Opinnäytetyö (AMK)

Tieto- ja viestintätekniikka

2022

Noora Kuorikoski

Lääkevarmennuksen toteutus kannettavaan keräilylaitteeseen

TURKU AMK
TURKU UNIVERSITY OF APPLIED SCIENCES

Opinnäytetyö (AMK) | Tiivistelmä

Turun ammattikorkeakoulu

Tieto- ja viestintätekniikka

2022 | 44 sivua

Noora Kuorikoski

Lääkevarmennuksen toteutus kannettavaan keräilylaitteeseen

Lääkelogistiikassa lääkevarmennus tarkoittaa sitä, että lääkkeen aitous varmennetaan tarkistamalla lääkepakkauksen turvaominaisuudet. Lääkevarmennus perustuu lääkeväärennösdirektiiviin 2011/62 (EU) ja sitä tarkentavaan komission delegoituun asetukseen 2016/161(EU). Lainsäädännön vaatimusten mukaisesti Marela-toiminnanohjausjärjestelmän kannettava keräilylaitesovellus sisältää toiminnallisuuden tavaran vastaanotossa tehtävälle lääkevarmennukselle.

Työn tavoitteena oli toteuttaa kannettavan keräilylaitteen sovellukseen tavaran toimitettaessa tehtävä lääkevarmennus. Tämän lisäksi sovellukseen toteutettiin 2D-koodin tietoja hyödyntäviä ominaisuuksia. Työn toteutukseen käytettiin JSF-teknologiaa ja sen kirjastoja, joilla keräilylaitesovellus on rakennettu. Lisäksi pohdittiin sovelluksen arkkitehtuuria ja jatkokehitysmahdollisuuksia.

Työn tuloksena keräilylaitesovelluksella voidaan suorittaa lääkevarmennus ja saada selkeät ilmoitukset lääkevarmennuksen toiminnasta. Sovellukseen lisätyt 2D-koodia hyödyntävät toiminnot toimivat vaatimusten mukaisesti.

Asiasanat:

lääkevarmennus, keräilylaite, ohjelmistokehitys, JSF, Java, MVC

Bachelor’s Thesis | Abstract

Turku University of Applied Sciences

Bachelor of Engineering, Information and Communications Technology

2022 | 44 pages

Noora Kuorikoski

Implementation of medicine verification in a mobile collection device

In medicine logistics, medicine verification means that the authenticity of medicine is verified by checking the safety features on the package. The verification of medicinal products is based on the Falsified Medicines Directive 2011/62 (EU) and the Commission Delegated Regulation 2016/161 (EU) specifying it. Under legal requirements, the Marela ERP system's mobile collection device application includes functionality for medicine verification upon reception of goods.

The work aimed to implement the medicine verification for the application of a mobile collection device upon delivery of the goods. In addition, features utilizing information in 2D code were implemented in the application.

The work was carried out using JSF technology and its libraries, on which the collection device application was built. In addition, the application architecture and further development possibilities were discussed. As a result of the work, the collection device application can be used to perform medicine verification and receive clear notifications about the operation of medicine verification. The added 2D code-enabled features of the application work as required.

Keywords:

medicine verification, collection device, application development, JSF, Java, MVC

Sisältö

Käytetyt lyhenteet

API	Application Programming Interface, ohjelmointirajapinta
EL	Expression Language, ohjelmointikieli
EMVO	European Medicines Verification Organisation
EMVS	European Medicine Verification System
ERP	Enterprise Resource Planning, organisaation ohjaamiseen käytettävä tietojärjestelmä
EU Hub	Eurooppalainen keskustietokanta
Fimea	Lääkealan turvallisuus- ja kehittämiskeskus
FiMVO	Suomen Lääkevarmennus Oy, yksi kansallisista lääkevarmennusorganisaatioista
HTTP	HyperText Transfer Protocol, protokolla tiedonsiirtoon
IDE	Integrated Development Environment, ohjelmointiympäristö
Java	Sun Microsystemsin luoma ohjelmointikieli
JPA	Java Persistence API, API olio-relaatiomallinnukseen
JSF	JavaServer Faces, nykyisin Jakarta Server Faces, standardi palvelinpuolen käyttöliittymien rakentamiseen
JSP	JavaServer Pages, kirjasto käyttöliittymäkomponenttien rakentamiseen
JVM	Java Virtual Machine, virtuaalikone
MVC	Model-View-Controller, sovellusarkkitehtuuri, jossa sovellus jaetaan malliin, näkymään ja kontrolleriin
NMVO	National Medicine Verification Organisation, kansallinen lääkevarmennusorganisaatio

NMVS	National Medicine Verification System, kansallinen lääkevarmennusjärjestelmä
ORM	Object Relational Mapping, olio-relaatio-mallinnus-ohjelmointitekniikka
PDA	Personal Digital Assistant, kädessä pidettävä tietokone
PL/SQL	Oraclen SQL-proseduurilaajennus
REST	REpresentational State Transfer, HTTP-protokollaan perustuva arkkitehtuurimalli
SQL	Structured Query Language, strukturoitu kyselykieli
XHTML	eXtensible Hypertext Markup Language, XML-kielen muotovaatimukset täyttävä HTML-kieli
WMS	Warehouse Management System

1 Johdanto

Lääkepakkauksien saapuminen turvallisesti apteekin hyllylle ja sieltä asiakkaalle on monimutkainen ja jatkuvan kehityksen alainen prosessi. Kehitystyön suuntaa ohjaavat asiakkaan tarpeet ja vaatimukset, mutta välillä sen määrittää myös uusi lainsäädäntö tai esimerkiksi pandemia.

Korona-aika on tuonut haasteita niin maailmanlaajuisesti kuin yksittäisten ihmisten arkeenkin. Sen myötä on jouduttu tekemään nopeita päätöksiä, ideoimaan ja innovoimaan ongelmien ratkaisemiseksi ja joustamaan aiemmista normaaleista käytännöistä poikkeustoimin. On vaadittu korkeaa organisointikykyä, jotta eri toimijat on saatu toimimaan yhdessä. Myös taustalla toimivat järjestelmät ovat joutuneet koetukselle, ja niitä on osittain jouduttu muokkaamaan uusien normien takia.

Pandemian aikana julkisen keskustelun aiheita ovat olleet esimerkiksi maskien ja lääkkeiden saatavuushäiriöt. Häiriöiden taustalla on viime aikoina ollut COVID-19-pandemia, jonka myötä lääkkeiden kysyntä on lisääntynyt ja kapasiteettiongelmat kasvaneet. (Paaskoski ym. 2020.) Muun muassa saatavuusongelmiin valmistautumiseen Suomessa on kehitetty yhteinen tietovaranto, joka kerää tietoa apteekkien ja lääketukkukauppojen varastotilanteesta sekä Kanta-palvelun reseptikeskuksen toimitusmerkinnöistä. Tätä tietovarantoa voidaan jatkossa hyödyntää varautumissuunnitelmissa, jotta myös tulevat pandemiat tai muut kriisit on helpompi selättää. (Rannanheimo ym. 2020.)

Kriisien lisäksi myös esimerkiksi lainsäädäntö voi luoda tarpeen kehittää uusia ja jo olemassa olevia järjestelmiä. Vuonna 2013 tuli voimaan uusi lääkeväärennösdirektiivi, joka toi keinoja lääkeväärennösten ehkäisyyn Euroopan talousalueen sisällä. Direktiivi loi tarpeen kehittää lääkelogistiikan rakenteissa olevia ohjelmistoja, jotta ne olivat turvaominaisuuksiltaan riittäviä kattamaan vaaditut säädökset ja lait. (Bucher ym. 2018.)

Tämän opinnäytetyön toimeksiantaja CGI Suomi Oy on kehittänyt Marela-toiminnanohjausjärjestelmän. Marela ja sen osasovellukset kattavat koko lääkelogistisen ketjun julkisen puolen sairaala-apteekeissa ja lääkekeskuksissa. Järjestelmässä on jo toiminnallisuus lääkevarmennuksen tekoon muussa tilanteessa, mutta on ilmennyt kehitystarve laajentaa lääkevarmennusmahdollisuutta sisälogistiikan toimintoon, lääkkeiden keräilyyn. Tämä opinnäytetyö on osa Marela-järjestelmään ja sen keräilylaitesovellukseen toteutettavaa muutostyötä, jonka päämuutos on tavaran toimittamisen yhteydessä suoritettava lääkevarmennus.

Tämän opinnäytetyön päätavoitteena on toteuttaa tavaraa toimitettaessa tehtävä lääkevarmennus keräilylaitesovellukseen. Lisäksi tavoitteena on toteuttaa muutos, jolla hyödynnetään 2D-koodinluennasta saatuja tuote-erää koskevia tietoja.

Tässä työssä käsitellään aluksi lääkelogistiikan ja sisälogistiikan teoriaa, jonka jälkeen käydään läpi lääkeväärennösdirektiiviä, joka jo aiemmin loi vaatimuksen toteuttaa tavaran saapumisen yhteydessä tapahtuvan lääkevarmennuksen. Sen jälkeen tehdään katsaus keräilylaitesovelluksen teknologioihin, työssä käytettyihin ohjelmointiympäristöihin sekä keräilylaitesovelluksen käyttöliittymään. Toteutusosiossa käsitellään keräilylaitesovelluksen muutosten toteutusta. Lopuksi tarkastellaan työn tavoitteiden täyttymistä ja pohditaan sovelluksen käytettävyyttä, arkkitehtuuria ja keräilylaiteohjelmiston jatkokehitysmahdollisuuksia.

2 Lääkelogistiikka

Lääkelogistiikan ongelmat, kuten esimerkiksi jakeluhäiriöt, aiheuttavat lääkkeiden saatavuushäiriöitä. Suomessa saatavuushäiriöt eivät yleensä johdu kansallisen kysynnän kasvusta, vaan häiriöiden juurisyyt löytyvät Suomen ulkopuolelta, sillä Suomi on riippuvainen lääkkeiden maahantuonnista. Samanlainen tilanne on käytännössä koko Euroopan talousalueella, koska se on riippuvainen niin tuotannon kuin tuonninkin puolesta kolmansista maista. Lääkkeiden saatavuushäiriöt ja korvaavien lääkkeiden puute aiheuttavat ongelmia niin potilaille, lääkäreille kuin apteekeillekin, kun juuri oikeaa, hoitoon sopivaa lääkettä ei ole välttämättä saatavilla. (Paaskoski ym. 2020.)

Fimean määräyksen lääkehuollon määrityksen mukaan (2012, 5) lääkehuollolla tarkoitetaan kokonaisuutta, jolla varmistetaan turvallisten, tehokkaiden ja kohtuuhintaisten lääkkeiden saatavuus. Kokonaisuuteen kuuluvat lääkkeiden hankintaa, valmistusta, käyttökuntoon saattamista, varastointia, toimittamista ja lääkeinformaation antamista lääkkeitä käyttäville toimintayksiköille. Toimintayksiköitä ovat avohuollon apteekit, sairaala-apteekit, lääkekeskukset, lääketehtaat ja lääketukkukaupat.

Euroopan talousalueella (ETA) lääkevalmisteet päästetään jakeluun lääketukuille ETA-alueella sijaitsevien lääketehtaiden kautta, riippumatta siitä, ovatko lääkevalmisteet valmistettu ETA-alueen sisä- vai ulkopuolella. Lääketehtaat vastaavat siitä, että lääkevalmisteiden tuotanto- ja testausvaiheet on tehty lääkevalmisteiden myyntiluvan ja EU:n lääkevalmistusta koskevien säädösten ja ohjeiden mukaisesti. (Junttonen 2017.)

Sairaala-apteekit ja lääkekeskukset voivat tilata lääkevalmisteita lääketukuilta tai Suomessa sijaitsevilta lääketehtailta suoraan. Sen lisäksi sairaala-apteekki voi tuoda lääkkeitä maahan yksittäistapauksissa esimerkiksi asianomaisen sairaanhoitopiirin omaa käyttöä varten. Kun lääkkeitä vastaanotetaan, lääkkeille tehdään vastaanottotarkistus, jossa varmistutaan saapuvan tavaran oikeellisuudesta ja ettei se ole vahingoittunut kuljetuksen aikana. Kun vastaanotetaan erityissäilytystä tai turvatoimenpiteitä vaativia lääkkeitä,

siirretään ne vastaanottotarkastuksen jälkeen asiaankuuluviin tiloihin. (Fimea 2012, 10.)

Sairaala-apteekeista ja lääkekeskuksista toimitetaan lääkkeitä niitä tilaavalle yksiköille, kuten sairaaloiden osastoille. Farmaseuttinen henkilökunta varmistaa tilausten ja toimitusten oikeellisuuden ennen niiden toimittamista. Lääkevalmisteet toimitetaan alkuperäispakkauksissaan tai tarpeen vaatiessa jaettuina pakkauksina. Jaetuissa pakkauksissa tulee olla kaikki lääkkeen jäljittämiseen ja tunnistamiseen liittyvät merkinnät sekä oikeaan käyttötapaan ja säilytykseen liittyvät tiedot, jotka löytyvät alkuperäisestäkin lääkepakkauksesta. Lääkkeiden laadun säilyttämiseksi lääkkeet tulee pakata, merkitä ja kuljettaa tietyillä tavoilla, esimerkiksi sinetöimällä kuljetuslaatikot niin, että vastaanottaja voi varmistua, ettei laatikkoa ole kuljetuksen aikana avattu sekä seuraamalla kuljetuksen lämpötiloja säännöllisin väliajoin. Lääkelain mukaan sairaala-apteekit ja lääkekeskukset voivat myös valmistaa lääkkeitä. (Fimea 2012, 11, 4.)

2.1 Sisälogistiikka

Logistiikan Maailma -verkkosivuston mukaan (2022a) logistiikasta on kehitelty useita hieman toisistaan eroavia määritelmiä. Yhden määritelmän mukaan logistiikalla tarkoitetaan hankintaan, varastointiin, kuljetukseen ja jakeluun liittyvien materiaalien ja palveluiden suunnittelua, toteutusta ja seurantaa asiakasvaatimukset huomioiden. Tämä erilaisten toimintojen sarja sisältää yrityksen tulo-, sisä- ja lähtölogistiikan.

Sisälogistiikka on materiaali- ja tietovirtojen hallintaa, joka tapahtuu esimerkiksi varaston tai terminaalin aitojen sisäpuolella. Käytännössä sisälogistiikan erilaisia toimenpiteitä ovat tavaran vastaanotto, hyllytys, siirto, keräily, pakkaaminen, lastaaminen ja kierrätys. (Logistiikan Maailma 2022b.)

Tavaraa vastaanottaessa tavara käydään jollain sovitulla tapaa läpi ja kuitataan käytössä olevaan järjestelmään, kuten varastonhallintajärjestelmään, vastaanotetuksi. Tavaraa voidaan vastaanottaa esimerkiksi siten, että luetaan

saapuneen kuorman kollitarrassa oleva SSCC-viivakoodi, eli sarjatoimitusyksikkökoodi, järjestelmään. SSCC-viivakoodin avulla saadaan tieto kuorman sisältämästä tavarasta ja sen määrästä sekä siitä, mille ostotilaukselle kyseinen tavara kuuluu. Kun viivakoodi on luettu, voidaan varmistaa esimerkiksi tabletin avulla järjestelmän kautta, että kuorma sisältää oikean sisällön ja kuitata se vastaanotetuksi. Tässä vastaanottotavassa lavoja tai laatikoita ei pureta tai käsitellä uudelleen, ja sen takia kyseisen vastaanottotavan käyttö edellyttää yhteistyötä ja luottamusta tavarantoimittajien kanssa. Toinen tapa tavaran vastaanotolle on käydä tavara läpi järjestelmästä saatavan vastaanottolistan avulla. Vastaanottolista käydään läpi kuittaamalla rivi kerrallaan tavara vastaanotetuksi. Vastaanoton kuittaamisen jälkeen järjestelmästä voidaan tulostaa vastaanottotarra, josta nähdään, että tavara on vastaanotettu ja usein myös sen sisältö ja aika, jolloin erä on saapunut. (Logistiikan Maailma 2022b.)

Vastaanoton jälkeen tavara voidaan käytännöistä riippuen hyllyttää joko saman tien tai jättää odottamaan tietylle alueelle, jotta myöhemmin voidaan hyllyttää samalla kerralla isompi määrä tavaraa. Tavaran hyllyttämisen paikka ja tapa riippuvat varastointitavasta. Dynaamisten varastopaikkojen avulla varastointi on joustavaa. Dynaamisissa varastopaikoissa tavaralla ei ole kiinteää varastopaikkaa, vaan järjestelmä määrittelee saapuneille erille parhaan vapaan varastopaikan huomioiden samalla tavaran kiertoluokan, eli sen keräyskertojen määrän. Silloin tavara, jota kerätään usein, on sijoitettu sellaiseen varastopaikkaan, joka on optimaalisella paikalla kerääjään nähden ja harvemmin kerätyt sijoitetaan kauemmaksi. Toinen tapa varastoida on käyttää kiinteitä varastopaikkoja, jolloin tavara on aina samalla paikalla. Näiden varastointitapojen lisäksi voidaan käyttää niiden välimuotoa, jolloin varastopaikat ovat dynaamisia. Välimuodossa järjestelmä ei ota huomioon kiertoluokkia, vaan tarjoaa ensimmäisen vapaan varastopaikan, ellei samanlaista tavaraa ole jo jollain varastopaikalla, jolloin kyseinen paikka määräytyy varastopaikaksi. (Logistiikan Maailma 2022b.)

Keräily tarkoittaa tavaran keräämistä varastopaikoilta jonkin listan tai muun ohjeen mukaisesti ja niiden viemistä ennalta määrättyyn paikkaan, kuten lähetysalueelle. Pelkässä valmistuotevarastossa on usein yhdenlainen keräilyprosessi, valmistuotteiden keräily lähetettäväksi. Tavaraa valmistavassa organisaatiossa on usein edellisen keräilyprosessin lisäksi keräilyä tuotantoon, jossa tavarasta vasta valmistetaan valmistuotteita. Itse keräily voidaan jakaa kahteen tapaan, joista toisessa kerätään manuaalivarastosta ja toisessa automaattivarastosta. Manuaalivaraston keräilyprosessissa keräilijä saa jossain muodossa, esimerkiksi paperisena tai sähköisenä kämmenlaitteelleen keräilylistan, josta nähdään kerättävien tuotteiden tiedot ja kerättävät määrät sekä varastopaikat, mistä ne löytyvät. Kämmenlaitteella kerätyt tavarat voidaan kuitata järjestelmään heti keräämisen aikana ja paperisen listan avulla kerättynä yleensä heti keräilyn jälkeen. Automaattivarastosta keräily vähentää keräilijän liikkumisen tarvetta, sillä yleensä kone hakee tavaran ja tuo sen kerääjälle. Kun tavara päätyy keräilyn päätteeksi lähetysalueelle lähetettäväksi, sen tilaus kuitataan toimitusvalmiiksi järjestelmään. (Logistiikan Maailma 2022b.)

2.2 Ohjausjärjestelmät

Toimitusketjun toimintoihin, kuten varaston hallintaan ja tiedonvälitykseen, käytetään erilaisia ohjausjärjestelmiä. Näitä ovat esimerkiksi toiminnanohjausjärjestelmä ja varastonhallintajärjestelmä. (Logistiikan Maailma 2022e.)

Toiminnanohjausjärjestelmä eli ERP-järjestelmä (Enterprise Resource Planning) tarkoittaa laajaa tietojärjestelmää, jota käytetään organisaation ohjaamiseen. Sen keskiössä on yhteinen tietokanta, jota toiminnanohjausjärjestelmään integroidut toiminnot käyttävät. Yhteisen tietokannan takia kaikki tieto on läpinäkyvää koko organisaatiossa ja sama tieto saavuttaa jokaisen toiminnon. (Logistiikan Maailma 2022c.)

Koko organisaation toiminnanohjausjärjestelmään sisältyy yleensä varastonhallintajärjestelmä. Varastonhallintajärjestelmä eli WMS (Warehouse

Management System) tarkoittaa varastotasojen hallintaa siten, että samalla otetaan huomioon palvelutasovaatimukset sekä varastointi- ja ohjauskustannukset. Järjestelmän avulla ohjataan ja hallitaan materiaalien ja tuotteiden siirtelyä, vastaanottoa, hyllytystä, keräilyä, pakkausta ja toimitusta. (Logistiikan Maailma 2022d.)

2.3 Kannettava keräilylaite

Varastonhallinta perustuu jokaisen sinne tulevan ja siellä olevan tavaran tunnistamiseen, jolloin jokaisen tavaran tunnisteen on oltava ajan tasalla järjestelmässä. Kaikki varaston toiminnot voidaan toteuttaa esimerkiksi kannettavalla tiedonkeruupäätteellä, jossa on lähiverkkoyhteys reaaliaikaisen tiedonsiirron saavuttamiseksi. (Finn-ID 2022.) Tavaran nopeassa ja luotettavassa tunnistamisessa voidaan hyödyntää viivakoodeja, joita voidaan lukea tiedonkeruupäätteellä, kuten kannettavalla keräilylaitteella (Logistiikan Maailma 2022d).

Monissa logistiikan tehtävien toteuttamiseen tarkoitetuissa kannettavissa tiedonkeruupäätteissä on kosketusnäyttö, fyysinen näppäimistö, viivakoodinlukija ja mahdollisuus lisätä erillinen kahva laitteen pitämiseen. Ne ovat tarkoitettu ammattilaiskäyttöön ja ovat siten kestäviä. (Honeywell 2022.) Opinnäytetyössä käsiteltävää keräilylaitesovellusta voidaan käyttää lisäksi tabletin tai työaseman kanssa, johon on liitetty viivakoodinlukija viivakoodien lukemiseen.

3 Lainsäädäntö

Euroopan unionin jäsenvaltiona Suomen lakiin vaikuttavat EU:n toimielinten antama johdettu oikeus eli oikeudellisesti sitovat säädökset. Säädöksiä voi luokitella tyyppien tai hierarkian mukaan. Hierarkkisesti ylimmällä tasolla ovat Euroopan parlamentin ja neuvoston antamat säädökset, jotka nojaavat suoraan johonkin jäsenvaltioiden ja EU:n väliseen perussopimukseen. Seuraavalla tasolla ovat edellä mainittuihin säädöksiin nojaavat säädökset, jotka annetaan yleensä komission toimesta. Tällöin puhutaan delegoidusta säädöksestä. Delegoidut säädökset eivät koske ylimmän säädöksen keskeisimpiä osia, vaan niillä täydennetään tai muutetaan ylimmän säädöksen muita osia. Säädösten tyypit ovat hierarkialtaan samantasoisia, ja niiden erot näkyvät niiden käyttötarkoituksessa. Säädökset voivat olla tyypiltään asetuksia, direktiivejä tai päätöksiä. Asetukset ja päätökset ovat siltä osin samankaltaisia keskenään, että ne ovat kaikilta osiltaan velvoittavia ja suoraan sovellettavia. Niiden voimaantulo ei edellytä erillisiä toimenpiteitä jäsenvaltiossa. Päätöksiä käytetään lähinnä hallinnollisessa menettelyssä yksittäisiä tapauksia ratkaistaessa, jolloin päätös osoitetaan tietylle taholle. Kolmas säädöstyyppi, direktiivi, ei ole suoraan sovellettava, vaan jäsenvaltioiden on tehtävä muutoksia tai lisäyksiä kansalliseen lainsäädäntöönsä, jotta ne ovat direktiivin mukaisia. Direktiivejä käytetään yleensä, kun halutaan jättää harkintavalta jäsenvaltioille, kuinka ne toteuttavat direktiivin vaatimat asiat. (Finlex 2022)

3.1 Lääkeväärennösdirektiivi

Euroopan komissio havahtui vuonna 2006 internetissä tapahtuvaan lääkeväärennösten kauppaamiseen, minkä johdosta alkoi lainsäädännön valmistelu lääkeväärennöksien ehkäisemiseksi (Paaskoski 2017). Tämän myötä syntyi lääkeväärennösdirektiivi 2011/62 (EU), joka tuli voimaan vuonna 2013 EU:n ja Euroopan talousalueella. Direktiivin yksi velvoite on se, että jäsenmaat ilmoittavat komissiolle tiedot kansallisista järjestelmistään heinäkuuhun 2013 mennessä. Näillä järjestelmillä pyritään estämään vaarallisiksi epäiltyjen

lääkkeiden päätyminen loppukäyttäjälle. (Lääkeväärennösdirektiivi 2011/62/EU.)

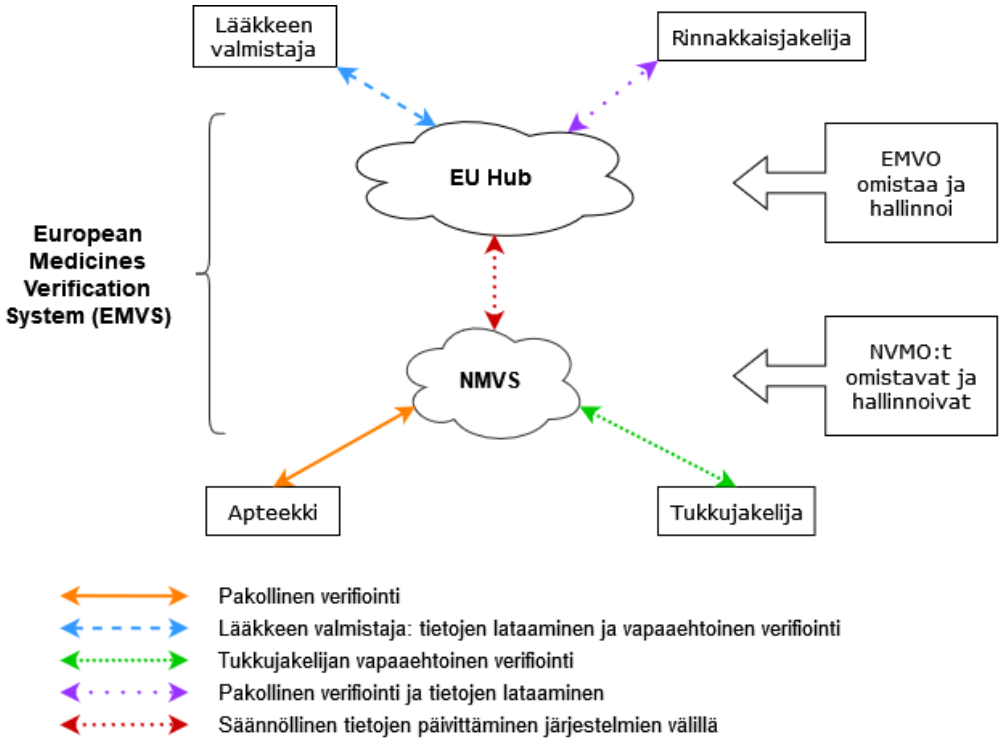
Lääkelaki ja lääkeasetus sisältävät keskeiset säädökset koskien Suomen lääkehuoltoa. Niissä säädetään lääkkeiden valmistuksesta, maahantuonnista, jakelusta, myynnistä ja myyntiluvista. (Sosiaali- ja terveysministeriö 2022.) Lääkeväärennösdirektiivin tarkentavassa delegoidussa asetuksessa täsmennetään turvaominaisuudet (Euroopan komissio n. d.). Nämä turvaominaisuudet takaavat sen, että lääkevalmisteen aitous voidaan tarkistaa ja yksittäisten lääkevalmisteiden tunnistaminen on mahdollista. Lääkelaissa säädetään komission delegoidun asetuksen mukaisesti ihmisille tarkoitettujen lääkkeiden pakkauksien turvaominaisuuksista. (Lääkelaki 23.2.2019/208.)

Uusien sääntöjen soveltaminen alkoi 9. helmikuuta 2019. Tämän jälkeen reseptilääkkeissä ja asetuksen vaatimissa reseptittömissä lääkkeissä on oltava peukaloinnin paljastava mekanismi ja yksilöllinen tunniste. (Linnolahti 2017.) Toimintavalmiutta odotettiin tuolloin myös tallennusjärjestelmältä, joka koostuu eurooppalaisesta keskuksesta ja kansallisista tietokannoista. Terveyden ja elintarviketurvallisuuden pääosaston, Euroopan lääkeviraston ja Euroopan lääkevirastojen yhteisen verkoston kirjeessä sidosryhmille esitettiin eri sidosryhmien vastuut, jotka väärennettyjä lääkkeitä koskevan direktiivin myötä muotoutuvat. Jotta lääkkeitä kansalaisille toimittavien toimijoiden, eli apteekkien, sairaala-apteekkien ja terveydenhuollon laitoksien keskeinen tehtävä lääkkeiden aitouden todentamisessa toteutuu, on heidän käyttämiensä tietokonejärjestelmien oltava myös toimintakunnossa delegoidun asetuksen voimaan tullessa. Toimintakuntoon saattaminen tarkoitti kehitystyötä, testausta ja pilotointia. (Bucher ym. 2018.)

3.2 Lääkevarmennusjärjestelmä

Käytännössä lääkeväärennösdirektiiviä ja sen delegoidun asetuksen turvaominaisuuksien toteuttamista varten luotiin eurooppalainen lääkevarmennusjärjestelmä EMVS (European Medicines Verification System),

joka koostuu kuvan 1 mukaisesti eurooppalaisesta keskustietokannasta, EU Hubista, sekä EU- ja ETA-maiden omista kansallisista tietokannoista. EU Hub sisältää kaikkien markkinoille tuotujen lääkepakkauksien yksilöivät tiedot, ja sieltä tiedot siirtyvät myös kansallisiin tietokantoihin, jotka ovat omia lääkevarmennusjärjestelmiään. (FiMVO 2022.)



Kuva 1. EMVS koostuu EMVO:n omistamasta EU Hub-keskustietokannasta ja NVMO:n omistamista NVMS:stä, kuten FiMVO (mukaillen FiMVO 2022).

Lainsäädännön mukaan lääkevarmennusjärjestelmiä hallinnoimaan tarvitaan voittoa tavoittelemattomat organisaatiot. Euroopan tasolla keskustietokannan hallinnoimista ja yhteistyöelimenä toimimista varten luotiin EMVO (European Medicines Verification Organisation), joka muodostui lääketeollisuuden, apteekkien, lääketukkujen ja sairaala-apteekkien keskusjärjestöjen toimesta. (FiMVO 2022.) Kansallisia järjestelmiä eli NMVS:jä (National Medicines Verification System) hallinnoimaan perustettiin myös omat kansalliset organisaatiot eli NMVO:t (National Medicines Verification Organisation).

Suomessa tätä varten perustettiin Suomen Lääkevarmennus OY, lyhyemmin FiMVO, vuonna 2016 (Linnolahti 2017). FiMVOn järjestelmää valvoo lääkealan turvallisuus- ja kehittämiskeskus Fimea (FiMVO 2022).

3.3 Lääkkeiden turvaominaisuudet

Lääkevalmisteen yksilöimiseksi ja aitouden tarkistamiseksi lääkepakkauksiin lisättiin pakkauskohtainen ja yksilöllinen tunniste ja pakkauksen peukaloinnin paljastava mekanismi. Pakkauskohtaisena tunnisteena toimiva kaksiulotteinen viivakoodi eli 2D-koodi mahdollistaa yksilöllisen tunnistamisen ja aitouden tarkistamisen. Sen yhteydessä on myös silmin luettava muoto koodin sisältämistä tiedoista. 2D-koodin tarkastuksella voidaan varmistaa, että lääke löytyy EU Hubista. Turvaominaisuudet koskevat ihmisille tarkoitettuja reseptilääkkeitä, pois lukien 14 reseptilääkeryhmää. (Linnolahti & Kankkunen 2018.) Delegoitu asetus sisältää listan näistä lääkeryhmistä. Poissuljettuja reseptilääkkeitä ja -lääkeryhmiä ovat esimerkiksi homeopaattiset lääkkeet, kitit, lääkkeelliset kaasut ja varjoaineet, joiden anatomis-terapeuttis-kemiallinen koodi, eli ATC-koodi alkaa V08 (Komission delegoitu asetus (EU) 2016/161).

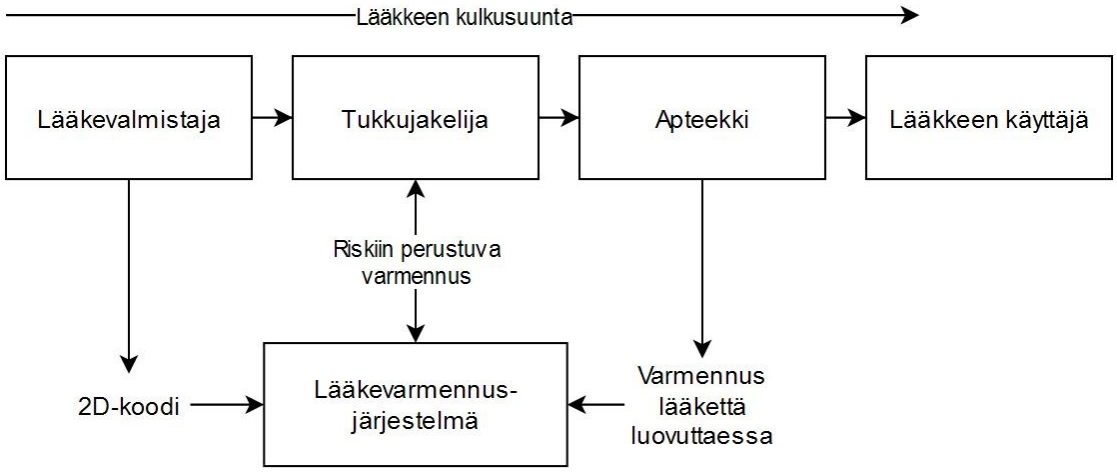
Lääkepakkauksien 2D-koodi noudattaa GS1-standardin mukaista ECC 200-version datamatriisia. Standardin mukaisesti 2D-koodi sisältää vähintään seuraavat neljä tietoa lääkepakkauksesta:

1. Tuotekoodi, yleensä GTIN
2. yksilöity sarjanumero
3. eränumero
4. viimeinen voimassaolopäivä.

Nämä tiedot on standardin mukaan oltava myös silmin luettavassa muodossa. (EFPIA ym. 2017, 4–5.)

3.4 Lääkevarmennus

Lääkevarmennus tarkoittaa menetelmiä, joilla varmennetaan lääkepakkauksen aitous ja turvallisuus. Kun esimerkiksi apteekissa luetaan lääkepakkauksen 2D-koodi, järjestelmä vertaa sen tietoja lääkevalmistajan lääkevarmennusjärjestelmään syöttämiin tietoihin, kuten kuvassa 2 on kuvattu. Jos tiedot eivät vastaa toisiaan eli havaitaan lääkeväärennösepäily, lääkevarmennusjärjestelmä tekee hälytyksen. Mikäli lääkepakkaus todennetaan, varmistetaan vielä peukaloinnin varmistavan mekanismin, kuten sinetin tai repäisynauhan, olevan ehjä. Lääkevarmennuksen jälkeen pakkaus voidaan luovuttaa eteenpäin, esimerkiksi apteekista lääkkeen käyttäjälle. (FiMVO 2022.)



Kuva 2. Kaavio lääkkeen kulusta ja lääkevarmennuksen toimimisesta taustalla (mukaillen EMVO 2022).

4 Keräilylaitesovellus

KaKe-keräilylaitesovellus on yksi Marela-toiminnanohjausjärjestelmän osasovelluksista. Sen avulla voidaan tehdä monia varastonohjaukseen liittyviä toimintoja, kuten tuotteiden keräilyä listan perusteella, saapuneiden tuotteiden kirjausta ja inventointia. Lisäksi sovelluksella voidaan tehdä muita varastonhallinnassa hyödyllisiä toimenpiteitä, kuten tulostaa toimituslistoja ja ylläpitää tuotteiden viivakoodeja. Koska sovellus on selainpohjainen, sitä voidaan käyttää käytännössä millä laitteella tahansa. Sovellus toimii siis niin tabletilla kuin perinteisimmilläkin keräilylaitteilla.

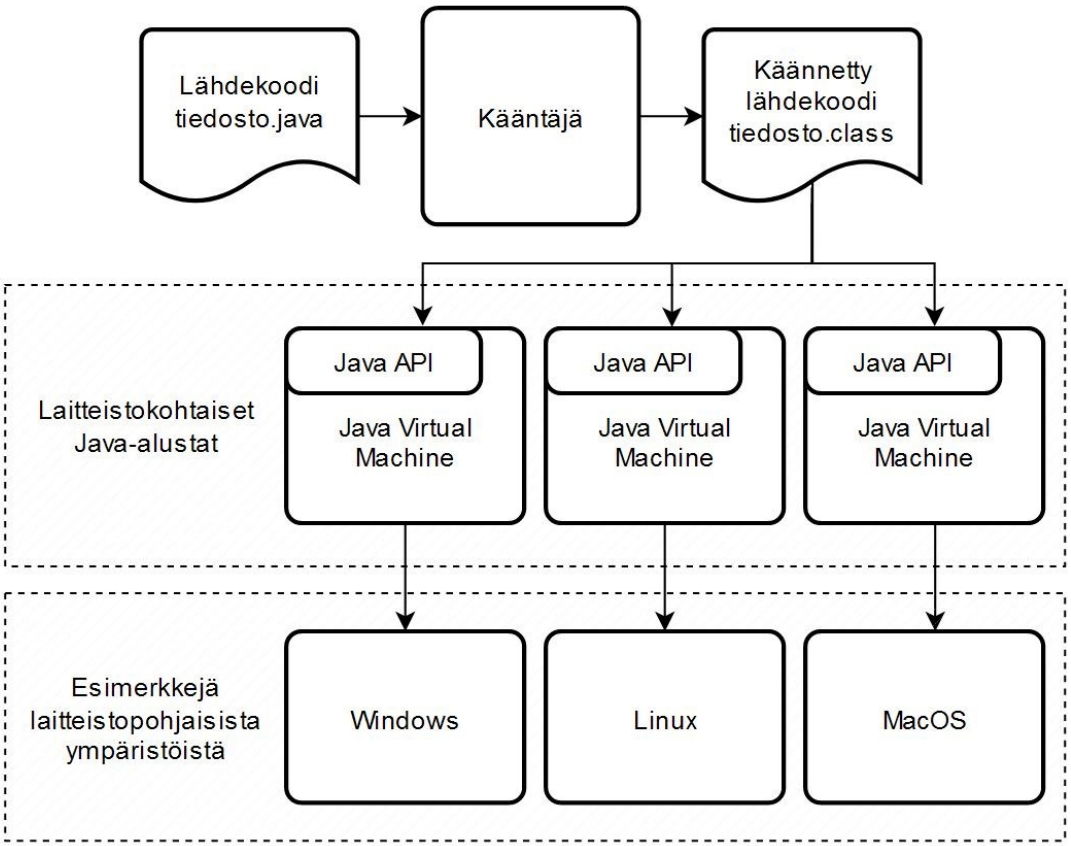
Keräilylaitesovellukseen tehtiin lääkeväärennösdirektiivin ja sitä tarkentavan komission delegoidun asetuksen myötä muutoksia, jotta vaatimusten mukainen lääkevarmennus oli mahdollista jokaisessa toimitusketjun vaiheessa. Sairaala-apteekeille tämä tarkoitti lääkevarmennuksen suorittamista tavaran vastaanottovaiheessa. Tästä syystä sovellukseen lisättiin tavaran vastaanotossa tehtävä lääkevarmennus, joka hyödyntää koodinluennasta saatuja datamatriisin tietoja mahdollistaen lääkevarmennuksen delegoidussa asetuksessa määritellyille lääkeryhmille.

Keräilylaitesovellus käyttää Java-alustaa. Sen käyttöliittymäteknologiana käytetään JavaServer Faces:ia (JSF). Sovelluksen taustajärjestelmänä käytetään Oraclen tietokantaa.

4.1 Teknologiat, ohjelmointikielet ja arkkitehtuuri

Java-teknologia tarkoittaa sekä alustaa (engl. platform) että ohjelmointikieltä. Alusta voi olla laitteisto- tai ohjelmistoympäristö, jossa ohjelma toimii. Java-alusta on ohjelmistoympäristö, joka toimii laitteistopohjaisen alustan, kuten Windows-käyttöjärjestelmän, päällä. Java-alusta sisältää kaksi komponenttia kuvan 3 mukaisesti: JVM:n (Java Virtual Machine) ja Java API:n (Java Application Programming Interface). JVM tarkoittaa Java-virtuaalikonetta, eli tietokonetta, joka on toteutettu ohjelmallisesti. Sen avulla Java-pohjaiset

ohjelmat toimivat monissa eri laitteistopohjaisissa ympäristöissä. API tarkoittaa ohjelmointirajapintaa, joka muodostaa kahden eri ohjelman välille rajapinnan. Java API on kokoelma valmiita ohjelmistokomponentteja, jotka ovat hyödyllisiä Java-kehityksessä. (Oracle 2022a.)



Kuva 3. Kuvio Java-alustasta, joka sisältää Java API:n ja JVM:n, ja joka eristää kääntäjällä käännetyn ohjelmakoodin laitteistopohjaisista ympäristöistä (mukaillen Oracle 2022a).

Java on korkean tason luokkapohjainen ja oliotyyppinen ohjelmointikieli. Java-kieli on Java-alustan ansiosta laitteisto- ja ohjelmistoriippumaton. Java-kieli on vahvasti ja staattisesti tyypitettyä. (Gosling ym. 2018, 1.) Staattisesti tyypitetty tarkoittaa sitä, että kielen tyypit tarkistetaan ennen ohjelman suoritusta. Java-

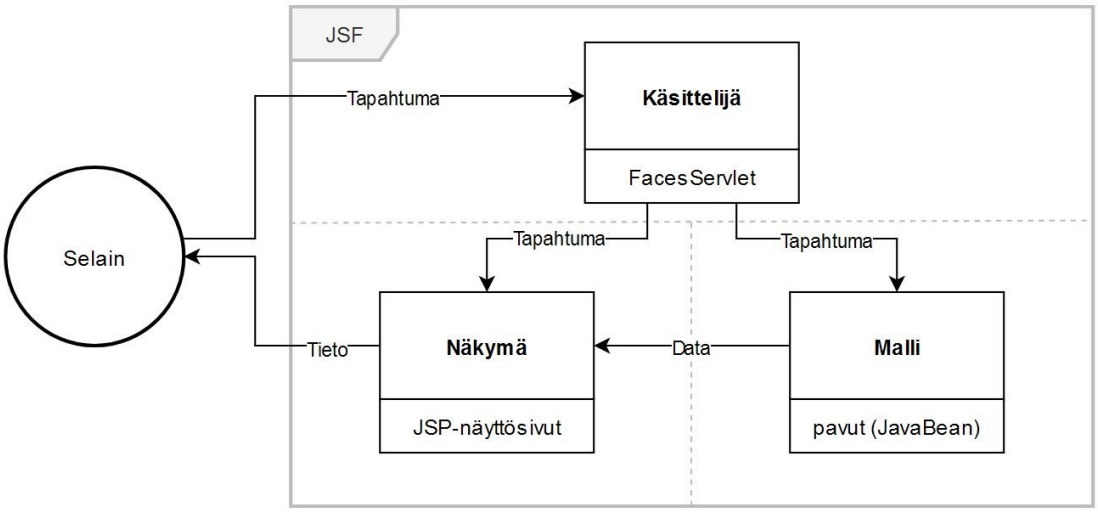
kielessä lähdekoodit kirjoitetaan tekstitiedostoihin, jotka tallennetaan java-päätteellä. Lähdekooditiedostot käännetään tämän jälkeen kääntäjällä class-tiedostomuotoon, kuten kuvassa 3 kuvataan. Käännetyt tiedostot sisältävät tavukoodia, joka on konekielen tason kieltä. Tietokoneen keskussuoritin ei osaa tulkita tavukoodia, vaan sen sijasta class-tiedostoja tulkitsee JVM. Koska JVM on saatavilla useimmille käyttöjärjestelmille, samat class-tiedostot toimivat niillä huolimatta käyttöjärjestelmien eroista. (Oracle 2022a.)

JSF-teknologia (JavaServer Faces, nykyisin Jakarta Server Faces) on standardi palvelinpuolen käyttöliittymien rakentamiseen. JSF-teknologia sisältää joukon ohjelmointirajapintoja, joiden avulla voidaan muodostaa käyttöliittymäkomponentteja ja hallita niiden tilaa. Näiden ohjelmointirajapintojen avulla voidaan myös käsitellä ohjelman tapahtumat, syötteiden validoinnit ja sivujen välisen navigoinnin. Lisäksi ne mahdollistavat kansainvälistämistämisen ja saavutettavuuden tukemisen. JSF-teknologian käyttöliittymäkomponenttiluokat kapseloivat komponentin toiminnallisuuden käyttöympäristökohtaisen esitystavan sijasta. JSF-teknologian käyttöliittymäkomponentteja voidaan siten renderöidä (engl. render) useille erilaisille käyttäjälaitteille. (Oracle 2022b.)

JSF-teknologia sisältää ohjelmointirajapintojen lisäksi mukautetun tunnistekirjaston nimeltään JSP (JavaServer Pages). JSP-kirjaston avulla voidaan näyttää JSF-pohjainen käyttöliittymä JSP-sivulla. (Oracle 2022b.) JSP on yksinkertainen, sivupohjainen ratkaisu dynaamisen HTML-kielen luontiin (Graham Mansfield 2015a). Versiosta 2.x eteenpäin JSF-teknologian oletustapa näkymien luomiseen vaihtui JSP:stä Facelets nimiseen web-mallijärjestelmään (engl. web template system), joka on hyvin samankaltainen JSP:n kanssa, mutta on tehokkaampi ja helpompi käyttää (Hookom 2016).

JSF-teknologia tukee puhdasta Model-View-Controller- eli MVC-arkkitehtuuria (Graham Mansfield 2015a). MVC-arkkitehtuuri jakaa ohjelmiston kolmeen rooliin: malliin, näkymään ja käsittelijään. MVC-arkkitehtuuri edellyttää, että koodi rajoittuu jokaisessa sovelluksen elementissä vain kyseisen elementin rooliin sovelluksen yleisessä arkkitehtuurissa. Toisin sanoen tietoja käsittelevän

koodin on oltava erillään koodista, joka näyttää tietojen käsittelyn tuloksen. Tietoja käsittelevä koodi kuuluu mallin rooliin ja tietojen käsittelyn tuloksen näyttävä koodi kuuluu näkymän rooliin. Käsittelijä, eli koodi, joka ohjaa muiden toimintojen välistä kulkua, delegoi ja koordinoi toimintoja näkymässä ja mallissa. (Cole ym. 2011, 7.) JSF-teknologiassa sovelluksen toimintaa ohjaava käsittelijä on kiinteä elementti nimeltään FacesServlet, joten sitä voidaan hyödyntää suoraan sovelluksessa. Malli ja näkymä eivät ole kiinteitä elementtejä, vaan niiden komponentit on ohjelmoitava erikseen. Kuvan 4 mukaisesti mallikomponentit koostuvat käyttöliittymän komponentteihin liitetyistä pavuista (bean), jotka ovat nimeämiskäytännöiltään JavaBeanin ohjeiden mukaisia JavaBean-komponentteja, sekä muista javaluokista. Näkymäkomponentit koostuvat xHTML-tiedostoista. (Graham Mansfield 2015b.) JSP tai Facelets, riippuen JSF:n versiosta, täyttää näkymän roolin, ja sitä kutsutaan yleisesti näyttösivuksi, kun sitä käytetään osana MVC-arkkitehtuuria (Cole ym. 2011, 7).



Kuva 4. MVC-arkkitehtuuri JSF-teknologiassa (mukaillen Kristály 2005, 76).

Relaatiotietokantojen ja olio-ohjelmoinnin välillä käytetään ORM-ohjelmointitekniikkaa (Object Relational Mapping) eli olio-relaatio-mallinnusta, jolla voidaan muuntaa olioita tietokantataulujen riveiksi ja takaisin. JPA (Java Persistence API) on olio-relaatiomallinnus-API, joka auttaa automatisoimaan

Java-olioiden yhdistämisen relaatiotietokantatauluihin. JPA:n mukainen kirjasto Hibernate on yksi olio-relaatiomallinnustyökaluista. (Heffelfinger 2011, 204.) JSF-teknologiassa JPA ja JavaBeans-komponentit täyttävät MVC-arkkitehtuurin mallin roolin (Heffelfinger 2011, 331).

Java Expression Language (EL) on olio-ohjelmointikieli, jota käytetään dynaamiseen kommunikaatioon JSF-pohjaisissa sovelluksissa. Käytännössä kieltä käytetään näkymäkomponenttien koodissa, mikä mahdollistaa kommunikoinnin sovelluksen logiikkakerroksen kanssa. (Leonard 2014, 27)

4.2 Tietokannan hallintajärjestelmä

Tietojärjestelmä on formaalinen systeemi eli systeemi, joka perustuu hyväksyttyihin ja etukäteen sovittuihin tietojen ja menettelytapojen määritelmiin tiedon tallentamiseen ja käsittelyyn. Tietojärjestelmien automatisoimiseen käytetään tietokantaa, joka on järjestetty kokoelma tietoja, joita käsitellään yhtenä yksikkönä. Tietokannan tarkoitus on kerätä, tallentaa ja noutaa asiaan liittyviä tietoja tietokantasovelluksen käyttöön. (Oracle 2022c.)

Tietomalli kuvaa tietokannan tietorakennetta ja tiedon välisiä yhteyksiä. Nykyään laajimmin hyväksytty tietomalli on E. F. Coddin vuonna 1970 kehittämä matemaattiseen joukkoteoriaan perustuva relaatiomalli. Relaatiomalli antaa mallin tietokannan rakenteelle, operaatioille ja eheyssäännöille. Relaatiotietokannat perustuvat relaatiomalliin. (Oracle 2022c.) Relaatiomallissa kaikki tieto on loogisesti järjestelty relaatioihin eli tauluihin. Relaatioilla on omat nimensä ja ne koostuvat nimetyistä kolumneista (attribuutti). Jokainen taulun rivi (tuple) sisältää jokaista kolumnia kohden yhden arvon. (Connolly 2014, 149)

Tietokannan hallintajärjestelmä (Database Management System) on ohjelma, jolla voidaan hallita tietojen tallennusta, järjestämistä ja hakua. Relaatiomalli antaa pohjan relaatiotietokannan hallintajärjestelmälle (Relational Database Management System). Oracle Database on yksi relaatiotietokannan hallintajärjestelmistä. Se sisältää lisäksi oliopohjaisia ominaisuuksia, kuten käyttäjän määrittämiä muuttujien tyypityksiä, periytymistä, ja polymorfismia.

Oracle Databasea voi kuvata siis oliorelaatiotietokannan hallintajärjestelmäksi. (Oracle 2022c.)

Relaatiotietokantoja käytetään strukturoidun kyselykielen (Structured Query Language, SQL) avulla. SQL ei ole proseduraalinen kieli, eli sillä ei proseduurikielen kuten Javan tavoin kirjoiteta eri vaiheita, miten ohjelman halutaan toimivan. Sen sijaan SQL:n voidaan suoraan kertoa mitä halutaan tehtävän ja SQL-kielen kääntäjä luo automaattisesti proseduurin tietokannassa hakemiseen ja tehtävän suorittamiseen. Oracle Database sisältää Oracle SQL:n proseduurilaajennuksen, PL/SQL:n. Se mahdollistaa SQL-käskyjen yhdistämisen ohjelmallisiin rakenteisiin, kuten proseduureihin, funktioihin ja paketteihin. PL/SQL:n avulla sovelluslogiikka saadaan paljon lähemmäksi tietokantaa. (Oracle 2022c.)

4.3 Ohjelmointiympäristöt

Ohjelmointityön tueksi on kehitetty erilaisia ohjelmointiympäristöjä eli IDE:ja (Integrated Development Environment), jotka helpottavat ja nopeuttavat työskentelyä. Eclipse on avoimen lähdekoodin IDE ja soveltuu ohjelmien kehittämiseen useilla eri ohjelmointikielillä ja on melko suosittu erilaisten Java-pohjaisten ohjelmien kehittämiseen. IDE:t sisältävät tekstieditorin koodin muokkaamista varten sekä työkaluja, jotka automatisoivat ohjelmointikielen kääntämisen ja ajamisen. Eclipse IDE tukee lisäksi ohjelman rakentamista julkaisuvalmiiseen muotoon, yksikkötestausta, versionhallintaa ja muita sovelluskehityksen työvaiheiden tehtäviä. (Kulkarni 2018, 7, 13.)

Oracle Databasea varten on olemassa monia omia IDE:ja, yhtenä esimerkkinä SQL Navigator. Kuten muissakin ohjelmointiympäristöissä, SQL Navigatorissa on tekstieditori PL/SQL-ohjelmien kirjoittamiseen ja muokkaamiseen. SQL Navigatorilla voidaan tehdä monia erilaisia tietokannan hallintatoimintoja, kuten rakentaa ja hallita tietokantaobjekteja. Se sisältää oman versionhallinnan ja koodinjaon muiden kehittäjien kanssa. (Quest Software Inc. 2017)

4.4 Käyttöliittymä ja käytettävyys

KaKe-keräilylaitesovelluksen käyttöliittymään tehdään ajoittain muutoksia uusien toimintojen ja vaatimuksien myötä. Keräilylaitesovelluksen käyttöliittymällä ei ole virallista tyyliopasta, mutta sen kehityksessä noudatetaan tiettyjä käytänteitä. Yksi käytänteistä on asiakaslähtöinen suunnittelu. Keräilylaitesovellusta kehitetään yhteistyössä asiakkaiden kanssa, jolloin myös käyttäjien mielipiteet ja toiveet tulevat kuulluksi.

Holtzblattin ja Beyerin (2017, 305–306) mukaan nykyään ihmiset odottavat muun muassa ohjelmistojen olevan suunniteltu siten, että niiden käyttäminen auttaa päivittäisiä aktiviteetteja ihmisiä hidastamatta. Heidän mukaansa hyvä tuotesuunnittelu tarkoittaa sitä, että ihmiset voivat käyttää tuotteita työ- ja vapaa-ajan eri aktiviteettien lomassa ilman eri ajatusta, koheltamista tai etsimistä. Tällöin käyttäjän tarvitsee miettiä ainoastaan, mitä haluaa tehdä tai saavuttaa eikä sitä, miten sen saa aikaan. Tuotteita tulisi pystyä käyttämään osittaisella keskittymisellä ja jatkuvien keskeytyksien keskellä. Hyvä tuotesuunnittelu pitää tuotteella tehdyn toiminnon yhdenmukaisena muun tuotteen toiminnallisuuden kanssa, mutta samalla myös johdonmukaisena muun elämän kanssa, kun käyttäjä siirtyy tuotteen käytöstä toiseen.

Aiemmin kun teknologia-alustat olivat joustamattomampia ja käyttäjät istuivat päätteen ääressä pidempiä aikoja niiden käyttöä opetellen, kaikki kohdeaktiviteettiin liittyvä toiminnallisuus ja informaatio pyrittiin tuomaan yhteen ainoaan käyttöliittymään yhdenmukaisuuden luomiseksi. Kaikkien toiminnallisuuksien hallinnoimiseksi luotiin monimutkaisia järjestelmiä, kuten valikkoja alivalikkoineen, hierarkkista tietoa, jota navigoitiin kategorioittain, monta työkalupalettia sekä ponnahdusikkunoita. Niiden avulla kaikki tiedot ja toiminnot olivat käyttäjän käytettävissä, jolloin käyttäjät joutuivat itse miettimään, missä tarvittavat asiat ovat ja kuinka niitä käytetään. Nykyään teknologian kehityksen myötä aiemmin runsaasti aikaa vievä käyttö tapahtuu elämän lomassa, esimerkiksi autossa parkkipaikalla tai suorittaessa jotain työtehtävää. (Holtzblatt & Beyer 2017, 306)

Holtzblatt ja Beyer (2017, 307) tiivistävät asian niin, etteivät käyttäjät halua liian monimutkaisia tuotteita, vaan he vaativat tuotteita, joiden käyttöä ei tarvitse opetella lainkaan ja niiden käyttäminen onnistuu saman tien.

5 Toteutus

Tämän työn tavoitteena oli toteuttaa KaKe-keräilylaitesovellukseen kaksi muutosta: lääkevarmennuksen teko keräilyvaiheessa sekä samalla 2D-koodinluennasta saatujen tuote-erätietojen hyödyntäminen. Keräilyvaiheella tarkoitetaan tässä tilanteessa sitä vaihetta, kun tavaraa keräillään toimitusta varten, eikä esimerkiksi tuotantoa varten.

5.1 Muutosten määrittely

Lääkevarmennuksen toteutus KaKe-keräilylaitesovellukseen vaati muutoksia niin taustalla toimivaan Marela-toiminnanohjausjärjestelmään, kuin keräilylaitesovellukseenkin. Kaikki muutokset oli kuvattu toimeksiantajan toimesta muutoskuvauksessa ja sen perusteella lähdettiin tässä opinnäytetyössä tehtäviä keräilylaitesovelluksen muutoksia tekemään.

Lääkevarmennus tuli toteuttaa vain sellaisille tuotteille, joille on erikseen määritetty toimitettaessa suoritettava varmennus Marela-toiminnanohjausjärjestelmässä. Lääkevarmennus tuli suorittaa samalla, kun luetaan tuotteen 2D-koodi.

Koska keräilylaitetta käytetään tiiviisti muiden työtehtävien ja tietojärjestelmien rinnalla, joutuvat käyttäjät usein siirtymään erilaisten tehtävien ja toimintaympäristöjen välillä. Siirtymisen sujuvoittamiseksi keräilylaitesovelluksen on oltava johdonmukainen ja samankaltainen niin sovelluksen eri versioiden, kuin muidenkin järjestelmien kanssa. Keräilylaitesovelluksen muutosten suunnittelun ja toteutuksen periaatteita ovat sovelluksen käytön säilyminen johdonmukaisena ja sellaisena, etteivät muutokset muuta käyttöliittymää voimakkaasti aiemmasta versiosta.

Sovelluksen koherenttius piti ottaa huomioon lääkevarmennukseen liittyvissä muutoksissa. Se tarkoitti sitä, että esimerkiksi sovelluksen käyttäjälle antama

palaute tuli olla samoilla säännöillä annettu kuten sovelluksen muutkin palautteet.

5.2 Työkalut

Muutokset tehtiin jo olemassa olevaan sovellukseen, joten teknologiat olivat etukäteen määrätyt.

Keräilylaitesovelluksessa on käytetty JSF-teknologiaa, jonka versio on 2.2.6. Tästä syystä näkymien luomiseen käytetään Facelets-web-mallijärjestelmää. Sovelluslogiikka toteutettiin Java-ohjelmointikielellä. Näkymien luomiseen käytetyissä XHTML-tiedostoissa käytettiin Expression Languagea, jotta sovelluslogiikka saatiin yhdistettyä näkymien toimintaan.

Tietokantayhteys on toteutettu JPA:lla, jonka toteutuskirjastona käytetään Hibernatea. Oraclen relaatiotietokantahallintajärjestelmässä sijaitsevia tallennettuja proseduureja (engl. stored procedures) kutsutaan JDBC-tietokantaliittymärajapinnan avulla.

Ohjelmointityökaluiksi valikoitui jo muiden kehittäjien käytössä olevat työkalut, kuten Eclipse IDE Java-kehitystä varten. Versionhallintatyökaluksi asennettiin Subversion, jota käytetään keräilylaitesovelluksen lähdekoodin hallintaan. Versionhallintatyökalu mahdollistaa monen kehittäjän samanaikaisen työskentelyn, koska jokainen kehittäjä voi hakea sovelluksen lähdekoodin kopion yhteisestä tallennuspaikasta, tehdä muutoksia siihen omassa ohjelmointiympäristössään vaikuttamatta muiden kehittäjien työhön ja lisätä tekemänsä muutokset takaisin yhteiseen versioon. Muut kehittäjät voivat sen jälkeen halutessaan hakea uudemman version, joka sisältää tehdyt muutokset.

Sovelluksen käännöstyökaluna käytettiin Maven-rakentamisautomaatiotyökalua (engl. build automation tool). Mavenilla voidaan rakentaa, julkaista ja saattaa toimintavalmiiksi (engl. deploy) Java-sovelluksia sovelluspalvelimelle. Yleisin Maven-komento, jota tässä työssä tarvittiin, oli `mvn clean install`

`wildfly:deploy`, joka rakentaa ja paketoi sovelluksen WAR-tiedostoksi ja samalla saattaa toimintavalmiiksi Wildfly-sovelluspalvelimelle

5.3 Alkuvalmistelut

Aloituksen yhteydessä muutoskuvaus käytiin läpi toimeksiantajan kanssa sekä luotiin työskentely-ympäristö keräilylaitesovelluksen kehitystä varten.

Ennen ohjelmoinnin aloittamista tutustuttiin sovelluksen rakenteeseen ja arkkitehtuuriin. Keräilysovelluksen kansiorakenne on jaettu sovelluksen toimintojen, kuten keräilyn ja inventoinnin perusteella omiin moduuleihinsa. Nämä sisältävät moduulin nimen mukaisesti omat toimintonsa, eli esimerkiksi keräilymoduulissa on kaikki keräilyssä tarvittavat toiminnot.

Moduuleihin on luotu lähdekoodipakkauksia sovelluksen lähdekooditiedostojen järjestämistä varten. Esimerkiksi keräilymoduulin kansio src/main/java sisältää muun muassa keraily-, model- ja restClient-pakkaukset. Keräilypakkaus sisältää sovelluslogiikkaan liittyvät java-luokat, modelpakkaus pavut muun muassa näkymien ohjaamiseen ja restClient REST-pyynnöt REST-moduuliin. Näkymien luomiseen tarvittavat XHTML-tiedostot sijaitsevat kansiossa src/main/webapp. Tyylitiedostot, joilla määrätään keräilylaitesovelluksen tyylit, ovat yleensä saman kansiossa XHTML-tiedostojen kanssa, mutta keräilylaitesovelluksessa ne on viety omaan moduuliinsa, josta muut moduulit käyttävät niitä sen sijaan, että jokaisella moduulilla olisi omat tyylitiedostonsa.

Myös sovelluksen REST-rajapinta on oma moduulinsa. Tämä REST-moduuli käsittelee sovelluksen kommunikoinnin tietokannan kanssa JPA:a ja Hibernatea hyödyntämällä. Olio-relaatiomallinnusta varten luodut entiteettiluokat sijaitsevat REST-moduulissa. REST-moduuli sisältää lisäksi luokat relaatiotietokantahallintajärjestelmän kanssa kommunikaatioon eli ohjelmakutsut tietokantaan tallennettuihin proseduureihin.

5.4 Lääkevarmennus

Sovelluksen keräilyrivinäkymään piti tuoda tieto siitä, onko keräiltävä tuote varmennettava, jotta nähdään, onko tuotteen tunnistaminen tehtävä lukemalla keräilylaitteella pakkauksen 2D-koodi. Tämä tieto oli lisätty jo tietokantaan, ja sovellukseen tuli toteuttaa tiedon ORM-käsittely ja näkyvyys oikeassa kohtaa.

Tieto lisättiin aluksi tietokantataulun ja Java-olioiden välisen muunnoksen toteuttavaan `KeräilyriviResponse`-entiteettiluokkaan. Java-luokkaan on määritelty annotaatioilla miten muunnos suoritetaan. Ohjelmassa 1 on kuvattu uusi luokkamuuttuja `toimLaakevarmennus`. Annotitaatiot `@Column` ja `@Type` määrittävät, mihin tietokantataulun kolumniin muuttuja liittyy sekä tiedon tyypin.

Ohjelma 1. KeräilyriviResponse.java -tiedostoon lisätty tieto.

```
@Column(name = "TOIM_VARM_TUOTE_KER831")

@Type(type = "fi.affecto.kake.sys.kanta.BooleanKETyyppi")

private boolean toimLaakevarmennus = false;
```

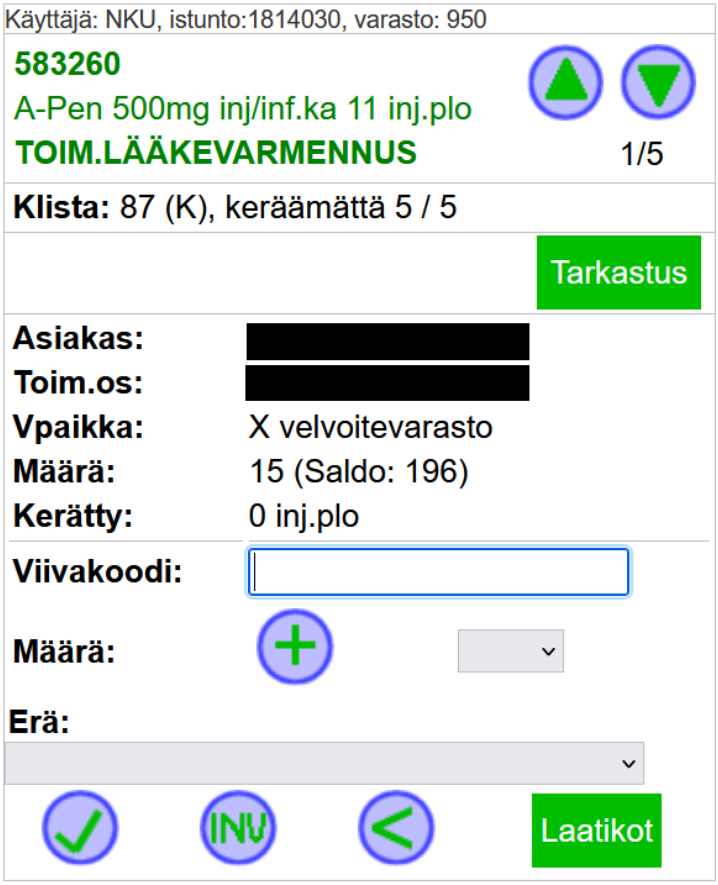
Entiteettiluokan lisäksi oli muokattava sitä vastaavaa `Keräilyrivi`-java-luokkaa, joka sijaitsee keräilymoduulin model-paketissa. Lisäämällä luokkaan vastaavan oliomuuttujan `toimLaakevarmennus` voidaan tietoa käyttää keräilyrivinäkymässä. Ohjelmassa 2 on ote keräilynäkymän muodostavasta XHTML-tiedostosta. Se osa koodia, joka on asetettu ristikkomerkin perään aaltosulkeiden väliin, on EL-ohjelmointikieltä. Koodin kohdalla `#{keräilynäyttö.rivi.toimLaakevarmennus}` luotiin viittaus luotuun keräilyrivi-luokan muuttujaan.

Ohjelma 2. Näkymään lisätty komponentti, joka näyttää ehdollisesti tiedon, onko tuote lääkevarmennettava toimitettaessa ja toinen komponentti, joka piilotetaan, mikäli edellinen tieto näytetään.

```
<h:outputText styleClass="otsikko"
```

```
            value="#{hoputteet['kake.keraily.toim_laakevarmennus']}"
            rendered="#{keräilynäyttö.rivi.toimLaakevarmennus}"/>
<h:outputText styleClass="otsikko"
            value="#{hoputteet['kake.keraily.eraseurattava']}"
            rendered="#{keräilynäyttö.rivi.toimLaakevarmennus ? false :
            keräilynäyttö.rivi.eräseuranta}"/>
```

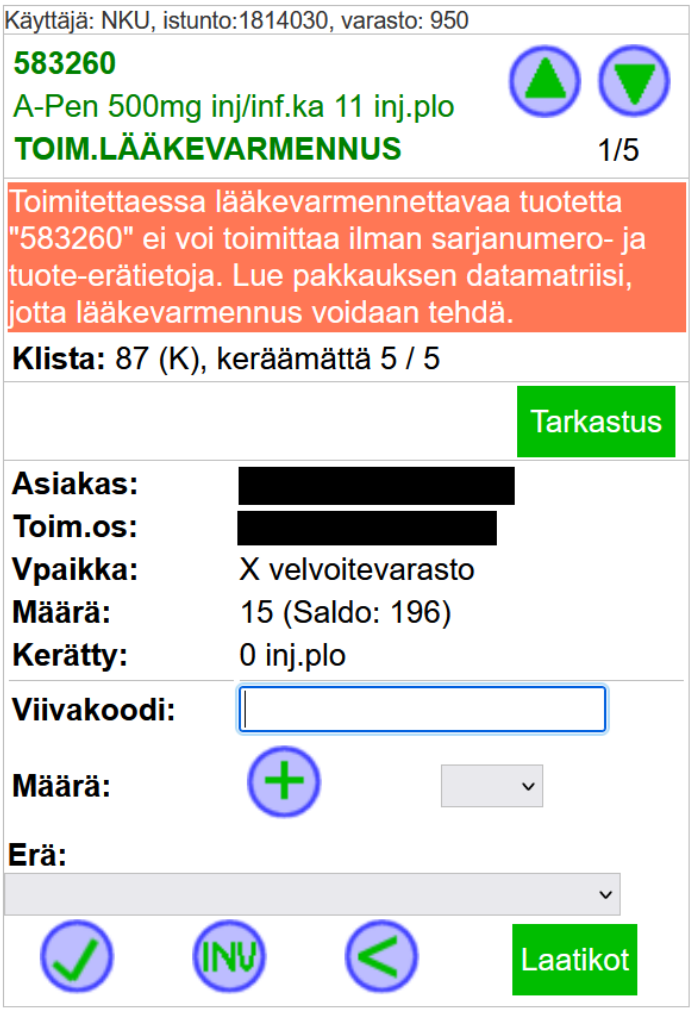
Kuvassa 5 on näkymä keräilyrivistä, jolla on toimitettaessa lääkevarmennettava tuote. Tuotenimen alla on tieto TOIM.LÄÄKEVARMENNUS indikoimassa toimitettaessa lääkevarmennettavaa tuotetta.



Kuva 5. Keräilylaitesovelluksen keräilyrivinäkymä, jossa näytetään käyttäjälle tieto, että tuote on toimitettaessa lääkevarmennettava.

Itse lääkevarmennus tapahtuu sovelluksen taustaohjelmassa eli tietokannan puolelle tallennettujen proseduurien avulla. Sovellukseen toteutettiin taustaohjelmasta palautuneiden arvojen käsittelyt ja niiden mukaisten ilmoitusten näyttäminen keräilynäkymässä.

Pelkille ilmoituksille oli jo rakenne sovelluslogiikassa, joten niiden toteutuksessa oli varmistuttava vain niiden toiminnasta. Sovellukseen on oma värikoodinsa erityyppisille ilmoituksille. Esimerkiksi seistyyppiset ilmoitukset näytetään omalla värillään, kuten kuvassa 6.



Kuva 6. Keräilylaitesovelluksen keräilyrivinäkymä, jossa näytetään käyttäjälle sovellusilmoitus. Tuotepakkaus on tunnistettu lukemalla jokin muu tunniste, kuin 2D-koodi, jolloin sovellus ilmoittaa asiasta seistyyppisellä ilmoituksella.

Käyttäjän erillistä kuittausta edellyttävät ilmoitukset toteutettiin samoin kuin sovelluksen muut vastaavat ilmoitukset. Kuitattavia ilmoituksia lisättiin kohtiin, joissa lääkeväärennösepäily oli havaittu joko 2D-koodia lukiessa tai kuitatessa koko keräilylistaa valmiiksi. Kuitattaville ilmoituksille on oma keräilyvahvistukset-luokka, joka sisältää boolean-arvot jokaisesta kuitattavasta ilmoitustyypistä. Luokkaan lisättiin oliomuuttujat `lääkeväärennösKuittaus` ja `käsittelyKertaValmisVirheKuittaus`. Jos keräilylaitesovellukseen tulee tieto lääkeväärennösepäilystä, asetetaan jompaankumpaan oliomuuttujaan arvoksi tosi riippuen siitä, missä tilanteessa lääkeväärennösepäily on havaittu.

Lääkevarmennustoiminnoille luotiin uusi java-luokka `Lääkevarmennustoiminnot`. Luokka sisältää metodin `lääkevarmennusepäilynKuittaksenVahvistus`, joka asettaa keräilyvahvistukset epätosiin ja siirtyy kuittausnäkymästä eteenpäin.

Ohjelma 3. Java-luokka lääkevarmennustoiminnoille.

```
@ViewAccessScoped
@Named
public class Lääkevarmennustoiminnot implements Serializable {
    private static final long serialVersionUID = 3803584321794775173L;

    @Inject
    Keräilytoiminnot keräilytoiminnot;
    @Inject
    Keräilyvahvistukset keräilyvahvistukset;

    public String lääkevarmennusepäilynKuittaksenVahvistus()
        throws JsonParseException, JsonMappingException, IOException {
           keräilytoiminnot.siirry(keräilytoiminnot.getHcrv(), false);
```

```
            keräilyvahvistukset.clear();

            return PageTargets.KERÄILY;

    }

}
```

Kuittausnäkymän luomiseksi keräilylaitesovelluksen Facelets-näkymäsivuihin tehtiin muutoksia. Keräily.xhtml-tiedosto sisältää varsinaisen keräilynäkymän. Sen rakenteeseen on sisällytetty `ui:include`-tagilla toinen XHTML-tiedosto, kerailyn_vahvistukset.xhtml. Keräilynäkymä piirretään normaalisti, mikäli kerailyvahvistukset-luokan muuttuja `vahvistuksia` on epätosi eli kuitattavia ilmoituksia ei ole. Muussa tapauksessa näkymään piirtyy se vahvistuksen vaativa ilmoitus, jonka kerailyvahvistukset-luokan muuttujan boolean-arvo on tosi. Ohjelmassa 4 nähdään lisätyt vahvistusilmoitukset. Ohjelmakoodin kohta `rendered` määrittää saamansa boolean-arvon perusteella, sisällytetäänkö kyseistä kuittausviestiä näkymään.

Ohjelma 4. Vahvistusviestit, jotka näytetään käyttäjälle, mikäli lääkeväärennösepäily on havaittu.

```
<!-- ***************************   -->
<!-- Lääkeväärennösepäilyn kuittausviesti -->
<ui:fragment rendered="#{keräilyvahvistukset.lääkeväärennösKuittaus}">
            <h:outputLabel
            value="#{hoputteet['kake.keraily.lvepaily_havaittu']}"
            styleClass="otsikko" />
            <kake:kehyksenValiviiva />
            <h:panelGrid columns="1"
            style="width:100%;text-align:center;">
                        <painike:kirjaa
```

```
                    action="#{lääkevarmennustoiminnot.lääkevarmennuse
          päilynKuittaksenVahvistus}" />

          </h:panelGrid>

</ui:fragment>

<!-- ***************************   -->

<!-- Tausta-ajossa ilmenneen lääkeväärennösepäilyn kuittausviesti -->

<ui:fragment

rendered="#{keräilyvahvistukset.käsittelykertaValmisVirheKuittaus}">

          <h:outputLabel

          value="#{hoputteet['kake.keraily.lvepaily_havaittu']}"

          styleClass="otsikko" />

          <kake:kehyksenValiviiva />

          <h:panelGrid columns="1" style="width:100%;text-
align:center;">

                    <painike:kirjaa

                    action="#{käsittelykerranVahvistustoiminnot.käsit
          telykerranKuittauksenVahvistus}" />

          </h:panelGrid>

</ui:fragment>
```

5.5 2D-koodin tietojen hyödyntäminen

Keräilylaitesovelluksen 2D-koodin lukemisen yhteyteen toteutettiin myös toimintoja, joilla 2D-koodista saatavia tietoja voitiin hyödyntää muutenkin kuin lääkevarmennuksen suorittamisessa ja saatiin toteutettua tarkempaa tiedon käyttöä. 2D-koodi sisältää pakkauskohtaisen sarjanumerotiedon lisäksi tarkan kestoajan, sekä valmistajan tuote-eränumeron. Näiden tietojen avulla

varastonhallintaa voidaan tarkentaa kerättyjen pakkauksien osalta. Varastosaldosta voidaan poistaa täsmälleen oikeaa tuote-erää, mikä ei ennen 2D-koodin olemassaoloa ollut mahdollista.

Talteen jäävän tuote-erätiedon ansiosta järjestelmään jää myös luotettava dokumentaatio toimitetusta erästä sen sijaan, että erä ratkaistaisi varastonohjauksessa käytettävin menetelmin, kuten first-in-first-out-menetelmällä (FIFO), jossa ensin saapunut erä kirjataan myös ensimmäisenä toimitetuksi keräillessä.

2D-koodin tietojen hyödyntämiseen liittyvä logiikka käsiteltiin pääosin PL/SQL-ohjelmointikielen avulla tietokantaan tallennetuin proseduurein. Sovellusilmoitukset näkyvät niiden tyypin mukaisella värillä ja ne kertovat käyttäjälle selkeästi ilmoitusaiheen.

6 Pohdinta

Opinnäytetyön tuloksena saatiin toimiva toimitettaessa varmennettavien lääkkeiden lääkevarmennuskokonaisuus, kun muutokset yhdistettiin taustajärjestelmän kanssa. Samalla toteutui myös 2D-koodin tietoja hyödyntäviä muutoksia, jotka lähinnä todettiin testauksella toimivaksi.

Vaikka itse lääkeväärennösdirektiivin ja sitä tarkentavan delegoidun asetuksen luomat muutostarpeet ovat olleet iso ja vaativa projekti kaikille lääkelogistiikan toimijoille lääketehtaasta apteekkiin, ovat lääkepakkauksiin lisätyt 2D-koodit ja yhteinen eurooppalainen lääkevarmennusjärjestelmä luonut suotuisan ympäristön lääkelogistiikan järjestelmien jatkokehitykselle. 2D-koodin sisältämiä tietoja, eli tuotekoodia, yksilöityä sarjanumeroa, eränumeroa ja viimeistä voimassaolopäivää voidaan hyödyntää monin tavoin, joista muutamaa tapaa on käsitelty tässä opinnäytetyössä. Uusia tapoja myös etsitään jatkuvasti lääkelogistiikan ohjelmistojen kehittäjien ja niitä käyttävien kesken.

Toteutuksen aikana tutustuttiin keräilylaitesovelluksen rakenteeseen ja samalla voitiin tarkastella, miten sovelluksen rakenteessa ilmenee malli-näkymä-ohjain- eli MVC-arkkitehtuuri. Koska Java ohjelmointikielenä ja JSF teknologiana olivat uusia työkaluja itselleni, oli aluksi hankala hahmottaa sovelluksen arkkitehtuuria ja toimintaa. Työn edetessä sain kuitenkin selkeän kuvan, miten sovellus toimii ja miksi se on rakennettu niin kuin on. Kun vertaillaan sovellusta MVC-arkkitehtuurin päätarkoitukseen, eli sovelluksen mallia, näkymiä ja ohjainta vastaavien ohjelmakoodien erillään pitämiseen, huomataan, että tämä toteutuu sovelluksessa. Keräilylaitesovelluksen tiedon tallennus- ja hakutoiminnallisuuden hoitavat entiteettiluokat ja niitä vastaavat keräilymoduulin java-luokat on sijoitettu erilleen muista, toiset kokonaan toiseen moduuliin, ja toiset omaan pakkaukseen. Mallin ja näkymän komponentit keskustelevat keskenään Expression Language -ohjelmointikielen kanssa, ja ovat siten erillään toisistaan. Sovelluslogiikka on myös omana ohjainkomponenttinaan, eikä siihen ole sekoitettu esimerkiksi mallin ohjelmakoodia.

Toteutuksessa huomioitiin keräilylaitesovelluksen käytettävyys pitämällä sovelluksen toiminnallisuus ja näkymä mahdollisimman samanlaisena kuten se aiemminkin on ollut. Esimerkiksi keräilyrivinäkymään ei lisätty erilaista näkymää lääkevarmennettaville lääkkeille, vaan tieto tuotiin samaan näkymään yhtenä otsikkona kohtaan, jossa on muutenkin tuotteen tuotetietoja. Holtzblattin ja Beyerin (2017, 307) tekstin sanoma, etteivät käyttäjät halua liian monimutkaisia tuotteita, heijastuu keräilylaitesovellukseen esimerkiksi sovelluksen yksinkertaisella käyttöliittymällä. Siinä on valikkopohjainen käyttöliittymä, eli sovellukseen kirjautuessa avautuu näkymä valikkoon, jossa näytetään yhden tason lista kaikista sovelluksen toiminnoista. Painikkeista päästään suoraan haluttuun toimintoon, eikä niissä ole käyttäjälle mahdollisesti monimutkaisia alivalikkoja.

Keräilylaitesovellukseen on helppo jatkossa toteuttaa lisää esimerkiksi lääkevarmennustoimintoja sitä varten luotuun java-luokkaan. 2D-koodiin mahtuu enemmän tietoa vaadittujen tunnistetietojen lisäksi. 2D-koodiin voidaan upottaa esimerkiksi verkko-osoite, joka osoittaa lääkkeen ulkonäköä näyttävään kuvaan tai lääketietoa antavaan verkkosivuun. Lisätietojen lisääminen 2D-koodiin vaatisi kuitenkin myös lääkeyritysten suunnalta muutoksia ja mahdollisesti yhteisen standardin luomista GS1-stadardin lisäksi. Tästä syystä tällainen jatkokehitysvaihtoehto ei ole kovin nopea tai helppo toteuttaa laajasti ja olemassa olevien 2D-koodin tietojen hyödyntäminen on järkevämpi suunta jatkokehitysideoille.

7 Yhteenveto

Opinnäytetyön tavoitteina oli toteuttaa toimitettaessa tapahtuva lääkevarmennus kannettavan keräilylaitteen sovellukseen. Tavoitteena oli myös hyödyntää lääkepakkauksen 2D-koodista saatavia erätietoja eri tavoin ja antaa käyttäjälle mahdollisia palauteviestejä sen mukaan.

Opinnäytetyössä tehtiin katsaus lääkelogistiikan ja lääkevarmennuksen takana olevan lainsäädännön aihealueisiin. Lisäksi käsiteltiin keräilylaitesovelluksessa käytettyjä teknologioita, ohjelmointiympäristöjä ja käyttöliittymän muutosten periaatteita. Toteutuksen aikana tutustuttiin muutoskuvaukseen eli määrittelyyn, työkaluihin, joita sovelluskehityksessä tarvittiin sekä sovelluksen rakenteeseen.

Opinnäytetyön tuloksena keräilylaitesovelluksella voidaan suorittaa lääkevarmennus ja lääkevarmennuksesta saadaan selkeät ilmoitukset. Sovellukseen lisätyt, 2D-koodia hyödyntävät toiminnot toimivat ja ovat muutoskuvauksen mukaisia. Opinnäytetyön muutokset toimitettiin asiakaskäyttöön muutosten valmistumisen jälkeen ja saatu käyttäjäpalaute oli positiivista.

Keräilylaitesovelluksen muutoksissa otettiin huomioon sovelluksen jatkokehitys, jotta esimerkiksi mahdolliset lääkevarmennustoimintoihin tehtävät lisäykset ovat helposti toteutettavia. Jatkossa keräilylaitesovellukseen voidaan toteuttaa lisää esimerkiksi kuittausviestejä ja ilmoituksia 2D-koodin tietojen perusteella.

Lähteet

Bucher, A., Rasi, G. & Senderovitz, T. 2018. Kirje sidosryhmille väärennettyjä lääkkeitä koskevan direktiivin 2011/62/EU mukaisten turvaominaisuuksien toteuttamisesta. Viitattu 20.2.2022. https://www.laakevarmennus.fi/sites/default/files/attachments/2018_letterstakeholders_safetyfeatures_fi.pdf

Cole, K.; McChesney, R. & Raszka, R. 2011. Advanced Java EE Development for Rational Application Developer 7.5 : Developers' Guidebook. Ketchum: MC Press. ProQuest Ebook Central. Saatavilla myös http://ebookcentral.proquest.com/lib/turkuamk-ebooks/detail.action?docID=739006

Connolly, T. 2014. Database Systems: a practical approach to design, implementation and management. 6., painos. Harlow: Pearson Education Limited. VLeBook. Saatavilla myös https://www.vlebooks.com/Product/Index/472772?page=0

EFPIA & Medicines for Europe 2017. European Pack Coding Guidelines. Versio 4. Viitattu 24.3.2022. https://www.medicinesforeurope.com/docs/European%20Pack%20Coding%20Guideline%20V4_0.pdf

EMVO 2022. European Medicines Verification System. Viitattu 31.5.2022. https://emvo-medicines.eu/mission/emvs/

Euroopan komissio n. d. Flasified medicines. Viitattu 28.2.2022. https://ec.europa.eu/health/medicinal-products/falsified-medicines_fi

Fimea 2012. Sairaala-apteekin ja lääkekeskuksen toiminta. Lääkealan turvallisuus- ja kehittämiskeskuksen määräys. Saatavilla myös https://www.fimea.fi/documents/160140/764653/22690_Maarays_6_2012.pdf

FiMVO 2022. Lääkevarmennusjärjestelmä. Viitattu 7.3.2022. https://www.laakevarmennus.fi/laakevarmennusjarjestelma

Finlex 2022. Lainkirjoittajan opas. Viitattu 21.2.2022. http://lainkirjoittaja.finlex.fi/6-euroopan-unionin-oikeus-osana-suomen-oikeusjarjestysta/6-3/

Finn-ID 2022. Opas onnellisempaan varastonhallintaan. Web-esite. Viitattu 26.5.2022. https://www.finn-id.fi/images/tiedostot/webesitteet/fi_opas_onnelliseen_varastonhallintaan_15-1.pdf

Gosling, J., Joy, B., Steele, G., Bracha, G., Buckley, A. & Smith, D. 2018. The Java® Language Specification. Oracle America, Inc. Saatavissa myös https://cr.openjdk.java.net/~iris/se/11/latestSpec/java-se-11-jls-draft-diffs.pdf

Graham Mansfield 2015a. v30 Introduction to JavaServer Faces (JSF). Viitattu 18.4.2022. https://youtu.be/GT-9c9E_fDE

Graham Mansfield 2015b. v31 MVC in JSF. Viitattu 18.4.2022. https://youtu.be/8GzAK8MmFqk

Heffelfinger, D. 2011. Java EE 6 Development with NetBeans 7. Olton: Packt Publishing, Limited. ProQuest Ebook Central. Saatavilla myös http://ebookcentral.proquest.com/lib/turkuamk-ebooks/detail.action?docID=948505

Holtzblatt, K. & Beyer, H. 2017. Contextual Design: Design for life. 2. painos. Amsterdam: Morgan Kaufmann (Interactive Technologies). Saatavissa myös https://search-ebscohost-com.ezproxy.turkuamk.fi/login.aspx?direct=true&db=nlebk&AN=1145291&site=ehost-live

Honeywell 2022. CK65 Series Mobile Computer powered by Android. Käyttöohje. Viitattu 26.5.2022. https://prod-edam.honeywell.com/content/dam/honeywell-edam/sps/ppr/en-us/public/products/mobile-computers/handheld-computers/ck65/documents/sps-ppr-ck65-a-en-ug.pdf?

Hookom, J. 2016. Facelets – JavaServer Faces View Definition Framework. Developer Documentation. Viitattu 26.5.2022. https://web.archive.org/web/20161231170917/https://facelets.java.net/nonav/docs/dev/docbook.html#intro

Junttonen, A. 2017. Lääkkeen toimitusketju vilisee eri toimijoita. Sic! -lehti 3-4/2017. Viitattu 23.5.2022. https://sic.fimea.fi/verkkolehdet/2017/3-4_2017/laakkeiden-saatavuus-ja-laakevaarennokset/laakkeen-tuotantoketju-vilisee-eri-toimijoita

Komission delegoitu asetus (EU) 2016/161. https://ec.europa.eu/health/system/files/2016-11/reg_2016_161_fi_0.pdf

Kristály, D.M.; Crăciun, A.V.; Pelcz & A. & Trican I. 2005. MVC Architecture in web applications development. Proceedings of the 14th ELECTRONICS. Sozopol: Tech. Univ. of Sofia & Tech. Univ. of Delft, Sofia. Saatavilla myös https://ecad.tu-sofia.bg/et/2005/pdf/Paper138-D_Kristaly.pdf

Kulkarni, R. 2018. Java EE 8 Development with Eclipse : Develop, Test, and Troubleshoot Java Enterprise Applications Rapidly with Eclipse. 3., painos. Birmingham: Packt Publishing, Limited. ProQuest Ebook Central. Saatavilla myös http://ebookcentral.proquest.com/lib/turkuamk-ebooks/detail.action?docID=5446039

Leonard, A. 2014. Mastering JavaServer Faces 2.2. Olton Birmingham: Packt Publishing, Limited. ProQuest Ebook Central. Saatavilla myös http://ebookcentral.proquest.com/lib/turkuamk-ebooks/detail.action?docID=1674863

Linnolahti, J. 2017. Lääkepakkausten uusilla turvaominaisuuksilla suojaudutaan lääkeväärennöksiä vastaan. Sic! -lehti 3-4/2017. Viitattu 7.3.2022. https://sic.fimea.fi/verkkolehdet/2017/3-4_2017/laakkeiden-saatavuus-ja-laakevaarennokset/laakepakkausten-uusilla-turvaominaisuuksilla-suojaudutaan-laakevaarennoksia-vastaan

Linnolahti, J. & Kankkunen, T. 2018. Lääkepakkausten uudet turvaominaisuudet käyttöön helmikuussa 2019. Sic! -lehti 4/2018. Viitattu 8.3.2022. https://sic.fimea.fi/verkkolehdet/2018/4_2018/palstat/laakepakkausten-uudet-turvaominaisuudet-kayttoon-helmikuussa-2019

Logistiikan Maailma 2022a. Logistiikka. Viitattu 10.5.2022. https://www.logistiikanmaailma.fi/logistiikka/

Logistiikan Maailma 2022b. Sisälogistiikka (Intralogistics). Viitattu 11.5.2022. https://www.logistiikanmaailma.fi/logistiikka/logistiikka-ja-toimitusketju/sisalogistiikka/

Logistiikan Maailma 2022c. Toiminnanohjausjärjestelmä. Viitattu 11.5.2022. https://www.logistiikanmaailma.fi/logistiikka/ohjausjarjestelmat/toiminnanohjausjarjestelma/

Logistiikan Maailma 2022d. Varastonhallintajärjestelmät. Viitattu 22.5.2022. https://www.logistiikanmaailma.fi/logistiikka/ohjausjarjestelmat/varastonhallintajarjestelmat/

Logistiikan Maailma 2022e. Ohjausjärjestelmät. Viitattu 22.5.2022. https://www.logistiikanmaailma.fi/logistiikka/ohjausjarjestelmat/

Lääkelaki 23.2.2019/208. Annettu Helsingissä 22.2.2019. Saatavilla https://www.finlex.fi/fi/laki/ajantasa/1987/19870395

Lääkeväärennösdirektiivi 2011/62/EU. Euroopan parlamentin ja neuvoston direktiivi. Euroopan unionin virallinen lehti 1.7.2011. Viitattu 23.2.2022. https://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2011:174:0074:0087:FI:PDF

Oracle 2022a. Java Documentation. The Java™ Tutorials. Viitattu 30.3.2022. https://docs.oracle.com/javase/tutorial/getStarted/intro/definition.html

Oracle 2022b. JavaServer Faces Technology. Viitattu 17.4.2022. https://www.oracle.com/java/technologies/javaserverfaces.html

Oracle 2022c. Help Center. Database Concepts. Viitattu 27.4.2022 https://docs.oracle.com/cd/E11882_01/server.112/e40540/intro.htm#CNCPT001

Paaskoski, S. 2017. Kansainvälisen lääkerikollisuuden vaikutukset näkyvät myös Suomessa. Sic! -lehti 3-4/2017. Viitattu 23.2.2022. https://sic.fimea.fi/verkkolehdet/2017/3-4_2017/laakkeiden-saatavuus-ja-laakevaarennokset/kansainvalisen-laakerikollisuuden-vaikutukset-nakyvat-myos-suomessa

Paaskoski, S., Linnolahti, J. & Luhtanen, P. 2020. Lääkkeiden saatavuus poikkeuskeväänä 2020. Sic! -lehti 3/2020. Viitattu 14.2.2022. https://sic.fimea.fi/arkisto/2020/3_2020/laakkeiden-saatavuus-ja-covid-19/laakkeiden-saatavuus-poikkeuskevaana-2020

Rannanheimo, Piia., Salminen, E. & Kuivamäki, M. 2020. Fimea jalosti lääkealan datan terveyden huollonvarautumissuunnitelmia varten. Sic!-lehti 3/2020. Viitattu 14.2.2022. https://sic.fimea.fi/arkisto/2020/3_2020/laakkeiden-saatavuus-ja-covid-19/fimea-jalosti-laakealan-datan-terveydenhuollonvarautumissuunnitelmia-varten

Sosiaali- ja terveysministeriö 2022. Lääkehuolto. Viitattu 6.3.2022. https://stm.fi/lainsaadanto/laakehuolto

Quest Software Inc. 2017. SQL Navigator. Viitattu 27.4.2022. https://www.quest.com/documents/sql-navigator-datasheet-75424.pdf