LAB University of Applied Sciences

Dockerin hyödyntäminen teollisuudenmitta-laitteissa

LAB-ammattikorkeakoulu
Insinööri (AMK), Tieto- ja viestintätekniikka
2022
Veera Koivisto

Tiivistelmä

Tekijä(t)	Julkaisun laji	Valmistumisaika
Koivisto, Veera	Opinnäytetyö, AMK	2022
	Sivumäärä 33	

Työn nimi

Dockerin hyödyntäminen teollisuusmittalaitteissa

Tutkinto ja koulutusala

Insinööri (AMK), tieto- ja viestintätekniikan koulutus

Toimeksiantajan nimi, titteli ja organisaatio

Lisker Oy

Tiivistelmä

Opinnäytetyö tehtiin toimeksiantona Lisker Oy:lle. Lisker Oy on Asikkalassa toimiva sahateollisuuden yritys, joka on erikoistunut mittaus- ja optimointijärjestelmiin.

Työn tavoitteena oli tutkia Docker-konttiteknologian hyödyntämistä sahateollisuuden mittalaitteissa. Tavoitteena oli kehittää kaksi käytännön sovellusta, joiden avulla testattiin Dockerin soveltuvuutta Lisker Oy:n tarpeisiin.

Teoriaosuudessa käsiteltiin yleisesti virtualisointia sekä konttiteknologioita. Teoriaosuudessa tutkittiin myös Oracle VM VirtualBox- ohjelmaan luodun virtuaalikoneen asentamista sekä Dockerin asentamista ja tutkittiin tarkemmin edellä mainittujen ohjelmien käyttöönottoa ja ominaisuuksia.

Tutkimuksessa käytettiin C#- sekä Python -ohjelmointikieliä, Oracle VM VirtualBox -virtuaalikonetta ja Docker -konttiteknologiaa. Näiden teknologioiden lisäksi tutkimuksessa käytettiin USB-webkameraa. Tutkimustulosten perusteella käytiin vielä läpi Dockerin hyötyjä ja haittoja Lisker Oy:n käyttöön.

Asiasanat

docker, virtualisointi, virtuaalikone, Oracle VM VirtualBox, konttiteknologia, dockerfile, docker image, docker compose, python, c#, visual studio code

Abstract

Author(s)	Type of Publication	Published
Koivisto, Veera	Thesis, UAS	2022
	Number of Pages	
	33	

Title of Publication

Utilization of the Docker in industrial measuring instruments

Possible subtitle(s)

Degree and field of study

Bachelor of Engineering, Information and communications Technology

Name, title and organisation of the client

Lisker Oy

Abstract

The work was commissioned to Lisker Oy. Lisker Oy is a sawmill industry company in Asikkala. The company is specializing in measurement and optimization systems.

The aim of the work was to investigate the utilization of Docker container technology in measuring instruments for the sawmill industry. The aim of the work was to develop two applications and search for the benefits of Docker. In the first section, an application was created, which runs through Docker on a computer with no internet connection. The second sections explored how Docker container technology could be used to analyze screenshot cards.

The study used the C# and Python programming languages, the Oracle VM VirtualBox virtual machine, and Docker container technology. In addition to these technologies, the study used a USB webcam. Based on the results of the research, advantages and disadvanges of Docker for Lisker Oy's use were researched.

Keywords

docker, virtualization, virtual machine, Oracle VM VirtualBox, container technology, dockerfile, docker image, docker compose, python, c#, visual studio code

Sisällys

1 Johdanto ... 1
2 Virtualisointi ja konttiteknologia ... 2
2.1 Virtualisointi ... 2
2.2 Konttiteknologia ... 2
2.3 Kontit vs virtualisointi ... 3
3 Oracle VM VirtualBox ... 5
3.1 Yleistä Oracle VM VirtualBox:sta ... 5
3.2 Oracle VM VirtualBox-ohjelman asentaminen ... 5
3.3 Oracle VM VirtualBox -ominaisuudet ... 6
3.4 Virtuaalikoneen luominen ... 7
4 Docker ... 13
4.1 Yleistä Dockerista ... 13
4.1.1 Dockerfile ... 14
4.1.2 Docker Image ... 15
4.1.3 Docker Compose ... 15
4.1.4 Docker registry ... 15
4.1.5 Docker volumes ja bind mounts ... 16
4.2 Docker Desktop ... 17
4.2.1 Asennus Windows-käyttöjärjestelmään ... 17
4.2.2 Docker Desktop -sovelluksen käyttöönotto ... 17
4.3 Kuvankaappauskortit ... 19
5 Dockerin soveltaminen teollisuudessa ... 20
5.1 Case 1 ... 20
5.1.1 Sovelluksen luominen ... 20
5.1.2 Docker Imagen jakaminen yhteydettömään koneeseen ... 24
5.2 Case 2 ... 25
5.2.1 Webkamera-sovellus ... 26
5.2.2 Webkamera-sovelluksen ajaminen Dockerissa ... 27
5.3 Docker-konttiteknologian soveltuvuus Lisker Oy:n käyttöön ... 29
5.3.1 Hyödyt ... 29
5.3.2 Haitat ... 30
6 Yhteenveto ja pohdinta ... 31

Lähteet 32

1 Johdanto

Opinnäytetyön tavoitteena on kamerasovelluksen toimivuuden varmistaminen konttiteknologiassa sekä sovelluksen suorittaminen internetyhteydettömästi kontissa. Opinnäytetyöhön valittuna konttiteknologiana toimii Docker, joka on avoimeen lähdekoodiin perustuva konttien ajoympäristö. Docker on yksi suosituimpia käytössä olevia konttiteknologioita ja soveltuu usealla eri käyttöjärjestelmällä käytettäväksi.

Työn toimeksiantajana toimii sahatavarateollisuuden yritys nimeltä Lisker Oy. Lisker Oy on erikoistunut mittaus- ja optimointijärjestelmiin. Yrityksen toimipiste on Asikkalassa ja Lisker Oy työllistää 8 henkilöä. Yritys toimii kansanvälisesti 16 eri maassa. Toimeksiantajan toiveena on tutkia, olisiko Docker-konttiteknologiasta hyötyä heidän käytössänsä.

Docker-konttiteknologian hyödyntämistä tulee testata kahdella eri toteutuksella. Ensimmäisessä toteutusosiossa luodaan .Net6 -konsoliapplikaatio, jota ajetaan internetyhteydettömän tietokoneen kautta Docker kontissa. Toisessa toteutusosiossa luodaan virtuaalikoneeseen webkamera -sovellus ja selvitetään, kuinka Docker-konttiteknologiaa voitaisiin hyödyntää sahatavaran laadun mittaamisessa käytettävien kuvankaappauskorttien kuvien käsittelyyn.

Tutkimuksen teoriaosuudessa käydään läpi virtualisointia ja konttiteknologioita yleisesti sekä virtualisoinnin ja konttiteknologioiden eroavaisuuksia, jotta saadaan käsitys tutkielmaan liittyvistä tekniikoista. Teoriaosuudessa perehdytään myös Oracle VM VirtualBox -ohjelmaan, virtuaalikoneeseen ja syvennytään tarkemmin Docker -konttiteknologiaan sekä Docker Desktop -ohjelmaan.

2 Virtualisointi ja konttiteknologia

2.1 Virtualisointi

Virtualisointi on tekniikkaa, jonka avulla voidaan luoda useita ympäristöjä tai erillisiä resursseja yhdestä fyysisestä laitteistosta. Virtualisoinnin avulla voidaan siis esimerkiksi ajaa useampia käyttöjärjestelmiä samanaikaisesti ilman fyysisiä asennuksia. (Redhat, b.)

Virtualisoinnin avulla voidaan luoda muun muassa erilaisia ohjelmistopohjaisia IT-palveluita, kuten esimerkiksi virtuaalisia sovelluksia. Virtuaalisien sovelluksien luomisessa käytetään fyysisen laitteiston resursseja. Virtualisoinnin avulla voidaan fyysiseltä laitteistolta jakaa laitteen ominaisuuksia useiden eri ympäristöjen ja käyttäjien välillä. (Redhat, b.)

Virtualisointitapoja on useita erilaisia. Näitä ovat esimerkiksi palvelimien virtualisointi, sovellusten virtualisointi ja työpöydän virtualisointi. Palvelimien virtualisoinnilla mahdollistetaan se, että fyysiselle laitteelle voidaan virtualisoinnin avulla asentaa eri käyttöjärjestelmiä. Palvelimien virtualisointi mahdollistaa siis sen, että ei tarvita useaa eri laitetta, mikäli halutaan käyttää eri käyttöjärjestelmiä. Tämä mahdollistaa myös sen, ettei käyttöjärjestelmiä tarvitse asentaa suoraan fyysiselle laitteelle. (CDW.)

Sovellusten virtualisoinnilla tarkoitetaan sitä, että virtualisoidut sovellukset voidaan jakaa suoraan palvelimelta eri laitteille kuten älypuhelimelle. Tämän myötä käyttäjät voivat käyttää sovellusta suoraan omalta internetyhteydessä olevalta laitteeltaan. (CDW.)

Työpöydän virtualisoinnilla fyysisestä laitteesta erotetaan työpöytäympäristö. Työpöydän virtualisointi mahdollistaa sen, että käyttäjä pääsee fyysisen laitteen tiedostoihin käsiksi miltä tahansa laitteelta, jolloin käyttäjä voi olla missä vain nähdäkseen haluamansa tiedostot. (CDW.)

2.2 Konttiteknologia

Konttiteknologia on menetelmä, jonka avulla voidaan paketoida erilaisia sovelluksia ja ohjelmistoja. Konttien avulla luodaan ympäristö, jossa voidaan kehittää, ylläpitää, siirtää ja testata sovelluksia ja ohjelmistoja. Konttiteknologiassa siis sovellukset tai ohjelmistot pakataan konttiin, jonka kautta sovellus- tai ohjelmistokehitystä toteutetaan. (Docker.)

Perinteisillä sovellus- tai ohjelmistokehitysmenetelmillä sovelluksia ja ohjelmistoja kehitetään tietyssä ympäristössä, joka voi aiheuttaa sen, että sovelluksia tai ohjelmistoja siirrellessä esiintyy virheitä. Konttiteknologiassa puolestaan mahdollistetaan sovellusten ja

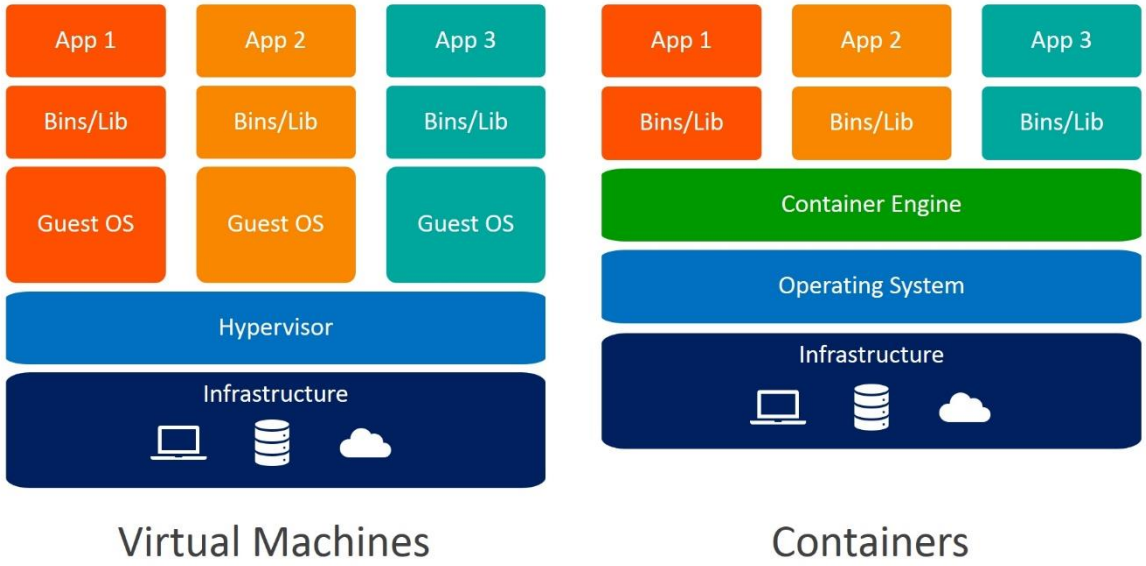
ohjelmistojen hallitseminen missä tahansa ympäristössä eli kontit eivät ole ympäristöriippuvaisia. Kontteja voidaan hallita millä tahansa alustalla. (IBM, a).

Kontit mahdollistavat nopean ja tehokkaan ympäristön sovellusten kehittämiseen (Docker). Kontit ovat kevyitä, helposti siirrettäviä, moderneja sekä kontit parantavat sovellusten ja ohjelmistojen käyttöastetta ympäristöstä riippumatta. (IBM, a).

2.3 Kontit vs virtualisointi

Kontteja verrataan usein virtuaalikoneisiin, sillä molemmat tekniikat sallivat ohjelmistojen ajamisen yhdessä ympäristössä ja mahdollistavat keskusyksikön paremman käyttöasteen. Tänä päivänä kuitenkin konttiteknologia on menossa virtualisoinnin edelle huomattavien etujensa takia. (IBM, a.)

Keskeinen ero konttien ja virtuaalikoneiden välillä on se, että virtuaalikoneet virtualisoivat koko koneen laitteistokerroksiin asti, kun taas kontit virtualisoivat ohjelmistokerrokset vain käyttöjärjestelmän tason yläpuolelle (kuva 1). Virtuaalipalvelimesta poiketen konttiteknologia tarjoaa standardoidun ympäristön. Standardoidun ympäristön avulla ohjelmistoja ja sovelluksia voidaan siirtää sellaisenaan esimerkiksi koneelta pilveen tai konesaliin, eikä ohjelmaa tai sovellusta tarvitse kääntää, muokata tai pakata. (IBM, a.)



Kuva 1. Virtuaalikoneet ja kontit (Weave works)

Konttiteknologian yhtenä isona hyötynä on myös se, että kontit vievät vähemmän tilaa kuin virtuaalikoneet. Tämän myötä konttiteknologiaa käyttäen kontit pystyvät laaja-alaisemmin käsittelemään sovelluksia ja ohjelmistoja, mutta vaativat vähemmän resursseja kuin virtuaalisoidut koneet. Kontit ovat myös virtuaalikoneita resurssitehokkaampia. Kontti käyttää

virtuaalikoneesta poiketen vain sen päälle rakennetun sovelluksen tai ohjelmiston kannalta pakollisia komponentteja. Tämä saattaa tarkoittaa huomattavasti vähemmän käyttöön tarvittavia resursseja, kuin mitä sama sovellus vaatisi virtuaalisella koneella ajettuna. (IBM, a.)

3 Oracle VM VirtualBox

3.1 Yleistä Oracle VM VirtualBox:sta

Oracle VM VirtualBox on ohjelma, joka perustuu avoimeen lähdekoodiin. Ohjelman avulla voidaan virtualisoida eri käyttöjärjestelmiä sekä muita ohjelmia. Oracle VM VirtualBox soveltuu monipuolisesti niin koti- kuin yrityskäyttöön. Oracle VM VirtualBox on yksinkertainen, mutta myös tehokas ohjelma. Oracle VM VirtualBox toimii lähes kaikkialla, sulautetuista järjestelmistä pilviympäristöihin asti. Oracle VM VirtualBoxiin voidaan asentaa virtuaalikoneita ja käyttää virtuaalikoneita niin montaa kuin halutaan. Virtuaalikoneiden ainoana rajana on muisti sekä levytila (VirtualBox, b.)

Oracle VM VirtualBox tukee muun muassa Linux-, Windows- sekä MacOs-käyttöjärjestelmiä. Esimerkkinä Oracle VM VirtualBox -ohjelma voidaan siis asentaa tietokoneeseen, jossa on käytössä Windows-käyttöjärjestelmä ja samaiseen tietokoneeseen voidaan luoda Oracle VM VirtualBox-ohjelman avulla virtuaalikone esimerkiksi Linux-käyttöjärjestelmällä. Fyysisen tietokoneen käyttöjärjestelmä toimii tällöin isäntäkäyttöjärjestelmänä (host OS) ja isäntäkäyttöjärjestelmään luotu virtuaalikone toimii vieras käyttöjärjestelmänä (quest OS). Oracle VM VirtualBoxin avulla voidaan siis käyttää toista käyttöjärjestelmää, ilman, että toinen käyttöjärjestelmä asennetaan fyysisesti käytössä olevalle laitteelle. (VirtualBox, b.)

Oracle VM VirtualBox-ohjelman yksi hyödyllisimpiä ominaisuuksia onkin juuri se, että sen avulla mahdollistetaan useiden eri käyttöjärjestelmien käyttäminen ilman fyysisiä asennuksia. Lisäksi kun käytössä on Oracle VM VirtualBox-ohjelmaan luotu virtuaalikone, vältytään koneen uudelleen käynnistämisiltä ottaessa toisen käyttöjärjestelmän käyttöön. (VirtualBox, b.)

3.2 Oracle VM VirtualBox-ohjelman asentaminen

Ohjelma asennetaan Oracle VM VirtualBox:in virallisilta kotisivuilta. Sivustolta löytyy Oracle VM VirtualBox -ohjelman latauspaketit eri käyttöjärjestelmille. Esimerkkinä Windows -käyttöjärjestelmälle sivustolla on tarjolla Windows hosts -niminen asennuspaketti. (VirtualBox, a.)

Sivustolta löytyy myös laajennuspaketti nimeltä Oracle VM VirtualBox Extensions Pack. Laajennuspaketti mahdollistaa muun muassa USB 2.0 sekä USB 3.0 laitteiden tuen sekä isäntäkoneen webkameran käyttämisen. Laajennuspakettia ladattaessa, tulee varmistaa, että paketin versio on sama, kuin asennetussa Oracle VM Virtualbox -ohjelmassa. (VirtualBox, a.)

Ennen Oracle VM VirtualBox -ohjelman asentamista ja virtuaalikoneen luomista, tulee varmistaa, että käytössä olevalla laitteella on tarvittavat resurssit ohjelman asennusta varten. Esimerkiksi Oracle VM VirtualBox -ohjelman asentamista varten tarvitaan kohtuullisen tehokas x86 -laitteisto. Tehokkaalla laitteistolla varmistetaan se, että ohjelman käyttäminen on sujuvaa. Ohjelman asennusta varten tulee myös varmistaa, että laitteellasi on muistia riittävästi. Riippuen siitä, mitä vieraskäyttöjärjestelmää halutaan käyttää, tarvitaan kuitenkin vähintään 512 Mt vapaata RAM-muistia. Lisäksi on huomioitava, että kiintolevyllä on tilaa riittävästi. Vaikka itse ohjelma ei kiintolevyltä tilaa juurikaan vie, virtuaalikoneet kuitenkin vaativat melko paljon tilaa. (VirtualBox, c.)

3.3 Oracle VM VirtualBox -ominaisuudet

Oracle VM VirtualBox -ohjelmalla on paljon erilaisia ominaisuuksia, joista keskeisimpiä ovat muun muassa siirrettävyys, vieraslisäykset (Guest Aditions), kattava laitteistotuki ja koneen etäkäyttö. Siirrettävyydellä tarkoitetaan sitä, että Oracle VM VirtualBox toimii useilla eri isäntäkäyttöjärjestelmillä, jotka ovat 64-bittisiä. Lisäksi virtuaalikoneita voidaan helposti tuoda ja viedä käyttämällä Open Virtualization Format -standardia. Open Virtualization Format -standardia käyttämällä voidaan virtuaalikoneissa käytössä olevia ohjelmistoja pakata ja jakaa. (VirtualBox, d.)

Vieraslisäykset ovat ohjelmistopaketteja, joita voidaan asentaa virtuaalikoneessa olevaan vieraskäyttöjärjestelmään. Vieraslisäyksien tarkoituksena on auttaa parantamaan esimerkiksi kommunikointia isäntäkäyttöjärjestelmän kanssa tai parantamaan suorituskykyä. Vieraslisäykset tarjoavat jaettuja kansioita, jotka mahdollistavat isäntäkäyttöjärjestelmän tiedostojen käyttämisen vieraskäyttöjärjestelmässä. Vieraslisäyksien avulla mahdollistetaan muun muassa videotarkkuuksien automaattinen säätäminen sekä nopeutettu 3D-grafiikka. (VirtualBox, d.)

Kattavaan laitteistotukeen kuuluu esimerkiksi USB-tuki, moniajo, yhteensopivuus ja täyden ACPI-tuen. USB-tuella mahdollistetaan erilaisten USB-laitteiden käyttäminen myös virtuaalikoneessa. USB-laitteena voi olla esimerkiksi webkamera, jonka käyttö mahdollistetaan kyseisen tuen avulla. (VirtualBox, d.)

Yhteensopivuudella tarkoitetaan sitä, että Oracle VM VirtualBox mahdollistaa virtuaalisten laitteiden laaja-alaisen virtualisoinnin. Oracle VM VirtualBoxin yhteensopivuus mahdollistaa myös levykuvien kopioimisen fyysisistä laitteista. (VirtualBox, d.)

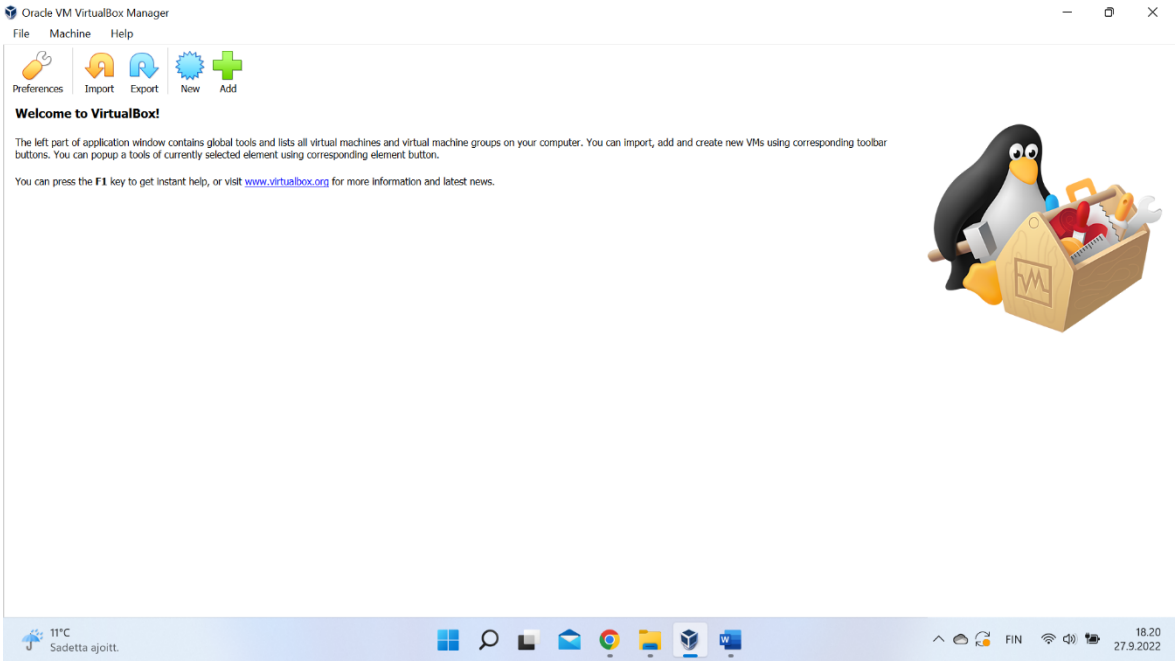
Täydellä ACPI-tuella tarkoitetaan sitä, että Oracle VM VirtualBox tukee täysin ACPI-liitäntää (VirtualBox, d). ACPI on lyhenne Advanced Configuration and Power Interface:sta, jonka

tarkoituksena on hallita esimerkiksi käytössä olevan laitteiston virransäästöä (Techopedia). ACPI-tuen avulla Oracle VM VirtualBox voi lähettää tiedon laitteiston tehon tilasta myös vieraskäyttölaitteille (VirtualBox, d).

VirtualBox Remote Desktop Extension eli lyhennettynä VRDE, mahdollistaa etäkäytön mihin tahansa käynnissä olevaan virtuaalikoneeseen. Laajennus tukee Remote Desktop Protocol:aa (RDP), joka on alun perin rakennettu Microsoft Windowsille. Oracle VM VirtualBox:ssa VRDE liitetään suoraan virtualisointikerrokseen, eikä se ole sisäänrakennettu. Tämä mahdollistaa sen toimivuuden myös muilla käyttöjärjestelmillä kuin Windowsilla. (VirtualBox, d.)

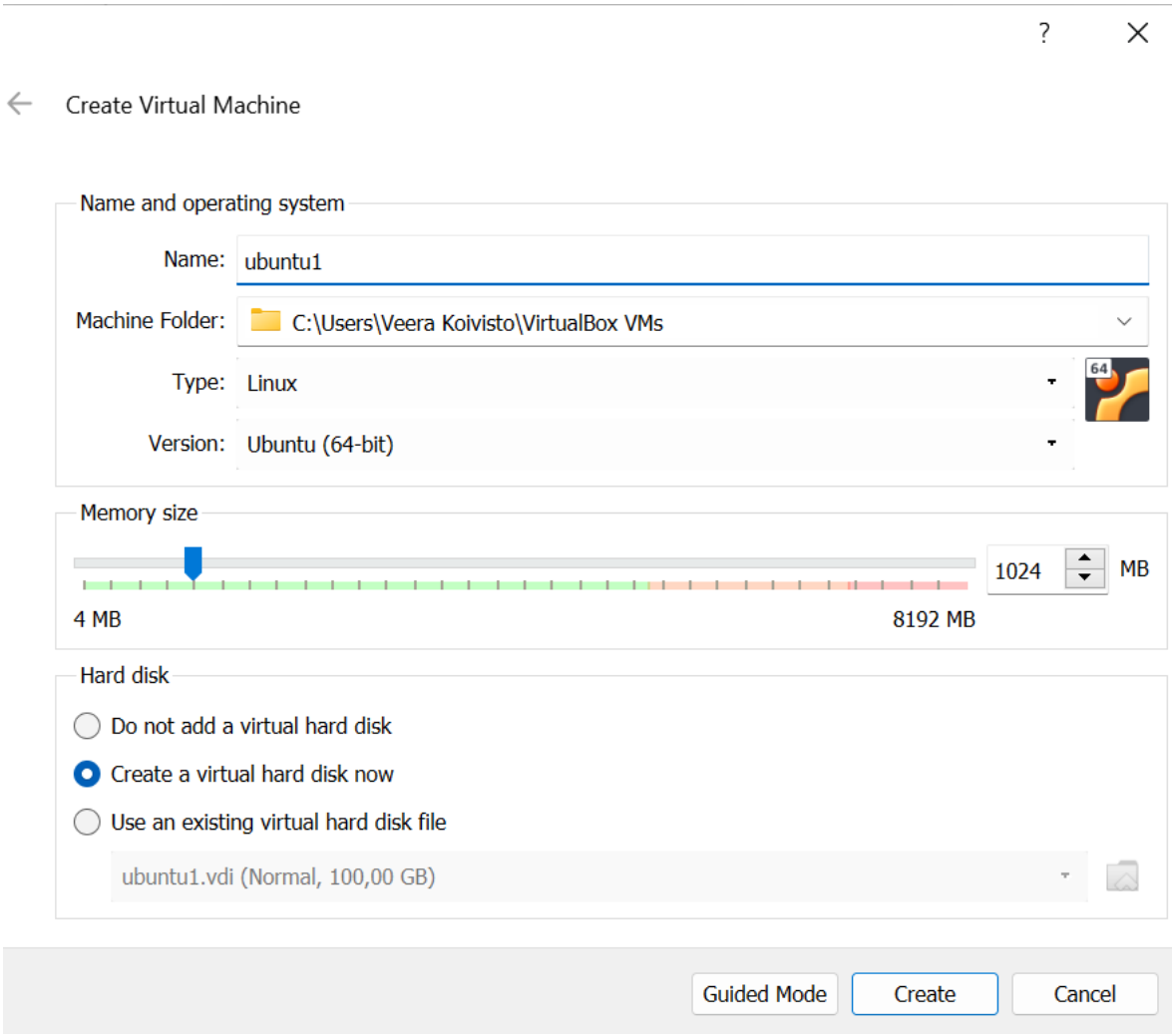
3.4 Virtuaalikoneen luominen

Kuvassa 2 esitellään Oracle VM VirtualBox-ohjelman aloitusnäkymä. Aloitusnäkymältä päästään muun muassa luomaan uusi virtuaalikone tai tarkastelemaan asetuksia.



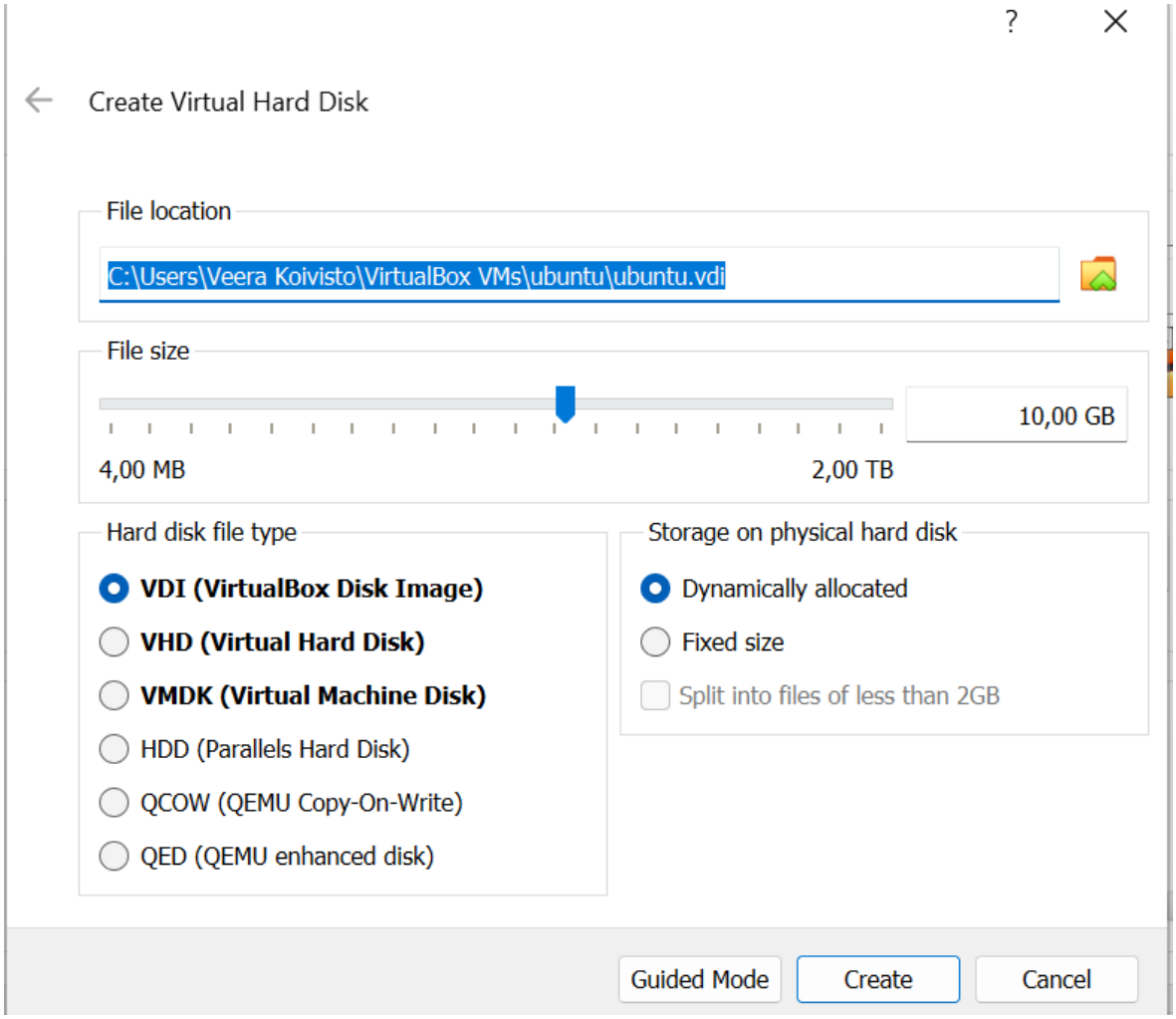
Kuva 2. Oracle VM VirtualBox-ohjelman aloitusnäkymä

Virtuaalikoneen luomisen ensimmäisessä vaiheessa valitaan virtuaalikoneelle nimi sekä käyttöjärjestelmä, jota halutaan käyttää (kuva 3). Asennuksessa määritellään myös se, kuinka paljon keskusmuistia virtuaalikoneelle annetaan sekä kiintolevy, jota virtuaalikoneessa käytetään. Virtuaalikoneelle voidaan luoda joko uusi kiintolevy tai sitten käyttää jo olemassa olevaa.



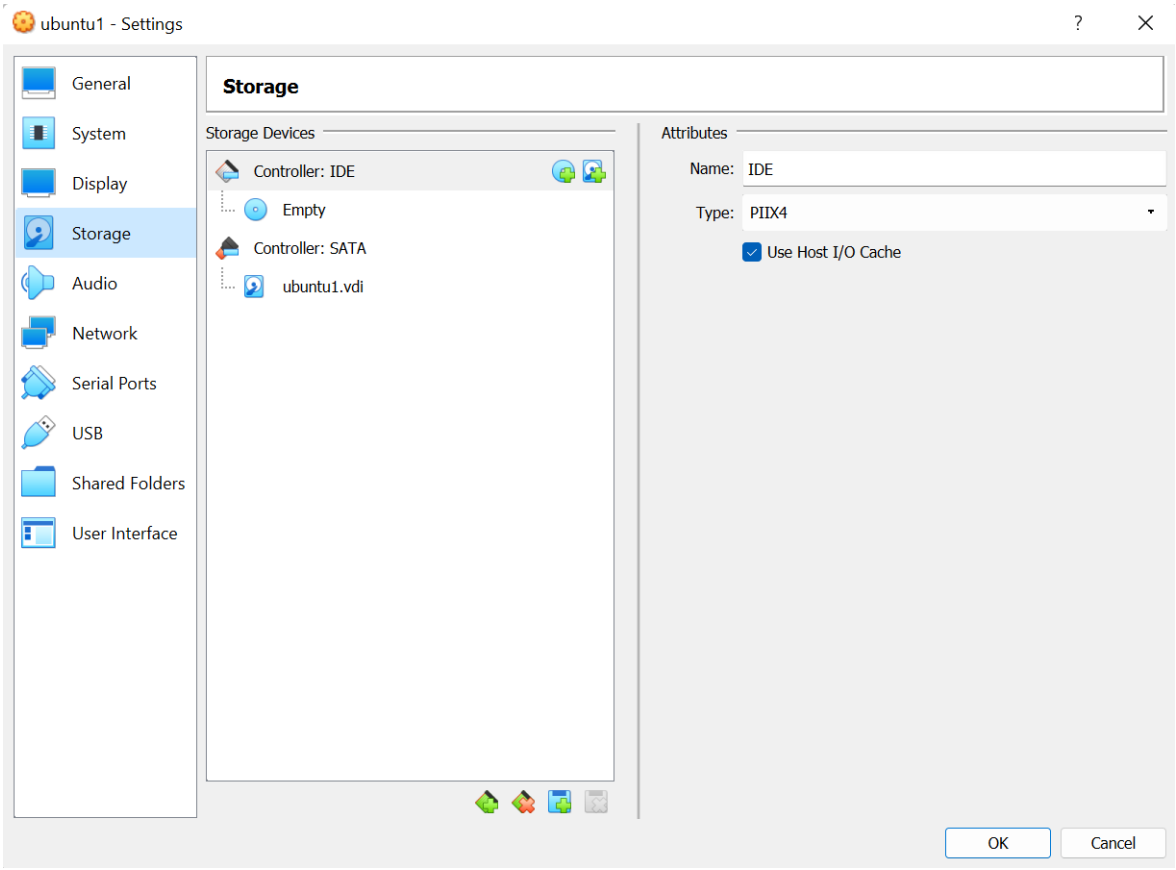
Kuva 3. Virtuaalikoneen luominen

Kuvassa 4 esitellään, kuinka luodaan uusi virtuaalinen kiintolevy. Virtuaaliseen kiintolevyyn määritellään tiedoston sijainti, koko, tyyppi sekä kiintolevyn varastoimistapa. Kun kiintolevy on saatu luotua, voidaan virtuaalikone ottaa käyttöön.



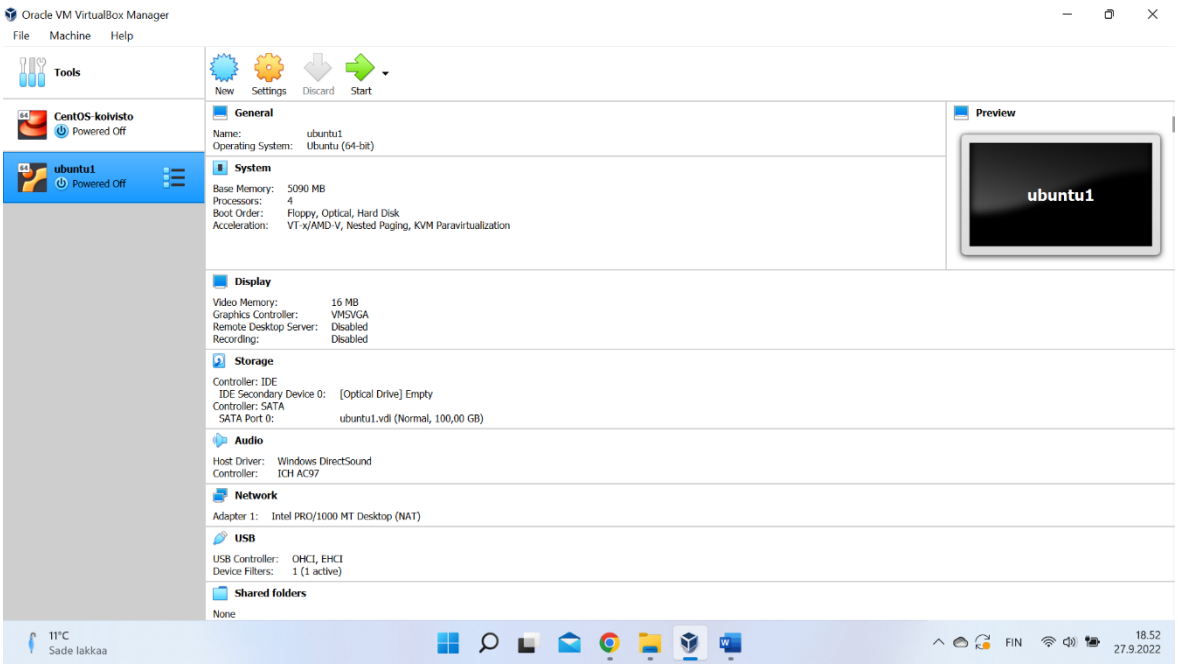
Kuva 4. Virtuaalisen kiintolevyn luominen

Kun virtuaalikone on luotu, tulee vielä virtuaalikoneen asetuksista määritellä asennuslevyn levykuva (kuva 5). Levykuva päästään asettamaan virtuaalikoneen muistiasetuksista.



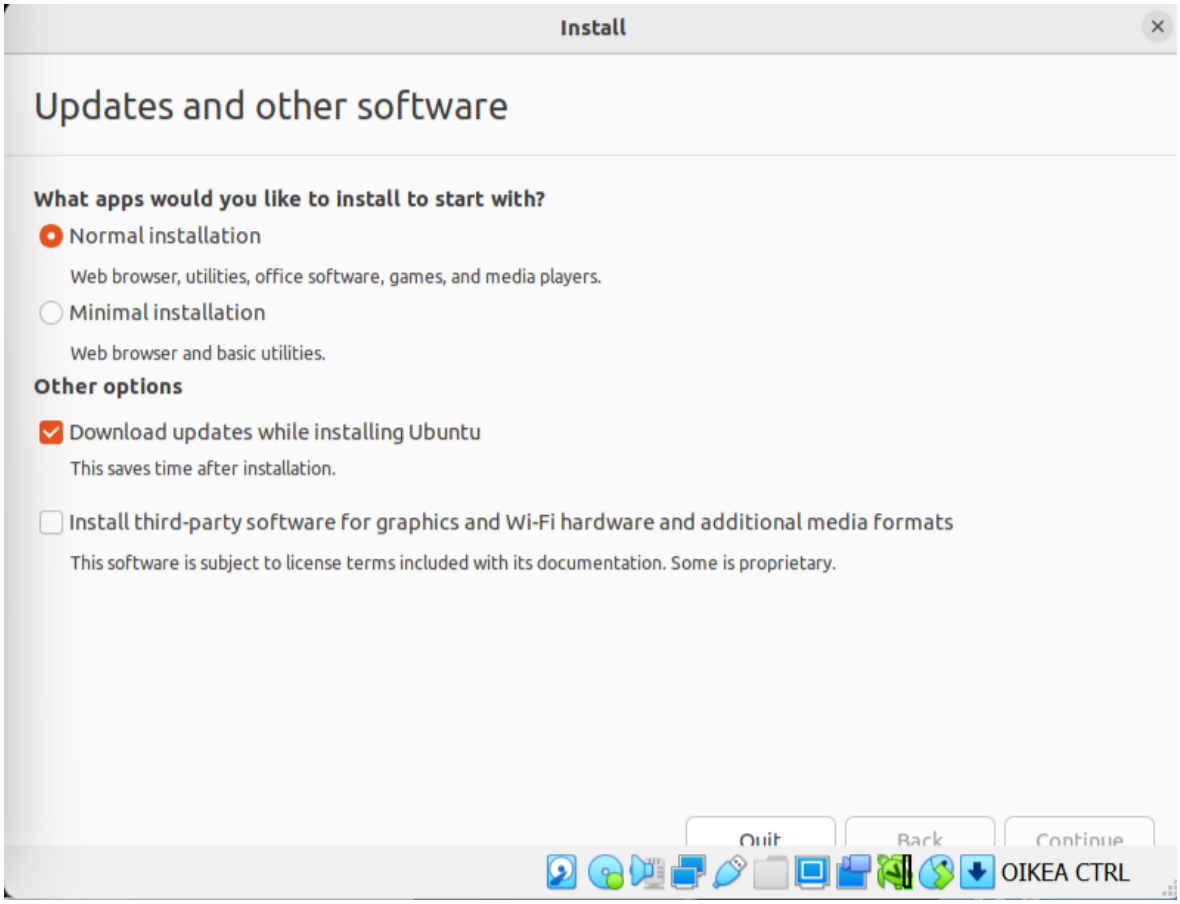
Kuva 5. Asennuslevyn levykuvan lisääminen

Kuvassa 6 esitellään luotu virtuaalikone, johon on määritelty tarvittavat asetukset. Kuvan 6 näkymältä voidaan myös käynnistää luotu virtuaalikone. Kun virtuaalikone käynnistetään, avautuu luotu virtuaalikone uuteen ikkunaan.



Kuva 6. Virtuaalikoneen käynnistysnäkymä

Kun virtuaalikone käynnistetään ensimmäistä kertaa, tulee valita mitä virtuaalikoneeseen halutaan asentaa (kuva 7). Virtuaalikoneelle voidaan valita joko normaali tai minimaalinen asennus. Mikäli valitaan virtuaalikoneen normaali asennus, asennuksen mukana virtuaalikoneeseen lisätään selain, toimisto-ohjelmia, pelejä ja mediasoitin, kun taas minimaalisessa asennuksessa asennetaan virtuaalikoneelle ainoastaan selain ja perushyödykkeitä.



Kuva 7. Virtuaalikoneen asennus

Virtuaalikoneelle määritellään myös kieliasetukset ja minkä kielinen näppäimistö otetaan käyttöön. Lisäksi virtuaalikoneelle määritellään käyttäjänimi ja salasana, joilla pystytään jatkossa kirjautumaan sisään. Kun nämä kaikki on luotu, voidaan käynnistää virtuaalikoneen lopullinen asennus ja aloittaa virtuaalikoneen käyttäminen.

4 Docker

4.1 Yleistä Dockerista

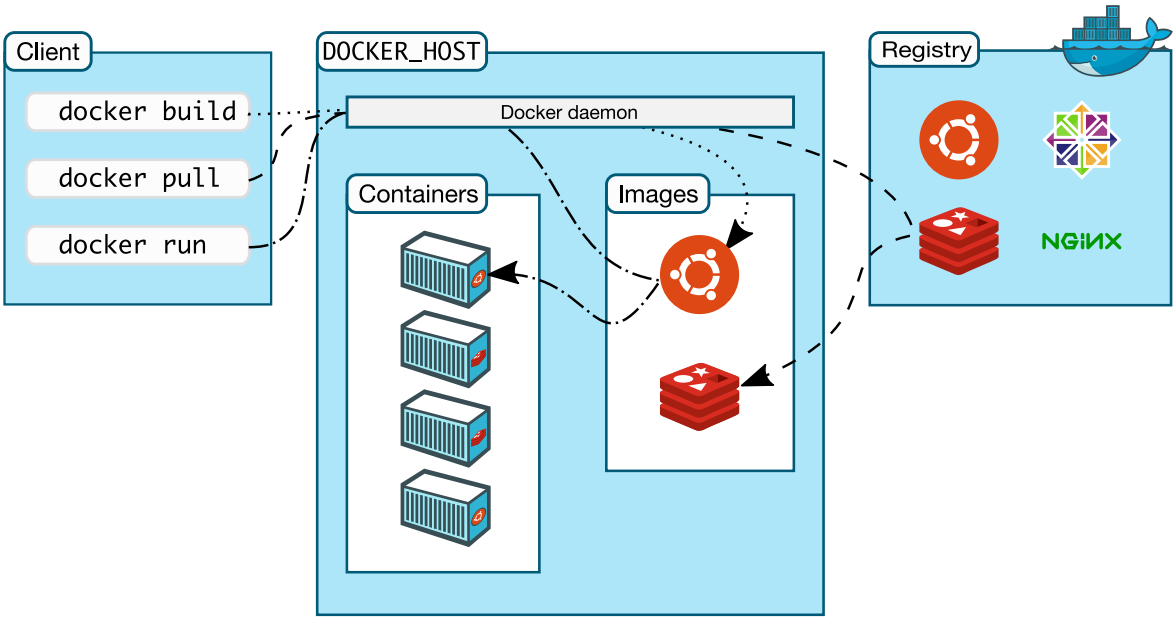
Konttien yleisimpiin ajoympäristöihin kuuluva Docker on saanut alkunsa avoimen lähdekoodin projektina vuonna 2013. Dockerin perustajana toimi yhdysvaltalainen Docker Inc. -niminen yritys. (Techtarget.)

Docker on alusta, jonka avulla voidaan luoda, ottaa käyttöön, poistaa käytöstä sekä hallita erilaisia konttisovelluksia. Dockerin avulla mahdollistetaan konttien siirtäminen ympäristöstä toiseen vaivattomasti ja nopeasti. (IBM, b.)

Docker-kontit eivät tarvitse omaa käyttöjärjestelmää, vaan kontit ovat erillään itse käyttöjärjestelmästä sekä myös muista konteista. Docker-kontit yksinkertaistavat sovellusten toimitusta, ja tämän yksinkertaisuuden myötä konteista on tullut yhä suositumpia (IBM, b.)

Docker-konttien eristyksen ja suojauksen ansiosta voidaan ajaa useita kontteja samanaikaisesti. Docker-kontit ovat kevyitä ja sisältävät kaiken, mitä sovelluksien tai ohjelmistojen suorittamiseen tarvittavan. Konteissa ei tarvitse ottaa huomioon isäntäkoneelle asennettuja tiedostoja tai ohjelmistoja, kun kaikki tarvittava on pakattu kontteihin. (Docker docs, g.)

Kuvassa 8 esitellään Docker-arkkitehtuuria. Dockerissa käytetään asiakas-palvelin-arkkitehtuuria. Docker-client vastaa siis vuorovaikutuksen Dockerin kanssa. Docker-client lähettää erilaisia komentoja Dockeriin ja tämän jälkeen Docker suorittaa clientin antamat komennot. Docker-daemon puolestaan kuuntelee Docker -sovelluksen pyyntöjä ja hallitsee objekteja, kuten kuvia, kontteja ja verkkoja. Docker-arkkitehtuurissa Docker-client kuuntelee Docker-daemonia niin kuin kuvassa 8 nähdään. (Docker docs, g.)



Kuva 8. Docker arkkitehtuuri (Docker docs, g)

4.1.1 Dockerfile

Dockerfile on tekstitiedosto, jonka avulla Docker Image rakennetaan. Dockerfile pitää sisällään komentokehoitteita, joita käyttäjä voi komentoriviltä kutsua docker imagen luomiseksi. Dockerfile -tekstitiedoston peruskomentoja ovat muun muassa FROM, RUN, CMD ja COPY. (Docker docs, b.)

Dockerfile aloitetaan aina FROM-komennolla. FROM-komennolla asetetaan kontin peruskuva. FROM-komennon peruskuva voi olla mikä tahansa kelvollinen kuva, joka voidaan hakea myös julkisesta arkistosta. (Docker docs, b.)

RUN-komennolla voidaan suorittaa mitä tahansa komentoja, kun Docker Imagea rakennetaan. RUN-komennolla voidaan myös asentaa tarvittavia paketteja kuvan rakentamista varten. (Docker docs, b.) RUN-komennolla asennettuja paketteja ovat esimerkiksi Python-opencv ja Ffmpeg. Asennuspaketti python3-opencv mahdollistaa OpenCV -moduulin käyttämisen myös Dockerin kontissa (Developer Toradex). Kun taas ffmpeg-asennuspaketti mahdollistaa esimerkiksi äänien ja videoiden toimimisen Docker kontissa (FFmpeg).

CMD-komennolla määrätään oletusarvot Docker-kontille. Oletusarvot voivat sisältää esimerkiksi suoritettavan tiedoston. Dockerfile voi sisältää vain yhden CMD-komennon, mikäli CMD-komentoja on useampia, tiedostossa viimeinen CMD astuu voimaan ja loput jätetään huomioimatta. (Docker docs, b.)

COPY-komennolla kopioidaan uusia hakemistoja ja tiedostoja. Kopioidut hakemistot tai tiedostot lisätään kontin tiedostojärjestelmään. Komennot voivat sisältää myös yleismerkkejä polkujen määrittelemiseksi. (Docker docs, b.)

4.1.2 Docker Image

Docker Image on tiedosto, jota käytetään Docker konttien luomiseen. Docker Image toimii niin sanotusti mallina Docker-konttien luomisessa. Docker Image sisältää ohjeita kontin luomiseen ja on ainoastaan luettavassa muodossa (Docker docs, g).

Docker Image koostuu kokoelmasta tiedostoja, jotka kasaavat yhteen kaiken tarvittavan, kuten esimerkiksi erilaiset asennukset ja sovelluskoodit. Docker Imageja voidaan luoda joko itse tai sitten voidaan käyttää rekisteriin julkaistuja kuvia. (Docker docs, g.)

Docker Imagea varten luodaan Dockerfile -niminen tiedosto, johon määritellään vaiheet Docker Imagen luomista ja suorittamista varten. Vaihtoehtoisesti Docker Image voidaan myös luoda suoraan kontista, jolloin käytetään commit -komentoa. (Docker docs, g.)

4.1.3 Docker Compose

Docker Compose on työkalu, jonka avulla luodaan ja hallinnoidaan erilaisia palvelukokonaisuuksia, jotka muodostuvat useista konteista. Docker Compose käyttää palvelujen määrittelemiseksi YAML-tiedostoa, jonka avulla voidaan yhden komennon taktiikalla ajaa sovelluksia ja yhtä lailla myös lopettaa sovellukset (Docker docs, a.)

Yksi Docker Composen isoimpia etuja on se, että käyttämällä Docker Composea voidaan kasata sovelluksia yhteen tiedostoon. Tämä tarkoittaa sitä, että koko projekti pysyy yhden tiedoston sisällä, jonka avulla on myös helppo antaa muiden osallistua projektiin. (Docker docs, a.)

4.1.4 Docker registry

Docker-rekisteri on Apache -lisenssin alla oleva avoimen lähdekoodin rekisteri. Docker-rekisteri on palvelinpuolen sovellus, jonka avulla voidaan tallentaa ja jakaa Docker Imageja. Rekisteriin tallennetaan ja jaetaan Docker Image aina tietyillä nimillä. Docker Imagesta voi olla useita eri versioita, mutta jokaisella kuvalla on oltava omat tunnistetiedot. (Docker docs, f.)

Docker-rekisterejä on olemassa sekä julkisia että yksityisiä. Yksi suosituimmista yksityisistä rekistereistä on Docker Hub, joka on Dockerin tarjoama palvelu. Docker-rekisteri voidaan

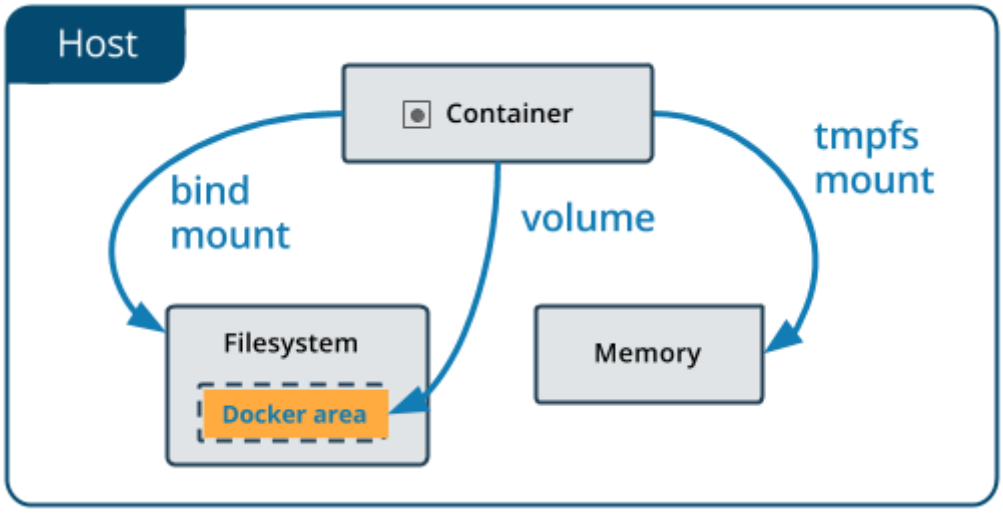
siis asentaa joko omaan palvelinsaliin tai vaihtoehtoisesti voidaan käyttää edellä mainittua Docker Hubia. (Docker docs, j.)

Docker-rekisteriä käytetään, kun halutaan turvallisesti hallita sitä, mihin Docker Image tallennetaan. Dockerin käyttäjät voivat hakea Docker Imagea rekisteristä ja siirtää uusia kuvia rekisteriin. (Geeks for Geeks.)

4.1.5 Docker volumes ja bind mounts

Oletusarvoisesti Docker-kontissa luodut tiedostot tallennetaan konttikerrokseen. Tämä tarkoittaa sitä, että konttiin tuodut tiedot eivät säily, kun kyseistä konttia ei enää ole. Tällaisessa tilanteessa kontin sisällön saaminen on vaikeaa, mikäli sitä tarvittaisiin. Dockerilla on kaksi vaihtoehtoa, volyymit ja liittämiset eli bind mounts, joiden avulla voidaan tallentaa tiedostoja isäntäkoneeseen. Näiden vaihtoehtojen avulla mahdollistetaan tiedostojen säilyvyys myös kontin hävittämisen yhteydessä. (Docker docs, h.)

Docker volyymit tallennetaan isäntätiedostojärjestelmän osaan, jota Docker hallinnoi (kuva 9), kun taas liittämiset (bind mounts) voidaan tallentaa mihin tahansa isäntäjärjestelmässä (kuva 9). Molemmissa tapauksissa vain Dockerin prosessit voivat muokata tiedostojen sisältöjä. (Docker docs, h.)



Kuva 9. Volumes ja bind mount (Docker docs, h)

Docker volyymeilla on monia etuja verrattuna liittämisiin (bind mounts). Docker volyymit ovat esimerkiksi helpompia varmuuskopioida ja siirtää kuin liittämiset (bind mounts). Volyymeja voidaan hallita Docker API:lla sekä Docker CLI -komennoilla. Volyymien käyttö on myös mahdollista niin Linux- kuin Windows-konteissa. (Docker docs, h.)

4.2 Docker Desktop

Docker Desktop -sovellus voidaan asentaa laitteisiin, joissa on käytössä Linux-, Mac- tai Windows-käyttöjärjestelmä. Docker Desktop -sovelluksen avulla voidaan nopeasti ja turvallisesti hallita kontteja, kuvia ja sovelluksia suoraan laitteelta. (Docker Docs, c.)

Docker Desktop -sovellusta voidaan hyödyntää erilaisten kehitystyökalujen sekä ohjelmointikielien kanssa. Sovellus mahdollistaa myös pääsyn Docker Hubiin, joka on Dockerin oma laaja kirjasto Docker Imageille. (Docker Docs, c.)

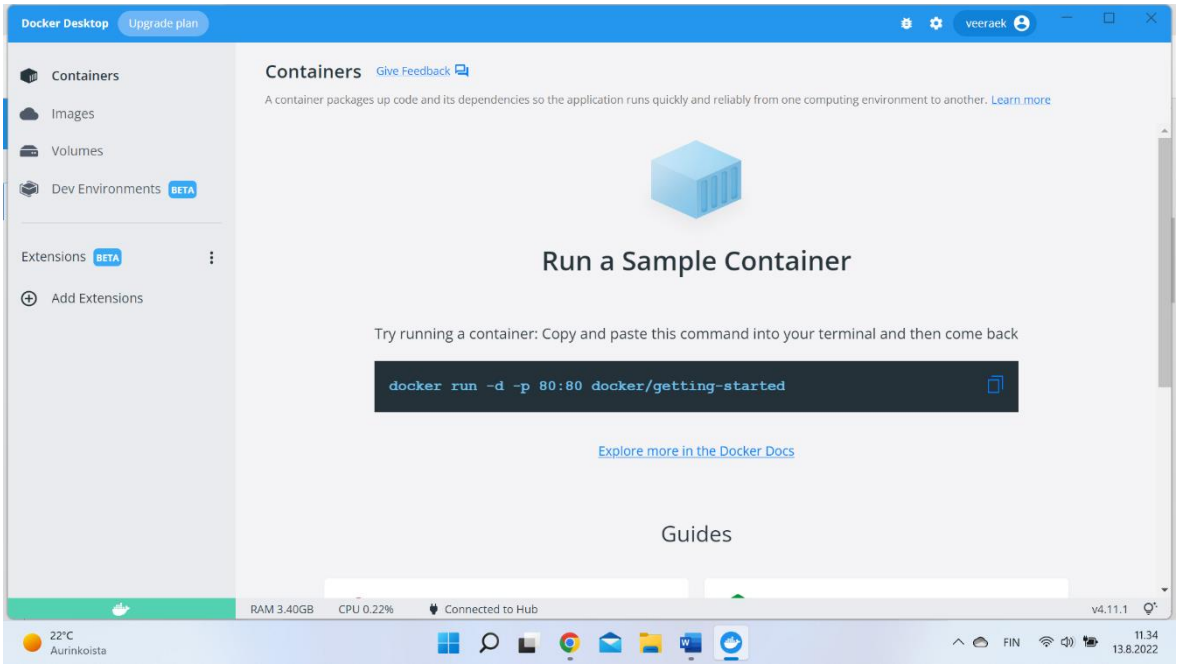
4.2.1 Asennus Windows-käyttöjärjestelmään

Docker Desktop-sovelluksen asentaminen vaatii tiettyjen vaatimuksien täyttymistä, jotta sovellus saadaan asennettua. Mikäli käytössä on Windows 10 tai Windows 11-käyttöjärjestelmä, vaaditaan, että Windows on 64-bittinen ja RAM-muistia on vähintään 4 GB. Lisäksi BIOS-asetuksista tulee varmistaa, että virtualisointituki on käytössä. (Docker docs, d.)

Docker Desktop -sovellus voidaan ladata interaktiivisesti Dockerin viralliselta sivustolta tai sitten se voidaan ladata käyttäen komentokehotetta. Mikäli käytetään komentokehotetta, tulee kuitenkin ensiksi ladata Docker Desktop Installer.exe -ohjelma käytössä olevalle laitteelle. (Docker docs, d.)

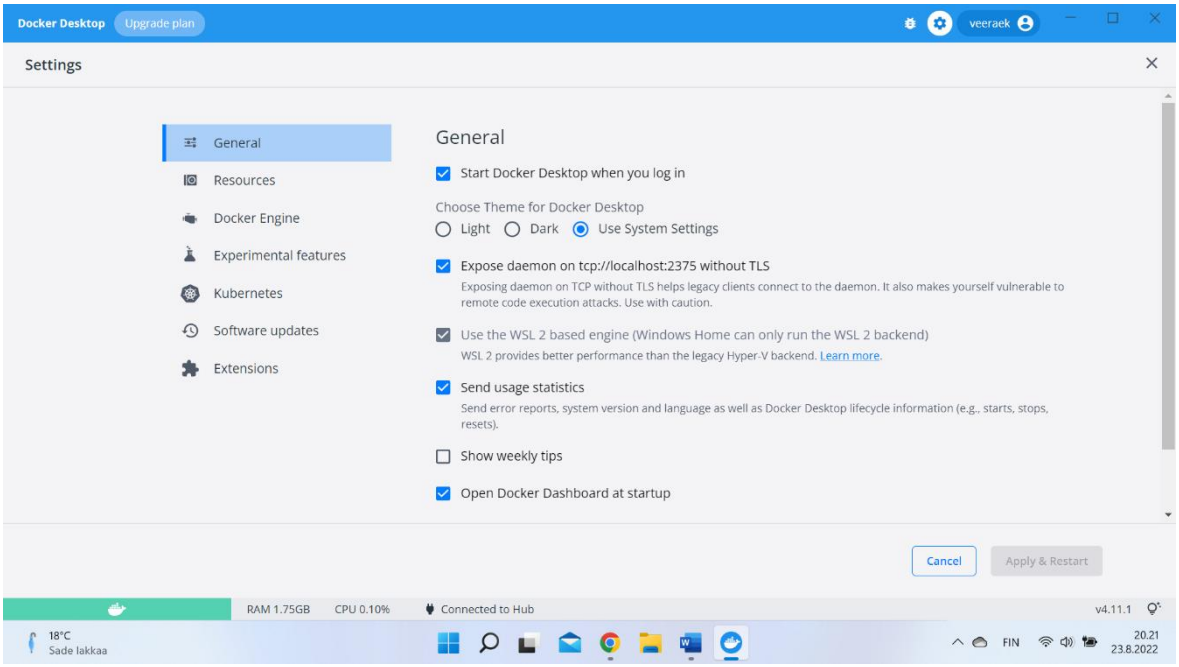
4.2.2 Docker Desktop -sovelluksen käyttöönotto

Kuvassa 10 esitellään Docker Desktop -sovelluksen aloitusnäkymä. Sovelluksen aloitusnäkymältä päästään muun muassa muuttamaan sovelluksen asetuksia sekä nähdään käyttöönotetut tai käytössä olevat kontit, kuvat ja volumet. Aloitusnäkymältä voidaan myös lisätä sovellukselle lisäosia. (Docker docs, d.)



Kuva 10 Docker Desktop -sovelluksen aloitusnäkymä

Kuvassa 11 esitellään Docker Desktop -sovelluksen asetukset -välilehti, josta päästään hallitsemaan sovelluksen käyttöön vaikuttavia asetuksia. Asetuksista voidaan esimerkiksi hallita sovelluksen päivityksiä ja lisäosia. Koska käytössä on WSL 2 eli Windows Subsystem for Linux, hallitsee Windows suoraan resurssiasetuksia ja asettaa resurssit suoraan oletusarvoihin. Tällöin Docker Desktop -sovelluksen kautta resurssiasetuksia ei voida muuttaa.



Kuva 11. Docker Desktop -sovelluksen asetusvälilehti

4.3 Kuvankaappauskortit

Lisker Oy käyttää kuvankaappauskortteja osana tukkien ja sahatavaran mittausjärjestelmiä. Kuvankaappauskortteja käytetään kuvainformaation lukemiseen teollisuuskameroilta. Tiedon perusteella sahatavaraa kuvannetaan ja luokitellaan laadun ja koon perusteella, (Toukola, 2022.)

Kuvankaappauskortteja on olemassa sekä sisäisiä että ulkoisia. Sisäiset kuvankaappauskortit asennetaan suoraan käytössä olevaan laitteeseen, kun taas ulkoiset kuvankaappauskortit liitetään USB:n kautta. Kuvankaappauskorttien yksi isoimpia hyötyjä on se, että niiden avulla voidaan tallentaa ja toistaa kuvamateriaalia suurella nopeudella teräväpiirtoisesti, joka mahdollistaa reaaliaikaisen ja tarkan mittauksen. (Make Use Of.)

5 Dockerin soveltaminen teollisuudessa

5.1 Case 1

Case 1 -toteutusosiossa luodaan .Net 6 -konsoliapplikaatio. Konsoliapplikaation luomisessa käytetään C# -ohjelmointikieltä. Työhön luotu konsoliapplikaatio lukee käytössä olevan laitteen vapaan muistin sekä kokonaismuistin ja tulostaa tiedot tavuina.

Luotu konsoliapplikaatio viedään Docker -konttiin ja testataan konsoliapplikaation toimivuus Docker -kontissa. Kun konsoliapplikaatio saadaan toimimaan internetyhteyttä käyttävällä laitteella, siirretään konsoliapplikaatio internetyhteydettömään laitteeseen. Siirron jälkeen testataan sovelluksen ajaminen ilman internetyhteyttä Docker -kontissa.

5.1.1 Sovelluksen luominen

Sovelluksen luomisessa käytetään Visual Studio Code -tekstieditoria. Visual Studio Code on avoimeen lähdekoodiin perustuva tekstieditori, jota käytetään laaja-alaisesti esimerkiksi ohjelmoinnissa (Visual Studio.)

.Net -konsoliapplikaatio luodaan kuvassa 12 esitetyllä komennolla. Dotnet new -komento luo uuden projektin, joka sisältää tiedostot sekä tiedostokansiot, joihin konsoliapplikaatiota lähdetään rakentamaan. Komennolla luotu sovellus tuo projektiin muun muassa Program.cs -tiedoston. Kun sovellus luodaan dotnet new -komennolla, Program.cs -tiedostossa on aluksi ”Hello World” -ohjelma. Kyseistä program.cs -tiedostoa lähdetään muokkaamaan niin, että saadaan ohjelma tulostamaan halutut tiedot.

```
$ dotnet new app1
```

Kuva 12. .Net -konsoliapplikaation luominen

Tässä konsoliapplikaatiossa haluttiin lukea tietokoneen vapaana olevan muistin määrä sekä tietokoneen muistin kokonaismäärä. Sovellukseen määritellään metodi, joka esitellään kuvassa 13. Metodin avulla voidaan lukea halutulta asemalta erilaisia tietoja. (Learn Microsoft, b.) Sovellukseen määritellään asema, josta halutaan, että sovellus lukee muistin tiedot.

```
DriveInfo[] allDrives = DriveInfo.GetDrives();
```

Kuva 13. Metodi aseman tietojen lukua varten

Konsoliapplikaation toimivuutta testataan dotnet run -komennolla, joka esitellään kuvassa 14. Dotnet run -komento tulostaa konsoliapplikaation Program.cs -tiedostoon luodun

tulosteen eli käytössä olevan laitteen vapaan tilan ja kokonaistilan tavuina C -asemalta. (Learn Microsoft, a.)

```
Veera Koivisto@LAPTOP-5P6HOA4P MINGW64 ~/Desktop/App1
$ dotnet run
Asema C:\
  Vapaa tila:       46645207040 tavua.
  Kokonaistila:     254721126400 tavua.
```

Kuva 14. Sovelluksen testaaminen

Kun ohjelma tulostaa halutun tulosteen, voidaan luoda Dockerfile. Dockerfile -tiedostoon määritellään tarvittavat komennot Docker Imagea varten. Dockerfile voidaan lisätä joko komennolla ”touch docker” tai sitten manuaalisesti lisäämällä haluttuun tiedostokansioon Dockerfile- niminen tiedosto.

Kuvassa 15 esitellään luotu Dockerfile. FROM-komentoon määriteltynä sovelluksen peruskuvana käytetään .Net 6 -sovellukselle tarkoitettua kuvaa, jota voidaan käyttää niin Linuxilla kuin Windowsilla. Peruskuvan lisäksi lisätään WORKDIR -komento, johon määritellään hakemisto, jossa työskennellään. Dockerfile -tiedostoon määritellään myös COPY -komennoilla projektitiedostot kopioitavaksi sekä lopuksi lisätään komento, jolla sovellus käynnistetään. Sovelluksen käynnistäminen määritellään ENTRYPOINT -komentoon.

```
Dockerfile > ...
1   FROM mcr.microsoft.com/dotnet/sdk:6.0 AS build-env
2   WORKDIR /App1
3
4   # Kopioidaan csproj
5   COPY *.csproj ./
6   RUN dotnet restore
7
8   # Kopioidaan kaikki muut
9   COPY . ./
10  RUN dotnet publish -c Release -o out
11
12  FROM mcr.microsoft.com/dotnet/aspnet:6.0
13  WORKDIR /App1
14  COPY --from=build-env /App1/out .
15  ENTRYPOINT ["dotnet", "App1.dll"]
```

Kuva 15. Konsoliapplikaation Dockerfile

Dockerfile -tiedoston luomisen jälkeen voidaan luoda Docker Image, jonka luomiseen käytetty komento esitellään kuvassa 16. Docker Image luodaan dockerfile-tiedostosta. Docker Imagen luomisen komennolla määritellään Docker Imagen nimi sekä ID ja asetetaan polku Dockerfile -tiedostoon. (Learn Microsoft, a.)

```
$ docker build -t new:1 .
[+] Building 0.9s (15/15) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 32B
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load metadata for mcr.microsoft.com/dotnet/sdk:6.0
 => [internal] load metadata for mcr.microsoft.com/dotnet/aspnet:6.0
 => [build-env 1/6] FROM mcr.microsoft.com/dotnet/sdk:6.0@sha256:ce977e0ce71ce4aeecde3917f3abf0dcffcf952e9ca138704b63e1a838b4700c
 => [stage-1 1/3] FROM mcr.microsoft.com/dotnet/aspnet:6.0@sha256:af97531c4f335ce9a039659b3ce4436a558494257d5520f8e515cc1ad5ad4eee
```

Kuva 16. Docker Imagen luominen

Docker Imagen luomisen jälkeen, voidaan tarkistaa käytettävissä olevat Docker Imaget. Kuvassa 17 esitellään komento, joka listaa kaikki Docker Imaget. Listauksessa nähdään jokaisen Docker Imagen tagi, id, ajankohta, milloin image on luotu sekä Docker Imagen koko. (Learn Microsoft, a.)

```
Veera Koivisto@LAPTOP-5P6HOA4P MINGW64 ~/Desktop/App1
$ docker images
REPOSITORY          TAG       IMAGE ID       CREATED          SIZE
new                 1         c3a34f0f47c2   4 minutes ago    208MB
```

Kuva 17. Docker Image listaaminen

Kun Dockerfile ja Docker Image on saatu luotua, voidaan luoda Docker-kontti. Docker-kontti luodaan kuvassa 18 näkyvällä komennolla. (Learn Microsoft, a.) Komennolla -d tarkoittaa sitä, että konttia ajetaan taustalla. Komennolla -p määrittää sen, että sovitetaan ulkoinen portti kontin sisäiseen porttiin. Lopuksi vielä –name avulla määritellään Docker-kontille nimi ja tämän jälkeen määritellään Docker Image, jota kontissa käytetään. (Softchris.)

```
Veera Koivisto@LAPTOP-5P6HOA4P MINGW64 ~/Desktop/App1
$ docker run -d -p 8080:80 --name myapp new:1
3bdefc8e61a38e4ecd32e5c88a4a64a2cbb63d3f2da240e07f80c26ebbfcdde3
```

Kuva 18. Kontin luominen

Käytössä olevat Docker-kontit voidaan tarkistaa kuvassa 19 esitetyllä komennolla. Docker-konttien listauksesta nähdään Docker-kontin ID, kuva, komennot, milloin kontti on luotu, kontin tila, portit sekä kontin nimi. (Learn Microsoft, a.)

```
Veera Koivisto@LAPTOP-5P6HOA4P MINGW64 ~/Desktop/App1
$ docker ps -a
CONTAINER ID   IMAGE     COMMAND             CREATED         STATUS                     PORTS     NAMES
3bdefc8e61a3   new:1     "dotnet App1.dll"   7 minutes ago   Exited (0) 7 minutes ago             myapp
```

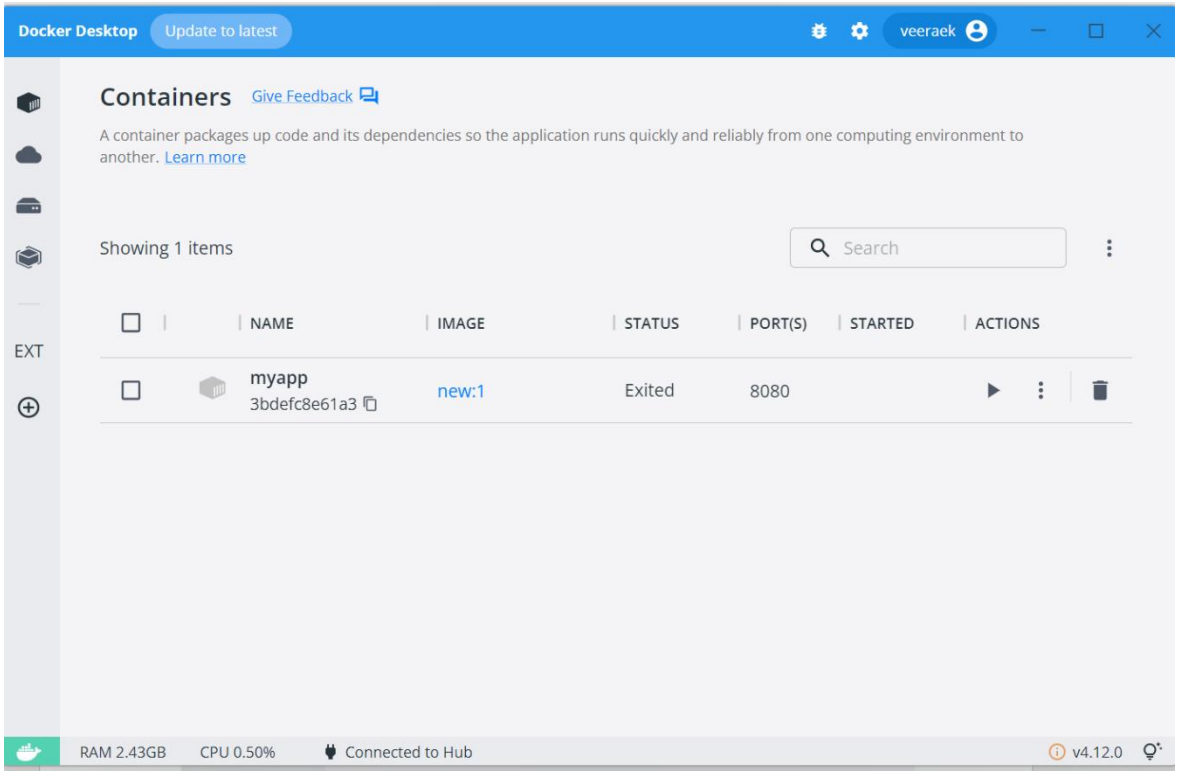
Kuva 19. Docker-konttien listaaminen

Docker-kontti voidaan käynnistää kuvassa 20 näkyvällä komennolla. Komentoon määritellään sen Docker-kontin nimi, joka halutaan käynnistää. (Learn Microsoft, a.)

```
Veera Koivisto@LAPTOP-5P6HOA4P MINGW64 ~/Desktop/App1
$ docker start myapp
myapp
```

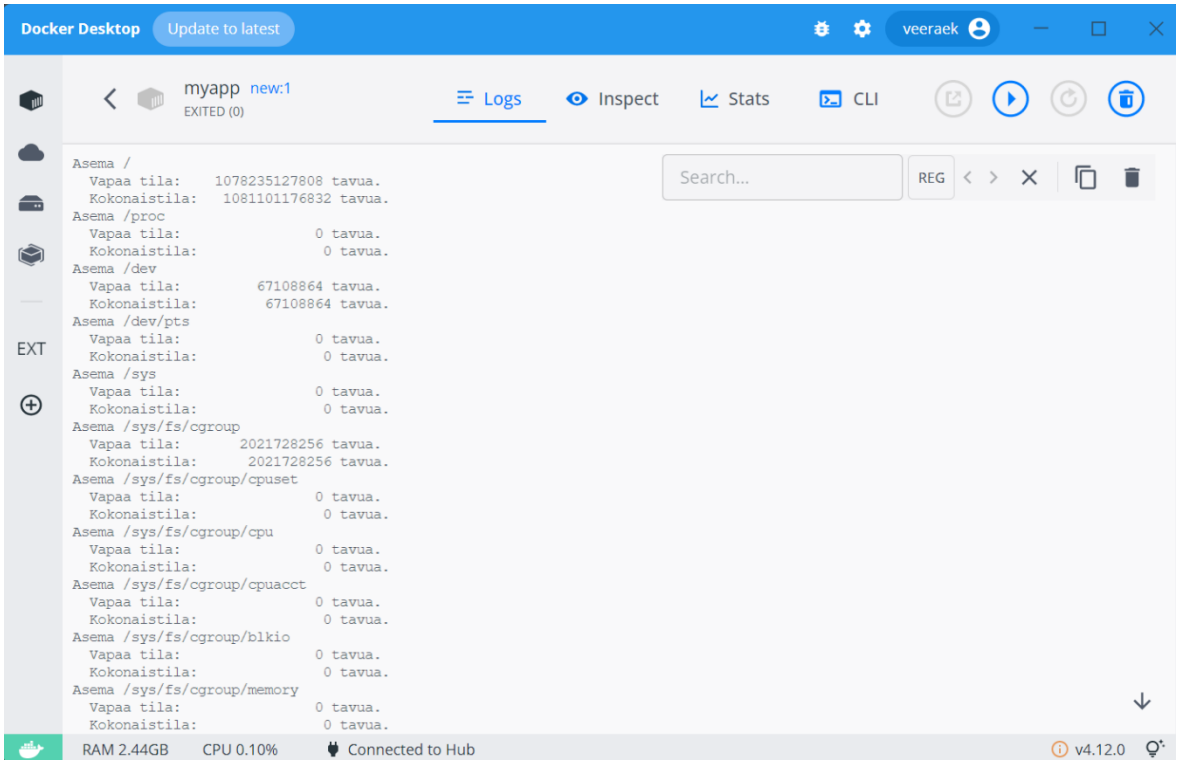
Kuva 20. Docker -kontin käynnistäminen

Kun Docker-kontti on käynnistetty, voidaan avata Docker Desktop -sovellus, josta nähdään luotu kontti ja kontin sisältö. Kuvassa 21 esitellään aikaisemmin luotu Docker-kontti nimeltä myapp. Docker-kontti käyttää Docker Imagea nimeltä new-image, jonka ID:nä on 1. Lisäksi nähdään Docker kontin status sekä portti, joka on käytössä.



Kuva 21. Docker-kontit Docker Desktop -sovelluksessa

Kuvassa 22 nähdään, että sovellus toimii Docker -kontissa ja tulostaa sovellukseen luodun tulosteen. Näkymän yläreunasta nähdään, että käytössä on myapp -niminen kontti, joka käyttää new -nimistä Docker Imagea tagilla 1.



Kuva 22. Sovelluksen tuloste Docker -kontissa

5.1.2 Docker Imagen jakaminen yhteydettömään koneeseen

Ensimmäisessä osiossa luotu App1 -konsoliapplikaatio jaetaan internetyhteydettömään koneeseen, sillä tarkoitus on, että sovellusta voidaan ajaa Dockerin kautta myös ilman internetyhteyttä. Konsoliapplikaation jakamiseen tarvitaan aikaisemmin luotua Docker imagea, jonka avulla konsoliapplikaatio voidaan siirtää yhteydettömään koneeseen ja tämän jälkeen yhdistää Docker-konttiin.

Docker Imagen jakamiseen voidaan käyttää muun muassa scp tai fcp tiedonsiirto-ohjelmia. Tai sitten voidaan käyttää esimerkiksi muistitikkua, jonka avulla Docker Image siirretään laitteesta toiseen. Tässä osioissa käytetään usb muistitikkua.

Aluksi tallennetaan haluttu Docker Image tar-muotoon. Kuvassa 23 esitellään komento, jossa Docker Image nimeltä new muutetaan tar-muotoon.

```
Veera Koivisto@LAPTOP-5P6HOA4P MINGW64 ~/Desktop/App1
$ docker save new:1 > new.tar
```

Kuva 23. Komento Docker Imagen tallentamiseksi tar tiedostoksi

Kun Docker Image nimeltä new on tallennettu .tar -tiedostoksi, voidaan siirtää kyseinen tiedosto internetyhteydettömään laitteeseen. Tiedostonsiirron jälkeen tulee tar -tiedosto purkaa, jotta tiedoston sisällä oleva Docker Image voidaan ajaa ja saada tulostamaan konsoliapplikaatioon luotu komento yhteydettömässä laitteessa. Kuvassa 24 nähdään docker load komento, jonka avulla saadaan tar -tiedosto purettua.

```
$ docker load < new.tar
Loaded image: new:1
```

Kuva 24. Tar -tiedoston purku

Kun tar-tiedosto, joka pitää sisällään konsoliapplikaation Docker Imagen on saatu purettua, testataan sovelluksen toimivuus internetyhteydettömässä laitteessa. Kuvassa 25 näkyvällä komennolla testataan new -kuvan toimivuutta ja niin kuin nähdään, tulosteena tulee teksti, joka sovellukseen on luotu.

```
$ docker run new
Asema /
  Vapaa tila:    1078234984448 tavua.
  Kokonaistila:   1081101176832 tavua.
Asema /proc
  Vapaa tila:                0 tavua.
  Kokonaistila:               0 tavua.
Asema /dev
  Vapaa tila:         67108864 tavua.
  Kokonaistila:        67108864 tavua.
```

Kuva 25. Sovelluksen toimivuuden testaaminen internetyhteydettömässä laitteessa

Docker Imagen julkaisu yhteydettömään koneeseen olisi mahdollista myös esimerkiksi Docker rekisterin avulla, mutta tällöinkin tarvitaan aina kuvaa päivittäessä yhteys internettiin, jotta päivitykset astuvat voimaan.

5.2 Case 2

Lisker Oy käyttää kuvankaappauskortteja sahatavaran laadun mittaamiseen. Tässä toiminnallisessa osiossa tutkitaan, onko mahdollista hyödyntää Docker-kontteja kuvankaappauskorttien kuvan lukuun.

Sovellus luodaan käyttäen Oracle VM VirtualBox -ohjelmaa. Oracle VM VirtualBox:iin luodaan Ubuntu-käyttöjärjestelmää käyttävä virtuaalikone. Virtuaalikoneeseen luodaan webkamera -sovellus, jonka toimivuutta testataan Docker-kontissa.

Sovelluksessa käytetään kuvankaappauskorttien sijaan USB-webkameraa, sillä mikäli USB-webkameran kuvan luku onnistuu Docker-kontissa, tulisi kuvankaappauskorttienkin lukemisenkin olla mahdollista Dockerissa. Valmis sovellus viedään Docker-konttiin ja testataan kuvan näkyminen kontissa.

5.2.1 Webkamera-sovellus

Webkamera-sovelluksen luomisessa käytetään Python-ohjelmointikieltä ja sovellus luodaan Visual Studio Code-tekstieditorilla. Sovelluksessa käytetään OpenCV-moduulia, joka mahdollistaa USB-webkameran kuvan näkymisen Python-sovelluksessa.

Kuvassa 26 esitellään komento, jolla asennetaan OpenCV -asennuspaketti sovelluksen luomista varten. Asennuspakettien jälkeen virtuaalikoneessa voidaan ottaa käyttöön OpenCV -moduuli, jota tarvitaan webkameran kuvan lukemiseen.

```
veeraek@veeraek-VirtualBox:~$ sudo apt-get install python3-opencv
```

Kuva 26. OpenCV-moduulin asennus virtuaalikoneeseen

Moduulien asentamisen jälkeen voidaan aloittaa sovelluksen luominen. Sovelluksen tiedostokansioon lisätään py -päätteinen tiedosto, johon sisällytetään Python-koodi. Sovellukseen tuodaan aluksi OpenCV -moduuli ja määritellään kamera, jota sovelluksessa käytetään (kuva 27).

```
import cv2

cam = cv2.VideoCapture(cv2.CAP_V4L2)
```

Kuva 27. OpenCV-moduulin importtaus ja kameravalinta

Kameravalinnan jälkeen määritellään mitä sovellukselta halutaan eli tässä tapauksessa määritellään webkamerakuvan avaus ja webkamerakuvan sulkeminen. Kameraruudulle voidaan myös määritellä esimerkiksi minkä kokoisessa tai muotoisessa ikkunassa kamerakuva aukeaa.

5.2.2 Webkamera-sovelluksen ajaminen Dockerissa

Jotta sovellusta voidaan testata Docker-kontissa, tulee ensiksi Docker asentaa virtuaalikoneeseen. Asennus tapahtuu terminaalissa komentokehotteella ”sudo apt install docker.io”. Terminaaliin kirjoittamalla komentokehotteen ”sudo docker –version” varmistetaan, että Dockerin asentaminen on onnistunut ja Docker voidaan ottaa käyttöön.

Webkamera-sovelluksen kontittaminen aloitetaan rakentamalla Dockerfile ja siihen tarvittavat komennot. Kuvassa 31 esitellään luotu Dockerfile. Kuvassa 28 näkyvän Dockerfile-tiedoston ensimmäisellä rivillä, FROM-komennolla määritellään, että käytetään pohjakuvana ubuntu:latest -kuvaa. FROM-komennon jälkeen asennetaan tarvittavia paketteja, jotta webkamera-sovellus saadaan toimimaan myös Docker -kontissa. Dockerfile-tiedostoon lisätään muun muassa python3 ja opencv -pakettien asennukset (Developer Toradex). Tiedostoon lisätään myös ffmpeg-asennuspaketti äänen ja videon lähettämistä varten (FFmpeg).

```
Dockerfile ●
Dockerfile > ...
1   FROM ubuntu:latest
2   RUN apt-get update
3   RUN apt-get update && apt-get install -y --no-install-recommends python3-pip
4   RUN apt-get update && apt-get install -y python3-opencv
5   RUN apt-get install ffmpeg libsm6 libxext -y
6   RUN apt-get install -y libxcb-util-dev
7
8   WORKDIR /webcam
9   COPY webcam.py .
10
11  CMD ["python3", "webcam.py"]
```

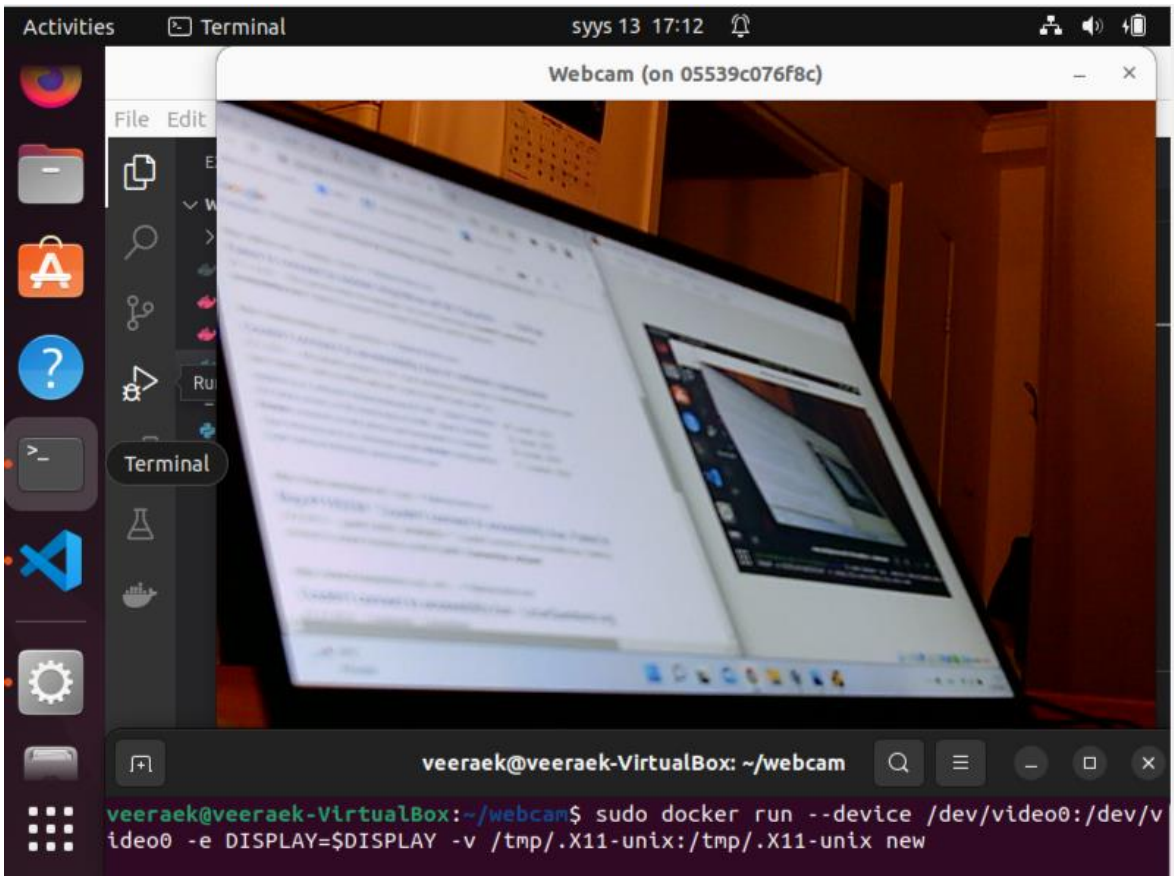
Kuva 28. Webkamera -sovellukseen luotu Dockerfile

Dockerfile -tiedoston luomisen jälkeen luodaan Docker Image, joka mahdollistaa sovelluksen ajamisen Docker -kontissa. Docker Imagen luominen esitetään kuvassa 29 näkyvällä komennolla, jossa ”new” on Docker Imagen nimi.

```
veeraek@veeraek-VirtualBox:~/webcam$ sudo docker build -t new .
Sending build context to Docker daemon   5.12kB
Step 1/10 : FROM ubuntu:latest
 ---> 2dc39ba059dc
Step 2/10 : RUN apt-get update
 ---> Using cache
 ---> df0e9b791e36
Step 3/10 : RUN apt-get update && apt-get install -y --no-install-recommends py
thon3-pip
 ---> Using cache
 ---> 65b50cc8e954
Step 4/10 : RUN apt-get update && apt-get install -y python3-opencv
 ---> Using cache
 ---> 0744c9ec5ed1
Step 5/10 : RUN apt-get install ffmpeg libsm6 libxext6 -y
 ---> Using cache
 ---> feeabe2a48ea
Step 6/10 : RUN apt-get install -y libxcb-util-dev
 ---> Using cache
 ---> c453245d0196
Step 7/10 : ENV QT_DEBUG_PLUGINS=1
 ---> Using cache
 ---> b2a32e210df0
Step 8/10 : WORKDIR /webcam
 ---> Using cache
 ---> afc15e19b1f5
Step 9/10 : COPY webcam.py .
 ---> efe01830dac4
Step 10/10 : CMD ["python3","webcam.py"]
```

Kuva 29. Docker Imagen luominen

Kun Docker Image on saatu onnistuneesti luotua, voidaan testata sovelluksen toimivuutta Docker -kontissa. Kuvassa 30 esitellään webkamera-sovelluksen käynnistyskomento, jolla webkamera-sovellus käynnistettiin Docker -kontissa. Sovellus käynnistyy uuteen ikkunaan, jossa nähdään webkameran kuva. Ikkunassa nähdään myös webkamera -sovelluksen nimi sekä Docker-kontti, jossa sovellus pyörii.



Kuva 30. Kamerasovelluksen testaaminen Docker-kontissa

5.3 Docker-konttiteknologian soveltuvuus Lisker Oy:n käyttöön

Viimeisessä osiossa selvitetään Docker-konttiteknologian soveltuvuutta toimeksiantajan tarpeisiin. Docker-konttiteknologian soveltuvuutta käydään läpi mahdollisien hyötyjen ja haittojen kautta.

Docker-konttiteknologiasta löytyy useista eri lähteistä tietoja hyödyistä sekä haitoista. Tässä tapauksessa hyötyjen ja haittojen selvittämisen perusteena käytetään kuitenkin Case 1 ja Case 2 -toteutuksia, joiden perusteella hyödyllisyyttä tai haitallisuutta tutkitaan.

5.3.1 Hyödyt

Case 2 toteutusosiossa selvitetään USB-webkameran kuvan lukemisen onnistuneisuutta Dockerissa. Mikäli toimeksiantaja ottaisi Docker konttiteknologian käyttöön, tulisi USB-laitteiden lukemisen olla mahdollista.

Toteutusosiota tehdessä huomattiin, että USB-webkameran kuvan ajaminen onnistuu Docker kontissa. Webkamera-sovellusta olisi mahdollista päivittää niin, että siihen lisätään webkameran kuvan tarkkailua varten ominaisuuksia, joita toimeksiantaja tällä hetkellä

käyttää laadunvalvonnassaan. Toiminnallisen osion perusteella voitaisiin päätellä, että Docker -konttiteknologian käyttöönottaminen olisi toimivuutensa puolesta mahdollinen ratkaisu kuvan tarkkailua ja laadunmittaamista varten.

5.3.2 Haitat

Haittapuoliksi voisi luokitella sen, että uutena käyttäjänä Dockeriin perehtyminen vie aikaa. Käyttäjän tulee perehtyä konttien toiminnallisuuksiin, jotta osataan hyödyntää konttiteknologiaa parhaalla mahdollisella tavalla.

Case 1 toteutusosioissa selvitetään Docker konttiteknologian hyödyntämistä internetyhteydettömällä laitteella. Internetyhteydettömällä laitteella Docker konttiteknologian hyödyntäminen on mahdollista, mutta kuitenkin Docker Imagen luomiseen tarvitaan internetyhteyttä.

Tämä tarkoittaa sitä, että aina sovellusta päivittäessä tai Docker Imagea päivittäessä, laite tulisi yhdistää internettiin, jotta päivitykset saadaan käyttöön myös internetyhteydettömällä laitteella. Eli täysin internetyhteydettömästi Docker konttiteknologian käyttäminen ei ole mahdollista, joten tämä olisi toisena huonona puolena toimeksiantajan tarpeita ajatellen.

6 Yhteenveto ja pohdinta

Tutkimuksen tavoitteena oli tutkia Docker -konttiteknologian hyödyntämistä Lisker Oy:n teollisuudenmittalaitteistoissa. Lisker Oy:lla mitataan sahatavaran laatua, jonka mittaamisen avuksi olisi suunnitteilla ottaa Docker -konttiteknologia käyttöön, mikäli Docker -konttiteknologia koetaan hyödylliseksi.

Docker-konttiteknologian hyödyntämistä Lisker Oy:n tarpeisiin testattiin kahdella eri toteutusosiolla. Ensimmäisessä toteutusosiossa luotiin C# -ohjelmointikielellä konsoliapplikaatio, joka siirrettiin internetyhteydettömään laitteeseen. Internetyhteydettömällä laitteella testattiin sovelluksen toimivuutta Docker-kontissa. Konsoliapplikaatio saatiin onnistuneesti luotua ja toimimaan myös internetyhteydettömällä laitteella.

Toisessa toiminnallisuusosiossa luotiin Oracle VM VirtualBox -ohjelmaan virtuaalikone, jossa käytettiin Ubuntu-käyttöjärjestelmää. Virtuaalikoneeseen luotiin Python-ohjelmointikieleltä käyttäen webkamera-sovellus, joka luki webkameran kuvaa. Webkamera-sovellus pakattiin Docker -konttiin ja testattiin webkamera-sovelluksen toimivuutta kontissa. Käytössä olevana webkameran toimi USB-webkamera, jotta voitiin varmistaa USB-laitteiden toimivuus Docker -kontissa. Webkamera-sovelluksessa saatiin webkameran kuva avautumaan halutusti uuteen ikkunaan ja varmistettua sovelluksen toimivuus myös Docker -kontissa.

Docker -konttiteknologian käyttöönottaminen olisi mahdollinen ratkaisu Lisker Oy:n tuotannossa. Tutkinnan myötä kuvankaappauskorttien kuvan lukemiseen Docker -kontteknologian hyödyntäminen olisi mahdollista. Internetyhteydettömän laitteen kanssa puolestaan Docker -konttiteknologian käyttäminen ei välttämättä ole kovin hyödyllistä, sillä täysin internetyhteydettömästi konttiteknologiaa ei voitaisi hyödyntää. Jatkokehityksenä voisi webkamera-sovelluksen toiminnallisuuteen päivittää lisää ominaisuuksia laadunmittaamista varten, tällä hetkellä, kun sovellus ainoastaan avaa kamerakuvan.

Lähteet

CDW. What is virtualization. Viitattu 20.04.2022. Saatavissa https://www.cdw.com/content/cdw/en/articles/datacenter/what-is-virtualization.html

Code. Visual Studio. Viitattu 15.08.2022. Saatavissa https://code.visualstudio.com/

Developer Toradex. Torizon sample using opencv for computer vision. Viitattu 25.09.2022. Saatavissa https://developer.toradex.com/torizon/how-to/machine-learning/torizon-sample-using-opencv-for-computer-vision/

Docker. What is a container? Viitattu 01.04.2022. Saatavissa https://www.docker.com/resources/what-container/

Docker docs, a. Using compose. Viitattu 20.04.2022. Saatavissa https://docs.docker.com/get-started/08_using_compose/

Docker docs, b. Dockerfile reference. Viitattu 21.04.2022. Saatavissa https://docs.docker.com/engine/reference/builder/

Docker docs, c. Desktop. Viitattu 10.07.2022. Saatavissa https://docs.docker.com/desktop/

Docker docs, d. Windows install. Viitattu 10.07.2022. Saatavissa https://docs.docker.com/desktop/install/windows-install/

Docker docs, f. Registry. Viitattu 10.07.2022. Saatavissa https://docs.docker.com/registry/

Docker docs, g. Overview. Viitattu 13.08.2022. Saatavissa https://docs.docker.com/get-started/overview/

Docker docs, h. Volumes. Viitattu 15.10. 2022. Saatavissa https://docs.docker.com/storage/volumes/

Docker docs, i. Container networking. Viitattu 15.10.2022 Saatavissa https://docs.docker.com/config/containers/container-networking/

Docker docs, j. Docker Hub. Viitattu 20.10.2022. Saatavissa https://docs.docker.com/docker-hub/

Ffmpeg. FFmpeg. Viitattu 01.10.2022. Saatavissa https://ffmpeg.org/

Geeks for geeks. What is Docker registry? 11. Mar, 2022. Viitattu 10.07.2022. Saatavissa https://www.geeksforgeeks.org/what-is-docker-registry/

IBM, a. Containers. Viitattu 01.04.2022. Saatavissa https://www.ibm.com/cloud/learn/containers

IBM, b. Docker. Viitattu 01.04.2022. Saatavissa https://www.ibm.com/cloud/learn/docker

Learn Microsoft, a. Containerize a .NET app. Viitattu 01.10.2022. Saatavissa https://learn.microsoft.com/en-us/dotnet/core/docker/build-container?tabs=windows

Learn Microsoft, b. DriveInfo.GetDrives Method. Viitattu 25.10.2022. Saatavissa https://learn.microsoft.com/en-us/dotnet/api/system.io.driveinfo.getdrives?view=net-6.0

Make Use Of. What is a capture card and how does it workd? Viitattu 26.07.2022. Saatavissa https://www.makeuseof.com/what-is-a-capture-card-how-does-it-work/

Techopedia. Advanced Configuration and Power Interface (ACPI). Viitattu 25.10.2022. Saatavissa https://www.techopedia.com/definition/4962/advanced-configuration-and-power-interface-acpi

Techtarget 2015. Application containerization. Viitattu 24.04.2022. Saatavissa https://www.techtarget.com/searchitoperations/definition/application-containerization-app-containerization

Toukola, 2022. Lisker Oy. Haastattelu 08.04.2022.

Redhat, a. What is virtualization? Viitattu 20.04.2022. Saatavissa https://www.redhat.com/en/topics/virtualization/what-is-virtualization#overview

Redhat, b. What is YAML? Viitattu 20.04.2022. Saatavissa https://www.redhat.com/en/topics/automation/what-is-yaml

Softchris. How You can Dockerize a .Net Core app. Viitattu 24.10.2022. Saatavissa https://softchris.github.io/pages/dotnet-dockerize.html#build-our-image-start-container

VirtualBox, a. Download VirtualBox. Viitattu 02.09.2022. Saatavissa https://www.virtualbox.org/wiki/Downloads

VirtualBox, b. VirtualBox. Viitattu 02.09.2022. Saatavissa https://www.virtualbox.org/

VirtualBox, c. End-user documentation. Viitattu 25.10.2022. Saatavissa https://www.virtualbox.org/wiki/End-user_documentation

VirtualBox, d. User Manual. Viitattu 25.10.2022. Saatavissa https://www.virtualbox.org/manual/UserManual.html#virtintro

Weave Works. A Pactical guide to choosing between Docker containers and vms. Viitattu 15.10.2022. Saatavissa https://www.weave.works/blog/a-practical-guide-to-choosing-between-docker-containers-and-vms