Windows-ohjelmointi II011300
KokkiMikko

Mikko Siloaho

14.4.2016

Ohjelmistotekniikan koulutusohjelma
Tekniikan ja liikenteen ala

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

SISÄLTÖ

1 JOHDANTO

Tässä raportissa on kuvattuna Jyväskylän ammattikorkeakoulun Windows-ohjelmointi (IIO11300) opintojakson harjoitustyö. Harjoitustyön teki Mikko Siloaho. Harjoitustyön tavoitteena oli suunnitella ja kirjoittaa C# WPF sovellus, jossa yksittäinen käyttäjä hallitsee ja käyttää yksinkertaista ruokaresepti tietokantaa. Käyttäjä voi valita erilaisista esimääritellyistä ruokatyypeistä yhden tai useamman parametrin hakua varten. Käyttäjä voi hakea kaikki mahdolliset reseptit tai rajata hakua hakusanalla. Käyttäjä voi halutessaan tulostaa reseptin paperille.

2 ASENNUS

Asennuspaketti toteutetiin Microsoftin Clickoncella. Ohjeet olivat selvät ja paketin kokoamisessa ei ollut muuta ongelmaa kuin ulkoisien tiedostojen lisääminen ja niiden käyttö koodissa.

Järjestelmä vaatimuksina ovat Windows 7, 8, 8.1, 10 (x86, x64) ja Microsoft .NET framework 4.5.2

Asennuspaketti löytyy Labranetin osoitteesta: \\ghost\temp\Install\G8499

3 TOIMINNALLISUUS

3.1 Toiminnalliset vaatimukset

3.1.1 Toteutetut

- Käyttäjä voi tehdä hakuja tietokantaan
- Käyttäjä voi valita yhden tai usemman reseptityypin haun parametriksi
- Käyttäjä voi asettaa hakusanan reseptin nimelle
- Käyttäjä voi valita vain yhden reseptin tarkasteltavaksi
- Käyttäjä voi muokata reseptin kaikkia tietoja
- Käyttäjä voi tulostaa reseptin paperille
- Käyttäjä voi tallentaa tekemänsä muutokset

- Käyttäjä voi luoda kokonaan uuden reseptin
- Käyttäjä voi poistaa reseptin
- Käyttäjä voi määrittää käytettävän tietokannan

3.1.2 Toteutumattomat

- Reseptin arpominen
- Reseptin luonnissa tai muokkaamisessa reseptityyppien määrittely
- Reseptin päättely raaka-aineiden perusteella
- Raaka-aineiden syöttäminen ohjelmalle

3.1.3 Rajaukset

- Sovellusta ja kantaa käyttää vain yksi henkilö

3.2 Ei toiminnalliset vaatimukset

3.2.1 Toteutetut

- Käyttäjän syötteiden tarkistus
- Ulkoasu on selkeä
- Layout on skaalautuva
- Tietokantana Mysql
- Ulkoisina tiedostoina XML
- Kolmikerros arkkitehtuuri
- Tilakone, joka säätelee käyttöliittymän toimintaa

3.2.2 Toteutumattomat

- Paikallinen SQLite –kanta
- Raaka-ainetiedot reseptille

3.2.3 Rajaukset

- Ulkoasu ei ole graafisesti näyttävä

4 KÄYTETYT TEKNIIKAT

4.1 WPF

Sovelluksen käyttöliittymä käyttää tavallisia WPF komponenteilla, kuten grideillä, dockpaneleilla, textboxeilla, datagrideillä ja katsottiin että ne ovat riittävät.

4.2 XAML

Komponenttien käsittely tehtiin pääasiallisesti XAML-puolella ja koodissa. XAML oli todella monipuolinen ja nopea tapa saada aikaan käyttöliittymä.

4.3 XML

Sovelluksen BL-tasolla on XML lukija, jonka tehtävä on lukea ulkoisesta XML-tiedostosta tietokantapalvelimen tiedot.

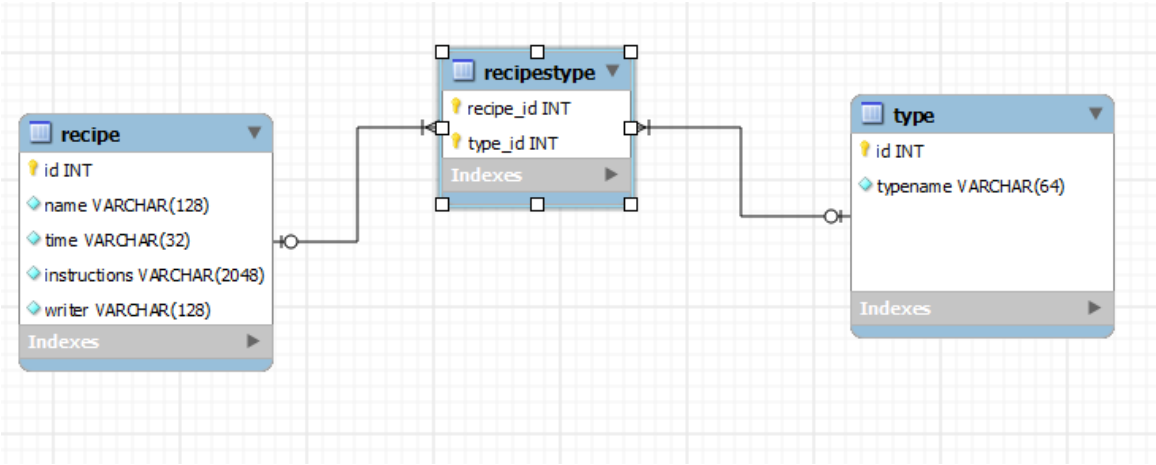
4.4 MySql

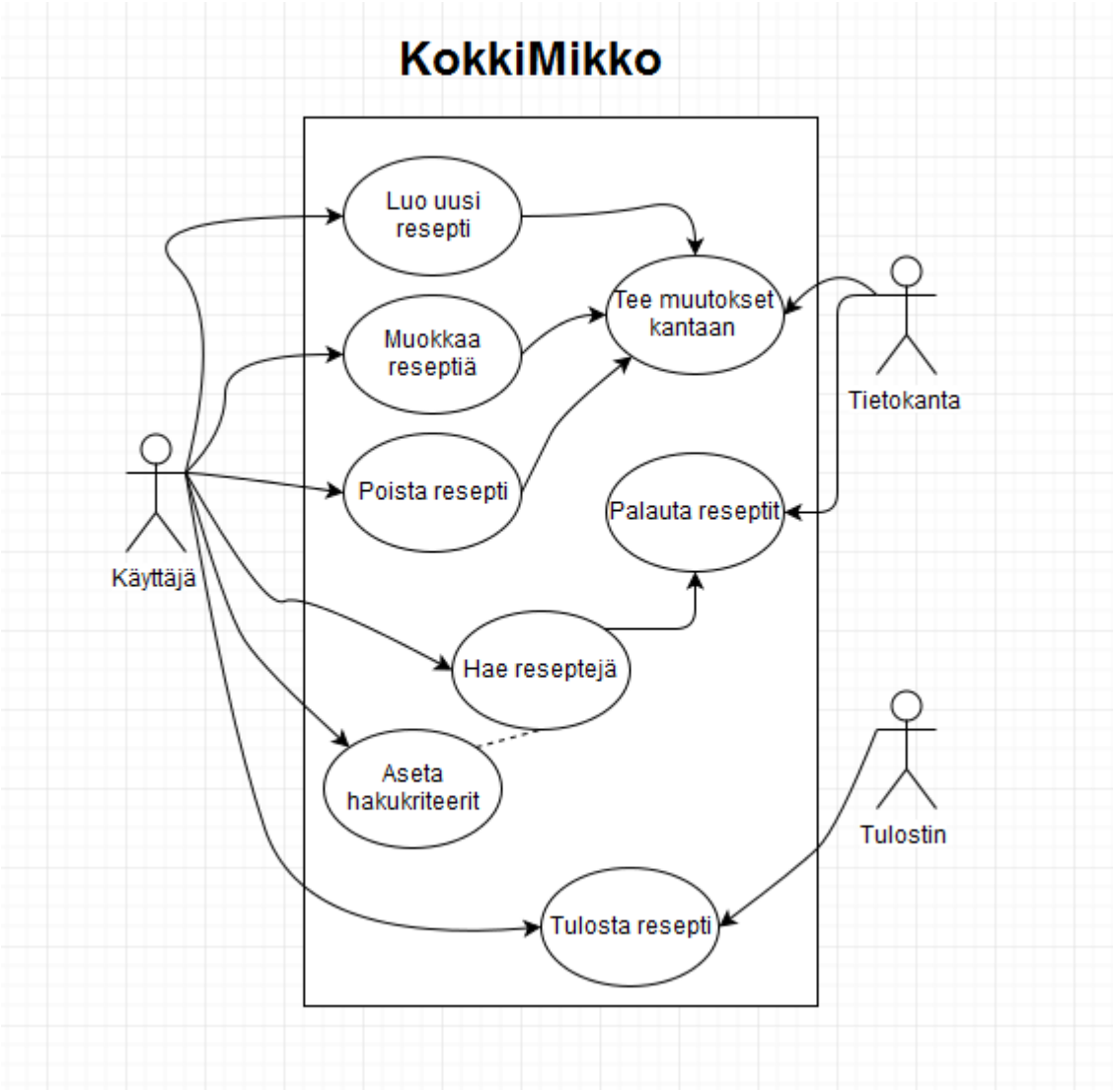
Tietokantana toimi Labranetin MySql kanta. Kanta suunniteltiin niin yksinkertaiseksi kuin mahdollista ja siitä löytyvät vain kaksi taulua ja yksi välitaulu. Ensimmäinen taulu on reseptit ja toinen taulu on reseptityypit ja näiden välillä on moni moneen yhteys. Sovelluksessa on käytössä staattinen luokka MySql.cs, jonka tehtävän on ylläpitää tietoa tietokantapalvelimesta ja sen asetuksista.

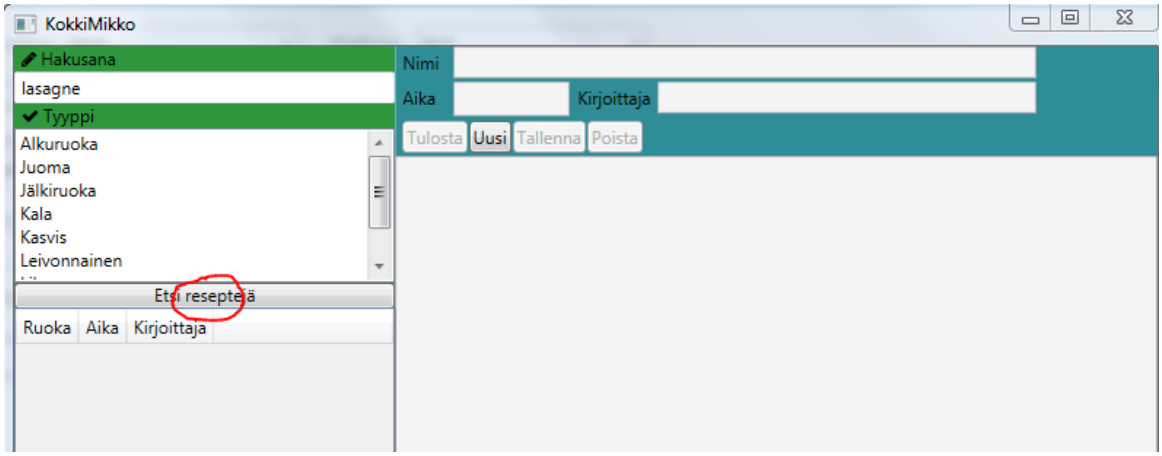
4.5 Arkkitehtuuri

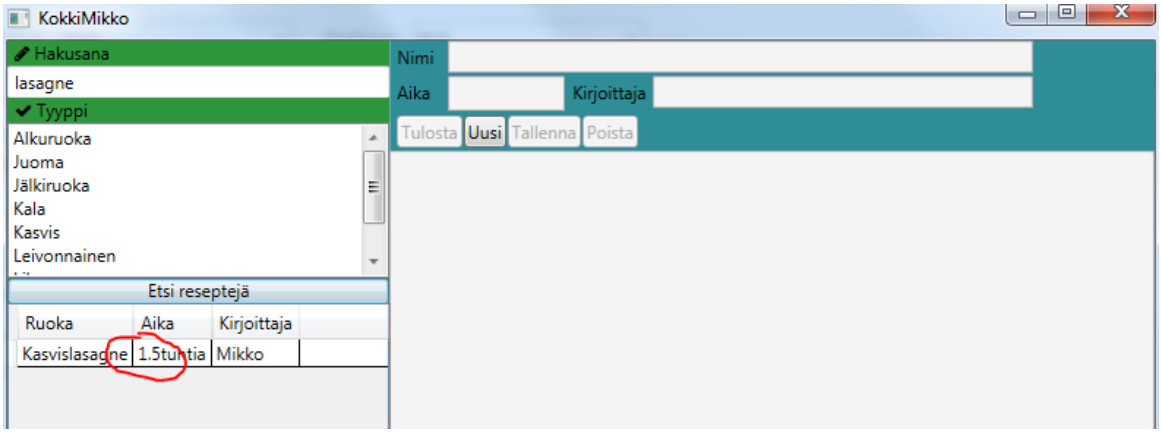
Sovellus käyttää perinteistä kolmikerrosarkkitehtuuria, joissa UI-kerros ottaa vastaan käyttäjän antaman syötteet ja tapahtumat. UI-kerros on niin sanotusti tyhmä ja sovelluksessa pyrittiin kaikki tiedon käsittely BL-tasolle. BL-taso, jossa käyttäjän syötteet tarkistetaan, ylimääräiset toiminnot, kuten XML-käsittely ja kutsut DB-tasolle. DB-tasossa tehdään kyselyt tietokantaan. sovelluksen virheenkäsittely toimi moitteetta.

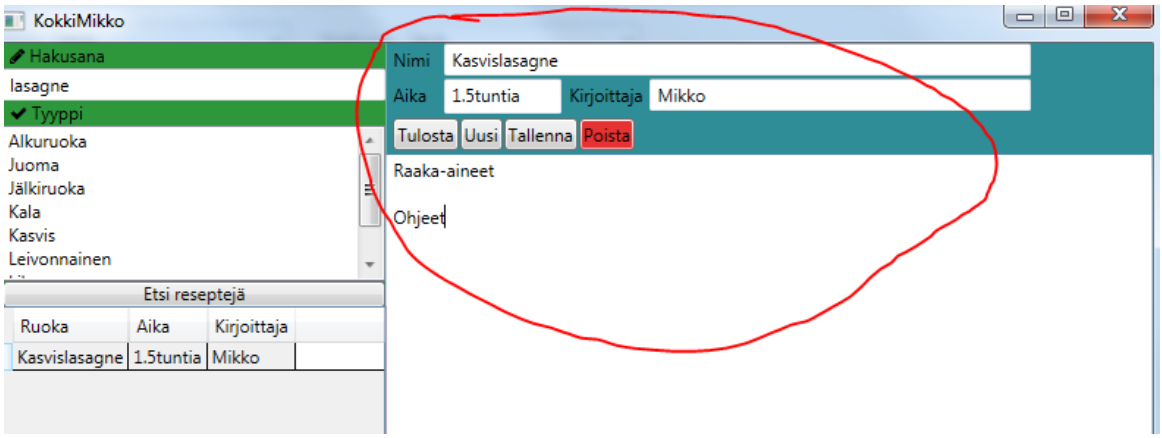
5 KUVARUUTUKAAPPAUKSET






KokkiMikko
Hakusana
lasagne
Tyyppi
Alkuruoka
Juoma
Jälkiruoka
Kala
Kasvis
Leivonnainen
Etsi reseptejä
Ruoka
Aika
Kirjoittaja
Nimi
Aika
Kirjoittaja
Tulosta
Uusi
Tallenna
Poista


KokkiMikko
Hakusana
lasagne
Tyyppi
Alkuruoka
Juoma
Jälkiruoka
Kala
Kasvis
Leivonnainen
Etsi reseptejä
Ruoka
Aika
Kirjoittaja
Kasvislasagne
1.5tuntia
Mikko
Nimi
Aika
Kirjoittaja
Tulosta
Uusi
Tallenna
Poista


KokkiMikko
Hakusana
lasagne
Tyyppi
Alkuruoka
Juoma
Jälkiruoka
Kala
Kasvis
Leivonnainen
Etsi reseptejä
Ruoka
Aika
Kirjoittaja
Kasvislasagne
1.5tuntia
Mikko
Nimi
Kasvislasagne
Aika
1.5tuntia
Kirjoittaja
Mikko
Tulosta
Uusi
Tallenna
Poista
Raaka-aineet
Ohjeet

6 TIETOKANNAN TESTIDATA

6.1 Reseptitaulu

```
INSERT INTO recipe (`name`, `time`, `instructions`, `writer`)
VALUES ('Karjalan piirakka', '2tuntia', 'Pyydä mummoa leipomaan', 'Jaska');
INSERT INTO recipe (`name`, `time`, `instructions`, `writer`)
VALUES ('Kalakeitto', '3tuntia', 'Keitä kalat ja vedä naamaan', 'Mr. Mister');
INSERT INTO recipe (`name`, `time`, `instructions`, `writer`)
VALUES ('Jallukola', '1tuntia', 'Heitä kokis menemään ja työnnä jallu naamaan', 'msilo');
INSERT INTO recipe (`name`, `time`, `instructions`, `writer`)
VALUES ('Lasagne', '6tuntia', 'Leivo ja syö', 'Jaska');
INSERT INTO recipe (`name`, `time`, `instructions`, `writer`)
VALUES ('Kasvislasagne', '1.5tuntia', 'Nom nom :3', 'Mikko');
INSERT INTO recipe (`name`, `time`, `instructions`, `writer`)
VALUES ('Kaljaa', '12tuntia', 'Avaa -> juo -> rince and repeat', 'Mikko');
INSERT INTO recipe (`name`, `time`, `instructions`, `writer`)
VALUES ('Läskisoosi', '6tuntia', 'Jauha hevonen -> keitä hevonen', 'Jaska');
INSERT INTO recipe (`name`, `time`, `instructions`, `writer`)
VALUES ('Smoothie', '1tuntia', 'Survo sekoiottimeen vedä hudlariin', 'Mr. Mister');
```

6.2 Tyyppitaulu

```
INSERT INTO `type` (`typename`)
VALUES ('Liha');
INSERT INTO `type` (`typename`)
VALUES ('Kala');
INSERT INTO `type` (`typename`)
VALUES ('Kasvis');
INSERT INTO `type` (`typename`)
VALUES ('Juoma');
INSERT INTO `type` (`typename`)
VALUES ('Leivonnainen');
INSERT INTO `type` (`typename`)
VALUES ('Alkuruoka');
INSERT INTO `type` (`typename`)
VALUES ('Pääruoka');
INSERT INTO `type` (`typename`)
VALUES ('Jälkiruoka');
```

6.3 Välitaulu

```
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (1, 3);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (1, 5);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (2, 2);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (2, 7);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (3, 4);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (4, 1);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (4, 7);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (5, 3);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (5, 7);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (6, 4);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (6, 7);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (7, 1);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (7, 6);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (7, 7);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (7, 8);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (8, 3);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (8, 6);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (8, 7);
INSERT INTO `recipestype` (`recipe_id`, `type_id`)
VALUES (8, 8);
```

7 JATKOKEHITYS

Sovellusta tullaan jatkokehittämään seuraavasti. Sovellus arpoo tai päättelee ja ehdottaa käyttäjälle reseptiä tietyin parametrein. Esimerkiksi käyttäjä voi ilmoittaa raaka-aine määränsä ja sovellus etsii sopivan reseptin. Käyttäjä voisi lisätä omia ruokatyyppejään ja muokata näitä. Käyttäjä voisi ryhmitellä reseptejä eri kategorioihin. Sovellus ottaisi yhteyden kantaan ja katosoisi onko kanta olemassa. Jos kantaa ei ole niin ohjelma luo kannan. Mahdollisuus käyttää SQLiteä ja pystyttää paikallisen tietokannan. BL ja DB luokkien refactorointi ja tekeminen muistinvaraukseltaan dynaamisiksi staattisten sijaan. Ja viimeisenä mutta ei vähäisempänä sovellus tulee käyttämään Entity frameworkia.

8 AJANKÄYTTÖ

Työn tekemiseen kului viikko ja ajallisesti arvioituna noin 40 tuntia. Vaikka tavoite oli selkeä,niin aikaa kului lähinnä tekniikoiden ja WPF ominaisuuksien opetteluun ja hyödyntämiseen. Kuten edellä mainittuna, harjoitustyö olisi pitänyt aloittaa ainakin viikkoa aikaisemmin.

9 MITÄ OPIN?

Työn suurin saavutus oli virheetön virheidenkäsittely ja käyttöliittymän toimivuus. Arkkitehtuurin kolme kerrosta toimivat ja kaikki virheet saatiin kehitysvaiheessa aina kiinni. Tämä nopeutti huomattavasti kehitystyötä. Ohjelma ei kaatunut kertaakaan kehitysvaiheessa. Ajan säästämisen vuoksi olisi pitänyt käyttää Entity Frameworkia, sillä tietokantojen käsittely on työlästä ja virhealtista. Sovellus olisi yksinkertaisuudesta huolimatta mielestäni pitänyt suunnitella paremmin. Lisäksi opin nuGet pakettien toiminnallisuutta ja niiden hyödyntämistä ja C# ohjelmointia. Tietokantojen tulosjoukkojen muuttaminen olioiksi oli tuttua. Onnistuin omasta mielestäni kohtuullisen hyvin tavoitteissani. Täten ehdotan itselleni 25/30 pistettä.