön. Kontit poistettiin ja luotiin uudelleen komennoilla `docker compose down && docker compose up -d`. Selaimella päästiin kirjautumaan instanssiin paikallisesti ja HTTPS-yhteyden kautta, joten määritys ei ainakaan suoraan estänyt palveluiden toimintaa. Myöskään konttien lokitiedoissa ei näkynyt virhesanomia.

Huomioi konttien välinen kommunikointi – tässä kohtaa listausta kehotetaan luomaan toistensa kanssa kommunikoiville konteille oma projektikohtainen Docker-verkko, sen sijaan, että käytettäisi oletuksena muodostuvaa verkkoa. Tällä tavalla mm. useiden eri verkkojen hallittavuus paranee, koska kullekin verkolle voidaan määritellä niiden tarvitsemat asetukset toisistaan riippumatta. Projektissa tämä kohta toteutettiin luomalla konteille oma verkko nimeltä ”projekti_network”.

Listauksessa tarkastettavia kohtia on yhteensä 13 kpl, joista edellä toteutettujen kohtien lisäksi vielä muutama olisi toteutettavissa myöhemmin tarpeen vaatiessa. Näitä keinoja olisi mm. rajoittaa kontin tietokoneen resurssien, kuten muistin ja prosessoritehon käyttöä, sekä kontin käynnistysyritysten ja prosessien määrää. Tällä keinolla vältyttäisi mm. *Denial of Service* (DoS) eli palvelunestohyökkäysten haittavaikutuksilta, ja konttien muiden virhetilanteiden aiheuttamalta järjestelmän ylikuormittumiselta.

Viimeinen listauksesta nostettu kohta on konttikuvien automaattinen tietoturvatarkastus esim. käyttämällä automatisoituja työkaluja. Tätä kohtaa voitaisiin soveltaa jatkossa osana järjestelmän kehitystä.

Automatisoituja työkaluja on saatavilla mm. Dockerfilen tarkistamiseen, Docker imagejen skannaukseen, sekä arkaluontoisten tietojen etsimiseen levykuvista. Työkalujen

toimintaperiaate on skannata eli tarkastaa edellä mainitut kohteet, kuten Dockerfile-tiedoston, etsien sieltä tunnettuja tietoturvahaavoittuvuuksia, arkaluontoisia tietoja ja virheellisesti konfiguraatioita. Työkalu muodostaa löydöksistä raportin ja tarjoaa käyttäjälle suositeltavia korjaustapoja löydettyjen ongelmien korjaamiseen.

7 Tuotteistamisen arviointi ja jatkokehitys

Opinnäytetyössä toteutettu järjestelmä olisi mahdollista tuotteistaa Home Assistantin käyttöönottoa helpottavaksi sovellukseksi tai esim. valmiiksi ohjelmistopaketiksi.

Järjestelmässä tehtiin reitittimeen porttiohjaukset, jotta verkkoliikenne internetistä päin on mahdollista kodin sisäverkossa toimiviin Docker-kontteihin ja lopulta Home Assistant-palveluun. Jatkokehitysideana tuotteelle, reitittimen porttien ohjauksen sijaan, mahdollisuuden liikennöintiin voisi toteuttaa käyttäen VPN-tekniikkaa (Virtual Private Network), joko tunneloimalla komentoriviltä tai Docker-kontissa toimivan sovelluksen avulla. Toisena toteutusvaihtoehtona voisi toimia esim. Cloudflaren tai muun kolmannen osapuolen tarjoama palvelu. Käyttäjä voisi valita haluamansa toteutustavan, jonka perusteella asennussovellus suorittaisi taustalla yhteyden muodostamiseen tarvittavat komennot komentorivillä.

Toinen kehitysidea tuotteelle kohdistuu Home Assistantin sisäisiin ominaisuuksiin. Docker-kontilla toteutettu versio, Home Assistant Container, ei sisällä yleisesti käytettyjä lisäosia. Kotiautomaatiojärjestelmien yhteydessä, joissakin tilanteissa lisäosia kuitenkin tarvittaisi, jotta esim. Bluetooth-, Zigbee- ja MQTT-protokollilla yhteyden muodostavien laitteiden liittäminen järjestelmään olisi mahdollista. Container-versiossa lisäosat otetaan tavallisesti käyttöön luomalla lisäosaa varten uusi Docker-kontti, jossa sovellus toimii. Käyttäjä voisi valita tarvitsemansa lisäosat, jotka asennussovellus lisäisi compose-tiedostoon uusina palveluina.

Viimeinen tuotekehitykseen liittyvä idea olisi lisätä asennussovellukseen toiminto, jonka avulla otettaisi käyttöön automaattiset varmuuskopiot koko järjestelmästä, jotta esim. Home Assistantiin lisättyjen laitteiden ja automaatioiden konfiguraatioita ei menetettäisi, vaikka järjestelmätiedostot menetettäisi jostain syystä.

Järjestelmän tuotteistamista pohdittaessa huomattiin, että ohjelmistojen ja Docker imagejen päivitysten pitäminen ajan tasalla voitaisiin toteuttaa myös erillisellä sovelluksella, joka

tarkistaisi saatavilla olevat ohjelmistopäivitykset ja asentaisi ne tarvittaessa. Tämä päivityksiin liittyvä sovellus voisi olla valittavissa asennettavaksi asennussovelluksessa järjestelmää käyttöönotettaessa.

8 Pohdinta

Opinnäytetyön tavoitteet saavutettiin odotetusti. Järjestelmään toteutettiin toimiva etäkirjautumisominaisuus kontissa ajettavilla palveluilla, joiden konfigurointi onnistui dokumentaatioita seuraamalla. Järjestelmän toteuttaminen syvensi myös omaa osaamistani Docker-pohjaisten järjestelmien rakentamisessa ja hallinnassa sekä verkkopalvelun yksinkertaisista suojaamistavoista.

Järjestelmä toteutettiin tavanomaiseen opiskelukäytössä olevaan kannettavaan tietokoneeseen, mikä ei ole käytännössä pidemmän päälle toimiva ratkaisu. Toteutettu järjestelmä, eli Docker-konteissa toimivat palvelut, tullaan siirtämään ainoastaan palveluiden käyttöön varattuun mini-PC:hen, joka on päällä 24/7 ilman virransäästö tai muita keskeyttäviä toimintoja. Tällöin palvelu on käytännössä lähes jatkuvasti saatavilla, mikä on riittävä toimintavarmuus omiin tarpeisiin kotikäytössä.

Työssä toteutettu järjestelmä olisi mielestäni tuotteistettavissa eri tavoin. Työssä ideoitiin asennussovellus, jonka tarkoitus on avustaa käyttäjää järjestelmän käyttöönotossa. Tuotekehitystä pidemmälle vietäessä, asennussovellus voisi olla osa tuotetta, kuten kuluttajamarkkinoille suunnattua *plug & play* -periaatteella toimivaa IoT-laitepakettia.

Tekoälyn ja laajojen kielimallien käyttö ovat tulleet jäädäkseen, ja ne tarjoavat uusia mahdollisuuksia myös älykotijärjestelmien kehittämiseen. Google AI:n ja OpenAI:n keskusteluagentit mahdollistavat jo nyt järjestelmän ohjaamisen tekoälyn avulla Home Assistantissa. Nämä yhdistettynä jatkuvaan kehitykseen ja integraatioiden lisäämiseen aktiivisen yhteisön toimesta tekevät Home Assistantista jatkuvasti suositumman älykotiratkaisun eri tarpeisiin myös tulevaisuudessa. (Nijhof, 2024)

Lähteet

Bugnion, E. (1.11.2012). *Bringing Virtualization to the x86 Architecture with the Original VMware Workstation.* ACM. https://dl.acm.org/doi/10.1145/2382553.2382554

Cloud_Devops. (11.12.2017). Para virtualization vs Full virtualization vs Hardware assisted Virtualization. UnixArena. https://tinyurl.com/4brc5bnn

Comodo. (n.d.). *HTTPS: What is HTTPS?* InstantSSL. Arkistoitu 12. helmikuuta 2015, Internet Archive Wayback Machine: https://tinyurl.com/2feh9d5e

Davidson, A. (26.3.2025). *These Are the 7 Best Ways to Run Home Assistant.* How To Geek. https://www.howtogeek.com/best-ways-to-run-home-assistant/

Delta Wye Electric. (n.d.). *Top 10 Industrial Automation Protocols Explained.* https://deltawye.com/industrial-automation-protocols/

Diener, D. (5.6.2024). *What Is Home Assistant? And Why Isn't Everyone Using It?* How To Geek. https://www.howtogeek.com/what-is-home-assistant-and-why-isnt-everyone-using-it/

Dockerdocs. (n.d.-a). *Docker container exec*. Docker documentation. Haettu 24.4.2025 osoitteesta: https://docs.docker.com/reference/cli/docker/container/exec/

Dockerdocs. (n.d.-b). *Docker Desktop.* Docker documentation. Haettu 31.3.2025 osoitteesta: https://docs.docker.com/desktop/

Dockerdocs. (n.d.-c). *Docker Engine security.* Docker documentation. Haettu 12.5.2025 osoitteesta: https://docs.docker.com/engine/security/

Dockerdocs. (n.d.-d). *Install Docker Desktop on Windows.* Docker documentation. Haettu 16.4.2025 osoitteesta: https://docs.docker.com/desktop/setup/install/windows-install/

Dockerdocs. (n.d.-e). *Networking overview.* Docker documentation. Haettu 20.5.2025 osoitteesta: https://docs.docker.com/engine/network/

Dockerdocs. (n.d.-f). *Services top-level elements.* Docker documentation. Haettu 10.4.2025 osoitteesta: https://docs.docker.com/reference/compose-file/services/

Dockerdocs. (n.d.-g). *Volumes.* Docker documentation. Haettu 2.6.2025 osoitteesta: https://docs.docker.com/engine/storage/volumes/

Easy2Patch. (24.5.2024). What are the Cybersecurity Risks Associated with Outdated Software and Operating Systems? *Easy2Patch Blog.* https://tinyurl.com/7h87ar

Fernandez, R. (20.6.2023). *What is Server Virtualization? How It Works, Types, and Examples.* ServerWatch. https://www.serverwatch.com/virtualization/server-virtualization/

FiTech101. (n.d.). *Internetin perusosat.* Haettu 19.5.2025 osoitteesta: https://fitech101.aalto.fi/fitech101/internet-and-browser-applications/1-principles-of-internet/

Fortinet. (n.d.). *What Is IoT Security? Challenges and Requirements.* Haettu 19.5.2025 osoitteesta: https://www.fortinet.com/resources/cyberglossary/iot-security

F-Secure. (13.8.2022). *Mikä on kyber-hyökkäys?* https://www.f-secure.com/fi/articles/what-is-a-cyber-attack

GeeksforGeeks. (4.3.2024-a). *What is Network Virtualization?* https://tinyurl.com/3dpru68j

GeeksforGeeks. (21.10.2024-b). *What is Dockerfile?* https://www.geeksforgeeks.org/what-is-dockerfile/

GeeksforGeeks. (2.4.2025-a). *Docker Compose.* https://www.geeksforgeeks.org/docker-compose/

GeeksforGeeks. (22.4.2025-b). *What is Docker?* https://www.geeksforgeeks.org/introduction-to-docker/

GeeksforGeeks. (21.3.2025-c). *What is Docker Image?* https://tinyurl.com/4wdfz9vw

GeeksforGeeks. (22.3.2025-d). *YAML Full Form.* https://www.geeksforgeeks.org/yaml-full-form/

Gunge, V. & Yalagi, P. (2016). *Smart home automation: a literature review.* International Journal of Computer Applications. https://www.ijcaonline.org/proceedings/rtdm2016/number1/24679-2568/

HiveMQ Team. (6.6.2023). MQTT Publish/Subscribe Architecture (Pub/Sub) – MQTT Essentials: Part 2. *HiveMQ Blog*. https://www.hivemq.com/blog/mqtt-essentials-part2-publish-subscribe/

Home Assistant. (n.d.-a). *Common tasks - installation independent.* Haettu 21.5.2025 osoitteesta: https://www.home-assistant.io/common-tasks/general/

Home Assistant. (n.d.-b). *HTTP*. Haettu 23.4.2025 osoitteesta: https://tinyurl.com/y77b3v6t

Home Assistant. (n.d.-c). *Linux.* Haettu 14.3.2025 osoitteesta: https://tinyurl.com/3a3uc4zw

Home Assistant. (n.d.-d). *Multi-factor authentication.* Haettu 28.4.2025 osoitteesta: https://www.home-assistant.io/docs/authentication/multi-factor-auth/

Home Assistant. (n.d.-e). *Onboarding Home Assistant.* Haettu 15.4.2025 osoitteesta https://www.home-assistant.io/getting-started/onboarding/

IBM. (2.8.2022). *1972: Introduction to VM/370* [video]. YouTube. https://tinyurl.com/5n862565

IBM. (12.5.2023). *What is the Internet of Things (IoT)?* https://tinyurl.com/2e6dv7pa

Jakhar, O. (n.d.). *Memory Virtualization in Cloud Computing.* Host IT Smart. https://tinyurl.com/5n6v8v86

Khvoynitskaya, S. (25.11.2019). The IoT history and future. *Itransition.* https://tinyurl.com/44hs2vc7

Kumar, R. (26.4.2025). *Step-by-Step Guide: Installing Docker on Windows 11 and 10.* TecAdmin. https://tecadmin.net/installing-docker-on-windows/

Lake. (23.5.2025). *What is a Smart Home?* https://www.lake.com/help/glossary/smart-home/

LinuxServer.io. (19.5.2025-a). *Linuxserver/duckdns.* https://docs.linuxserver.io/images/docker-duckdns/

LinuxServer.io. (17.5.2025-b). *Linuxserver/swag.* https://docs.linuxserver.io/images/docker-swag/

LinuxServer.io. (17.1.2025-c). *SWAG.* https://docs.linuxserver.io/general/swag/

LinuxServer.io. (17.10.2023). *Understanding PUID and PGID.* https://tinyurl.com/8kx9au9a

Löppönen, H. (20.4.2021). Datan virtualisoinnista haastaja perinteisille tekniikoille. *IBM: n Think-blogi.* https://tinyurl.com/4rnxsceu

McKay, D. (17.7.2020). *How to set up basic HTTP authentication in NGINX.* How To Geek. https://www.howtogeek.com/devops/how-to-setup-basic-http-authentication-on-nginx/

Microcontrollers Lab. (n.d.). *IoT Protocols: A Comprehensive Guide to Different Smart Home Protocols.* https://microcontrollerslab.com/iot-protocols-types/

Microsoft. (20.11.2023). *Set up a WSL development environment.* https://tinyurl.com/3bw38xyd

Microsoft. (2024-a). *Comparing WSL Versions.* https://tinyurl.com/mvt8nj8m

Microsoft. (2024-b). *How to install Linux on Windows with WSL.* https://tinyurl.com/y88fekej

Mudadla, S. (7.12.2023). *What is the use of .env file in projects? How to store sensitive information like API keys in .env file?* https://tinyurl.com/29wzuawv

Nabu Casa. (n.d.). *Remote UI.* Haettu 3.5.2025 osoitteesta: https://www.nabucasa.com/config/remote/

Nijhof, F. (5.6.2024). *2024.6: Dipping our toes in the world of AI using LLMs.* Home Assistant. https://www.home-assistant.io/blog/2024/06/05/release-20246

Osnat, R. (10.1.2020). *A Brief History of Containers: From the 1970s Till Now.* https://www.aquasec.com/blog/a-brief-history-of-containers-from-1970s-chroot-to-docker-2016/

OWASP. (n.d.). *Docker Security Cheat Sheet.* Haettu 15.5.2025 osoitteesta: https://tinyurl.com/bdf553vv

Randal, A. (2020). *The Ideal Versus the Real: Revisiting the History of Virtual Machines and Containers.* https://dl.acm.org/doi/pdf/10.1145/3365199

Riffid. (n.d.). *Mikä RFID?* Haettu 16.5.2025 osoitteesta: https://www.riffid.fi/mika-rfid

RouteThis. (2.12.2024). *How to Be Smarter About the IoT Smart Home Experience.* IoT For All. https://www.iotforall.com/how-to-be-smarter-about-the-iot-smart-home-experience

Simard, E. (n.d.). Chmod Command Examples. Linux Handbook. https://tinyurl.com/4e5wt837

Storage Tutorials. (n.d.). *What is Storage Virtualization?* Haettu 7.3.2025 osoitteesta: https://www.storagetutorials.com/what-storage-virtualization/

Susnjara, S. & Smalley, I. (9.4.2025). *What is virtualization?* IBM. https://tinyurl.com/mr33njhd

taichikuji. (n.d.). Get a DDNS up and running with DuckDNS – step by step. GitHub Gist. Haettu 16.4.2025 osoitteesta: https://gist.github.com/taichikuji/6f4183c0af1f4a29e345b60910666468

TEPA-termipankki. (n.d.). *Instanssi*. https://termipankki.fi/tepa/fi/haku/instanssi

The Geek Diary. (n.d.). *Htpasswd Command Examples.* Haettu 24.4.2025 osoitteesta: https://www.thegeekdiary.com/htpasswd-command-examples/

TheLinuxCode. (5.11.2024). *Chroot: the magical healing powers of the original Linux virtualization tool.* https://tinyurl.com/7nrz29sw

Traficom. (19.3.2025). *Kyberkestävyyssäädös (Cyber Resilience Act, CRA).* https://tinyurl.com/2xyu8s8x

Valencia, J. (2011). *Using rsync and cron to automate incremental backups*. JVEWeb. https://tinyurl.com/52mj7a3m

Liite 1. Tiedoston docker-compose.yml sisältö

```
services:
  homeassistant:
    image: ghcr.io/home-assistant/home-assistant:stable
    container_name: homeassistant
    #network_mode: host
    ports:
      - 8123:8123

    environment:
      - TZ=${TZ}
    volumes:
      - ./homeassistant/config:/config
    networks:
      - projekti_network
    restart: unless-stopped

  duckdns:
    image: lscr.io/linuxserver/duckdns:latest
    container_name: duckdns
    network_mode: host
    environment:
      - TZ=${TZ}
```

```
    - PUID=${PUID}
    - PGID=${PGID}
    - SUBDOMAINS=${DUCKDNS_SUBDOMAIN}
    - TOKEN=${DUCKDNS_TOKEN}
    - LOG_FILE=false
  volumes:
    - ./duckdns:/config
  networks:
    - projekti_network
  restart: unless-stopped

swag:
  image: lscr.io/linuxserver/swag:latest
  container_name: swag
  cap_add:
    - NET_ADMIN
  environment:
    - TZ=${TZ}
    - PUID=${PUID}
    - PGID=${PGID}
    - URL=${DOMAIN_NAME}
    - SUBDOMAINS=${SUBDOMAINS}
    - VALIDATION=${VALIDATION}
```

```
      - DUCKDNSTOKEN=${DUCKDNS_TOKEN}
      - DNSPLUGIN=${DNSPLUGIN}
      - EMAIL=${LETSENCRYPT_EMAIL}
      - ONLY_SUBDOMAINS=${ONLY_SUBDOMAINS}
    volumes:
      - ./swag:/config
    networks:
      - projekti_network
    ports:
      - 80:80
      - 443:443
    restart: unless-stopped
networks:
  projekti_network:
    name: projekti_network
    driver: bridge
    ipam:
      driver: default
      config:
        - subnet: 172.30.0.0/24
```

Liite 2. Tiedoston .env sisältö

```
# Yleiset asetukset

TZ=Europe/Helsinki

PUID=1000

PGID=1000



# DuckDNS

DUCKDNS_SUBDOMAIN=oppariha

DUCKDNS_TOKEN=abc1234yourtoken



# SWAG

DOMAIN_NAME=oppariha.duckdns.org

LETSENCRYPT_EMAIL=username@example.com

VALIDATION=dns

DNSPLUGIN=duckdns

SUBDOMAINS=

ONLY_SUBDOMAINS=false
```

Liite 3. Tiedoston Homeassistant.subdomain.conf sisältö

```
server {
    listen 443 ssl;
    server_name oppariha.duckdns.org;

    include /config/nginx/ssl.conf;

    location / {
        auth_basic "Restricted";
        auth_basic_user_file /config/nginx/.htpasswd;

        proxy_pass http://homeassistant:8123;
        include /config/nginx/proxy.conf;
    }
}
```

Liite 4. Tiedoston default.conf sisältö

```
## Version 2024/12/17 - Changelog: https://github.com/linuxserver/docker-swag/commits/master/root/defaults/nginx/site-confs/default.conf.sample



# redirect all traffic to https

server {

    listen 80 default_server;

    listen [::]:80 default_server;



    location / {

        return 301 https://$host$request_uri;

    }

}



# enable subdomain method reverse proxy confs

include /config/nginx/proxy-confs/*.subdomain.conf;
```

Liite 5. Tiedoston configuration.yaml sisältö

```
# Loads default set of integrations. Do not remove.
default_config:

# Load frontend themes from the themes folder
frontend:
  themes: !include_dir_merge_named themes

# Added for trust 22.4.
http:
  use_x_forwarded_for: true
  trusted_proxies:
    - 172.30.0.0/24

automation: !include automations.yaml
script: !include scripts.yaml
scene: !include scenes.yaml
```