2019

ULI-9000
Varastonhallintasovellus

LOPPURAPORTTI

JOONATAN HÄMÄLÄINEN, JOEL LAINE, TORSTI LAINE, EERO TAKANEVA

Sisällys

1.Lyhyt kuvaus

ULI-9000-sovellus helpottaa, nopeuttaa ja tehostaa varaston hallintaa ja organisointia. Graafisen käyttöliittymän ansiosta varastotyöntekijä pystyy luomaan, muokkaamaan ja poistamaan varastoja, seiniä ja hallinnoimaan hyllyjä ja niissä olevia tuotteita reaaliajassa. Syötetyt tiedot lisätään tietokantaan, mikä mahdollistaa useiden päätelaitteiden käytön sekä varaston hallinnan mistä ja milloin tahansa.

2. Toiminnallisuudet

ULI-9000 varastonhallintaohjelmalla käyttäjä voi luoda useita varastopohjapiirroksia tarpeidensa mukaan. Luotuaan uuden varaston käyttäjä pääsee muokkaamaan sen pohjapiirrosta lisäämällä tilaan hyllyjä, tavaroita ja seiniä. Kun varastoon on lisätty ainakin yksi tavara ja hylly, käyttäjä voi alkaa siirtämään tavaroita hyllyille, minkä jälkeen tavaroiden määrää on mahdollista muuttaa käyttöliittymän ponnahdusikkunan kautta.

2.1 Uuden varaston luonti

Varasto luodaan painamalla ”Luo Uusi Varasto” painiketta, joka sijaitsee käyttöliittymän vasemmassa yläreunassa. Luonnin yhteydessä varastolle annetaan nimi, osoite sekä leveys ja pituus. Käyttäjän syötteet tarkastetaan, ja täten varmistetaan, että syötteet ovat kelpoja: nimi ja osoite ovat vähintään yhden merkin pituisia (ilman välilyöntejä) sekä pituus ja leveys ovat syötetyt positiivisina kokonaislukuina.

Tämän jälkeen siirrytään pohjapiirrokseen ja painetaan näkymän oikeassa yläreunassa näkyvää ”Muokkaa Seiniä” painiketta, jolloin on mahdollista poistaa soluja varaston pohjapiirroksesta. Tilasta poistutaan painamalla samaa painiketta uudelleen.

Varaston poisto ja tietojen muokkaus aloitetaan valitsemalla ”Sinun Varastosi” listasta kohde varasto, minkä jälkeen poistetaan varastokenttien lukitus valintaruutua painamalla. Varaston pystyy poistamaan käyttämällä ”poista” painiketta, ja muutokset tallentuvat painamalla ”Tallenna Muutokset” painiketta.

2.2 Varastonhallinta

Hyllyjen lisääminen uuteen varastoon käy helposti. Käyttäjä valitsee pohjapiirroksesta haluamansa solut hiiren napin painalluksella, jolloin solut muuttavat värinsä harmaaksi osoittaakseen tehtyä valintaa. Valittuaan solut, käyttäjä napauttaa ”Lisää Hylly” painiketta, jolloin harmaat solut muuttuvat mustiksi (tyhjää hyllyä merkitään mustalla).

Tavaroita voi lisätä varastoon ”Uusi Tuote” painikkeella. Käyttäjän on syötettävä tuotteelle nimi, paino, määrä sekä yksikkö- ja myyntihinta. Kuten uuden varaston luonnin yhteydessä syötteet testataan, ja siinä tapauksessa, että syöte on virheellinen, käyttäjä saa virheilmoituksen, jossa osoitetaan, minkä kenttien syötteet eivät menneet läpi.

Luodut tuotteet löytyvät ”Tuotteet Varastossa” pudotuslistalta, ja tuotteen voi asettaa hyllylle valitsemalla kyseisen tuotteen edellä mainitusta listasta, ja hylly joko pohjapiirroksesta tai ”Valitse Hylly” pudotuslistasta, minkä jälkeen painetaan ”Lisää Tuote Hyllyyn” painiketta.

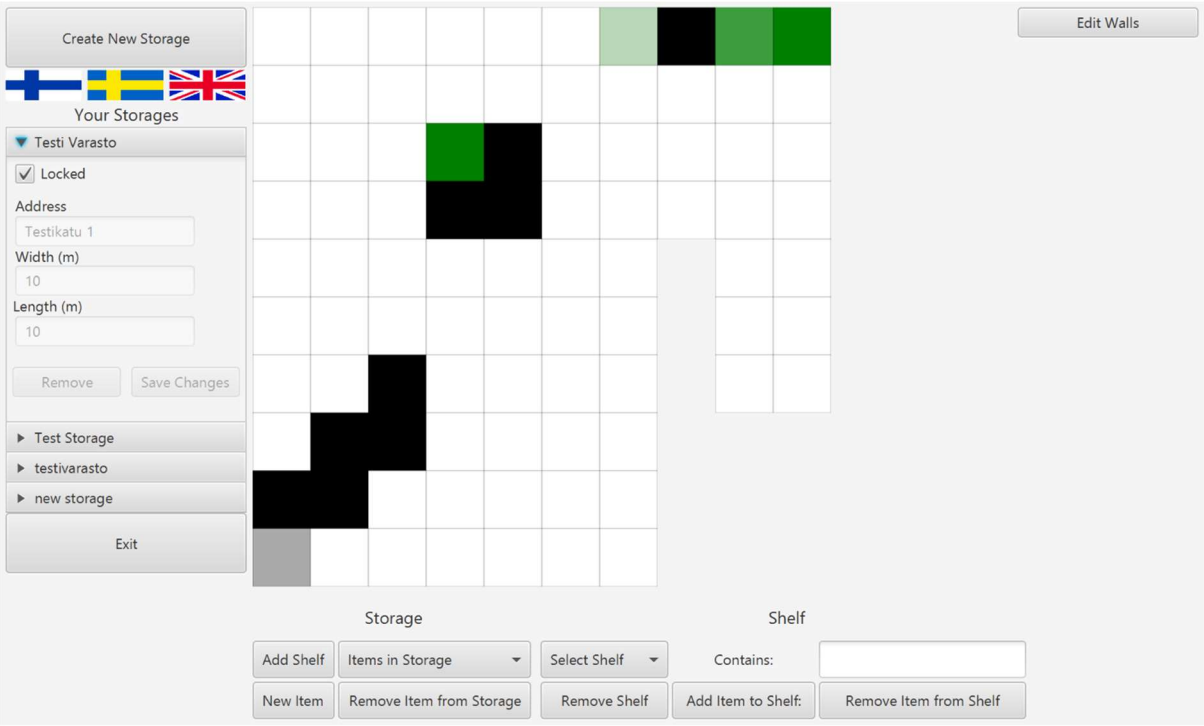
Hyllyn tai tavaran poisto tapahtuu valitsemalla poiston kohde aiemmassa kappaleessa mainitulla tavalla, minkä jälkeen painetaan ”Poista Tuote Varastosta” tai ”Poista Hylly” painiketta.

2.3 Tavaroiden tiedot

Hyllyssä olevan tavaran tietoja pääsee tutkimaan painamalla hyllyä käyttöliittymässä, tai valitsemalla tavara ”Tuotteet Varastossa” pudotuslistasta. Valinnan jälkeen pohjapiirroksessa hyllyn viereen

avautuu ponnahdusikkuna, jossa on näkyvissä tavaran tiedot, sekä mahdollisuus lisätä tai vähentää tavaran määrää.

Käyttöliittymässä tuotteiden määrää hyllyssä ilmaistaan värikoodein [kuva 1]. Hyllyn ollessa täysi se on tumman vihreä, mutta väri alkaa haaleta määrän laskiessa. Musta väri tarkoittaa, että hyllyllä ei ole tavaraa.



Kuva 1. Varaston pohjapiirros

2.4 Kielivalinta

Kielen pystyy vaihtamaan painamalla haluamansa kielen lippupainiketta. Kaikki paitsi tietokannan tekstit vaihtuvat haluamalleen kielelle, kun nappia painaa.

Kielenvaihto on tällä hetkellä tehty painikkeilla, joka ei ole niin helposti laajennettavissa, jos haluaa lisätä uuden kielen. Tämä korvataan *”Combobox”* -painikkeella, joka saisi kielivalinnat suoraan *”properties”* -tiedostosta. Tällöin kun lisää uuden kielen, ei tarvitse koskea koodiin ollenkaan vaan ainoastaan *”properties”* -tiedostoihin.

2.5 Toteutetut käyttäjätarinat

Tässä luvussa aiemmissa osioissa mainitut toiminnot toteuttavat seuraavat Agilefantissa olevat käyttäjätarinat:

- Yrittäjänä minulle on tärkeää, että pystyn tekemään ohjelmalla varastoni pohjapiirroksen
- Varastotyöntekijänä haluan pystyä lisäämään ja poistamaan tuotteita valikoimasta
- Varastotyöntekijänä haluan tuotteen määrän ilmaistavan värikoodilla
- Varastotyöntekijänä minun on pystyttävä muokkaamaan myös varaston muotoa. Pelkkä neliö ei ole osuva kuvaus
- Yrittäjänä haluan käyttää sovellusta haluamallani kielellä
- Varastotyöntekijänä haluan saada tarkempaa tietoa tuotteesta

- Varastotyöntekijänä haluan pystyä muuttamaan tuotteiden määrää hyllyssä
- Käyttäjänä ohjelmiston toiminta on hankalaa, koska tekemäni muutokset eivät päivity käyttöliittymään heti vaan joudun lataamaan ikkunan aina uudelleen
- Varastotyöntekijänä minulle on tärkeää, että saan hyllyn tiedot esiin painamalla sitä käyttöliittymässä
- Varastotyöntekijänä haluan pystyä muuttamaan tuotteiden määrää sitä mukaa kun varastossa tapahtuu muutoksia
- Varastotyöntekijänä haluan luomani varaston seinien tallentuvan myös seuraavaa käyttökertaa varten
- Käyttäjänä haluan, että ohjelmalla on ainutlaatuinen kuvake

3. Keskeneräiset ja toteuttamattomat toiminnallisuudet

Tuotteita voi tällä hetkellä vain lisätä ja poistaa varastoista ja hyllyistä. Tuotteiden määrää hyllyssä pitäisi pystyä muuttamaan. Tuotteet eivät myöskään vielä tallennu tietokantaan.

Tiettyä hyllyä pitäisi pystyä painamaan ruudukossa, joka toisi esille näkymän, jossa näkyisi hyllyssä olevan tuotteen tarkat tiedot.

Käyttöliittymä tällä hetkellä ei päivity automaattisesti, kun siinä lisää tai poistaa hyllyn tai tuotteen, vaan pitää näkymä päivittä klikkaamalla varastoa uudestaan.

Toteuttamattomat käyttäjätarinat:

- Varastovastaavana haluan sovelluksen ilmoittavan minulle, jos joku tuotteista on loppumassa tai loppunut
- Varastotyöntekijänä haluan, että voin tilata lisää tuotteita varastoon ohjelman avulla
- Varaston vastaavana haluan, että voin tarkistaa varaston tilanteen reaaliajassa
- Varastotyöntekijänä haluan pystyä tulostamaan inventaariotilanteen paperiversiona myöhempään tarkasteluun

Nämä käyttäjätarinat jäivät toistaiseksi toteuttamatta. Ne tullaan toteuttamaan tulevaisuudessa.

Tuotteiden saatavuutta pystyisi helposti hallitsemaan ominaisuudella, jossa tuotteiden hyllyjen väri muuttaisi väriään ensin vihreästä, keltaiseksi, sitten punaiseksi, ja lopulta mustaksi, riippuen kuinka paljon tuotetta olisi jäljellä hyllyssä.

Ohjelma ilmoittaisi ilmoituksella käyttäjälle, jos jokin tuote on loppumassa tai loppunut.

Ohjelmassa tulee olemaan tuotteiden tilausnappi, jolla työntekijä pystyisi tilaamaan lisää haluamiaan tuotteita.

Varaston inventaariotilanteen pystyisi tulostamaan.

4. Arkkitehtuuri

Tässä osiossa käydään läpi ohjelman eri osien arkkitehtuuri. Sovelluksen tarvittava koodi löytyy sivulta: https://github.com/TorstiL/OTProjekti1.

Yhdistääkseen Jenkins-palvelimeen käyttäjän tarvitsee tunneloida yhteys metropolia.fi verkkoon ja PuTTY-sovelluksen avulla kirjautua shell.metropolia.fi-palvelimeen. Shell.metropolia.fi-palvelimen portti on 22 ja lisätä tunneli, jonka lähdeportti on L2206 ja kohde on 10.114.32.46:80. Kun tunneli on lisätty palaa takaisin Session-osioon, josta voit avata luodun istunnon. Kirjaudu käyttämällä Metropolian tunnuksia, jonka jälkeen voit avata selaimen ja kirjoittaa osoitehakuun 10.114.32.46:8080. Jenkins-sivun pitäisi näkyä, johon voit kirjautua ja katsoa kokoelmien toimivuutta.

Tietokantaan yhdistäminen tapahtuu myös PuTTY-sovelluksen avulla. Niin kuin Jenkins-palvelimen kanssa muodosta yhteys osoitteeseen 10.114.32.46:80. Sovelluksen toimivuuden kannalta tämä yhteys pitää aina, kun haluaa käynnistää ULI-9000 sovelluksen. Tietokannan avaaminen ei onnistu ilman tätä yhteyttä. Kun olet kirjautunut PuTTY-sovelluksessa metropolia tunnuksilla sisään, kirjoita komentoriville: ssh 10.114.32.46 -l ”käyttäjänimi”. Tämän jälkeen syötä salasana. Nyt olet kirjautunut virtuaalikoneelle, jonka jälkeen kirjoitat komentoriville: mysql -u ”käyttäjänimi”. Nyt voit käyttää tietokannan eri tauluja, miten haluat.

4.1 Sovelluksen arkkitehtuuri

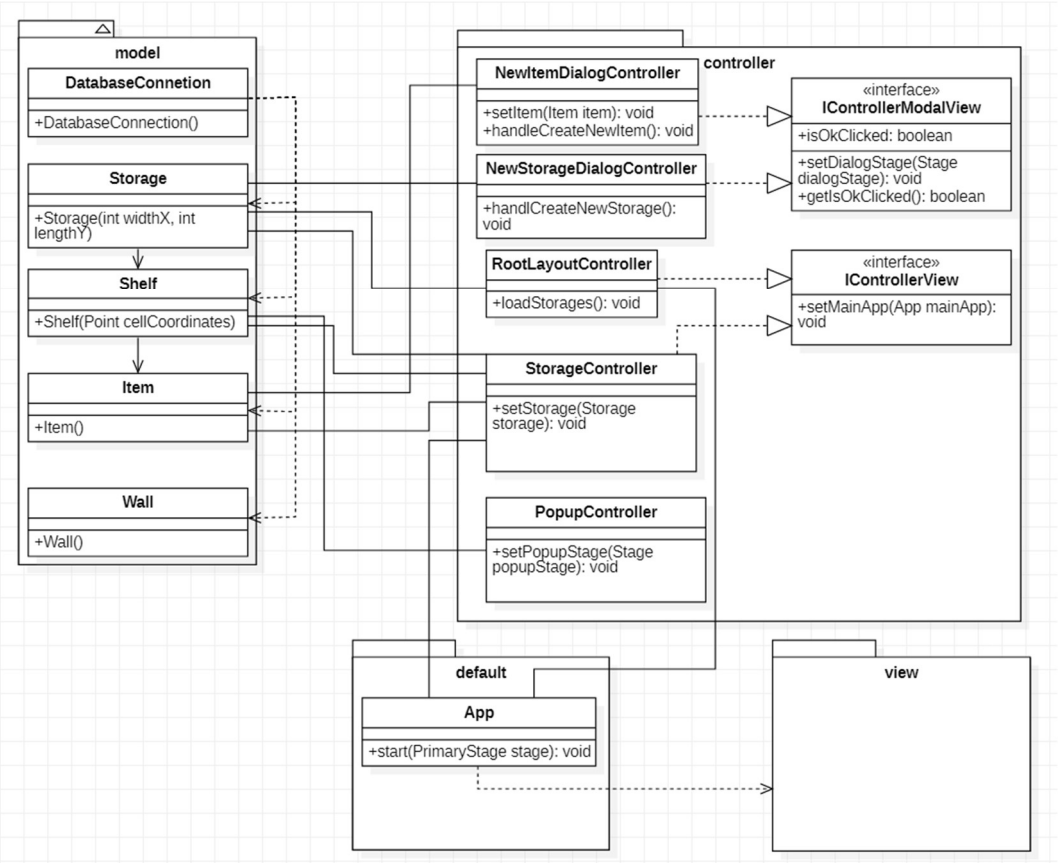
Sovelluksen tuottamisessa on käytetty MVC-mallia [kuva 2]. Sovelluksen tärkeimmät kohdat ovat selkeä ja graafinen käyttöliittymä, sekä tämän kanssa toimiva tietokanta. Graafisen käyttöliittymän tuottamiseen on käytetty SceneBuilderiä, joten *View*-paketin sisältö perustuu fxml-tiedostoihin. App.java toteuttaa nämä graafiset tuotokset.

RootLayoutController ja *StorageController* toteuttavat *InterfaceControllerView*-rajapinnan ja *NewStorageDialogController* sekä *NewItemDialogController* toteuttavat *ControllerInterfaceModalView*-rajapinnan. *Popup*-, *RootLayout*-, sekä *StorageController*it riippuvat *DatabaseConnection* malliluokasta.

Data Access Object-luokka *DatabaseConnection* käyttää kaikkia muita malliluokkia: *Storage*, *Shelf*, *Item* ja *Wall*. *Storage*-luokkaa käytetään *NewStorageDialogController*issa, *RootLayoutController*issa ja *StorageController*issa. *StorageController* käyttää *Shelf*, ja *Item*-luokkia, ja *NewItemDialogController* käyttää myös *Item*-luokkaa.

Lokalisointi on toteutettu *properties* -tiedostojen avulla. Ohjelma hakee *language.properties* -tiedostosta tämän hetkisen valitun kielen, jonka avulla *ResourceBundle* valitsee oikean kielisen *TextResources.properties* -tiedoston. FXML-dokumenteissa on käytetty kovakoodatun tekstin sijasta kansainvälistettyjä merkkijonoja kuten: *%itemedit.shelf* sen sijaan, että siinä lukisi pelkkä *Shelf* tai hylly. Nämä merkkijonot toimivat avaimina, joilla oikea teksti haetaan *ResourceBundle:*n avulla *properties* -tiedostosta.

Graafisen käyttöliittymän ulkomuoto on suunniteltu käyttäjäkeskeisesti, jotta sovellusta ensimmäistä kertaa käyttävä käyttäjä saa mahdollisimman paljon sovelluksesta irti.



Kuva 2. ULI-9000 UML-kaavio

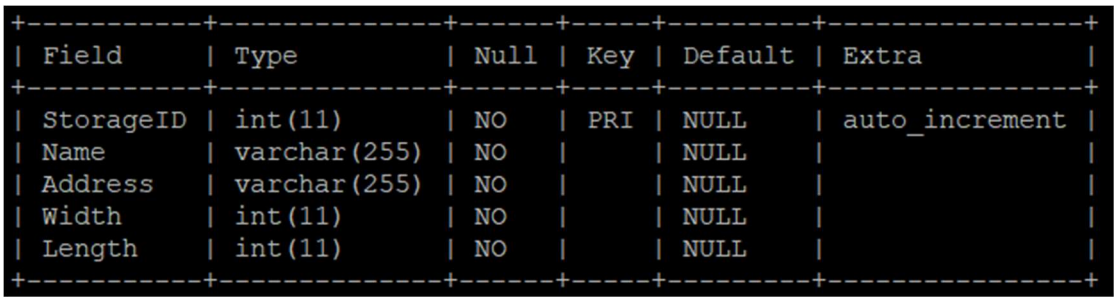
4.2 Tietokannan arkkitehtuuri

Sovellus käyttää tiedon säilömiseen yksinkertaista tietokantaa, joka sisältää taulut varastoille, hyllyille, seinille ja tuotteille [Kuva 3]. Tiedon hakemiseen ja tallentamiseen käytetään sovelluskehys Hibernate ORM:ia, joka tarjoaa oliopohjaisen työkalun tietokannan käsittelyyn.

```
+---------------------+
| Tables_in_WAREHOUSE |
+---------------------+
| Item                |
| Shelf               |
| Storage             |
| Wall                |
+---------------------+
```

Kuva 3: Tietokannan taulut

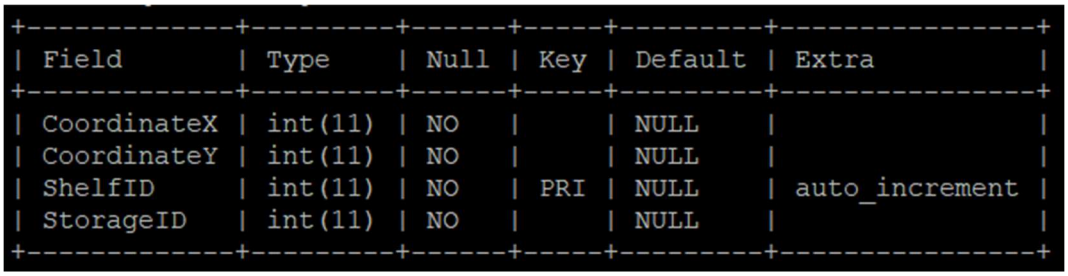
Storage-tauluun voidaan tallentaa varaston nimi, osoite, leveys ja pituus [Kuva 4]. Käyttäjä antaa nämä tiedot käyttöliittymässä varastoa luodessa. Varaston tunnus saadaan automaattisesti, kun varasto tallennetaan tietokantaan.



Field	Type	Null	Key	Default	Extra
StorageID	int(11)	NO	PRI	NULL	auto_increment
Name	varchar(255)	NO		NULL	
Address	varchar(255)	NO		NULL	
Width	int(11)	NO		NULL	
Length	int(11)	NO		NULL	

Kuva 4: Tietokannan Storage-taulu

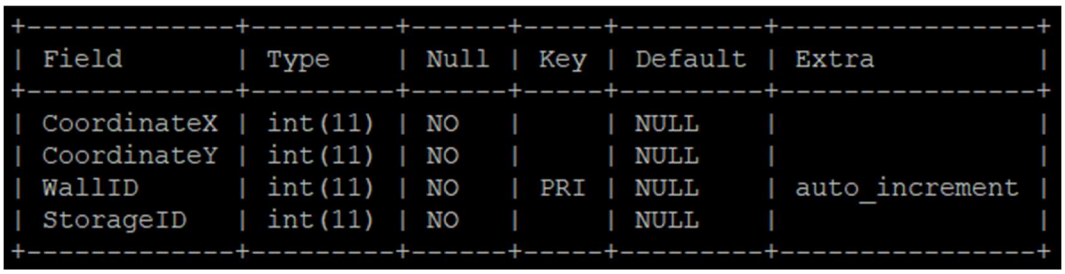
Shelf-tauluun voidaan tallentaa hyllyn nimi, pituus- ja leveyskoordinaatit sekä sen varaston tunnus, jossa hylly sijaitsee [Kuva 5]. Käyttäjä antaa nämä tiedot käyttöliittymässä hyllyä luodessa. Hyllyn tunnus saadaan automaattisesti, kun varasto tallennetaan tietokantaan.



Field	Type	Null	Key	Default	Extra
CoordinateX	int(11)	NO		NULL	
CoordinateY	int(11)	NO		NULL	
ShelfID	int(11)	NO	PRI	NULL	auto_increment
StorageID	int(11)	NO		NULL	

Kuva 5: Tietokannan Shelf-taulu

Hyllyt ja seinät käsittelevät samoja varaston luonnin yhteydessä rakennettuja soluja. Tämän takia niiden tietokantataulut ovat lähes identtisiä [Kuva 5 ja 6]. Vaikka molemmat rakenteet olisi saanutkin mahdutettua samaan tauluun, sisällön helpomman hahmottamisen ja nopeamman testaamisen takia kyseinen ratkaisu oli parempi.



Field	Type	Null	Key	Default	Extra
CoordinateX	int(11)	NO		NULL	
CoordinateY	int(11)	NO		NULL	
WallID	int(11)	NO	PRI	NULL	auto_increment
StorageID	int(11)	NO		NULL	

Kuva 6: Tietokannan Wall-taulu

Tuotteita varten on neljästä taulusta sisällöltään isoin taulu [Kuva 7]. Tuote saa käyttäjältä luonnin yhteydessä tuotenimen, -määrän ja -aloitusmäärän, -myyntihinnan, -yksikköhinnan, -painon sekä automaattisesti sen varaston tunnuksen, johon se luodaan. Tuotteen hyllytunnus saraketta päivitetään, kun tuote siirretään johonkin varaston hyllyistä kyseisen hyllyn tunnuksella.

Field	Type	Null	Key	Default	Extra
Name	varchar(255)	NO		NULL	
Amount	int(11)	NO		NULL	
Salesprice	double	NO		NULL	
Unitprice	double	NO		NULL	
Weight	int(11)	NO		NULL	
ShelfID	int(11)	YES		NULL	
StorageID	int(11)	NO		NULL	
HighestAmount	int(11)	YES		NULL	
ItemID	int(11)	NO	PRI	NULL	auto_increment

Kuva 7: Tietokannan Item-taulu

Data Access Object-luokkana tietokannan ja kontrollereiden välillä toimii *DatabaseConnection*-luokka. Luokasta löytyy Hibernatea käyttävät metodit tietokantatauluihin tallentamiseen, niistä hakemiseen ja poistamiseen sekä tiedon muokkaamiseen, joita kontrollerit käyttävät asiakkaan pyynnöstä.

4.3 Testiarkkitehtuuri

Testiluokat sijaitsevat ”src/test/java” kansiosta. Jokaisella MVC-mallin osalla on oma testi kansionsa, ja ne ovat nimetty vastaavin englanninkielisten nimiensä mukaan: *model, view* ja *controller.*

Malliluokkien (*model*) testit käsittävät olion luonnin, poiston sekä primitiivisiä operaatioita kuten määrän ja koon muutokset sekä johdettujen ominaisuuksien hakufunktiot. Perusominaisuuksien *get/set* -funktioita ei testattu niiden yksinkertaisen rakenteensa vuoksi.