Olli Kolkki, Juuso Lahtinen, Severi Reivinen

Varastonhallintaohjelmisto

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tieto- ja viestintäteknologian tutkinto-ohjelma

Toteutusdokumentti

16.12.2021

Sisällys

1 Johdanto

Toteutusdokumentin tarkoitus on kertoa ohjelman toiminnasta ja toteutuksesta, sekä antaa ohjeet siihen, miten ohjelmaa käytetään. Aluksi dokumentissa esitellään käyttäjäroolit, käyttötapaukset ja lähtökohdat, jonka jälkeen käydään läpi varastonhallintaohjelmaa ja erityisesti sitä, mitä hyötyjä se tuo ja miten se tulee rakentaa yrityksien käyttöön sopivaksi. Dokumentissa kuvataan myös projektin kehitysprosessi, käytetyt ohjelmistot ja tekniikat. Dokumentin lopussa on ohjelman käyttöohje.

Varastonhallintasovellus on tarkoitettu helpottamaan verkkokauppapohjaisia yrityksiä. Sovelluksen avulla yritykset pystyvät hallitsemaan verkkokauppaan tulleita tilauksia sekä verkkokaupassa olevia tuotteita.

2 Tuotteen vaatimukset

Sovelluksen avulla verkkokauppaan pystytään luomaan uusia tuotteita, muokkaamaan vanhoja tuotteita sekä tarkastelemaan ja käsittelemään tilauksia. Lisäksi ohjelma mahdollistaa työvuorojen suunnittelun ja tarkastelun. Ohjelman asu on yksinkertainen, mutta ohjelma pyrkii tarjoamaan kaikki välttämättömät ominaisuudet verkkokaupan hallintaan.

Sovellus tarjoaa mahdollisuuden muokata kaikkien luotujen tuotteiden tietoja, kuten hintaa, kuvausta ja varastopaikkaa. Tuotteita voi myös hakea erilaisten parametrien mukaan. Sovellukseen on mahdollista luoda käyttäjiä sekä työntekijöille sekä esimiehille. Työntekijät pystyvät selaamaan työvuorolistojansa ja esimiehillä on mahdollisuus lisätä sekä poistaa työntekijöiden työvuoroja.

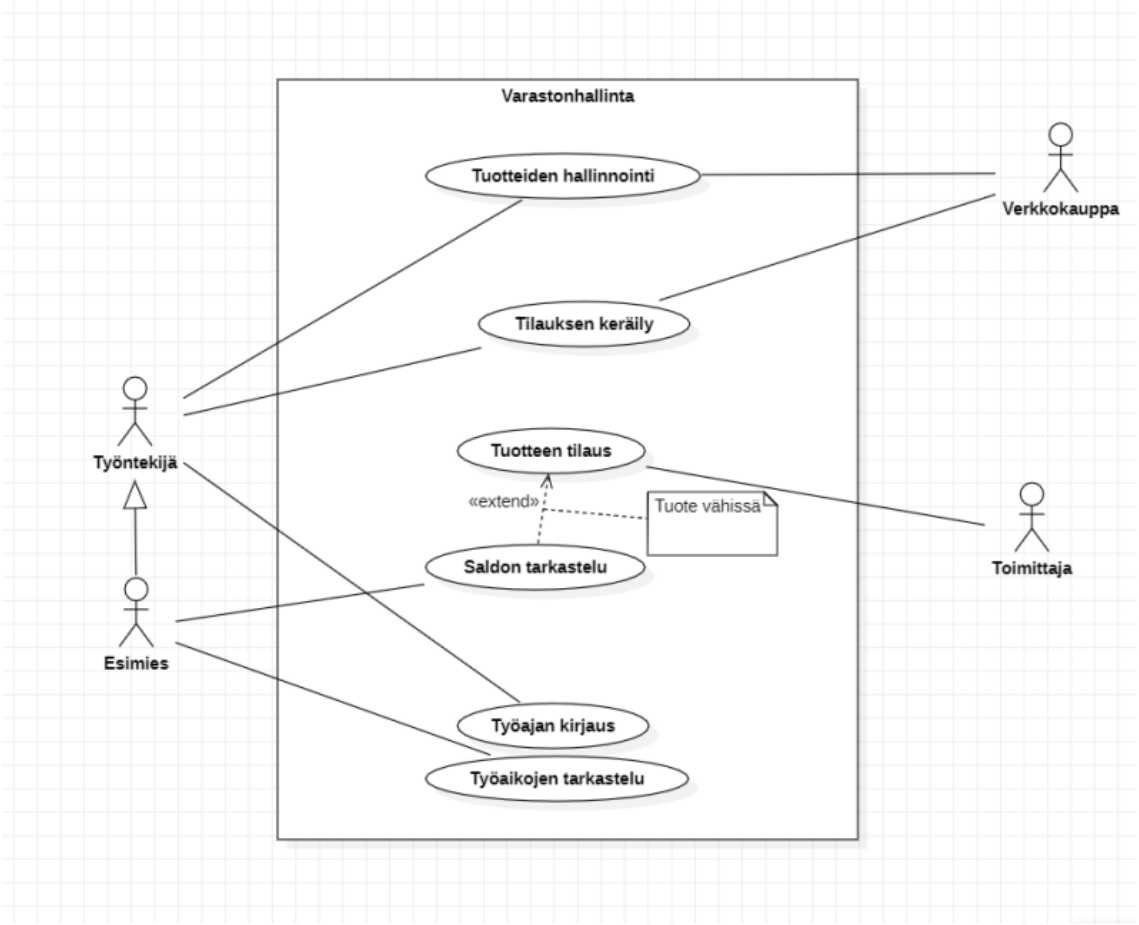
Tilausten hallinta -ikkunan avulla voi tarkastella verkkokauppaan saapuneita tilauksia sekä jo käsitellyt tilaukset. Jokaista tilausta voidaan tarkastella erikseen, jolloin saadaan yksityiskohtaisempia tietoja tilauksesta, kuten tilauksen

sisältämät tuotteet ja tilaajan tiedot. Tilaus voidaan merkitä käsitellyksi, jolloin tilauksen sisältämien tuotteiden määrä vähennetään varaston saldosta.

Sovelluksen avulla pystyy hallitsemaan suurta tietomäärää varaston tarpeen mukaan. Mahdollisista virhetilanteista ilmoitetaan käyttäjälle yksityiskohtaisesti.

3 Käyttäjäroolit ja käyttötapaukset

Sovelluksella on pääasiassa neljä erilaista käyttäjää (aktoria). Alla olevassa kaaviossa nähdään visuaalisesti mitä kukin käyttäjä tekee.



Kuva 1 Varastonhallintasovelluksen käyttötapauskaavio

Varastossa työskentelevä työntekijä käyttää järjestelmää tuotteiden sekä tilausten hallinnointiin. Lisäksi työntekijällä on mahdollisuus tarkastella tulevia sekä menneitä työvuorojaan.

Varaston esimies tekee samoja asioita mitä työntekijä tekee sovelluksella, mutta lisäksi esimies voi korjata tuotteiden saldoja ja hallinnoida työntekijöiden työaikoja.

Verkkokauppa lähettää varastolle tilauksia, joka käsitellään varaston järjestelmän sisällä.

Toimittaja, joka on tietolähde tuotteille.

Aiemmin mainitussa kaaviossa nähdään myös eri käyttötapauksia. Työntekijän käyttötapauksiin kuuluu tuotteiden hallinnointi, tilausten keräily sekä työajan kirjaaminen. Esimies tarkastelee saldoa, ja tilaa tuotetta tarvittaessa lisää, jos tuote on vähissä tai loppunut. Esimies voi myös luoda, poistaa sekä tarkastella työaikoja.

Aktoreilla, jotka eivät sinänsä näy ohjelmistossa (verkkokauppa ja toimittaja), on myös käyttötapaukset sovelluksessa. Verkkokauppa on ns. sovelluksen tilaaja, eli käyttää sovellusta oman verkkokauppansa tuotteiden hallinnoinnissa ja tilauksen keräilyssä. Toimittaja taas toimittaa tuotteita varastoon.

4 Käsitteet, määritelmät ja ohjelmiston tietomalli

Projekti toteutettiin perinteisen ohjelmistokehityksen vaiheiden mukaisesti (määrittely, suunnittelu, toteutus, testaus ja käyttöönotto) siinä määrin kuin se katsottiin hyödylliseksi projektille.

4.1 Tekniset käsitteet

Tekniset vaatimukset varastonhallintasovellukselle asetettiin seuraavasti: ohjelmiston tulisi olla helppokäyttöinen, sovelluksessa tulee käyttää hyväksi todettuja teknologioita - Java -ohjelmointikieli ja PostgreSQL objekti-relaatiotietokanta. Sovelluksessa hyödynnetään eri olio-ohjelmoinnin suunnittelumalleja ja MVC-arkkitehtuuria.

4.2 Kehitysympäristö

Eclipse

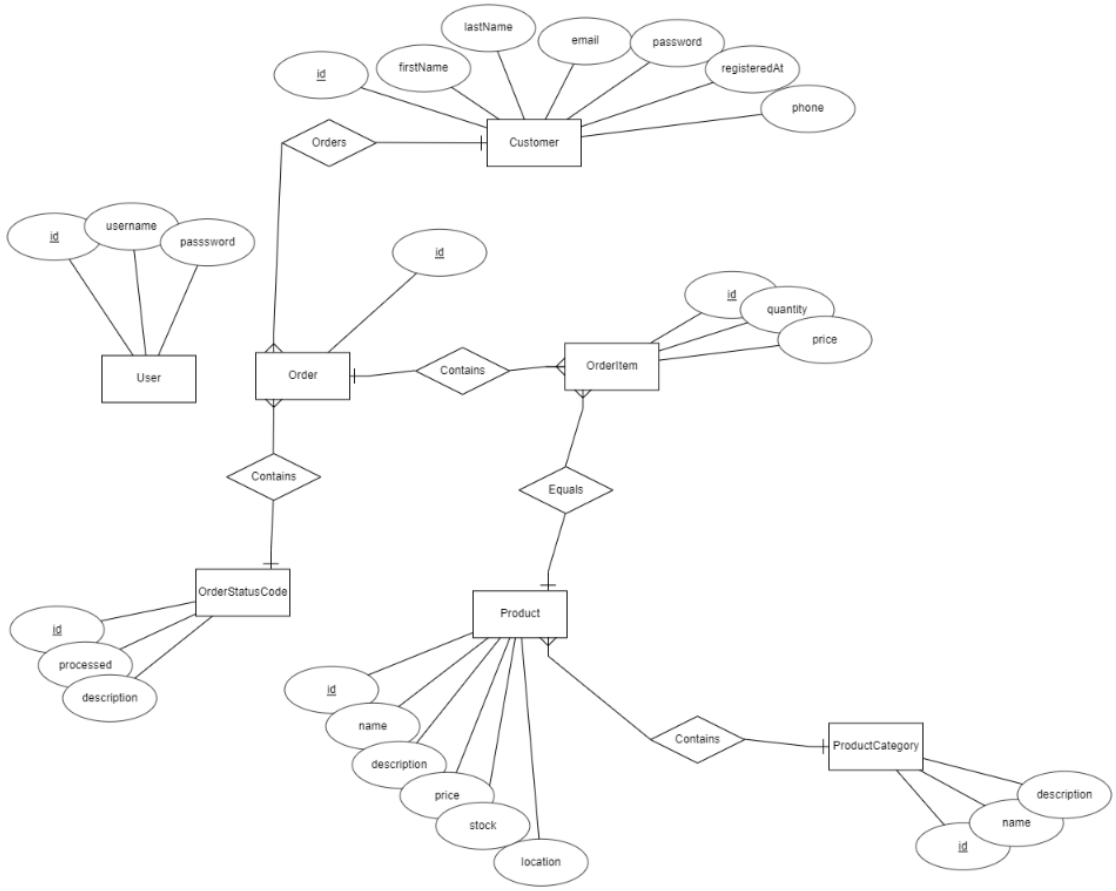
Ohjelmointiympäristönä käytettiin Eclipseä, joka helpottaa ja nopeuttaa Java -sovellusten kehitystä. Eclipse:ssa on lukuisia toimintoja, jotka auttavat monissa eri tapauksissa.

Maven

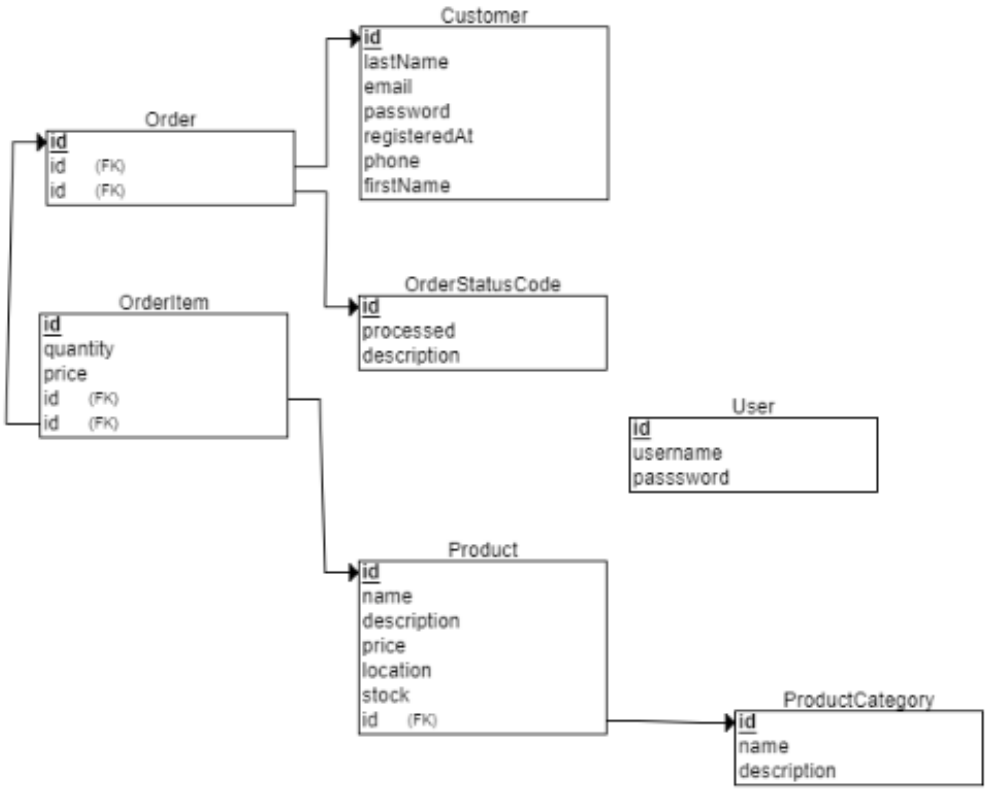
Maven on Apache kehittämä Java-sovellusten rakentamiseen ja ylläpitämiseen helpottava työkalu. Maven helpottaa ylläpitämään sovelluksen riippuvuuksia. Maven-projektiin liittyy POM-tiedosto, joka hallitsee rakennetta sekä kertoo tietoja projektista. POM:iin lisätään projektin perustiedot, riippuvuudet ja resurssit.

4.3 Sovelluksen tietomalli

Järjestelmästä voi nähdä mitä tuotteita varastossa on. Tuotteeseen on liitetty tietoja, kuten varastosaldo ja hyllypaikka. Järjestelmään kirjataan tilauksen kulku tuotteiden vastaanotosta, käsittelyyn ja toimitukseen asiakkaalle.



Kuva 2 Sovelluksen tietomalli (ER-malli)

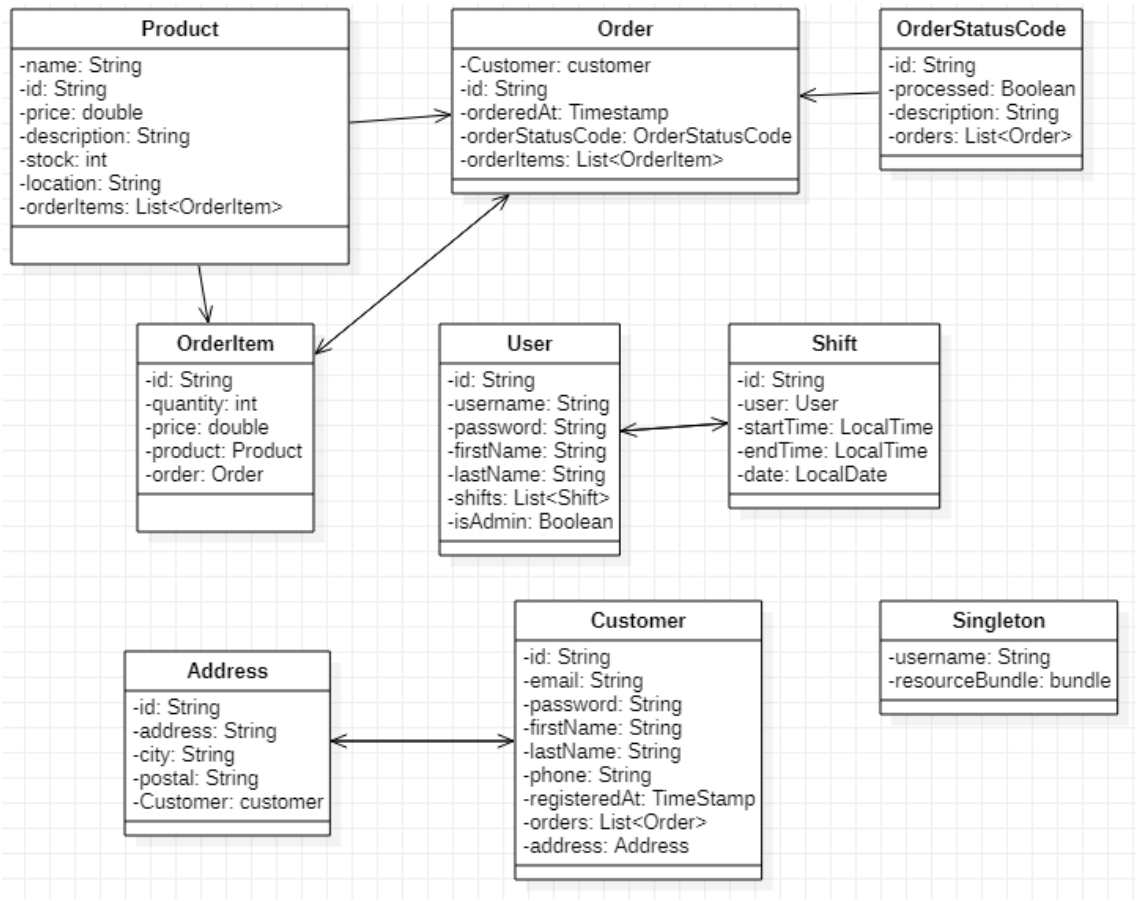


Kuva 3 Sovelluksen tietomalli (relaatiotietokantakaavio)

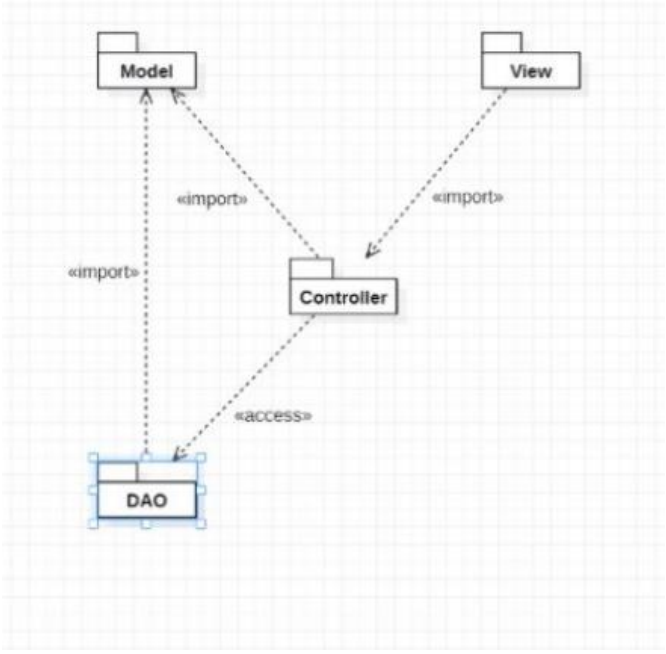
Kuvat 2 ja 3 visualisoivat tietokannan rakennetta.

5 Ohjelmiston rakenne

Sovellus on suunniteltu noudattamaan MVC-mallia (View, Model, Controller). Näkymät, mallit sekä näkymien kontrollerit on jaettu erilleen toisistaan omiin pakkauksiinsa. Näin sovelluksen rakenne pysyy siistinä ja ymmärrettävyys on helpompaa.



Kuva 4 Varastonhallintasovelluksen model-pakkauksen luokkakaavio



Kuva 5 Varastonhallintasovelluksen pakkauskaavio

Pakkauskaavio sisältää neljä pakkausta, joihin on asetettu näihin kuuluvat luokat.

Model – pakkauksen sisältö

- **Product** on tuote, joita verkkokauppa sisältää. Tuotteen instanssimuuttujat sisältävät konkreettiset tiedot tuotteesta kuten nimen, hinnan ja lisätiedot. Lisäksi malli sisältää tiedon varastopaikasta sekä varastosaldosta.

- **Customer** on verkkokaupan asiakas, joka tekee tilauksia verkkokaupasta. Asiakas sisältää välttämättömät tiedot tilauksen tehneestä henkilöstä, kuten nimen, sähköpostiosoitteen sekä puhelinnumeron.

- **Address** sisältää tiedon osoitteesta, ja kenelle se kuuluu. Osoitteeseen kuuluu lähiosoite, postinumero, postitoimipaikka sekä viite asiakkaaseen.

- **OrderItem** on tilauksen sisältämä tuote. OrderItem sisältää viitteen Product-luokkaan ja saa sitä kautta tiedon tilatusta tuotteesta. Erona on kuitenkin se, että OrderItemin ominaisuuksiin kuuluu myös tieto tilatun tuotteen määrästä, yhteishinnasta sekä viite tilaukseen, johon kyseinen OrderItem kuuluu.

- **Order** on asiakkaan tekemä tilaus verkkokaupasta. Tilaus koostuu aiemmin mainituista OrderItemeistä. Order sisältää myös tiedon tilauksen tehneestä asiakkaasta, tilauspäivämäärän sekä tiedon siitä, onko tilaus käsitelty varastolla.

- **OrderStatusCode** on malli, jota tilaus käyttää määrittääkseen tiedon tilauksen tilasta. Tällä hetkellä mahdolliset tilat ovat käsitelty ja käsittelemätön, mutta tulevaisuudessa näitä tiloja voisi olla lisää.

- **ProductCategory** mahdollistaa tuotteiden (Product) kategorioimisen varastolla. Malli sisältää kategorian nimen sekä lisätietoa kategoriasta. Jokaisen varastossa olevan tuotteen voi kategorioida omaan

kategoriaansa. Esimerkiksi housut voitaisiin asettaa alaosa – kategoriaan ja paidat ja hupparit yläosa – kategoriaan. Ominaisuus tullaan kehittämään myöhemmin.

- **User** on varastonhallintasovelluksen käyttäjä. Tällä hetkellä malli sisältää vain käyttäjätunnuksen sekä salasanan, mutta tulevaisuudessa tämä malli sisältää laajempaa tietoa sovelluksen käyttäjästä. Lisäksi käyttäjälle tullaan tulevaisuudessa asettamaan rooli (esim. työntekijä, esimies), joka määrittää sovelluksen käyttäjän käyttöoikeudet.

- **Shift** on sovelluksen käyttäjän työvuoro. Jokaisella käyttäjällä on lista, johon lisätään tai josta poistetaan työvuoro-olioita. Shift-olio sisältää tiedon käyttäjästä, aloitusajasta, lopetusajasta sekä päivämäärästä.

- **Singleton** on apuna käyttäjänimen ja locale-tiedoston säilyttämiseksi, jota hyödynnetään muissa luokissa. Esim. locale-tiedostoa on käytettävä kaikissa sovelluksen sceneissä, jotta sovelluksen kieli on oikein kaikkialla.

DAO – pakkauksen sisältö

- **OrderAccessObject** mahdollistaa tietokantakutsut liittyen tilausluokkaan (Order).

- **OrderItemAccessObject** mahdollistaa tietokantakutsut liittyen tilauksen tuotteen luokkaan (**OrderItem**).

- **ProductAccessObject** mahdollistaa tietokantakutsut liittyen verkkokaupan tuote luokkaan (**Product**).

- **ProductCategoryAccessObject** mahdollistaa tietokantakutsut liittyen tuotteen kategorioimiseen (**ProductCategory**).

- **UserAccessObject** mahdollistaa tietokantakutsut sovelluksen käyttäjiin liittyvään luokkaan (**User**).

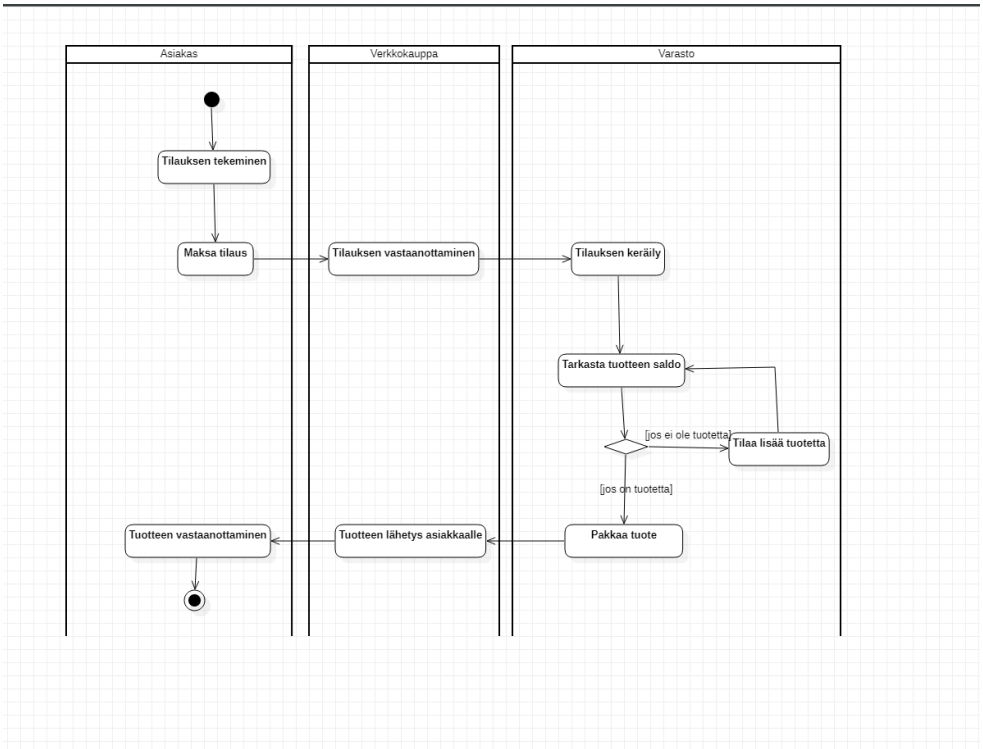
- **ShiftAccessObject** mahdollistaa tietokantakutsut sovelluksen käyttäjien (**User**) työvuoroihin (**Shift**) liittyen.

Controller – pakkaus sisältää sovelluksen kontrollerit. MVC – mallissa näkymät on eritelty sovelluksen toiminnallisuudesta ja näkymiä voidaan hallita kontrollereiden avulla. Jokaiselle näkymälle on luotu oma kontrollerinsa, jonka avulla näkymässä tapahtuvat toiminnallisuutta vaativat operaatiot määritellään.

View – pakkaus sisältää sovelluksen näkymät. Näkymä tarkoittaa käytännössä sovelluksen käyttäjän näkemää ikkunaa sovellusta käyttäessä. Esimerkiksi sovellukseen kirjautuminen on oma ikkunansa, mistä siirrytään seuraavaan käyttäjän syöttäessä oikeat tunnistautumistiedot.

6 Ohjelmiston toiminta

Ohjelmisto helpottaa varaston toimintaa toimimalla tuotteiden ja tilausten läpikulun kirjanpitovälineenä.



Kuva 6. Varastonhallintasovelluksen aktiviteettikaavio

Tilanne alkaa asiakkaan tekemästä tilauksesta. Asiakas-uimaradalla on asiakkaan tarpeelliset aktiviteetit tuotteen saamiseksi. Verkkokauppa-uimaradalla vastaanotetaan ja lähetetään tilaus, joka toteutuu varasto-uimaradalla olevissa aktiviteettikohdissa. Seuraavassa listassa on kuvattu ohjelmistomme toiminnan kannalta aktiviteettikaavion olennaisimmat osat.

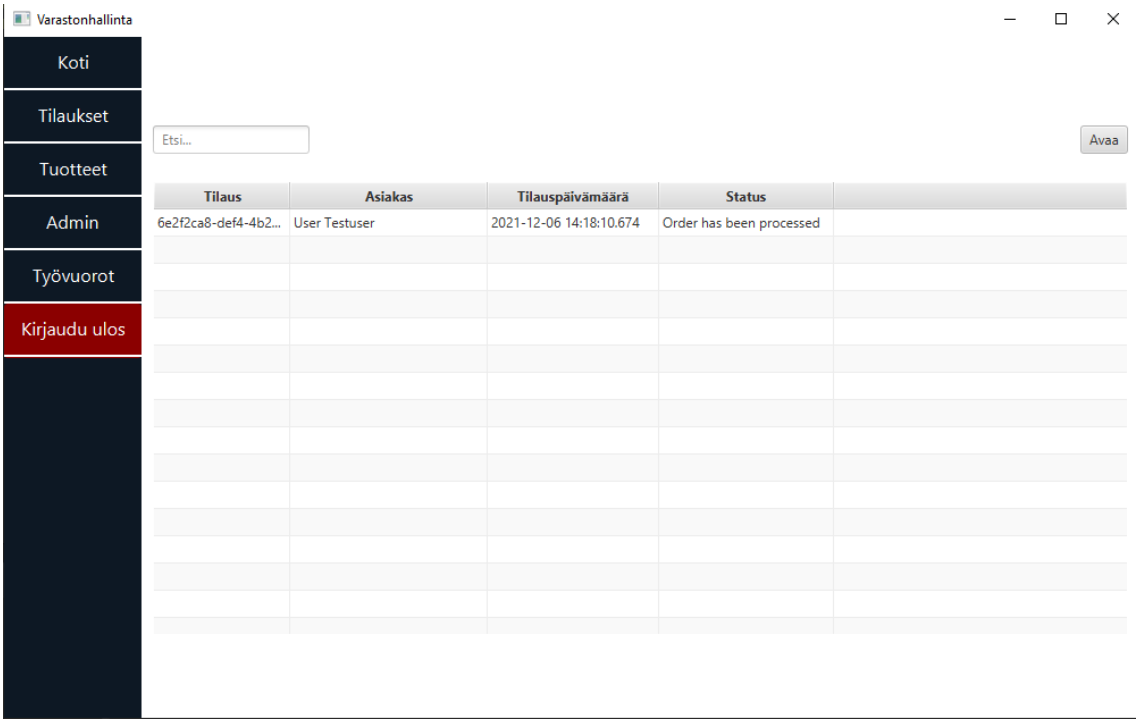
- **Tilauksen tekeminen** Asiakas tekee tilauksen verkkokaupan nettisivulla.

- **Tilauksen vastaanottaminen** Verkkokauppa saa tilauksen, ja lähettää keräilypyynnön varastoon. Tässä vaiheessa tilaus saapuu varastonhallintaohjelmaan.

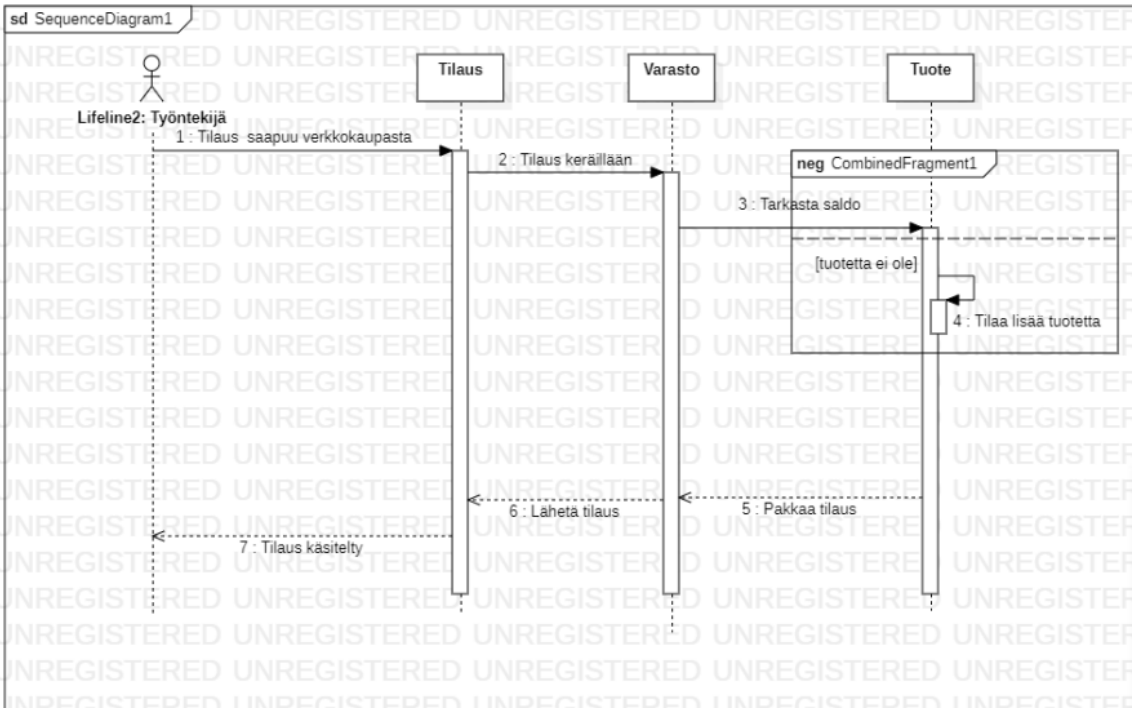
- **Tilauksen keräily** Työntekijä tarkastaa tilattujen tuotteiden tiedot järjestelmästä ja keräilee ne.

- **Tarkasta tuotteen saldo** Työntekijä varmistaa, että tuotteen saldo on riittävä, ja tilaa tuotetta tarvittaessa lisää.

- **Tuotteiden pakkaus ja lähetys** Tässä vaiheessa tilaus lähetetään varastolta ja merkitään ohjelmaan käsitellyksi



Kuva 7. Tilausnäkymä



Kuva 8. Varastonhallintasovelluksen sekvenssikaavio

Sekvenssikaavion käyttötapauksessa tarkastellaan myös tilauksen käsittelyä työntekijän näkökulmasta. Tapaus lähtee liikkeelle verkkokaupasta saadusta tilauksesta, jonka työntekijät keräilevät. Jos tuotteiden saldo ei riitä, tuotteita tilataan lisää. Jos saldo on riittävä ja keräily onnistuu, niin tuotteet pakataan ja tilaus lähetetään. Tilaus merkitään lopuksi käsitellyksi.

Tilausten hallintanäkymässä työntekijä voi tarkastella tilausten tietoja, kuten lähetysosoitetta, tai tilauksella olevia tuotteita. Tilauksen prosessointi, eli sen merkitseminen tehdyksi, vaatii seuraavien kriteereiden täyttymistä: tilaus ei saa olla jo prosessoitu, ja varaston saldon on oltava yhtä suuri tai suurempi, kuin tilauksella olevien tuotteiden määrä.

Tilausten hallinnan lisäksi sovelluksessa voi hallinnoida varaston tuotteita. Käyttäjä voi lisätä, poistaa ja muokata niitä. Esimerkiksi tuotteen saldoa voi vähentää, tai varastopaikkaa muuttaa.

Sovelluksella esimies (admin-käyttäjä) voi luoda ja poistaa työvuoroja eri työntekijöille ”Admin” -ikkunassa. Työntekijät voivat sitten tarkastella omia työvuorojaan ”Työvuorot” -ikkunan taulukkonäkymässä.

7 Kehitysprosessi ja kehitysvaiheen tekniikat

Projekti aloitettiin kartoittamalla mitä toimintoja varastonhallintajärjestelmän ensimmäisen versioon tarvitaan. Sen lisäksi kartoitettiin, mitä tietoja tallennetaan tietokantaan. Varastonhallintajärjestelmältä vaadittiin seuraavat toiminnot:

- Tuotteiden lisäys, muokkaus, poisto
- Käyttäjäystävällinen ulkoasu
- Tilauksien hallinta

- Työaikojen hallinta

7.1 Ensitoimet tilauksien hallintaan

Sovelluksen ensimmäisessä versiossa huomio kohdistettiin verkkokaupan tuotteiden käsittelyyn ja tilauksien hallintaan. Toimivuuden takaamiseksi taustalle kehitettiin mock-tyylinen verkkokauppa, jossa verkkokaupan asiakas pystyy selaamaan tuotteita, lisäämään näitä ostoskoriin ja lopulta lähettämään tilauksen. Varastonhallintasovellus vastaa jälkitoimista, joita tilauksen käsittely vaatii.

7.2 Verkkokaupan varasto ja tuotteet

Verkkokaupan tuotteiden vaihtuessa jatkuvasti on sovelluksessa pyritty yksinkertaistamaan näiden hallinta. Sovelluksen käyttäjällä on mahdollisuus lisätä uusia tuotteita, muokata jo varastossa olevia sekä poistaa tuotteita. Näitä muutoksia tehdessä varaston tuotevalikoima sekä verkkokaupan sisältö päivittyvät reaaliajassa.

7.3 Työaikojen hallinta työntekijöille

Uusimpana ominaisuutena sovellukseen lisättiin mahdollisuus työaikojen hallinnointiin. Järjestelmään lisätyt esimiehet pystyvät suunnittelemaan varaston työntekijöiden työvuorolistoja. Esimiehillä on mahdollisuus luoda, poistaa sekä tarkastella yksityiskohtaisesti jokaisen työntekijän työvuoroja. Työntekijät pystyvät sovelluksen sisällä tarkastelemaan omia tulevia sekä menneitä työvuorojaan.

7.4 Käyttöliittymäsuunnittelu ja toteutus

Käyttöliittymän suunnittelussa pyrittiin ottamaan huomioon visuaaliset seikat, yksinkertaisuus sekä toimintojen käytettävyys. Ainoat vaatimukset käyttöliittymälle olivat käyttäjäystävällinen ulkoasu sekä helppo opittavuus. Piirrettiin

alustavia kuvia eri näkymistä Figma:ssa. Näkymät toteutettiin Java-FX hyödyntäen SceneBuilder –ohjelmassa.

Järjestelmä oli tarkoitus toteuttaa itsenäisenä sovelluksena. Sovelluksen ohjelmointikieleksi valittiin Java.

Sovelluksen toteuttamiseen on käytetty Scrum -ketterää menetelmää, ja sprinttien suunnittelun apuna projektinhallintasovellus Nektionia. Nektioniin on kirjattu sovelluksen ominaisuuksia ns. käyttäjätarinoina ja näitä ominaisuuksia on toteutettu sprinttien aikana. Kaikkia ominaisuuksia ei ole vielä tässä vaiheessa toteutettu.

Ohjelmointiympäristönä on käytetty Eclipseä, josta luotu Maven-projekti. Projektissa on käytetty eri liitännäisiä: JavaFX (SceneBuilder), JUnit, PostgreSQL ja Hibernate. Versionhallintana käytettiin GitLabia, ja tulevaisuutta varten asennettiin jatkuvan kehityksen työkalu Jenkins.

Sovelluksen testaukseen käytettiin JUnit-kirjastoa. Testejä tehtiin tietokannan apuluokille, joissa testataan, ettei tietokantaan voi syöttää virheellistä tietoa tai duplikaatteja.

7.5 Testaus

Sovelluksen testaus tehtiin pääosin manuaalisesti, mutta JUnit-testit ovat olemassa tuotteiden ja käyttäjien tietokantaluokille. Testeissä tarkistetaan, että tietokantaan syötetty ja sieltä haettu data vastaa todellisia syötteitä.

Testaus tehtiin pääosin manuaalisesti, ”trial & error” -tyyppisesti tekemällä koodimuutosten jälkeen toistuvia yrityksiä, kunnes haluttu lopputulos saavutettiin.

8 Käyttöohje

Sovelluksen avatessa kirjaudutaan sisään ennalta määritetyillä tunnuksilla. Alkuvalikossa on navigaationäppäimet tilausten- ja tuotteidenhallintaan. Tilausten hallintaikkunassa voi valita tilauksen, jota haluaa tarkastella. Valinnan jälkeen välilehdillä on vaihtoehtona merkitä tilaus käsitellyksi ja tarkastella sen tietoja. Tuotteiden hallintaikkunassa on vastaavat toiminnallisuudet, joiden lisäksi käyttäjällä on mahdollisuus lisätä ja muokata tuotteita.

”Työajat” -ikkunassa työntekijä tai admin-käyttäjä voi tarkastella omia työvuorojaan halutulta aikajaksolta. Admin-käyttäjälle on lisäksi pääsy ”Admin” -näkymään, jonka kautta hän voi hakea ja tarkastella jokaisen työntekijän työvuoroja, luoda tai poistaa niitä.

9 Jatkokehitys

Sovellukselle tulevaisuudessa voisi kehittää muutamia uusia toimintoja. Yksi tärkeä uusi toiminto olisi ainakin uusien käyttäjien luominen suoraan käyttöliittymästä. Tämä toiminto näkyisi vain esimies (admin) -käyttäjälle.
Toinen tärkeä toiminto on salasanan palautus, joka ei ole tällä hetkellä ollenkaan mahdollista.
Sovellukseen voisi olla hyvä myös toteuttaa tuotekategoriat, jota voi hyödyntää laajemmissa verkkokaupoissa, joissa on paljon erilasia tuotekategorioita.
Käyttöliittymää pystyy myös aina parantamaan, esim. saavutettavuuden parantamista.

10 Yhteenveto

Tämän dokumentin tarkoitus on avata sovelluksen tilaajalle sovelluksen teknistä toteutusta sekä ohjeistaa sovelluksen käytössä. Teknisen toteutuksen avaamiseksi on hyödynnetty sovelluksen kehityksessä käytettyjä kaavioita. Kaaviot on myös selitetty sanallisesti, jotta dokumentin lukija ymmärtää kehityksen perusidean.

Dokumentissa esitellään sovelluksen tärkeimmät toiminnallisuudet, eli tilausten, tuotteiden ja työvuorojen hallinta. Sovellus auttaa tilausten toimitusketjussa tarjoamalla näkymät keräilyyn ja tilauksen käsittelyyn. Lisäksi sovellus toimii apuvälineenä varaston yleiseen kirjanpitoon ja työntekijöiden työvuorosuunnitteluun ja -tarkasteluun.

Lopputuloksena saatiin toimiva varastonhallintajärjestelmän toinen versio. Järjestelmästä pyrittiin saamaan helposti laajennettava, ja siinä onnistuttiin.