OULUN SEUDUN
AMMATTIKORKEAKOULU

Sami Karsikas

QT-DAISY-PERHEPÄIVÄHOITOSOVELLUS

QT-DAISY-PERHEPÄIVÄHOITOSOVELLUS

Sami Karsikas
Opinnäytetyö
Syksy 2012
Tietotekniikan koulutusohjelma
Oulun seudun ammattikorkeakoulu

TIIVISTELMÄ

Oulun seudun ammattikorkeakoulu
Tietotekniikan koulutusohjelma, ohjelmistojen kehitys

Tekijä(t): Sami Karsikas
Opinnäytetyön nimi: Qt-Daisy-perhepäivähoitosovellus
Työn ohjaaja(t): Juha Alakärppä
Työn valmistumislukukausi ja -vuosi: Syksy 2012 Sivumäärä: 44

Opinnäytetyön päämäränä oli rakentaa perhepäivähoitajille puhelimessa toimiva seurantaohjelmisto. Ohjelmaan kirjataan hoitoon tulevat lapset NFC-tagin avulla sisään ja ulos. Kirjaukset voidaan tehdä myös käsin. Ohjelmiston ansiosta manuaaliset virheet jäävät pois ja työaikojen seuranta saadaan tarkaksi ja reaaliaikaiseksi, koska puhelin kommunikoi jatkuvasti tietokannan kanssa. Lisäksi ohjelmisto tarjoaa kaiken lapsen hoitoon tarvittavan informaation perhepäivähoitajalle.

Ohjelma rakennettiin pääosin käyttäen Qt Creatorin versiota 4.7.4, joka oli sillä hetkellä uusin versio. Käyttöliittymä toteutettiin QML-ohjelmoinnilla. Sovellus testattiin pääasiassa Nokian 700-mallilla.

Projektissa saatiin tehtyä toimiva ohjelma, joka vastasi asiakkaiden vaatimuksia. Tietokantayhteydet toimivat ja käyttöliittymä rakennettiin asiakkaiden toiveiden mukaisesti. Ohjelman laajuus myös kasvoi projektin aikana. Ohjelmistoon toteutettiin automaattinen päivitys puhelimen datayhteyden kautta, jolloin se voidaan päivittää helposti esimerkiksi sadoille käyttäjille.

Asiasanat: Qt, QML, NFC, mobiiliohjelmat

ABSTRACT

Oulu University of Applied Sciences
Information Technology, Programming

Author(s): Sami Karsikas
Title of thesis: Qt-Daisy family day care program
Supervisor(s): Juha Alakärppä
Term and year when the thesis was submitted: Autumn 2012 Pages: 44

The target of the thesis was to build manager program for family nurses. Day care children can be checked in and out with NFC-tag. Checking is also possible to do via program UI. When using program manual mistakes are eliminated and working hours follow-up is exactly up to date because of real time database connections. The program offers also all needed information of children for family nurse.

The program was built mainly using Qt Creator version 4.7.4, which was the latest version. User interface was implemented with QML-programming. Testing is mainly done with Nokia model 700.

The final result of the project was working program which was made according to the customer's requirements. Database connections are working and user interface is built according to customer's expectations. The extent of program also increased during the project. Automatic update with mobile device data connection in the program makes updating fast and it can be done easily for example for hundreds of users.

Keywords: Qt, QML, NFC, mobile programs

ALKUSANAT

Opinnäytetyöni oli erittäin monipuolinen ja opettava opintojani vastaava projekti. Se kehitti ohjelmistotaitojani erityisesti Qt- ja QML-ympäristössä, josta onkin tulossa todennäköisesti erittäin tehokas työkalu tulevaisuuden ohjelmistotarpeisiin.

Kiitos SPDesign Oy:lle sekä työkavereille mukavasta työilmapiiristä.

6.9.2012

Sami Karsikas

SISÄLLYS

SANASTO

C++	Yksi suosituimmista ohjelmointikielistä
Client	Asiakasohjelma eli käyttöliittymä asiakasta varten
Funktio	Luokassa määritelty aliohjelma
GUI	Graphical User Interface, graafinen käyttöliittymä
JavaScript	Pääasiassa Web-ympäristössä käytetty komentosarjakieli
Luokka	Määrittelee jonkun tietyn oliojoukon yhteiset piirteet
NFC	Near Field Communication, radiotaajuutta käyttävä tiedonsiirtostandardi lyhyillä etäisyyksillä
Olio	Sisältää joukon loogisesti yhteenkuuluvaa tietoa ja toiminnallisuutta
Parserointi	Merkkijonon pilkkominen halutun tiedon esiin saamiseksi
QML	Qt Modeling Language, JavaScript-pohjainen kuvaava käyttöliittymien kehittämiseen tarkoitettu kieli
Qt-Creator	Alustariippumaton kehitysympäristö
Qt Quick	Käyttöliittymien suunnittelupaketti, johon kuuluu QML, Qt Creator sekä mahdollisuus C++-käyttöön
Rajapinta	Määritelmä, jonka mukaan eri ohjelmat voivat keskustella keskenään
SDK	Software Development Kit eli ohjelmiston kehitysympäristö

Symbian	Yksi Nokian puhelinten käyttöjärjestelmistä
Tietokanta	Kokoelma tietoja, joilla on yhteys toisiinsa
Widget	GUI:n komponentti, esimerkiksi painike tai ikkuna

1 JOHDANTO

Opinnäytetyöni aiheena on perhepäivähoitajille tehtävä Qt-mobiilisovellus, jolla hallitaan päivähoidossa olevien lasten sisään- ja uloskirjaukset NFC (Near Field Communication) -tagin avulla. Kirjaukset voidaan tarvittaessa merkitä myös suoraan käyttöliittymän kautta. Ohjelmiston avulla kaikki manuaalisen kirjanpidon virheet poistuvat ja perhepäivähoitajan työajat sekä hoitolasten tiedot saadaan luotettavasti ja reaaliaikaisesti tietokannasta.

Mobiilisovellus automatisoi toimintoja ja kirjanpitoa niin, että hoitajien ei tarvitse käytännössä keskittyä kuin työhönsä, lasten hoitamiseen. Etukäteen kirjatut lasten poissaolot helpottavat työaikojen suunnittelua sekä muun muassa ruokatilauksia. Kaikki tiedot ovat sähköisessä muodossa eli pysyvät tallessa ja ne ovat aina saatavilla tietokannasta. Tiedot ovat ajan tasalla eli jälkikäteen tehtäviä korjauksia ei tarvitse tehdä.

Ohjelmisto pitää sisällään allergiatiedot, muut erityistiedot, vanhempien tiedot, erilaiset poissaolokirjaukset lapsille sekä hoitajille sekä lapsien syömien aterioiden kulut lapsikohtaisesti ja kootusti hoitajakohtaisesti. Tiedot lisäävät turvallisuutta, kun esimerkiksi lasten erityistiedot kuten allergiat löytyvät helposti ja päivitettynä suoraan mobiilisovelluksesta.

Ohjelmisto toteutetaan Qt-sovelluskehitystyökaluilla. Käyttöliittymät toteutetaan JavaScript-pohjaisella QML (Qt Modeling Language) -kielellä graafikon suunnittelemien käyttöliittymäkuvien mukaisesti. Tietokanta- sekä NFC-yhteydet luodaan käyttäen Qt:n omia kirjastoja. Testaus hoidetaan Nokian Symbian-puhelimilla.

Ohjelmisto on osa suurempaa kokonaisuutta. Lisäksi tehdään erillisinä projekteina yrityksessä DaisyNet- ja DaisyManager-sovellukset. DaisyNet on päivähoidossa olevien lasten vanhemmille tarkoitettu web-pohjainen sovellus, jolla vanhemmat voivat editoida lastensa tietoja sekä tarkastella hoitoaikoja erilaisten raporttien ja graafien muodossa. DaisyManager on päiväkodin

johtajalle tarkoitettu hallintasovellus, jolla hallitaan perhepäivähoitajan työaikasuunnitelmat, palkanmaksun hyväksymiset sekä koko ohjelmistokokonaisuuden järjestelmänvalvojan osuus. (Daisy.)

Koska opinnäytetyöni oli asiakasprojekti, kaikkea opinnäytetyössä esitettyä toimintaa ei voida havainnollistaa koodiesimerkeillä salassapitovelvollisuuden takia.

2 VAATIMUSMÄÄRITTELY

Ohjelmiston on kirjattava lapset perhepäivähoitajalle tuotaessa NFC-tagin avulla sisään sekä haettaessa tagin avulla ulos. Tarvittaessa kirjaukset on voitava tehdä myös käyttöliittymän kautta käsin. Tiedot kirjauksista menevät tietokantaan. Samalla ohjelmisto pitää kirjaa hoitajan todellisesta työajasta sekä vertaa sitä ennalta määriteltyyn suunnitelmaan. Työaika lähtee käyntiin, kun ensimmäinen lapsi kirjataan sisään, ja loppuu, kun viimeinen lapsi kirjataan päivän päätteeksi ulos. Näin hoitaja näkee koko ajan jakson aikana tehdyt tunnit sekä sen, onko kokonaistilanne ylityön puolella vai onko työtunneissa vajaata. Sovelluksen kautta pitää pystyä lisäämään myös uusi lapsi NFC-tagille.

Ohjelmiston on hallittava kaikki lapsen informaatio, jonka hoitaja voi tarvita. Lapsen allergiatietojen sekä luvallisten lapsen hakijoiden yhteystietojen täytyy olla selkeästi saatavilla, jotta lapsen turvallinen hoito varmistuu. Myös ateriakulut täytyy listata ohjelmistossa niin, että hoitajan ei tarvitse pitää enää manuaalisesti ylhäällä lasten syömisten kustannuksia.

Käyttöliittymän tulee olla selkeä ja informatiivinen. Erilaiset värikoodit ilmaisevat lapsen ja perhepäivähoitajan tilan. Vihreä väri kertoo paikallaolosta, punainen poissaolosta ja sininen lomasta ja sairaslomasta. Navigoinnin sivujen välillä pitää olla selkeää ja toimia sujuvasti. Alasvetovalikoiden selauksen tulee olla mahdollista sekä hipaisemalla että nuolella.

Mobiiliohjelmiston on toimittava reaaliajassa ja oltava jatkuvasti yhteydessä tietokantaan päivittäen näkymiä säännöllisesti. Mahdollisten virhetilanteiden varalta puhelin tallentaa edellisen käyttökerran tiedot, jolloin esimerkiksi datayhteyden puuttuessa saadaan kuitenkin oikeat tiedot näytölle.

Äänet toimivat ohjelmistossa käyttöä selkeyttäen ja tukien. Sisään- ja uloskirjauksessa äänimerkki sekä värinä ilmoittavat kirjauksen onnistumisesta tai epäonnistumisesta.

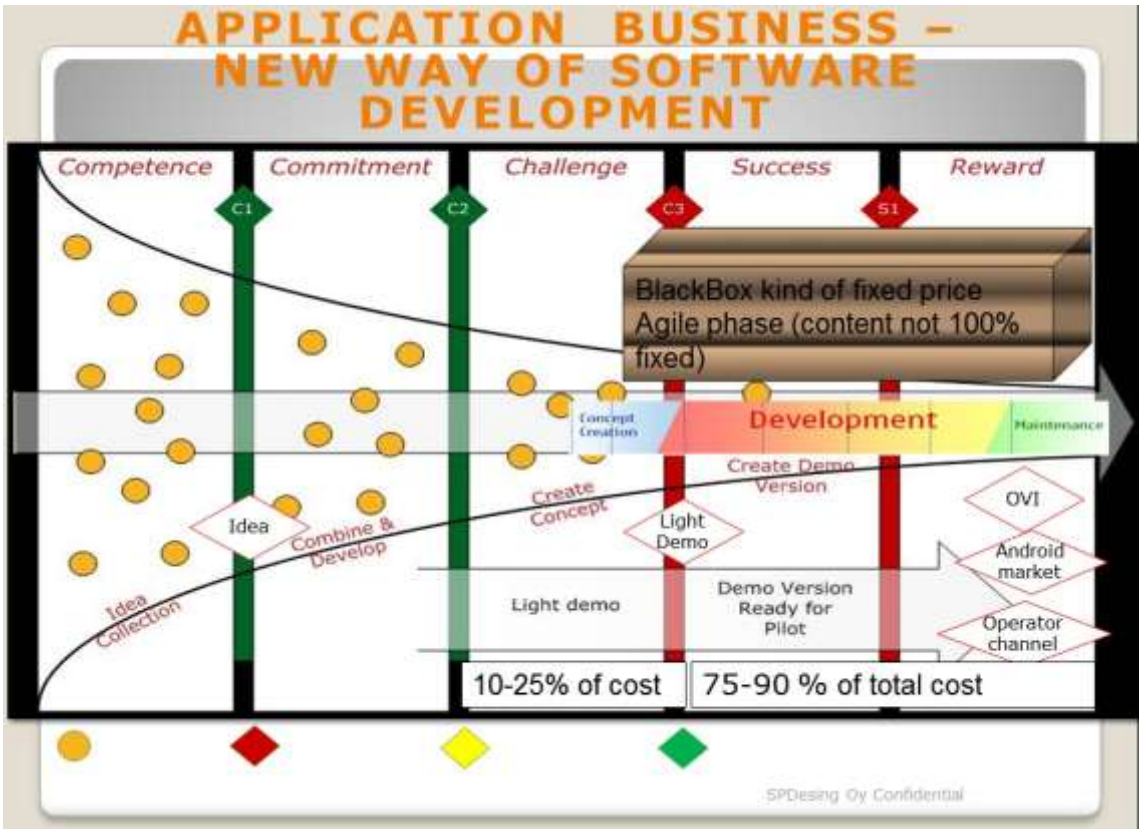
3 FAST VISUAL DEMO

Ohjelmisto rakennettiin käyttäen tehokasta Fast Visual Demo -menetelmää. Menetelmässä mietitään ensin sovellusidea, joka on toteutuskelpoinen. Seuraavassa vaiheessa rakennetaan ensin tavoiteohjelmaa vastaava käyttöliittymä ilman oikeaa toiminnallisuutta. Tämän avulla todellisen ohjelman toimivuutta on helppo simuloida ja esitellä esimerkiksi asiakkaalle. Ohjelmiston hyödyllisyyden tai asiakkaan ostopäätöksen varmistuttua rakennetaan lopullinen ohjelmisto oikealla toiminnallisuudella. (Application business - New way of software development.)

3.1 Edut

Fast Visual Demo -menetelmässä tehdään ensin ohjelmistosta visuaalinen esittelyversio asiakkaalle. Hyvän graafikon ja tehokkaan tiimin avulla näyttävä ja toimivan näköinen käyttöliittymä tehdään nopeasti ilman oikeasti toimivaa tietokantaa tai muuta lopputuotteen toiminnallisuutta. Tällä tavalla asiakas näkee heti konkreettisesti, mitä on tilaamassa. Näin tilaaja voi tehdä mahdollisen ostopäätöksen hyvinkin nopeasti. (Application business - New way of software development.)

Tyypillisesti tällaisen visuaalisen esittelyversion kehittämiseen ja tekemiseen menee vain 10–25 % lopputuotteen kustannuksista. Menetelmän avulla saadaan siis melko pienellä riskillä lopputuotteen näköinen käyttöliittymä. (Kuva 1.)



KUVA 1. Fast Visual Demo prosessi (Application business - New way of software development)

3.2 Riskit

Periaatteessa on mahdollista, että toimivan käyttöliittymän rakentamisen jälkeen ohjelman oikeassa toteuttamisessa tuleekin vastaan ongelma. Se voi olla mahdoton ratkaista tai sitten se vaatisi liikaa resursseja ohjelman kokonaisuuteen nähden. Näin ohjelman kehittämisestä voidaan joutua luopumaan. (Application business - New way of software development.)

4 QT JA QML

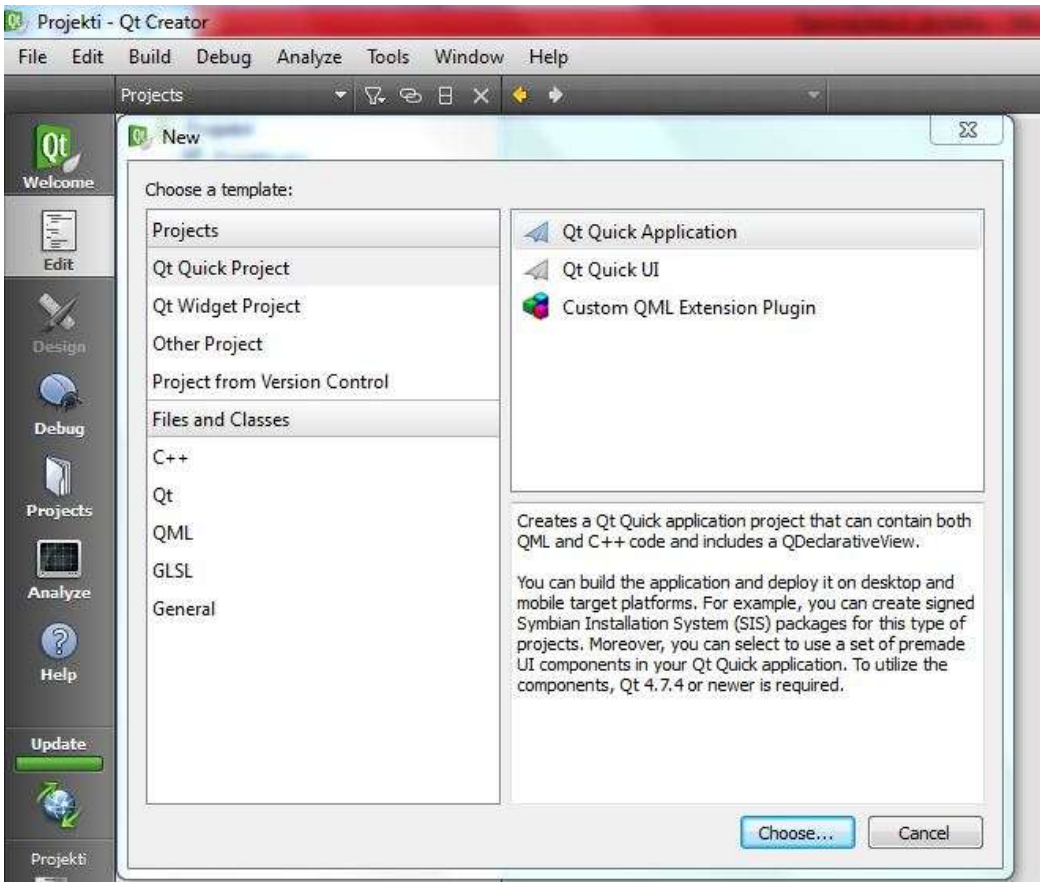
4.1 Yleisperiaatteet ja Qt:n tulevaisuus

Qt on alustariippumaton ohjelmistojen ja graafisten käyttöliittymien kehitysympäristö. Graafisen kehittämisen takia Qt onkin luokiteltu widget-työkaluksi. Ohjelmistoympäristöä käytetään myös palvelinpuolen konsoliohjelmistojen kehittämisessä sekä perinteisissä komentoriviltä ajettavissa ohjelmistoissa. (Qt-kehitysympäristö. 2012.)

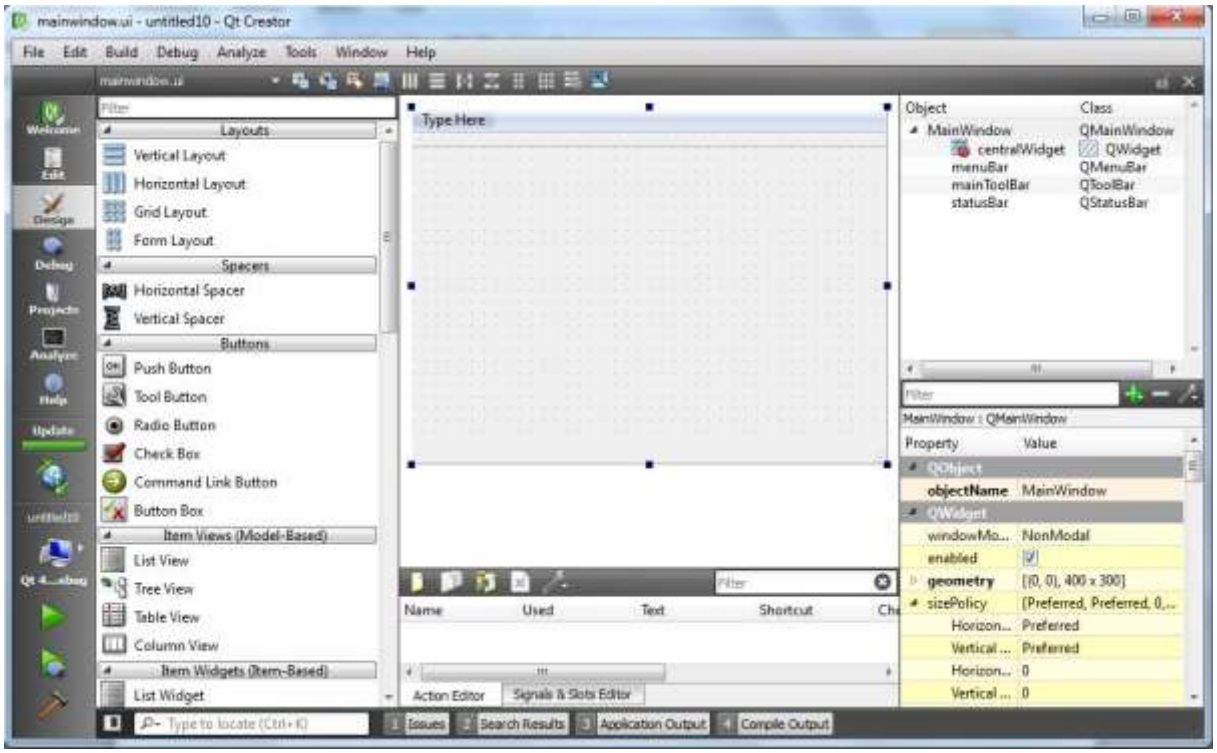
Qt-sovelluskehitysympäristöä käyttää maailmanlaajuisesti yli 450 000 sovelluskehittäjää ja tuhansia eri yhtiöitä, joten kyseessä on erittäin merkittävä kehitysympäristö. Nokia on aikaisemmin omistanut Qt:n ja vastannut sen kehittämisestä, mutta 9.8.2012 Nokia myi Qt-ohjelmistoteknologian ja liiketoiminnan kokonaisuudessaan Digialle. Tällä hetkellä Qt:llä on mahdollista kääntää sovellukset Windows-, Linux-, FreeBSD- sekä Mac OS -työpöytäkäyttöjärjestelmille, mobiilipuolelta Symbianille sekä Linux-pohjaisille Maemo- ja Meego-käyttöjärjestelmille. Digian tarkoituksena on laajentaa Qt:tä tukemaan myös Android-, Applen iOS- sekä Windows 8 -käyttöjärjestelmiä. Näin Qt:n alustariippumattomuus ja käyttökohteet tulevat kasvamaan entisestään. (Digia ostaa koko Qt-kehitysympäristön Nokialta. 2012.)

4.2 Qt Creator

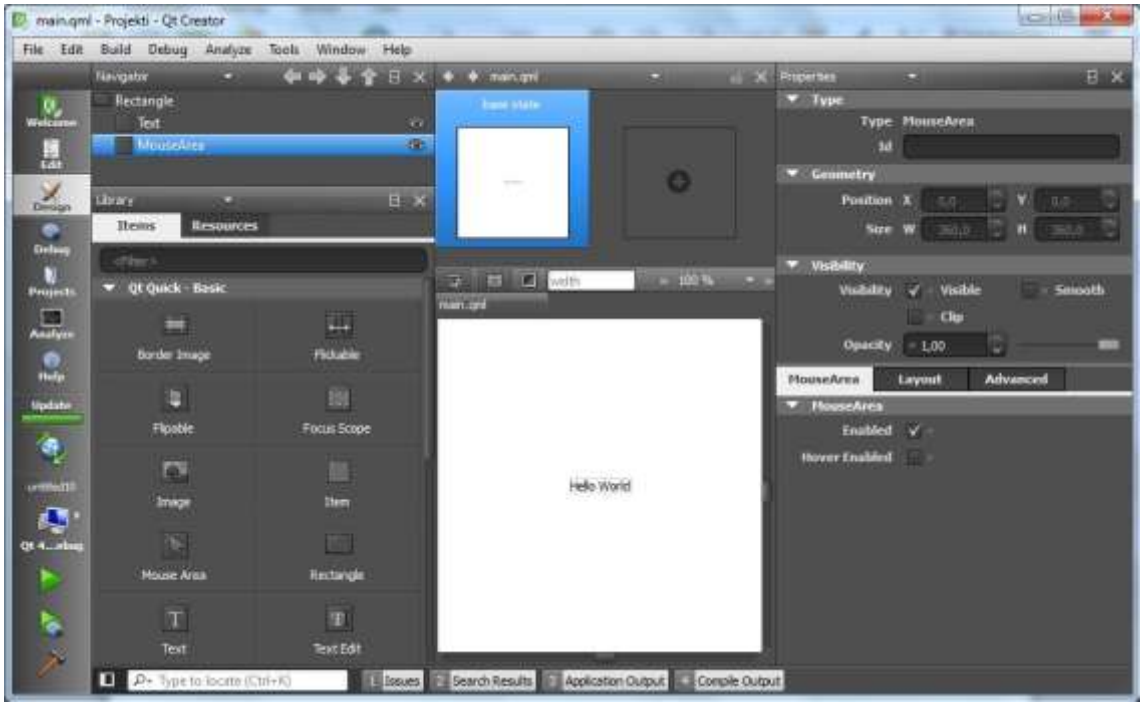
Qt Creator on ohjelmisto, jolla varsinainen Qt-ohjelmointi tapahtuu. Ohjelmalla luodaan aluksi halutunlainen projekti. Qt Quick Projectista saadaan luotua projekti, jossa käyttöliittymät voidaan ohjelmoida käyttäen QML:ää, ja siihen saa integroitua myös Qt C++ -kirjastot. (Kuva 2.) Qt Widget Projektista taas luodaan ohjelmisto, jossa graafinen käyttöliittymä saadaan rakennettua Qt:n omilla kirjastoilla ja komponenteilla. Qt Creatorissa on molempia graafisia käyttöliittymiä varten olemassa myös graafinen suunnittelutyökalu, joka helpottaa ja nopeuttaa käyttöliittymien rakentamista huomattavasti. (Kuva 3 ja kuva 4.)



KUVA 2. Qt-projektin luominen



KUVA 3. Qt Widget-tyyppisen GUI:n graafinen editori



KUVA 4. Qt QML-tyyppisen GUI:n graafinen editori

4.3 QML

Qt Modeling Language on JavaScript-pohjainen käyttöliittymien rakentamiseen suunniteltu kieli. QML on pääasiassa käytössä mobiilikäyttöliittymien rakentamisessa, jossa tavoitteena on rakentaa näyttäviä animoituja käyttöliittymiä, joita ohjataan kosketusnäytöltä. (QML 2012.) Koska QML on JavaScript-pohjainen, koodin sekaan voidaan lisätä normaalia JavaScript-koodia suoraan tai sisällytettyjen JavaScript-tiedostojen avulla. Tätä käytettiin myös projektin tekemisessä. Yleisimmin käytetyt funktiot lisättiin erilliseen JavaScript-tiedostoon, joka sisällytettiin aina siihen QML-tiedostoon, jossa funktioita haluttiin käyttää. Esimerkki erillisen JavaScript-tiedoston käyttämisestä nähdään kuvissa 5, 6 ja 7.

```
function changeDateFormat(strDate) {
    var temp = strDate.split(".");
    var year = temp[2];
    var month = temp[1];
    var day = temp[0];
    var strFormattedDate;
    strFormattedDate = year + "-" + month + "-" + day;
    strFormattedDate = strFormattedDate + " 00:00:00";
    return strFormattedDate;
}
```

KUVA 5. Funktio päivämäärän formaatin muuttamiseen erillisessä comfunctions.js -tiedostossa

```
import com.nokia.symbian 1.1
import Qt.labs.components 1.1
import com.nokia.extras 1.1
import QtQuick 1.1
import "comfunctions.js" as ComFunctions
```

KUVA 6. QML-tiedostossa JS-tiedosto sisällytetään rivillä 5

```
onClicked: {
            var dateStart;
            dateStart=ComFunctions.changeDateFormat(dateText.text);
```

KUVA 7. JavaScript-funktio voidaan nyt suorittaa QML-tiedostossa

5 OHJELMOINNIN TOTEUTUS

Ohjelmiston tekeminen toteutettiin kuuden hengen ryhmässä. Versionhallinta mahdollisti tehokkaan yhtäaikaisen työskentelyn. Versionhallintaohjelmistona toimi TortoiseSVN, joka integroitiin Qt Creatoriin. Taskit eli tehtävät pidettiin jaetussa Google Docs -dokumentissa, josta päivä- ja viikkopalaverissa jaettiin taskit ja katsottiin sen hetkinen tilanne. Käyttöjärjestelmänä oli 64 bittinen Windows 7. Tekstieditorina käytettiin apuna Notepad++-versiota 5.9.3. Tiedostojen jakaminen ryhmän kesken tapahtui Dropboxin avulla. Ohjelmiston testaus suoritettiin Qt Creatorissa olevalla simulaattorilla sekä tarkemmin Nokian 700-, 500- sekä C7-malleilla. Varsinainen ohjelmistotyö suoritettiin uusilla tehokkailla kannettavilla tietokoneilla.

5.1 Käyttöliittymä

Ohjelmiston tekeminen aloitettiin käyttöliittymän tekemisellä QML:llä. Graafikon piirtämien käyttöliittymäkuvien ja grafiikan avulla toteutettiin nykyaikainen graafinen käyttöliittymä. Qt Creatorissa on myös graafinen editori QML:ää varten, mikä nopeuttaa käyttöliittymien toteuttamista.

QML mahdollistaa komponenttien käytön. Esimerkiksi jos käytetään paljon vakiotyylisiä näppäimiä, on järkevää rakentaa siitä komponentti omaan QML-tiedostoon. Tätä rakennetta käytettiin ohjelmistossa muutaman vakiokomponentin tekemiseen. Komponentti määritetään halutuilla ominaisuuksilla ja se luodaan sitten varsinaiseen käyttöliittymään komponentin QML-tiedoston nimellä. Esimerkki Button-komponentista on kuvissa 8, 9 ja 10.

CommonButton.qml* Image

```
1  import QtQuick 1.1
2
3  Image {
4      id: commonButton
5      property string buttonText: ""
6      signal clicked
7      width: 160
8      height: 55
9      source: "nappula.png"
10
11
12     Text {
13         text: commonButton.buttonText
14         anchors.fill: parent
15         verticalAlignment: Text.AlignVCenter
16         horizontalAlignment: Text.AlignHCenter
17         font.pixelSize: 20
18         wrapMode: Text.WordWrap
19     }
20
21     MouseArea {
22         anchors.fill: parent
23         onClicked: {
24             commonButton.clicked();
25         }
26     }
27 }
28
```

KUVA 8. QML-komponentti

```
CommonButton{
    id: confirmationYesButton
    anchors.bottom: parent.bottom
    anchors.bottomMargin: 10
    x:5
    buttonText: "Kyllä"
    onClicked: {
        //Tässä kohdalla kirjoitetaan
        // mitä "Kyllä"-näppäimen painalluksesta
        // tapahtuu
    }
}
```

KUVA 9. Komponentin käyttö koodissa

KUVA 10.Varmistusikkunassa oleva Kyllä-painike komponenttina

Ohjelmaan tuli paljon alasvetovalikkoja. Ne rakennetaan QML:ssä ListView-elementillä. Alasvetovalikoiden sisältöjä täytyi pystyä painamaan ja sinne täytyi saada varsinainen data yleensä tietokannasta. ListView käsittää kolme osaa. Varsinainen ListView määrittää alasvetovalikon koon ja sijainnin. ListModel pitää sisällään datan, jonka ListView näyttää. ListDelegate määrittelee, kuinka sisältö näytetään. (QML ListView Element. 2012.)

QML-näkymän peruskomponentti on Rectangle. Käyttöliittymän pohjalle tehtiin ensin puhelimen resoluution kokoinen vihreä suorakulmio. Sen päälle tehtiin jälleen suorakulmio tummemman vihreällä värillä, johon Text-komponenttiin tuli teksti Perhepäivähoitaja sekä myös hoitajan nimi. Käyttöliittymän painikkeet tehtiin Image-komponentilla, jonka sisälle asetettiin Image-komponentin kokoinen MouseArea, joka reagoi painalluksiin. Image-komponentin sisälle tuli myös Text-komponentti, johon asetettiin painikkeen nimi. Pääsuorakulmion päälle lisättiin yläreunaan päivämäärä ja aika JavaScriptin Date-oliolla valmiina header-komponenttina. Lapsien listaus tehtiin ListView-elementillä, joka soveltuu tarkoitukseen erinomaisesti. ListViewiin voidaan lisätä ja muokata tietoa ja ominaisuuksia dynaamisesti ja sitä voi selata kosketusnäyttöä hipaisemalla. Listview-komponentin elementteihin asetettiin vielä MouseArea, jotta lapsen nimeä painamalla päästään lapsen tiedoista kertovalle käyttöliittymäsivulle. (Kuva 11.) Kuvissa 12-16 esitellään kuvan 11 käyttöliittymänäkymän toteutusta.

KUVA 11. Esimerkki QML-käyttöliittymästä

```
Rectangle {
    width: 360 //leveys
    height: 640  //korkeus
    id: main_rect //suorakulmion id
    color: "#värikoodi" //suorakulmion värin asettaminen
```

KUVA 12. Pääsuorakulmion tekeminen

```
Rectangle{
        id: rectangle2  //suorakulmion id
        width: 360   //leveys
        height: 200  //korkeus
        color: "#värikoodi" //suorakulmion värin asettaminen
       anchors.top: header1.bottom  //ankkuri joka asettaa kulmion
       //yläreunan header1-komponentin alareunan tasolle
```

KUVA 13. Tummemman suorakulmion tekeminen pääsuorakulmion sisään

```
        Image {
            id: image1  //kuvan id
            x: 9 //kuvan x-suuntainen vasemmasta reunasta
            y: 48  //kuvan y-suuntainen sijainti ylhäältä
            width: 165 //kuvan leveys
            height: 47  //kuvan korkeus
            anchors.bottom: image3.top  //ankkuri joka asettaa kuvan
           //alareunan image3:n yläreunan tasolle
            anchors.bottomMargin: 6 //marginaali
            source: "nappula.png" //käytettävän kuvatiedoston nimi
            MouseArea {        //painallukseen reagoiva MouseArea
                id: kulutYht   // MouseArean id
                anchors.fill: parent//koko tehdään image1:n kokoiseksi
                onClicked: { //klikattaessa suoritetaan seuraava rivi
                    myQmlObject.setQml(1)
                }
            }
            Text {
                id: totalMealText // tekstin id
                x: 35 //tekstin x-suuntainen sijainti
                y: 10 //tekstin y-suuntainen sijainti
                text: "KULUT YHT." //tekstin teksti
                anchors.horizontalCenter: parent.horizontalCenter
                anchors.verticalCenter: parent.verticalCenter
               //tekstin keskitys parentin eli image1:n mukaan
                font.pixelSize: 20 //tekstin koko
            }
        }
```

KUVA 14. Yhden näppäimen tekeminen tummemman suorakulmion sisälle

ListView {

```
        id: peopleList //lisviewin id
        model: peopleModel //listviewin käyttämä model
        delegate: peopleDelegate // listviewin käyttämä delegate
        x: 0 //listviewin x-suuntainen sijainti
        y: 246 //listviewin y-suuntainen sijainti
        height: 300 //listviewin korkeus
        width: 360 //listviewin leveys
       clip:true  //listviewin sisältö pysyy tarkalleen listviewin
       //sisällä
    }
```

KUVA 15. ListView-komponentin tekeminen

```
Item{
        Timer {
            interval: 500; running: true; repeat: true
            onTriggered: {
                dateText.text = Qt.formatDateTime(new Date(),
"dd.MM.yyyy");
                timeText.text = Qt.formatDateTime(new Date(),
"HH:mm");
            }
        }
    }
```

KUVA 16. Yläreunan ajan ja päivämäärän päivittäminen 500 ms:n välein

5.2 QML:n ja C++:n yhteistoiminta

Ohjelmisto oli melko laaja ja siksi heti alusta alkaen oli selvää, että QML:n sisältämät kirjastot ovat täysin riittämättömiä koko ohjelman kehittämiseen. Niinpä piti selvittää, miten QML ja Qt C++ kommunikoivat keskenään. Seuraavissa kuvissa (kuva 17, kuva 18 ja kuva 19) on toteutuksesta koodiesimerkki.

```
view->rootContext()->setContextProperty("myQmlObject",this);
```

KUVA 17. QML-objektin luominen, jolla C++-funktioita voidaan kutsua QML:stä

```
Q_INVOKABLE void setQml(int qmlNumber);
```

KUVA 18. Funktio esiteltävä header-tiedostossa Q_INVOKABLE-etuliitteellä

```
MouseArea {
                id: kulutYht
                anchors.fill: parent
                onClicked: {
                    myQmlObject.setQml(1)
                }
            }
```

KUVA 19. Qt C++ -funktion kutsuminen(setQml) myQmlObjectin avulla QML-koodista

5.3 Tietokanta

Tietokantayhteydet rakennettiin ohjelmiston ja palvelimen välillä Qt:n omia kirjastoja käyttäen. QNetworkAccessManager mahdollisti tämän yhteydenpidon. Palvelimen palautetta piti vielä parseroida kovasti, jotta saatiin kulloiseenkin tilanteeseen oikeat tiedot.

Varsinaisena tietokantajärjestelmänä toimii Microsoftin Sql Server 2008 R2, jota suoritetaan Microsoftin Windows Server 2008 R2 käyttöjärjestelmässä. Kannan hallintaa varten rakennettu ohjelmisto on rakennettu asp.net C# -ohjelmistokielellä. Lisäksi käytössä on .NET framework 3.5, joka on Microsoftin kehittämä ohjelmistokomponenttikirjasto. (NET Framework. 2012.)

Tietokantayhteys muodostetaan QnetworkAccessManager-luokan oliolla network, jonka vastaus tallennetaan QnetworkReply-luokan olioon reply. Samalla käytetään Qt:n signal-slot-periaatetta eli kun palaute on saatu, suoritetaan slot nimeltään slotRequestFinished. Siellä saatu tieto käsitellään oikeaan muotoon ja siirretään esimerkiksi QML-näkymissä oikeisiin paikkoihin. Kuvassa 20 esitellään tietokantahaun funktio. (QNetworkAccessManager Class Reference. 2012.)

```
    QNetworkAccessManager *network;
    QNetworkReply *reply;

void HttpHandleri::httpGet()
{
    QNetworkRequest request;
    request.setUrl(QUrl(url));
    reply = network->post(request,postData.encodedQuery());
    QObject::connect(reply,
    QObject::connect(network, SIGNAL(finished(QNetworkReply *)),this,
                          SLOT(slotRequestFinished(QNetworkReply *)));
}
```

KUVA 20. Tietokantahaun toteuttaminen sovelluksessa

5.4 NFC

Near Field Communication on RFID:hen pohjautuva radiotaajuisen etätunnistuksen mahdollistava tekniikka.NFC toimii molempiin suuntiin. Se voi olla lukijalaitteena ja tunnisteena, mikä onkin suurin ero verrattuna RFID-laitteisiin. (Near Field Communication. 2012.)

NFC:tä tukevia puhelimia löytyy nykyään jo useilta valmistajilta. Opinnäytetyön testauksessa käytettiin pääosin Nokian 700- ja 701-malleja, joilla ohjelmistoa on myös tulevien asiakkaiden tarkoitus käyttää.

Kirjattaessa lasta sisään tai ulos NFC-tagilta saadaan lapsen id, jonka perusteella suoritetaan tietokantaan lapsen statuksen vaihto. Jos lapsi on ensimmäinen, joka kirjataan sisään sinä päivänä, aloitetaan hoitajan työaika. Lapsen ollessa viimeinen, joka kirjataan ulos, lopetetaan hoitajan työajan kertyminen.

Qt:ssa on QNearFieldManager-luokka, joka mahdollistaa NFC:n käyttämisen. Luokan funktiolla startTargetDetection() aloitetaan NFC-lukijan kuunteleminen. Luokan olio kytketään signal-slot periaatteella erilliseen funktioon, jossa QnearfieldTarget-oliolta saadaan uid()-funktiolla tagilta id sovelluksen käytettäväksi. Näin sovellus tarkkailee puhelimen NFC-lukijaa koko sen käynnissä olevan ajan. Puhelimen lukijan havaittua NFC-tagin lukuetäisyydellä, tagi luetaan välittömästi. (QNearFieldManager Class Reference. 2012.)

5.5 Päivitykset

Ohjelmisto tulee yhtä aikaa käyttöön jopa sadoille käyttäjille. Siksi ohjelmiston päivitykset oli saatava toimimaan automaattisesti kaikille puhelimen datayhteyden kautta. Nyt ohjelmisto tarkistaa aina käynnistyessään, löytyykö uutta versiota. Jos uusi versio löytyy, ohjelmisto kysyy, haluaako käyttäjä päivittää sovelluksen. Käyttäjän hyväksyttyä päivityksen uusi sis-tiedosto ladataan automaattisesti palvelimelta sekä asennetaan se puhelimeen.

5.6 Kalenteri

Koska ohjelmistoa käytetään puhelimella, oli järkevää tehdä päivämäärien valinnat visuaalisesti kalenteria klikkaamalla. Sopiva kalenteri löytyikin Qt:n QCalendarWidget-luokasta. Sen avulla kalenteri rakennettiin ja nyt käyttäjä voi valita haluamansa päivämäärän selkeästä kalenterista valitsemalla. Kalenteri on toteutettu Qt-C++ -koodissa. Se on ohjelmiston taustalla koko ajan käynnissä ja se saadaan esille piilottamalla näkymä, joka näyttää sen hetkistä QML-tiedostoa. Käyttäjän painettua haluttua päivämäärää se asetetaan päivämääräkenttään ja näytetään näkymä, jolloin kalenteri siis jää piiloon taakse. (Kuva 21.)

KUVA 21. Päivämääräkenttää painamalla aukeaa kalenteri josta haluttua päivää painamalla päivämäärä asettuu kenttään

5.7 Äänet ja värinät

Ohjelmiston käyttämisen selkeyttämiseksi ja helpottamiseksi siihen oli lisättävä informoivat äänet. Ohjelmistoa saatetaan käyttää esimerkiksi kirkkaassa päivänvalossa, jolloin puhelimen näytöltä voi olla vaikea seurata ohjelmiston toimintaa. Siksi lapsen kirjautuessa sisään tai ulos äänimerkin on ilmoitettava joko kirjauksen onnistumisesta tai epäonnistumisesta.

Qt sisältää QSound-luokan, jonka avulla voidaan soittaa äänitiedosto halutuissa kohdissa (kuva 22). Projektissa tämä toteutettiin erillisellä luokalla, jossa äänifunktiot hallinnointiin. Funktiot määritettiin Q_INVOKABLE-etuliitteellä, jolloin niitä voidaan suorittaa mistä tahansa myös QML-koodista. (QSound Class Reference. 2012.)

```
QSound *soundplayer = new QSound(“sound.wav”);

soundplayer->play();
```

KUVA 22. QSound olion tekeminen ja äänen soittaminen

Ohjelmistoa saatetaan myös käyttää hyvin kovaäänisessä ympäristössä, jolloin äänimerkkikään ei anna informaatiota kirjauksen onnistumisesta. Sitä varten ohjelmistoon laitettiin myös värinä indikoimaan kirjauksen toimintaa. Qt:n kirjasto QFeedbackHapticsEffect sisältää funktiot ja pääsyn puhelimen värinän hallinnointiin (kuva 23). Värinälle voidaan antaa erilaisia parametrejä, kuten kesto ja intensiteetti. Myös tämä toteutettiin projektissa erillisellä luokalla ja Q_INVOKABLE-funktiolla, jotta värinä saatiin asetettua haluttuihin kohtiin ohjelman suorituksen mukaan. (QFeedbackHapticsEffect Class Reference. 2012.)

```
QFeedbackHapticsEffect shakeEffect;

shakeEffect.start();
```

KUVA 23. QfeedbackHapticsEffect-olion tekeminen ja värinän suorittaminen

5.8 Testaus

Ohjelmistoa testattiin jatkuvasti aina, kun tehtävälistalla olevia ominaisuuksia saatiin integroitua sovellukseen. Lopullinen käyttöliittymän testaus tapahtui näkymä kerrallaan. Jokainen näppäin ja navigointi sivujen välillä testattiin ja havaittiin toimiviksi. Tietokantafunktioiden toiminta testattiin tekemällä kirjauksia eri henkilöille sekä käyttöliittymän kautta että NFC-tagilla. Testaus tapahtui pääosin Nokian 700-puhelimella.

Palvelimen kapasiteetin testausta varten kehitettiin erikseen testausohjelmisto, jolla voidaan simuloida halutun kappalemääräisen mobiilisovellusjoukon yhteiskäyttöä. Samalla kun palvelinta rasitetaan erilaisilla kirjaus- ja hakufunktioilla, tutkitaan palvelimen resursseja, kuten muistinkäyttöä ja suorittimen käyttöä. Näin selviää palvelinpuolen resurssien riittävyys, kun tulee tarve mitoittaa palvelimen suorituskyky tietylle asiakkaalle.

6 LOPULLISEN OHJELMAN ESITTELY

Qt-Daisy-Perhepäivähoitosovellus toimii Symbian-käyttöjärjestelmällä. Sovelluksesta otetut käyttöliittymäkuvat on kaapattu Qt Creatorin simulaattorista, jossa puhelinmallina on Nokian N8. Sovelluksen resoluutio on Symbian malleille hyvin yleinen 360 x 640 ja sovellus on lukittu pystyasentoon, jolloin se ei kääntyile puhelinta käännellessä.

Aloitusnäkymässä (kuva 24) näkyy yläreunassa hoitajan nimi. Taustalla oleva vihreä väri kertoo hoitajan olevan tällä hetkellä töissä eli työaikaa kertyy. Tällä hoitajalla on kaksi lasta, joiden taustalla olevat värit kertovat ylemmän lapsen olevan hoidossa parhaillaan (vihreä) ja alemman olevan poissa (punainen). Kulut yht. -painikkeen takaa löytyy hoitajan ryhmässä olevien lasten syömät ateriat yhteensä lapsikohtaisesti. Uusi tunniste -painiketta painamalla pääsee lisäämään lapsen NFC-tagille, jonka jälkeen tagilla voi kirjata lapsen sisään ja ulos käyttämällä sitä puhelimen takana olevassa NFC-lukijassa. Vaihto-painikkeen takaa voi vaihtaa eri hoitajan käyttämään sovellusta. Oma-painiketta painamalla pääsee katsomaan omia työaikatietoja sekä lisäämään työaikakirjauksia. Lapsen nimeä painamalla aukeaa lapsen tietoja kokoava näkymä.

KUVA 24. Aloitusnäkymä

Testaaja Tiinalla on kuvan 25 tilanteessa ryhmässään kaksi lasta ja näkymä näyttää heidän syömiensä aterioiden yhteissumman näytöllä lapsikohtaisesti ja alla yhteensä koko ryhmän syömien aterioiden korvausten yhteissumman. Lapsen nimeä painamalla saa esiin lapsikohtaisesti tarkemman erittelyn.

KUVA 25. Kulut yhteensä -näkymä

Näkymä (kuva 26) näyttää lapsen syömät ateriat listattuna ja järjestettynä aikajärjestykseen uusin ensin. Aterian nimeä painamalla kuluja voi muokata sekä poistaa. Ohjelma varmistaa kuitenkin käyttäjältä ennen poistoa että poisto halutaan varmasti tehdä. Ateriat tulevat kannasta reaaliajassa, ja lista päivittyy heti, kun muutoksia lapsen aterioihin tehdään. Punaiset reunukset lapsen nimen ympärillä kertovat, että lapsella on jotain allergioita.

KUVA 26. Lapsen syömät ateriat -näkymä

Hoitajanvaihtonäkymässä (kuva 27) hoitaja vaihtaa itsensä puhelimen käyttäjäksi. Ensin valitaan alue, josta hoitaja on, ja sen jälkeen tietokanta näyttää sen alueen hoitajat. Hoitajan valittua itsensä käyttäjäksi puhelin hakee tietokannasta hoitajan hoitolapset ja hoitajan omat tiedot.

KUVA 27. Hoitajanvaihtonäkymä

Näkymä (kuva 28) kertoo hoitajan tämän päivän suunnitelman, toteutuman ja kertymän. Koska tälle päivälle ei ole suunnitelmaa ja toteutuma on positiivinen, saadaan kertymäksi neljä minuuttia. Tämä sivu näyttää myös hoitajan tekemät muut työaikakirjaukset listattuna päivämäärän mukaan järjestykseen. Työaikakirjauksia hoitaja voi merkata lisää siirtymällä muu työaikakirjausnäkymään.Työaikakoonti-painikkeen takaa löytyvät hoitajan työaikatiedot nykyisen jakson ajalta, joka voi olla esimerkiksi 3 viikkoa.

KUVA 28. Hoitajan omat tiedot -näkymä

Hoitaja voi merkata itselleen tarvittaessa muita työaikakirjauksia. Ensin valitaan haluttu kirjaustyyppi. Seuraavaksi päivämääräkenttää painamalla aukeaa kalenteri, josta haluttu päivämäärä hipaisemalla näyttöä siirtyy päivämääräkenttään. Ok-painikkeella kirjaus menee tietokantaan ja tulee automaattisesti näkyviin hoitajan omiin tietoihin. (Kuva 29.)

KUVA 29. Hoitajan muu työaikakirjaus -näkymä

Koontisivu (kuva 30) näyttää hoitajan työaikatiedot nykyisen jakson ajalta. Testaaja Tiinalla ei ole tällä hetkellä varsinaista suunnitelmaa, joten suunnitelmasarake on tyhjänä. Sen takia jakson kertymä on yhtä suuri kuin tehdyt tunnit. Päivämäärät ja kellonajat ovat melko pienellä fontilla, joten ohjelmaan on rakennettu mahdollisuus suurentaa sekä pienentää fonttia suurennuslasia painamalla. Suurentamisen jälkeen tunteja voi vielä liikutella pystysuunnan lisäksi sivusuunnassa kosketusnäyttöä pyyhkäisemällä. Lähetä-painiketta painamalla tunnit hyväksytään ja lähetetään palkanmaksua varten.

KUVA 30. Hoitajan työaikojen koontinäkymä

Aloitusnäkymästä lapsen nimeä painettaessa ohjelma siirtyy lapsen tietoihin (kuva 31). Yläreunassa on lapsen nimi, ja jälleen taustaväri kertoo lapsen olevan paikalla(vihreä). Lapsen voi tästä kirjata ulos joko lukemalla lapsen NFC-tagin tai painamalla Ulos-painiketta. Lapsen tiedoissa näkyvät myös lapsen syömien aterioiden yhteishinta kuukauden ajalta sekä lapselle merkatut poissaolot ja lomat. Lisää kulu -painikkeesta pääsee lisäämään lapselle ruokakuluja, Poissaolo-painikkeesta voi lisätä lomia ja poissaoloja, Allergia-painikkeesta aukeaa lapsen allergiatiedot ja Yhteystiedot-painikkeesta aukeaa lapsen vanhempien tai muiden päivähoidosta hakijoiden yhteystiedot. Allergia- ja Yhteystieto-painikkeen ympärille tulee punainen reunus jos on allergioita tai yhteyshenkilöillä esimerkiksi lähestymiskielto. Allergia- ja yhteystietoja ei pääse tietosuojalain takia lukemaan, ennen kuin puhelimella luetaan hoitajan NFC-tagi.

KUVA 31. Lapsen tiedot -näkymä

Kuvassa 32 lapsen nimi näkyy jälleen yläreunassa, ja vihreä taustaväri kertoo lapsen olevan parhaillaan hoidossa. Käyttäjä voi valita, minkä ateriakokonaisuuden haluaa lapselle lisätä, ja mille kalenteripäivälle kulu lisätään. Tallenna-painike tekee kirjaukset tietokantaan, jolloin kulut päivittyvät lapsen tietoihin.

KUVA 32. Näkymä lapsen kulun lisäämiseksi

Lapsen allergiatiedoissa (kuva 33) näkyy lapsen allergiat kuvina sekä tekstinä. Puhelimen näytölle mahtuu kuvassa olevalla koolla kuusi allergiatietoa, mutta ohjelmisto skaalaa kuvat pienemmiksi, jos allergioita on merkattu tietokantaan lapsen kohdalle enemmän. Allergioiden takia lapsen ruokakulut muuttuvat, ja tältä sivulta nähdään myös korvausluokka uuden ateriahinnan laskemiseksi.

KUVA 33. Allergianäkymä

Yhteystiedot näkymässä (kuva 34) ovat listattuna henkilöt, joilla on oikeus hakea lapsi hoitopaikasta ja joille ilmoitetaan lapseen liittyvissä asioissa tarvittaessa.

KUVA 34. Yhteystiedot-näkymä

7 POHDINTA

Opinnäytetyön päämäränä oli rakentaa perhepäivähoitajille seurantaohjelmisto. Ohjelmaan kirjataan hoitoon tulevat lapset NFC-tagin avulla sisään ja ulos. Sovelluksen avulla manuaaliset virheet jäävät pois, ja työaikojen seuranta saadaan tarkaksi ja reaaliaikaiseksi. Lisäksi ohjelmisto tarjoaa kaiken lapsen hoitoon tarvittavan informaation perhepäivähoitajalle.

Ohjelmiston laajuus yllätti kokonaisuudessaan. Sovellusta rakentaessa tuli vastaan monia ongelmia. Esimerkiksi NFC:n luvun kanssa oli pitkään ongelmia, mutta niistä selvittiin tietoa hakemalla ja kokeilemalla erilaisia koodiratkaisuja. Tietokantojen liittäminen sovellukseen oli myös monimutkainen prosessi, mutta lopulta kaikki toimii niin kuin pitääkin.

Käyttöliittymä rakennettiin uudehkolla QML-ohjelmoinnilla. JavaScript-pohjaisena ja kuvaavana kielenä se oli helppo omaksua, ja omien funktioiden tekeminen onnistui hyvin. Myös sisäänrakennettuja JavaScriptin olioita voi käyttää suoraan. Esimerkiksi JavaScriptin Datea käytettiin kovasti myös tässä projektissa, kun piti saada määritettyä päivämääriä ja nykyistä ajanhetkeä. Graafinen käyttöliittymänsuunnittelutyökalu helpotti ja nopeutti käyttöliittymien rakentamista. QML-käyttöliittymä rakentuu erilaisista komponenteista, joiden ominaisuuksia voidaan sitoa toisiinsa. Esimerkiksi suorakulmion leveys tai sijainti voidaan sitoa toisen suorakulmion ominaisuuksiin. Kun ensimmäisen kulmion ominaisuutta muutetaan, muutos näkyy välittömästi toisessa.

Projektin aikana käytetty versionhallinta oli ehdottoman tärkeä sovelluksen laajuuden takia. Jos joku asia ei onnistukaan, voi helposti palata aikaisempaan versioon. Lisäksi useamman ohjelmoijan tehdessä samaa ohjelmaa yhtä aikaa kaikki muutokset voidaan päivittää halutessa hetkessä kaikille käyttäjille. Joka tapauksessa muutokset päivitetään joka päivä kun edistystä tapahtuu. Versionhallinnasta on myös kätevä tarkistaa ongelmien ilmetessä, mitä kohtaa on viimeksi muokattu, jolloin mahdollinen virhe löytyy nopeasti.

Opinnäytetyönä tehtävä projekti antoi hyvän kokonaiskuvan oikean laajemman ohjelmiston tekemisestä. Jatkuvat palaverit ja selkeät taskilistat auttavat tavoitteiden saavuttamisessa ja helpottavat työn tekemistä. On myös todella tehokasta, että tiimin kesken voidaan nopeasti jakaa tietoa. Jos joku asia ei onnistu, niin vierestä voi löytyä nopeasti valmis ratkaisu kysymällä.

Perhepäiväsovellusta tehdessäni opin todella paljon työskentelystä Qt-ympäristössä. Tulevaisuuden näkymät ovatkin hyvät, varsinkin jos Digia laajentaa Qt:n alustariippumattomuuden lupaustensa mukaisesti Androidille, Applen iOS:lle, Windows 8:lle ja toivottavasti myös Windows Phone 8:lle. Näin tässä opitussa ympäristössä pystyisi tekemään ohjelmistoja melkein mille tahansa alustalle.

Opinnäytetyö oli opettava paketti tämän päivän ohjelmistoalasta. Perusohjelmointitaidot karttuivat paljon, Qt-ohjelmointi todella paljon. Ohjelmiston testaus tuli myös hyvin tutuksi, koska asiakasprojektina toteutetun sovelluksen täytyy toimia ehdottoman varmasti myös varsinaisessa laitteessa.

LÄHTEET

Application business - New way of software development. PowerPoint-esitys. SPDesign Oy. Yrityksen sisäinen materiaali.

Daisy. Esite. While on the Move.

Digia ostaa koko Qt-kehitysympäristön Nokialta. 2012. Saatavissa: http://www.digia.com/fi/Digia/Yritys/Uutiset/Digia-ostaa-koko-Qt-kehitysympariston-Nokialta/. Hakupäivä 24.8.2012.

Near Field Communication. 2012. Saatavissa: http://fi.wikipedia.org/wiki/Near_Field_Communication. Hakupäivä 7.8.2012.

NET Framework. 2012. Saatavissa: http://fi.wikipedia.org/wiki/.NET_Framework. Hakupäivä 23.8.2012.

Olio-ohjelmointi. 2012. Saatavissa: http://fi.wikipedia.org/wiki/Olio-ohjelmointi. Hakupäivä 7.8.2012.

QFeedbackHapticsEffect Class Reference. 2012. Saatavissa: http://doc.qt.nokia.com/qtmobility/qfeedbackhapticseffect.html. Hakupäivä 16.8.2012.

QML. 2012. Saatavissa: http://en.wikipedia.org/wiki/QML. Hakupäivä 15.8.2012.

QML ListView Element. 2012. Saatavissa: http://doc.qt.nokia.com/4.7-snapshot/qml-listview.html. Hakupäivä 16.8.2012.

QNearFieldManager Class Reference. 2012. Saatavissa: http://doc.qt.nokia.com/qtmobility/qnearfieldmanager.html. Hakupäivä 15.8.2012.

QNetworkAccessManager Class Reference. 2012: Saatavissa: http://doc.qt.nokia.com/4.7-snapshot/qnetworkaccessmanager.html. Hakupäivä 16.8.2012.

QSound Class Reference. 2012. Saatavissa: http://doc.qt.nokia.com/4.7-snapshot/qsound.html. Hakupäivä 7.8.2012.

Qt kehitysympäristö. 2012. Saatavissa: http://fi.wikipedia.org/wiki/Qt_(kehitysympäristö). Hakupäivä 23.8.2012.