Riina Antikainen
Tuukka Mytty
Janne Valle
Erja Nikunen, Simo Silander
Ohjelmistotuotantoprojekti 2
10.5.2019

Varasto-sovellus

Johdanto

Tässä loppuraportissa käsitellään ryhmä 8:n Ohjelmistotuotantoprojekti 2- kurssille tekemää projektia niin kuin se on toteutunut loppuun kevään toisen periodin aikana. Raportissa käydään läpi mitä teknologioita projektin toteuttamiseen on käytetty esim. ohjelmointikieli. Kuvataan mihin tarpeeseen sovellus tulee eli sen käyttötarkoitus. Kerrotaan mitä toiminnallisuuksia tähän mennessä on toteutettu. Miten ohjelmaa on testattu sen tekemisen aikana. Myöskin kuvataan lyhyesti sovelluksen arkkitehtuuria.

Teknologiat

Suurin osa sovelluksen koodista on tehty Java 8 ohjelmointikielellä. Tämän lisäksi, koska sovellukseen kuuluu graafinen käyttöliittymä (GUI), joka on toteutettu JavaFX:llä, niin projektissa tuli käytettyä myös XML-kieltä. GUI:n suunnittelu ja toteutus oli tehokasta, kun käytimme sen rakentamiseen SceneBuilderiä, joka on graafinen käyttöliittymä JavaFX:n generoimiseen. Sovellukseen kuuluu oleellisena osana tietokanta, joka pyörii MariaDB:n päällä Apache-palvelimella.

Sovelluksen kehityksessä on pyritty hyödyntämään jatkuvan integraation (CI) prosesseja käyttäen Jenkinsiä. Projekti rakennetaan Maven-projektina. Aina kun muutokset työnnetään Gitlab repositorioon, jota Jenkins-palvelin tarkkailee, niin se koostaa ja ajaa testit projektille. Projektissamme Jenkins-palvelin tarkkailee Master ja Development-haaroja.Jenkins-palvelimen kautta käytetään SonarQube-ohjelmaa staattisen analyysiin tekemiseen, joka tapahtuu automaattisesti sen aikana, kun Jenkins kokoaa uuden version ohjelmasta. SonarQube kuvaa muun muassa miten paljon bugeja koodista löytyi, “code smellsien” lukumäärän ja testien kattavuuden.

Sovelluksen kuvaus

Varaston inventaario sovellus, jota käytetään varaston tuotetilanteen seuraamiseen ja hallinnointiin. Varastotyöntekijät voivat lisätä, poistaa sekä päivittää varaston tilannetta graafisen käyttöliittymän avulla. Tämän lisäksi he voivat tarkastella tuotteiden lukumääriä ja yksittäisten tuotteiden tietoja. Tuotetiedot ovat kriittisimmät tiedot, jotta voidaan kertoa, mitä kaikkea se sisältää. Admin-oikeuksilla varustettu henkilö voi lisätä käyttäjiä ohjelmaan sekä lisätä ja poistaa tuoteryhmiä. Kirjautumissivu avaa sovellukseen käyttäjän rooliin perustuvan näkymän. Sovellus myöskin on lokalisoitu eri kielivaihtoehdoille, joita ovat suomi, englanti ja malaiji. Ohjelman koodia on refaktoroitu myöskin pyrkimyksenä poistaa niin sanotut “code smells” joita on esimerkiksi duplikaatti koodi ja suunnittelumalleja on pyritty hyväksi käyttämään myöskin.

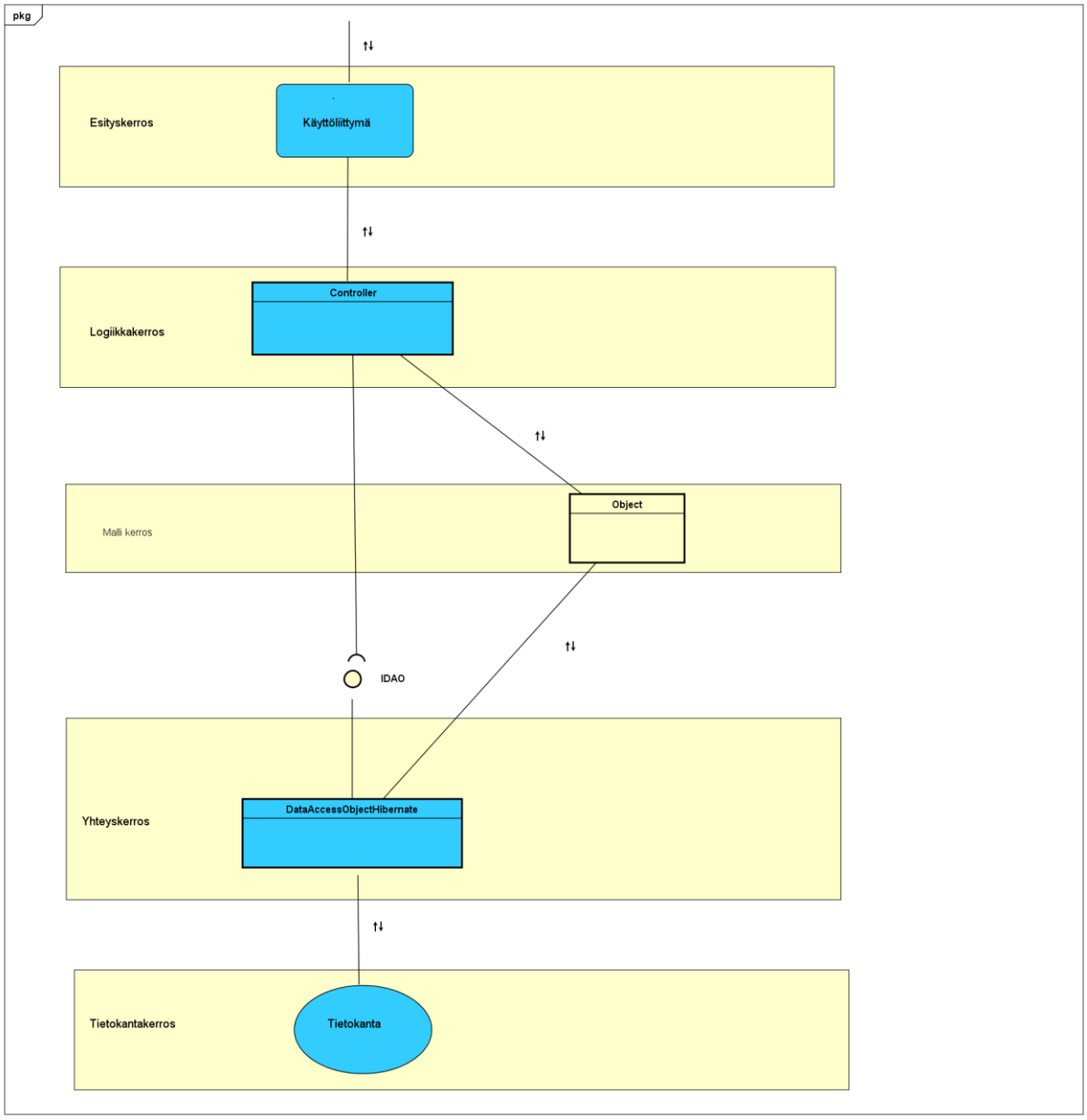
Toiminnallisuudet

Tässä luvussa kerrotaan mitä toiminnallisuuksia on saatu sovellukseen toteutettua eli nämä ovat ne käyttäjätarinat, jotka kahden periodin kuluessa on saatu tehtyä ja jotka löytyvät Agilefantista.

- Varastotyöntekijänä haluan lisätä uuden tuotteen varastoon. Tuote lisätään täyttämällä lomake, jonka sovellus tarkastaa, että se on validi ja jos niin tuote tallentuu tietokantaan jos ei niin puutteelliset tiedot tulee korjata.
- Varastotyöntekijänä haluan poistaa tuotteen varastosta. Tuotteen poisto tapahtuu antamalle sen Id. Sovellus tarkastaa, että annettu Id on numero kuten vaaditaan ja sen jälkeen poistaa kyseisen tuotteen tietokannasta.
- Varastotyöntekijänä haluan päivittää tuotteen tietoja. Muokkaus tapahtuu valitsemalla tuote listasta jonka jälkeen sen tietoja voi muokata. Muokattavat tiedot tarkistetaan ja jos ne ovat oikein niin muutokset tallennetaan tietokantaan.
- Varastotyöntekijänä haluan nähdä varastossa olevat tuotteet. Käyttöliittymä näyttää listauksen tietokannassa olevista tuotteista.
- Varastotyöntekijänä haluan saada tuotteen lukumäärän tietooni, jotta tiedän täytyykö niitä tilata/tehdä lisää. Käyttöliittymä näyttää jokaisen tuotteen lukumäärän.
- Myyntiedustajana haluan kirjautua sovellukseen. Käyttöliittymä näyttää kirjautumisen jälkeen myyntiedustajan näkymän.
- Varastotyöntekijän haluan kirjautua sovellukseen. Käyttöliittymä näyttää kirjautumisen jälkeen varastotyöntekijän näkymän.
- Johtajana haluan lisätä käyttäjän sovellukseen. Käyttäjän lisääminen onnistuu johtajalta.
- Johtajan haluan hallita tuoteryhmiä. Johtajalta onnistuu tuoteryhmien lisäys ja poisto.
- Käyttöliittymän muokkaus käytettävämmäksi. Tooltipit lisätty input kentille. Käyttäjille tulevat ilmoitukset kun toiminto on suoritettu loppuun. Käyttäjien toiminnoille on tehty varmistukset esimerkiksi ohjelman lopetuksen varmistus.
- Tuotteen lokalisointi. Ohjelmaan lisätty kielivaihtoehdot: englanti, suomi ja malaiji.
- Refaktoirointi. Koodista poistettu “bad smells” sekä koodia muokattu käyttämään suunnittelumalleja.

Arkkitehtuuri

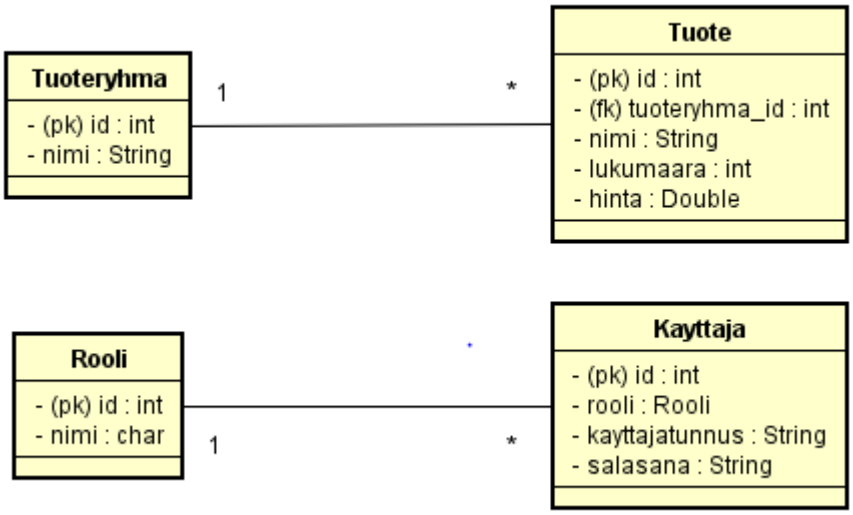
Sovelluskerrokset



Kuva 1. Arkkitehtuuri

Sovellus on toteutettu MVC-mallin mukaisesti ja se sisältää malli-, näkymä- (esitys) ja kontrolleri (logiikka) kerrokset. Sovelluksen bisneslogiikka on eristetty logiikkakerrokseen, jonka lähdekoodit löytyvät com.ro8.varastosofta.application.controller-paketista. Tämän lisäksi koodattua toiminnallisuutta löytyy Dao-tietokantarajapinnan toteuttavissa TuoteDao ja TuoteryhmaDao-luokissa. Näiden luokkien tehtävän on tarjota perus CRUD-operaatiot Hibernaten avulla käytettävälle MariaDB-tietokannalle. 2

Luokkakaavio



Kuva 3 Luokkakaavio

Testaus

Ohjelmoitaessa on pyritty noudattamaan testilähtöistä (TDD) työtapaa, jossa testit kirjoitetaan ennen varsinaista toiminnallisuuden toteuttamista. Testaaminen on suoritettu sekä automaattisesti että manuaalisesti. Automaattinen testaus tapahtuu Java JUnit5 yksikkötesteillä. Yksikkötestejä on ajettu sekä paikallisesti koodaajan omalta tietokoneelta että automaattisesti Jenkins-palvelimella jokaisen koonnin yhteydessä. Manuaalinen käyttöliittymän testaus on suoritettu aina, kun ohjelmaa on muutettu tai toiminnallisuuksia lisätty. Tällä on varmistettu, että ohjelma toimii muuutosten jälkeen halutulla tavalla.

Testitapaukset

Validaattori

METODI	Annettu arvo	Odotettu tulos
onkoLisattavaTuoteValidi(String id, String nimi, String lkm)	1) “”, “test”, “10” 2) “10”, “”, “12”, 3) “10”, “testi”, “” 4) “14”, “testiNimi”, “54” 5) “19”, “liianpitkanimitestataanjaakokiinnivaimeneeklapiheittamalla”, “98”	1) false 2) false 3) false 4) true 5) false
onkoPoistettavaIdValidi(String id)	1) “” 2) “ei ole numero” 3) “12”	1) false 2) false 3) false
onkoNumero(String strNum)	1) “ei numero” 2) “12”	1) false 2) true
onkoTuoteryhmaValidi(String tuoteryhma)	1) “vihannekset” 2) “” 3) “rewqrewrqrqewrqqreqwrerqwqrqrqqeqiyfyfkuf”	1) true 2) false 3) false
onkoLisattavaKayttajaValidi(String tunnus, String salasana)	1) “testi;”, “salasana” 2) “testi”, “salasana;” 3) “testi”, “salasana”	1) false 2) false 3) true

Tooltipit

METODI	Annettu arvo	Odotettu arvo
asetaTooltip(Control control, String viesti)	1) TextField, “testi 2) null, null 3) Button, “testi” 4) null, null	1) true 2) false 3) true 4) false

Näiden lisäksi koonnin yhteydessä ajetaan staattinen analyysin SonaQube työkalu, joka pystyy kertomaan ohjelman bugit, haavoittuvuudet, “Bad smells”:it, testien kattavuuden sekä duplikaatit. Sen avulla on saatu poistettua huonoja ohjelmointi tapoja koodista.

SonarQube:n tulokset