VAASAN AMMATTIKORKEAKOULU
VASA YRKESHÖGSKOLA
UNIVERSITY OF APPLIED SCIENCES

Jussi Antero Tuori

LOHKOKIRJANPITO-SOVELLUS

Tekniikka ja liikenne
2012

VAASAN AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma

TIIVISTELMÄ

Tekijä	Jussi Tuori
Opinnäytetyön nimi	Lohkokirjanpito-sovellus
Vuosi	2012
Kieli	suomi
Sivumäärä	30
Ohjaaja	Pirjo Prosi

Tämä opinnäytetyö tehtiin Vaasan ammattikorkeakoulun tietotekniikan koulutusohjelman päättötyönä omaan käyttöön. Työn tarkoituksena oli kehittää maatalouden kasvulohkojen kirjanpitosovellus, johon pystytään tallentamaan peltolohkoille suoritettuja tapahtumia ja tarkastella niitä kasvukausittain. Kasvukauden päätteeksi käyttäjä pystyy myös tulostamaan lohkoraportin lohko- ja kasvukausikohtaisesti.

Sovellus toteutettiin NetBeans -kehitysympäristössä Java-ohjelmointikielellä ja tietojen tallennukseen käytettiin Microsoft Access – tietokantahallintaohjelmaa. Tietokanta suunniteltiin niin, että siihen pystytään tulevaisuudessa lisäämään helposti lisää tauluja uusia ominaisuuksia varten. Käyttöliittymä suunniteltiin myös modulaariseksi.

Sovelluksen käyttöliittymä on valmis ja toiminnassa, mutta kaikkia vaatimusmäärittelyssä esitettyjä vaatimuksia se ei vielä sisällä. Sovellus vaatii vielä kehitystä, että kaikki vaatimukset saadaan sisällytettyä.

Avainsanat Java, Microsoft Access

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Tietotekniikan koulutusohjelma

ABSTRACT

Author	Jussi Tuori
Title	Segment Bookkeeping -Application
Year	2012
Language	Finnish
Pages	30
Name of Supervisor	Pirjo Prosi

The purpose of this thesis was to develop a Segment Bookkeeping –application for agriculture. In the application the user can add events performed for field segments and view them by growing season. At the end of the growing season the user can also print out segment report by growing season.

The application was made with the NetBeans develop environment using Java-programming language and for storing data was used Microsoft Access –database management program. The database was designed in that way you can easily add extra features in future. The user interface was also designed modular.

The application interface is up and running, but all the features of the requirement analysis are not implemented yet.

Keywords Java, Microsoft Access

ALKUSANAT

Tämä opinnäytetyö tehtiin Vaasan ammattikorkeakoulun tietotekniikan koulutusohjelman päättötyönä omaan käyttöön. Työn tarkoituksena oli kehittää sovellus, jolla maatalouden peltolohkojen tapahtumien tietoja voidaan tallentaa tietokantaan sekä tarkastella ja muokata tallennettuja tietoja käyttöliittymän kautta. Sovellus oli tarkoitus toteuttaa selkeäksi ja yksinkertaiseksi.

Työn valvojana toimi Vaasan ammattikorkeakoulun tekniikan ja liikenteen yksiköstä yliopettaja Pirjo Prosi. Yliopettaja Kalevi Ylinen konsultoi työn alkuvaiheessa.

Haluan kiittää tyttöystävääni Johannaa, joka kannusti minua tämän työn tekemisessä sekä yliopettaja Pirjo Prosia työn valvojana toimimisesta.

Vaasassa 23.5.2012

Jussi Tuori

SISÄLLYS

MERKINNÄT JA LYHENTEET

SQL	Structured Query Language on standardoitu relaatiotietokannan kyselykieli, jonka avulla annetaan käskyjä tietokannalle.
JDBC	Java Database Connectivity on ohjelmointirajapinta, jonka avulla voidaan luoda yhteys tietokannan ja Java – sovelluksen välillä sekä suorittaa SQL–kyselyitä Java – luokkien sisällä./1/
JDK	Java Development Kit on kehitystyökalupaketti, joka sisältää Java – ohjelmointiin tarvittavat ohjelmat./1/
NetBeans	Integroitu ohjelmointiympäristö Java-ohjelmointiin.
Swing	Graafisen käyttöliittymän luontiin tarkoitettu kirjasto Javalle./1/
JRE	Java Runtime Environment on ajoympäristö käännettyjen Java-ohjelmien ajamiseen./1/

1 JOHDANTO

Maataloudessa suoritetaan peltolohkoille erilaisia toimenpiteitä ja näistä laitetaan merkinnät lohkokirjanpitoon. Tämän opinnäytetyön tarkoituksena on muuttaa kirjanpito sähköiseen muotoon, josta on myös helppo sitten tarkastella ja muuttaa tietoja sekä tarvittaessa tulostaa ne. Sovellus on tarkoitus tehdä mahdollisimman helppokäyttöiseksi, jotta käyttäjän ei tarvitse opiskella sen käyttöä kauan. Sovelluksessa käyttäjä pystyy lisäämään maatilan tiedot ja sen jälkeen eri peltolohkojen perustiedot tietokantaan. Kun nämä tiedot ovat lisättynä, voidaan alkaa lisätä eri tapahtumia lohkoille. Käyttöliittymä näyttää valitun lohkon tiedot sekä lisätyt tapahtumatiedot lohkokohtaisesti, tapahtumia pystyy poistamaan tietokannasta sekä lohkojen valittuja tietoja voi muuttaa.

Esimerkkinä tallennettavasta tapahtumasta voi olla kylvötapahtuma, josta käyttäjä antaa sovellukseen vaadittavat tiedot. Kylvötapahtumasta halutaan tallentaa päivämäärä, kylvötyyppi, siemenlajike, siemenmäärä hehtaaria kohti, lannoitelajike sekä lannoitteen määrä hehtaaria kohti. Viljelijä pystyy näiden tietojen avulla suunnittelemaan seuraavan kasvukauden kylvön.

Ohjelma toteutetaan Java-ohjelmointikielellä NetBeans-ohjelmointiympäristössä ja tiedot tallennetaan Access-tietokantaan.

2 OHJELMISTOT JA TEKNIIKAT

2.1 Ohjelmointiympäristö

Ohjelmointiympäristöksi valitsin NetBeans IDE 7.1.1, koska se on avoimen lähdekoodin kehitysympäristö sekä siinä on sisällytettynä valmiiksi helppokäyttöinen graafisen käyttöliittymän editori. Työssä käytettiin JDK 7:ää eli uusinta versiota kyseisellä hetkellä.

NetBeans tukee monia eri ohjelmointikieliä ja sovelluskehyksiä sekä siinä on laaja työkaluvalikoima. NetBeans on myös yhteensopiva valitun tietokannan kanssa, yhteyden luonnissa käytetään perus JDBC-ODBC – siltaa.

2.2 Ohjelmointitekniikka

Ohjelman käyttöliittymän ulkoasun toteutuksessa käytettiin Java Swing-käyttöliittymäkirjaston komponentteja. Swingin komponentit on kirjoitettu kokonaan Javalla, eivätkä ne käytä käyttöjärjestelmän omia graafisia komponentteja Javan alkuperäisen graafisen kirjaston AWT:n tapaan. Tästä syystä Swing-komponentteja kutsutaankin ”kevyiksi”.

2.3 Tietokannat

Sovellukseen tarvittiin tehokas, helppokäyttöinen ja nopeasti omaksuttava tietokanta. Microsoft Access-relaatiotietokantasovellus sopi tämän sovelluksen tarpeisiin, käyttäjäyhteyksiä on vain yksi kerrallaan eli ei tarvita yhteysallasta ja Access riittää hyvin sovelluksen aineistojen käsittelyyn.

Java-ohjelmointikielen ja Accessin tietokannan välinen yhteys muodostetaan JDBC-ajureilla. JDBC tarjoaa rajapinnan, jonka avulla tietokannan hallintajärjestelmälle välitetään SQL-lauseita. Käytettäessä JDBC-protokollaa tietokannan URL muodostetaan niin, että ensin kirjoitetaan käytettävän protokollan nimi, seuraavaksi aliprotokolla ja lopuksi tietokannan nimi, eli esimerkiksi *jdbc:odbc:Tietokannan_nimi.*

Seuraavassa on esimerkki kuinka yhdistäminen tapahtuu JDBC-ajureita käyttäen. Aluksi ladataan JDBC-tietokanta-ajuri, ja jos ajuri saadaan onnistuneesti ladattua, otetaan yhteys tietokantaan./4/

```
try {
Class.forName (”sun.jdbc.odbc.JdbcOdbcDriver”);
} catch (java.lang.ClassNotFoundException ex) {
System.out.print (“Ajurin lataamisessa virhe:”+ex);
}
try {
yhteys =
DriverManager.GetConnection(“jdbc:odbc:Tietokannan_nimi”);
} catch (SQLException ex) {
System.out.print (“Virhe tietokantayhteyden avaamisessa: ”+ex);
```

3 VAATIMUSMÄÄRITTELY

3.1 Yleiskuvaus

Valmista sovellusta käytetään suoraan asiakkaan tietokoneelta ja myös sovelluksessa käytettävä Access-tietokanta sijaitsee suoraan asiakkaan koneella. Sovellusta käytetään viljelytilan peltolohkoille tehtyjen toimenpiteiden kirjanpitoon. Sovelluksessa käyttäjä pystyy lisäämään toimenpiteitä eri lohkoille sekä muokkaamaan ja poistamaan niitä.

3.2 Toiminnallisuusvaatimukset

3.2.1 Toimintovaatimukset

Sovellus toimii viljelijän peltolohkojen kirjanpito-ohjelmana, johon viljelijä merkitsee peltolohkoille suoritettuja toimenpiteitä. Sovelluksessa viljelijä pystyy lisäämään lohkoja tietokantaan, jonka jälkeen lohkolle voidaan lisätä tapahtumia kasvukausikohtaisesti. Sovelluksessa on myös käyttäjän rekisteröinti, jossa käyttäjä antaa maatilan perustiedot sekä keksii salasanan kirjautumista varten. Taulukossa 1 ovat sovelluksen toimintovaatimukset.

Taulukko 1. Toimintovaatimukset

Toiminto	**Kuvaus**	**Prioriteetti**
T1	Kasvulohkon lisäys tietokantaan	1
T2	Kasvulohkon poistaminen tietokannasta	1
T3	Kasvulohkon tietojen muokkaus	1
T4	Kasvukauden valinta	1
T5	Lohkon tapahtuman lisäys	1
T6	Lohkon tapahtuman poisto	1
T7	Käyttäjän rekisteröinti	1
T8	Lohkokortin tulostus	2
T9	Maanäytteiden lisäys	2

3.2.2 Yhteensopivuus ja toimintaympäristö

Sovellusta tullaan käyttämään Windows-käyttöjärjestelmällä varustetuissa tietokoneissa, joten käyttäjän ei tarvitse huolehtia kuin siitä, että tietokoneella on uusimmat Java-päivitykset.

3.2.3 Käytettävyysvaatimukset

Sovelluksen käyttöliittymän tulee olla nopeasti omaksuttava sekä yksinkertainen. Toimintojen täytyy myös olla loogisia sekä helppoja käyttää.

Sovelluksen käyttöliittymän ulkoasun tulee olla selkeä, komponentit tulee sijoittaa loogisiin paikkoihin sekä eri sivujen välinen navigointi tulee olla johdonmukaista.

4 TOIMINNALLINEN MÄÄRITTELY

4.1 Johdanto

Opinnäytetyön tavoitteena on toteuttaa työkalu, jonka avulla viljelijä pystyy tallentamaan peltolohkoille tehtyjä tapahtumia. Sovelluksessa on mahdollisuus lisätä erilaisia tapahtumia sekä poistaa niitä.

4.2 Yleiskuvaus

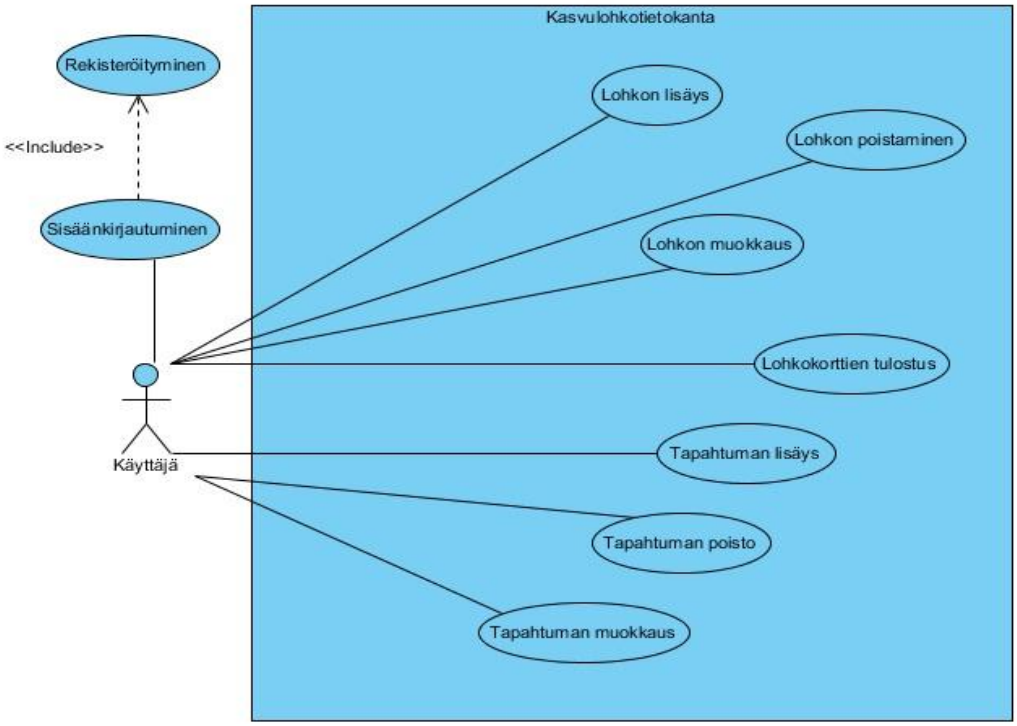
4.2.1 Käyttäjät

Sovellus on suunniteltu sellaisille käyttäjille, jotka tarvitsevat kirjanpito-ohjelmaa maatilan peltolohkoille. Sovelluksen avulla viljelijä pystyy suunnittelemaan seuraavaa kasvukautta ja hyödyntämään tallennettuja tietoja.

Sovelluksen käyttäminen on suunniteltu helppokäyttöiseksi silmälläpitäen käyttäjäryhmää, jonka atk-taidot ovat kohtuulliset. Käyttäjien aika ei kulu tietojen etsimiseen sekä ohjelman opetteluun vaan he pystyvät tallentamaan tai muokkaamaan tietoja nopeasti.

4.3 Sovelluksen toiminnot

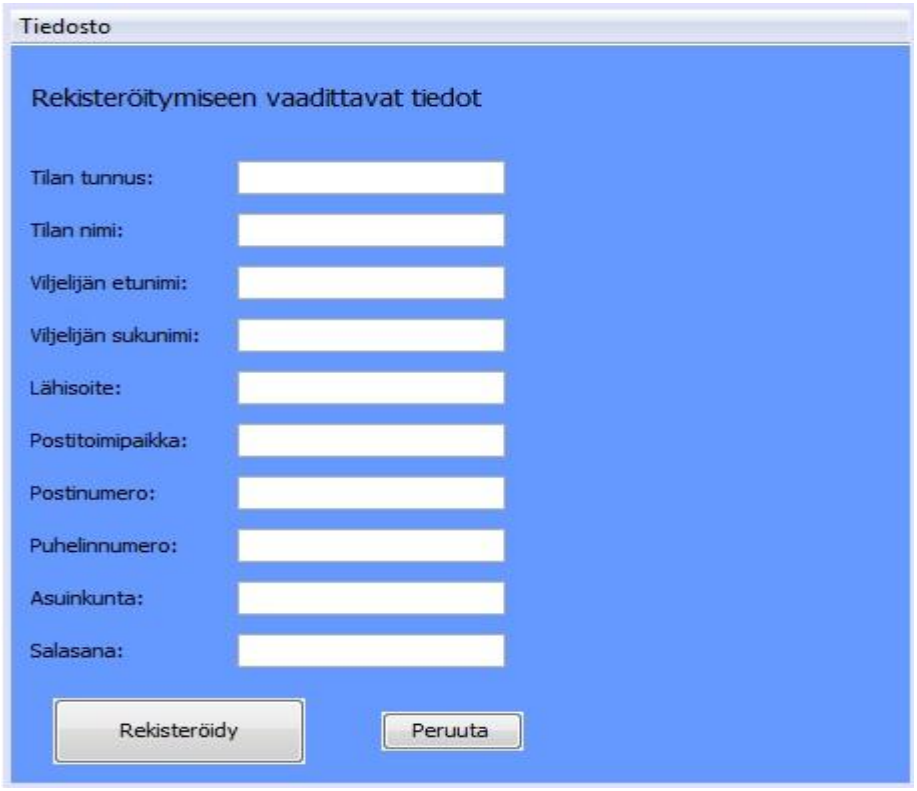
Sovellusprojektin toiminnot ovat kuvattuina kuvassa 1.



Kuva 1. Käyttötapakaavio

4.3.1 Rekisteröityminen

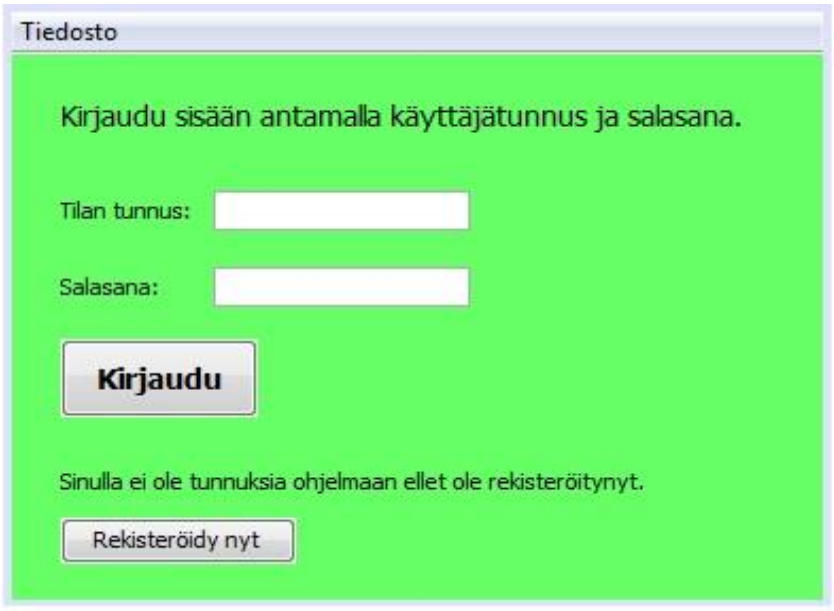
Sovellukseen täytyy rekisteröityä ennen kuin käyttäjä voi alkaa käyttää ohjelmaa, rekisteröinnissä käyttäjää pyydetään syöttämään maatilan tiedot. Rekisteröintilomakkeeseen täytetään maatilan tunnus, tilan nimi, viljelijän nimi, viljelijän sukunimi, lähisosoite, postitoimipaikka, postinumero, puhelinnumero, asuinkunta sekä salasana. Maatilan tunnus toimii sovelluksessa käyttäjätunnuksena. Kun käyttäjä painaa Rekisteröidy –painiketta, sovellus tallentaa annetut tiedot tietokantaan. Sen jälkeen sovellus kirjautuu sisään annetuilla tunnuksilla ja avaa päänäkymän, jossa käyttäjä näkee annetut maatilan perustiedot. Jos taas kaikkia kohtia ei oltu täytetty, sovellus antaa huomautuksen, että osa kentistä on tyhjiä. Jos käyttäjä haluaa palata päänäkymään hän voi painaa Peruuta-nappia tai mennä Tiedosto-valikkoon ja painaa sieltä Exit-kohdasta, jolloin ikkuna sulkeutuu. Kuvassa 2 on käyttöliittymän rekisteröitymislomake.



Kuva 2. Rekisteröityminen

4.3.2 Sisäänkirjautuminen

Sisäänkirjautumisessa käyttäjä antaa tilan tunnuksen sekä salasanan, jotka käyttäjä antoi rekisteröitymisen yhteydessä ja jotka tallennettiin tietokantaan. Kun käyttäjä antaa tiedot lomakkeelle ja painaa Kirjaudu –painiketta, sovellus vertaa annettuja tietoja tietokantaan tallennettuihin. Jos tunnukset ovat oikein, sovellus ilmoittaa tunnuksien oikeellisuudesta sekä ohjaa sen jälkeen käyttäjän päänäkymään. Jos taas tunnukset eivät täsmää, sovellus ilmoittaa, että annetut tunnukset olivat väärät. Sisäänkirjautumislomakkeella on myös Rekisteröidy nyt –painike, jota painamalla käyttäjä ohjataan rekisteröitymislomakkeelle. Ohjelman lopettaminen onnistuu menemällä Tiedosto-valikkoon ja painamalla sieltä kohdasta Exit. Kuvassa 3 on käyttöliittymän sisäänkirjautumislomake.



Kuva 3. Sisäänkirjautuminen.

4.3.3 Kasvulohkon lisäys

Kun käyttäjä on rekisteröitynyt sovellukseen, täytyy käyttäjän lisätä maatilan peltolohkot tietokantaan. Lomakkeeseen käyttäjä syöttää tiedoiksi lohkon tunnuksen, lohkon nimen, lohkon pinta-alan, maankäyttötarkoituksen ja tiedon onko lohko luomuviljelyksessä. Lohkon tunnus-kentän sovellus täyttää automaattisesti hakemalla päänäkymästä valitun kasvulohkon tunnuksen. Lisää kasvulohko –painiketta painaessa lomakkeelle annetut tiedot tallennetaan tietokantaan sekä ikkuna sulkeutuu. Kuvassa 4 on kasvulohkon lisäyslomake.

Kuva 4. Kasvulohkon lisäys

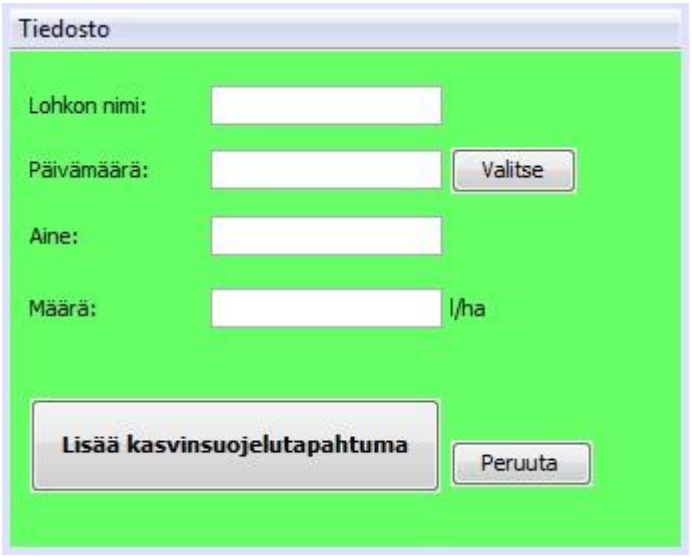
4.3.4 Kylvötapahtuman lisäys

Kylvötapahtuman lisäämisessä käyttäjää pyydetään antamaan lomakkeelle tapahtumasta tietoina lohkon nimi, päivämäärä, kylvötyyppi, siemenlajike, siemenmäärä, lannoitelajike ja lannoitteen määrä. Kun kaikki kohdat on täytetty, käyttäjä painaa Lisää kylvötapahtuma –painiketta ja sovellus tallentaa tiedot tietokantaan. Käyttäjä voi myös painaa Peruuta –painiketta, jolloin sovellus sulkee ikkunan ja palaa päänäkymään. Kuvassa 5 on kylvötapahtuman lisäyslomake.

Kuva 5. Kylvötapahtuman lisäys.

4.3.5 Kasvinsuojelutapahtuman lisäys

Kasvinsuojelutapahtuman lisäämisessä käyttäjää pyydetään antamaan lomakkeelle tapahtumasta tietoina lohkon nimi, päivämäärä, käytetty kasvinsuojeluaine ja ruiskutusmäärä hehtaaria kohden. Kun kaikki kohdat on täytetty, käyttäjä painaa Lisää kasvinsuojelutapahtuma –painiketta ja sovellus tallentaa tiedot tietokantaan. Käyttäjä voi myös painaa Peruuta –painiketta, jolloin sovellus sulkee ikkunan ja palaa päänäkymään. Kuvassa 6 on kasvinsuojelutapahtuman lisäyslomake.



Kuva 6. Kasvinsuojelutapahtuman lisäys.

4.3.6 Sadonkorjuutapahtuman lisäys

Sadonkorjuutapahtuman lisäämisessä käyttäjää pyydetään antamaan lomakkeelle tapahtumasta tietoina lohkon nimi, päivämäärä ja sadon määrä hehtaaria kohden. Kun kaikki kohdat on täytetty, käyttäjä painaa Lisää sadonkorjuutapahtuma –painiketta ja sovellus tallentaa tiedot tietokantaan. Käyttäjä voi myös painaa Peruuta –painiketta, jolloin sovellus sulkee ikkunan ja palaa päänäkymään. Kuvassa 6 on sadonkorjuutapahtuman lisäyslomake.

Kuva 7. Sadonkorjuutapahtuman lisäys.

4.3.7 Muokkaustapahtuman lisäys

Muokkaustapahtuman lisäämisessä käyttäjää pyydetään antamaan lomakkeelle tapahtumasta tietoina lohkon nimi, päivämäärä ja muokkaustoimenpide. Kun kaikki kohdat on täytetty, käyttäjä painaa Lisää muokkaustapahtuma –painiketta ja sovellus tallentaa tiedot tietokantaan. Käyttäjä voi myös painaa Peruuta –painiketta, jolloin sovellus sulkee ikkunan ja palaa päänäkymään. Kuvassa 6 on muokkaustapahtuman lisäyslomake.

Kuva 8. Muokkaustapahtuman lisäys.

4.3.8 Maanäytteiden lisäys

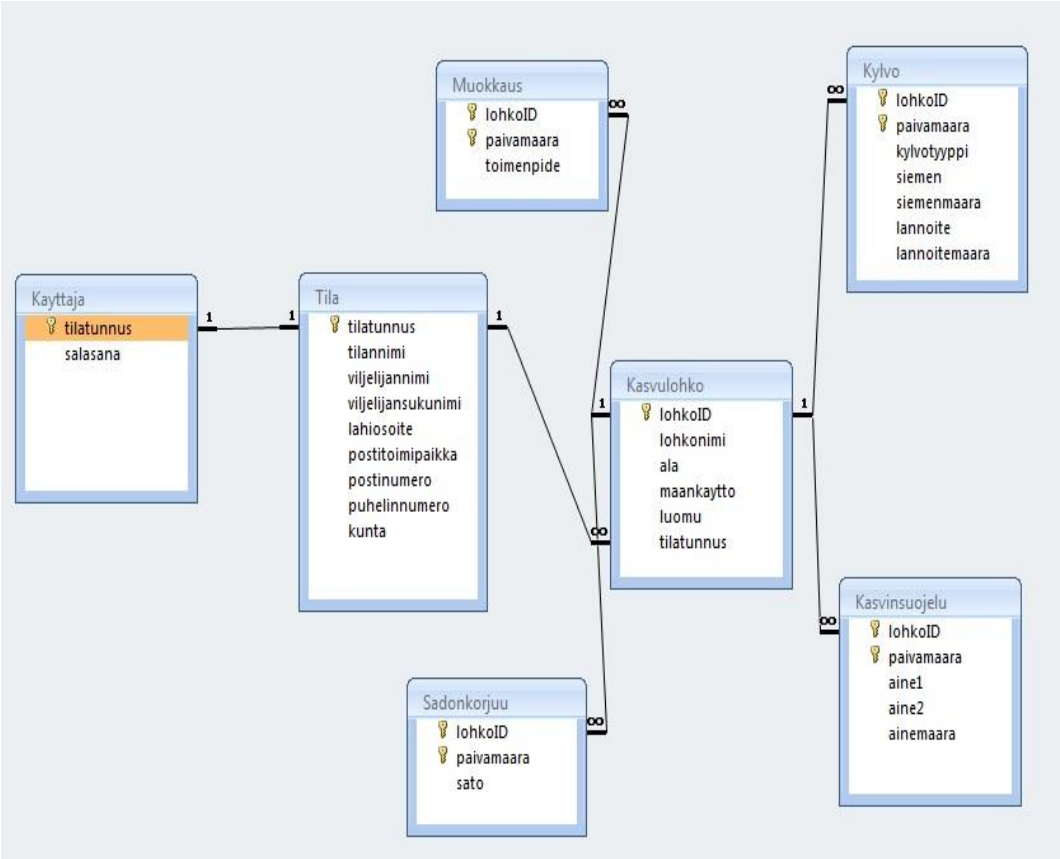
Maanäytteiden lisääminen tapahtuu myös erillisen lomakkeen kautta. Maanäytteiden lisäyksessä käyttäjä syöttää lomakkeelle peltolohkon ravinnearvot ja sen jälkeen sovellus tallentaa ne tietokantaan.

4.3.9 Lohkokortin tulostus

Lohkokortin tulostamisessa käyttäjä valitsee peltolohkon ja kasvukauden, jolta halutaan tulostaa lohkokortti. Kun käyttäjä painaa Tulosta –painiketta, sovellus noutaa näillä hakuehdoilla tarvittavat tiedot tietokannasta ja asettaa nämä lohkokorttipohjaan. Esikatselussa käyttäjä voi tarkastella tulostettavaa lohkokorttia sekä peruuttaa työn tai sitten jatkaa tulostukseen.

4.4 Tietokanta

Sovelluksen tietokanta on relaatiotietokanta. Tietokanta suunniteltiin vaatimusmäärittelyn pohjalta, jonka jälkeen tehtiin normalisointi, jossa tietojen toistaminen minimoidaan. Näin tietokanta on helppo pitää yhdenmukaisena, se on helppo päivittää ja se on muutosjoustava. Sovelluksen tietokanta on kuvassa 9. /2/



Kuva 9. Tietokanta.

5 TEKNINEN MÄÄRITTELY

5.1 Johdanto

Maatalouden lohkokirjanpitosovellus on työkalu, jonka avulla viljelijä pystyy pitämään kirjaa peltolohkoille tehdyistä toimenpiteistä. Viljelijä voi myös muokata tapahtumien tietoja jälkeenpäin sekä pystyy poistamaan tapahtuman tietokannasta.

Sovellus toteutetaan Java-ohjelmointikielellä NetBeans-ympäristössä ja sovellusta ajetaan suoraan käyttäjän koneelta. Sovelluksesta tehdään .jar paketti, joka sisältää kaikki vaadittavat kirjastot ja luokat ohjelmaa varten. Sovelluksen käyttämiseen ei siis tarvita verkkoyhteyttä.

5.2 Rakenne

Sovelluksen käyttöliittymä on kooditasolla jaettu kolmeen eri osaan. Tietokantapaketissa on tarvittava luokka pelkästään tietokantayhteyden muodostamista varten, logiikkapaketissa on luokka päivämäärävalitsintoiminnolle sekä käyttöliittymäpaketissa on luokat jokaiselle eri toiminnolle, jotka sisältävät myös tietokantatoimenpiteet sekä graafiset komponentit.

5.3 Moduulisuunnittelu

Sovellus suunniteltiin modulaariseksi, jolloin siihen on helppo lisätä tai poistaa toimintoja jälkeenpäin. Jokaisesta tietokantaan lisäystoiminnosta on esimerkiksi tehty oma luokkansa. Kuvassa 10 on sovelluksen käyttöliittymäpaketin luokkakaavio.

Kuva 10. Sovelluksen käyttöliittymäpaketin luokkakaavio.

6 TOTEUTUS

Seuraavassa esitellään sovelluksen toteutusta kylvötapahtuman lisäyksen osalta sekä miten kyseisen tapahtuman tiedot haetaan näkyviin pääikkunaan.

Kuva 11. Sovelluksen pääikkuna.

Aluksi käyttäjä painaa päänäkymässä Lisää kylvötapahtuma –painiketta, jonka seurauksena avautuu kylvötapahtuman lisäyslomake erilliselle sivulle. Lomakkeelle käyttäjä täyttää vaaditut tiedot, kuten lohkon nimi, päivämäärä, kylvötyyppi, siemenlajike, siemenmäärä, lannoitelajike ja lannoitteen määrä.

Lohkon nimen sovellus täyttää automaattisesti tarkistamalla, mikä kasvulohko on valittuna päänäkymässä ja kopioimalla tiedon Lohkon nimi –kenttään. Päivämäärän valinnassa käyttäjä painaa päivämääräkentän vierestä Valitse –painiketta, jolloin avautuu pieni kalenterinäkymä, josta käyttäjä pystyy valitsemaan halutun päivämäärän. Seuraavassa on koodeja PvmValitsin-luokan asetaValittuPvm-funktiosta.

```
public String asetaValittuPvm(){
    if(day.equals(“”))
      return day;
  java.text.SimpleDateFormat sdf = new java.text. SimpleDateFor
mat(“dd.MM.yyyy”);
  java.util.Calendar cal = java.util.Calendar.getInstance();
  cal.set(year, month, Integer.parseInt(day));
  return sdf.format(cal.getTime());
}
```

PvmValitsin-luokan konstruktorissa *day*-muuttujaan on tallennettu valittu päivä. Nyt asetaValittuPvm-funktiossa tarkistetaan ensin onko *day*-muuttuja tyhjä, ja jos on, palautetaan muuttujan arvo. Sen jälkeen muotoillaan päivämäärän muoto ja luodaan Calendar-luokasta instanssimuuttuja. Sitten muuttujaan asetetaan arvoksi valittu päivämäärä ja lopuksi funktio palauttaa valitun päivämäärän.

Seuraavassa on koodeja Lisaakylvo-luokan konstruktorista.

```
final JFrame f = new JFrame();
valitseButton.addActionListener(new ActionListener(){
    public void actionPerformed(ActionEvent ae){
         pvmTextField.setText(new PvmValitsin(f).asetaValittuPvm());
}});
```

Sovelluksen konstruktorin tehtävänä on luoda sovelluksen näkymä ja asettaa käyttöliittymän komponentteihin tapahtumankuuntelijoita. Tässä kyseessä olevassa konstruktorissa *valitseButton-komponentille* asetetaan tapahtumankuuntelija *addActionListener*-metodilla. Rajapintaluokka *ActionListener* sisältää metodin *actionPerformed*, jossa kerrotaan miten tapahtumaan reagoidaan. Tapahtumassa päivämäärä-kentän arvoksi asetetaan päivämäärä-valitsimesta valittu arvo. Päivämäärä-valitsimelle on annettu parametriksi *f*, joka viittaa luotuun *JFrame*-ikkunaan.

Kylvötyypin käyttäjä valitsee alasvetovalikosta, jonne on asetettu ennalta kaksi kylvötyyppiä, rivi- ja suorakylvö. Seuraavassa koodinpätkä, siitä miten alasvetovalikosta valittu arvo viedään tyyppi nimiseen muuttujaan. *getSelectedItem()*-funktio palauttaa valitun arvon ja *toString()*-funktio muuntaa sen vielä tavalliseksi merkkijonoksi.

```
tyyppi = kylvotyyppiComboBox.getSelectedItem().toString();
```

Loput kentistä käyttäjä täyttää syöttämällä käsin lukuarvot. Näiden viemisessä tekstikentästä muuttujaan käytetään *getText()*-funktiota sekä poistetaan *trim()*-funktiolla kenttään syötettyjen merkkijonojen alusta ja lopusta turhat välilyönnit.

```
siemen = siemenTextField.getText().trim();
siemenmaara = siemenmaaraTextField.getText().trim();
lannoite = lannoiteTextField.getText().trim();
lannoitemaara = lannoitemaaraTextField.getText().trim();
```

Muuttujien *siemenmaara* ja *lannoitemaara* arvoalueet tarkistetaan, ja jos ne ovat sallituilla alueilla, muuttujien tiedot viedään tietokantaan SQL-lausekkeen avulla. Tässä käytetään *Statement*-luokan *executeUpdate*-metodia. Metodille annetaan parametrina suoritettava SQL-lause *String*-muodossa. Metodi palauttaa int-tyyppisen tiedon, joka kertoo lauseen käsittelemien rivien määrän tai arvon nolla, mikäli lause ei palauttanut mitään.

```
String sql = "INSERT INTO Kyl-
vo(lohkoID,paivamaara,kylvotyyppi,siemen,siemenmaara,lannoite,lannoitemaa
ra) VALUES
('"+lohko+"','"+pvm+"','"+tyyppi+"','"+siemen+"','"+siemenmaara+"','"+lan
noite+"','"+lannoitemaara+"')";

st=yhteys.createStatement();
rivit = st.executeUpdate(sql);
```

Rivien määrä tarkistetaan päivityksen jälkeen ja jos metodi palauttaa käsiteltyjen rivien määräksi ”1”, sovellus ilmoittaa onnistuneesta lisäyksestä ja sulkee sen jälkeen lisäysikkunan ja palaa päänäkymään.

```
if(rivit==1){
JOptionPane.showMessageDialog(null,"Lisäys onnistui." );
dispose();
}
```

7 TESTAUS

7.1 Johdanto

Testausvaiheessa halutaan löytää virheitä ohjelmakoodista ja ohjelman käytöksestä. Yleensä ohjelmalle laaditaan testisuunnitelma, joka muodostuu testitapauksista. Tässä ohjelmointiprojektissa sovelluksen testausta on suoritettu koko projektin toteutuksen aikana sekä käytettävyystestauksena./3/

Toteutuksenaikaista testaamista suoritettiin aina jokaisen lisätyn toiminnon kohdalla kokeilemalla syöttää kenttiin hyväksyttyjä ja hylättäviä arvoja, esim. tyhjä syöte tai vääränmuotoinen syöte.

Käytettävyystestauksessa tuleva käyttäjä testasi sovellusta ja kokeili kaikki toiminnot läpi, näin ohjelmasta saatiin erilaisia korjausta vaativia virhetilanteita selville. Korjaukset ja parannukset veivät jonkin verran aikaa projektin toteutuksesta, joten kaikkia toimintoja ei ole vielä ehditty toteuttaa siihen.

7.2 Testausympäristö

Ohjelmaa testattiin eri Windows-käyttöjärjestelmillä, koska tulevan käyttäjän tietokoneella on tällä hetkellä Windows 7-käyttöjärjestelmä ja haluttiin varmistaa, että se toimii myös muilla versioilla, kuten XP ja Vista. Käyttäjän pitää varmistaa, että uusin JRE on asennettu tietokoneelle. Ohjelmakoodin kääntäminen suoritettiin NetBeansin tarjoamilla välineillä.

8 LOPPUYHTEENVETO

Opinnäytetyönä toteutettu sovellus vastasi hyvin asetettuihin tarpeisiin. Sovellus oli riittävän helppokäyttöinen ja nopeasti omaksuttava. Kaikkia vaatimusmäärittelyssä lueteltuja toimintoja ei ole vielä ehditty toteuttaa, mutta sovelluksen jatkokehitys jatkuu. Käyttöliittymä saatiin valmiiksi ulkoasultaan sekä tietokantapohjaa on helppo laajentaa jatkokehitystä ajatellen. Käyttöliittymään on helppo lisätä tulevaisuudessa lisäominaisuuksia tarpeen tullessa.

Projekti oli mielenkiintoinen ja haastava kokemus. Sain kehitettyä omaa osaamistani Java-ohjelmointikielen parissa sekä sain lisäkokemusta tietokantojen suunnittelusta ja niiden käyttämisestä. Opin myös ohjelmistotuotannon periaatteet, suunnitelmallisuuden ja rutiininomaisuuden.

LÄHTEET

/1/ Harju, J. & Juslin, J. 2009. Java-ohjelmointi. Jyväskylä. Gummerus Kirjapaino Oy.

/2/ Hovi, Huotari & Lahdenmäki 2003. Tietokantojen suunnittelu ja indeksointi. Porvoo. WS Bookwell.

/3/ Haikala, I. & Märijärvi, J. 2000. Ohjelmistotuotanto. Pieksämäki. RT-Print Oy.

/4/ Lambert, S. 2008. Microsoft Office Access 2007 Tehokas hallinta. Jyväskylä. Gummerus Kirjapaino Oy.