SAVONIA

OPINNÄYTETYÖ - **AMMATTIKORKEAKOULUTUTKINTO**
Tekniikan ja liikenteen ala

JÄÄKIEKKO

Tilastointisovellus Android-tableteille

TEKIJÄ Matti Ilvonen

Koulutusala Tekniikan ja liikenteen ala	
Koulutusohjelma Tietotekniikan koulutusohjelma	
Työn tekijä(t) Matti Ilvonen	
Työn nimi Jääkiekko tilastointisovellus Android-tableteille	
Päiväys 1.3.2013	Sivumäärä/Liitteet 29 + 8
Ohjaaja(t) lehtori Sami Lahti, lehtori Keijo Kuosmanen	
Toimeksiantaja/Yhteistyökumppani(t) Tmi Jussi Koistinen	

Tiivistelmä

Opinnäytetyön aiheena on luoda jääkiekko tilastointisovellus Android-tablet-tietokoneille. Sovelluksen tilaaja oli Tmi Jussi Koistinen.

Sovelluksen avulla haluttiin tallentaa jääkiekko-ottelun aikana syntyviä tilastomerkintöjä reaaliajassa. Tällaiselle sovellukelle on kysyntää, sillä ottelupöytäkirjojen tarjoamat tilastotiedot eivät ole riittävän kattavia eikä vastaavanlaista sovellusta ole saatavilla. Sovellus haluttiin kehittää Android-käyttöjärjestelmää käyttäviä tablet-tietokoneita varten niiden yleisyyden sekä saatavilla olevien ilmaisten kehitystyökalujen vuoksi.

Ohjelmistokehityksessä työvälineinä käytettiin Eclipse-kehitysympäristöä, johon oli asennettu ADT-lisäosa, mikä mahdollisti Android-sovelluksen kehittämisen. Ohjelmointikielenä käytettiin Javaa. Sovelluksen tietokantana toimi SQLite-tietokanta. Sovelluksen tietokanta synkronoidaan ulkoisen tietokannan kanssa JSON-tiedonsiirtomuotoa käyttäen.

Opinnäytetyön päätteeksi valmistui toimiva POC-versio. Sovellukseen saatiin kaikki määritellyt toiminnot, mutta kunnollista lopputestausta ei ehditty suorittaa. Sovelluksen kehittämistä on tarkoitus jatkaa vielä kesän yli, jolloin suoritetaan lopputestaus, ohjelmointivirheiden korjaaminen sekä hyväksymistestaus.

Avainsanat
jääkiekko, tilastointi, Android-tablet-taulutietokonesovellus

Field of Study Technology, Communication and Transport			
Degree Programme Degree Programme in Computer Science			
Author(s) Matti Ilvonen			
Title of Thesis Ice Hockey Statistics Application for Android Tablets			
Date	1 March 2013	Pages/Appendices	29 + 8
Supervisor(s) Mr Sami Lahti, Lecturer, Mr Keijo Kuosmanen, Lecturer			
Client Organisation /Partners t/a Jussi Koistinen			

Abstract

The purpose of this thesis was to create a hockey statistics application for Android tablet computers. The project was comissioned by t/a Jussi Koistinen.

The application allows saving statistics of a hockey game in real-time. There was a need for this kind of application because the statistics provided in score sheets are not comprehensive enough and there was no similar application available. The application was developed for tablets using Android-OS for their frequency and availability of development tools.

Eclipse IDE with additional ADT plugin was used for developing the application. Java was used as the programming language. The application database is a SQLite database. The application database is synchronized with an external database using the JSON data format.

As a result of this thesis a working POC version was created. All defined functions were created but no final testing was done due to lack of time. The project is intended to be continued for the summer. Final testing, error correction and acceptance testing will be done during the summer.

Keywords
ice hockey statistics android tablet software

SISÄLTÖ

TERMIT JA LYHENTEET

SDK	Software development kit sisältää ohjelmiston kehitystyökalut. Eri alustoille on olemassa omat kehitystyökalut.
ADT	Android development tools on lisäosa Eclipse sovelluskehittimeen, jonka avulla voidaan luoda Android-sovelluksia.
IDE	Integrated development environment eli ohjelmointiympäristö eli työkalut ohjelmiston suunnitteluun ja toteutukseen.
ER	Entity-relationship on kuvaus relaatiotietokannasta. Puhekielinen ilmaus: ER-malli.
OS	Operating system eli käyttöjärjestelmä.
POC	Proof of Concept eli konseptiversio.

1 JOHDANTO

Opinnäytetyön aiheena on luoda sovellus, jolla voidaan helposti reaaliajassa tallentaa jääkiekko-ottelun aikana syntyviä tilastomerkintöjä. Opinnäytetyön tilaaja on tmi Jussi Koistinen, ja sovellus on tarkoitettu seuroille, joiden alaisuudessa toimii nuorten jääkiekkojoukkueita. Sovellus sopii hyvin joukkueiden valmentajille ja huoltajille. Sovelluksen avulla valmentajat voivat tallentaa reaaliajassa syntyviä tilastomerkintöjä paljon tarkemmin kuin ottelupöytäkirjaan merkitään. Sovelluksella pystytään muun muassa merkitsemään aloitusvoitot tai vaihtojen määrä ja pituus pelaajittain. Syntyneitä tilastoja voidaan tarkastella vaikkapa eri näkökulmista, eri aikaväleiltä, otteluittain tai pelaajittain.

Tällaiselle sovellukselle on kysyntää, sillä tähän asti merkintöjä ei ole tehty ollenkaan tai niitä on tehty käsipelillä, mikä on todella hidasta ja hankalaa. Lisäksi tilastojen käsitteleminen on lähes mahdotonta, koska ei ole olemassa mitään tietokantaa tai sovellusta johon tiedot voisi jälkikäteen syöttää käsittelyä varten. Myös tiettyjen toimintojen toteuttaminen esim. pelaajien jääajan mittaaminen käsin on hankalaa tai lähes mahdotonta.

Vastaavanlaista sovellusta on yritetty luoda jo useana vuonna Savonia AMK:ssa opiskelijoiden kurssiharjoitustöinä. Tähän mennessä sovellukset ovat olleet joiltakin osin puutteellisia. Syynä puutteisiin todennäköisesti on ollut käytettävissä ollut aika. Suurin puute, joka huomattiin aikaisemmista versioista, oli tietokannan kyky varastoida tilastoja riittävän tarkasti. Tietokannan suunnitteluun varattiin riittävästi aikaa heti projektin alussa, jotta ongelmia ei syntyisi tietokannan kanssa.

Kiinnostus tähän aiheeseen heräsi mobiiliohjelmoinnin projektikurssilla, kun ryhmälle annettiin tehtäväksi etsiä ja korjata virheitä aikaisemman vuosikurssin aikaansaannoksesta. Ohjelman jatkokehityksen aikana löytyi paljon virheitä, jotka kaatoivat sovelluksen tai oleellisesti haittasivat sen toimintaa. Kurssin edetessä huomattiin, ettei ohjelmasta saisi toimivaa pelkästään virheitä korjailemalla. Sovelluksesta saisi toimivan, mutta työ olisi aloitettava kokonaan alusta, suunnittelusta lähtien.

Opinnäytetyössä käydään läpi, mitä tekniikoita ja työkaluja ohjelman suunnittelussa ja toteutuksessa käytettiin sekä kuinka ohjelma ja tietokanta suunniteltiin, määriteltiin ja toteutettiin. Lisäksi esittellään ohjelman päätoiminnot. Lopuksi kerron, kuinka työ testattiin ja kuinka työ mielestäni onnistui.

2 KÄYTETYT TEKNIIKAT JA TYÖKALUT

Sovellusalustaksi valittiin Android-käyttöjärjestelmää käyttävät tablet-tietokoneet. Syinä tähän olivat kyseisten laitteiden yleisyys, käytettävyys sekä helposti saatavilla olevat kehitysympäristö ja dokumentaatio. Android-sovellus puolestaan pitkälti määräsi kehitysympäristön ja siinä käytettävät työkalut ja ohjelmointikielen. Android-sovelluksen kehittämiseen täytyi ladata Android yhteisön kotisivuilta Android SDK (http://developer.android.com/sdk/index.html#win-bundle), joka pitää sisällään sovelluskehitin Eclipsen, johon on valmiiksi asennettu ADT-lisäosa. Ohjelmointikielenä käytetään Javaa. On olemassa myös muita työkaluja, kuten NetBeans IDE, johon voidaan asentaa lisäosa Android-sovelluksen kehittämistä varten. Googlen tuotepäällikkö Ellie Powers ilmoitti 16. toukokuuta uudesta ohjelmointiympäristöstä. Android Studio on puhtaasti Android-sovelluskehitystä varten kehitteillä oleva ohjelmointiympäristö. Tällä hetkellä ladattavissa on vasta versio 0.1, eli valmiista tuotteesta ei vielä voida kuitenkaan puhua. Koska sovelluksella pitää pystyä tallentamaan paljon tietoa, luonnollinen valinta oli käyttää Androidin tukemaa SQLite-tietokantaa. Tietokanta suunniteltiin SQLite-kantaa tukevalla Navicat Data Modeler -ohjelmalla, jolla kannasta ensin tehtiin ER-malli, josta sitten johdettiin tietokannan luontilauseet.

2.1 Android SDK

Android SDK on aloittavalle ohjelmoijalle sopiva paketti Android-sovelluksen kehittämiseen, sillä se sisältää muun muassa ADT lisäosan Eclipse ohjelmointiympäristöön sekä pakettien hallinnan, joista tärkeimpinä eri Android alustat eli API kerrokset, SDK työkalut sekä SDK alustatyökalut. Myös emulaattori ohjelman ajamiseen ja debuggaukseen tietokoneella tulee Android SDK:n mukana. Android SDK on saatavilla Windows, Linux ja Mac OS -käyttöjärjestelmille. (Android SDK.)

2.2 Eclipse IDE + ADT-lisäosa

Eclipse on suositelluin, käytetyin sekä ennen kaikkea ilmainen ohjelmointiympäristö. Eclipse on alun perin IBM:n kehittelemä ohjelmointiympäristö, joka julkaistiin avoimen lähdekoodin lisenssille vuonna 2003. Vuodesta 2004 lähtien ohjelmointiympäristön kehityksestä on vastannut Eclipse Foundation-säätiö. (Eclipse IDE.)

Eclipse käyttää hyväkseen lisäosia tarjotakseen toiminnallisuudet. Lisäosien ansioista Eclipsellä voidaan koodata erittäin monella ohjelmointikielellä. Erikoisimpina kielinä mainittakoon muun muassa Ada, COBOL, Fortran ja Haskell – ohjelmointikielet ja tunnetuimmista kielistä ainakin Java, C/C++ sekä PHP. Lisäosilla saadaan myös aikaiseksi muutakin toiminnallisuutta, kuten UML-kaavioiden käsittelyä tai projektien versionhallintaa. (Eclipse IDE.)

ADT-lisäosa on tarkoitettu Android-projektien kehittämistä varten Eclipse-ohjelmointiympäristöllä. ADT laajentaa Eclipsen ominaisuuksia ja antaa mahdollisuuden luoda nopeasti uusia Android-projekteja ja käyttöliittymiä. Lisäosa tarjoaa ohjatun projektin luonnin lisäksi työkalujen integroinnin, XML-

editoreja sekä debug-ruudun. Halutessaan ADT-lisäosan voi asentaa Eclipseen manuaalisesti. ADT-lisäosan asennusohjeet Eclipselle löytyvät Android-kehittäjäyhteisön kotisivuilta (http://developer.android.com/sdk/installing/installing-adt.html). On kuitenkin helpompi ladata aiemmin mainittu Android SDK, joka sisältää Eclipse-ohjelmointiympäristön sekä valmiiksi asennetun ADT-lisäosan. (ADT-lisäosa.)

2.3 Java

Java on yleiskäyttöinen, rinnakkainen, luokka-pohjainen, olio-ohjelmointikieli. Sen tarkoituksena on antaa ohjelmistokehittäjien ”kirjoitaa kerran, ajaa kaikkialla”, mikä tarkoittaa, että koodia, joka toimii yhdellä alustalla, ei tarvitse kääntää uudelleen toimiakseen toisella alustalla. Java-sovellukset on tyypillisesti koottu class-tiedostoihin, jotka toimivat kaikissa Java-virtuaalikoneissa (JVM) riippumatta tietokoneen arkkitehtuurista. Java on yksi suosituimmista käytössä olevista ohjelmointikielistä. Java on suosittu erityisesti asiakas-palvelin-internet-sovelluksissa, joissa käyttäjiä on raportoitu olevan jopa 10 miljoonaa. (Silander, Ollikainen, Peltomäki & Kosonen 2010.)

Javan kehitti alun perin James Gosling työskennellessään Sun Microsystemsillä, ja se julkaistiin vuonna 1995 keskeisenä osana Java-alustaa. Javan syntaksi muistuttaa paljon C ja C++ -kielien syntakseja, mutta Javalla ei pääse käsiksi yhtä syvälle tietokoneen muistiin kuin esim. C-kielellä. (Silander, Ollikainen, Peltomäki & Kosonen 2010.)

Javan kehittäjillä oli viisi pääperiaatetta luodessaan Java-kieltä. Ensimmäisen periaatteen mukaan Javan tuli olla yksinkertainen, olio-pohjainen ja tuttu. Toisen periaatteen mukaan Javan piti olla vahva ja turvallinen. Kolmannen periaatteen mukaan kielen piti olla arkkitehtuuri-neutraali ja helposti siirreltävä. Neljäntenä periaatteena oli, että Javan piti toimia tehokkaasti. Viimeisen periaatteen mukaan Javan piti olla tulkattava, säikeistetty ja dynaaminen. (Silander, Ollikainen, Peltomäki & Kosonen 2010.)

Aivan viime vuosina Java on saanut paljon negatiivista huomiota. Javasta on nimittäin löytynyt paljon vakavia tietoturva-aukkoja, ja niiden paikkaaminen Javan nykyiseltä kehittäjältä Oraclelta on kestänyt usein hyvinkin kauan. (Silander, Ollikainen, Peltomäki & Kosonen 2010.)

2.4 Versionhallinta

Projektin aikana jouduttiin työskentelemään usealla eri tietokoneella sovellusta kehittäessä. Tätä varten etsin tietoa saatavilla olevista versionhallintamahdollisuuksista. Löysinkin ohjeet Eclipse Subversive -lisäosan asentamiseen ja käyttämiseen. Subversive-lisäosa on tarkoitettu yhdistämään Subversion (SVN) -versiohallinta Eclipse-ohjelmointiympäristöön. Lisäosan avulla Eclipsestä voidaan ottaa yhteys suoraan projekteihin, jotka sijaitsevat Subversion-säilytyspaikoissa. Tällaisia paikkoja tarjoavat ilmaiseksi muun muassa Google ja GitHub. Päätin perustaa projektille säilytyspaikan Googlen tarjoamaan palveluun code.google.com.

Versionhallinnan avulla voin työskennellä useilla koneilla ja käytössäni on aina viimeisimmät versionhallintaan lataamani projektit. Projekti on myös varmassa turvassa mahdollisia laitevikoja silmällä pitäen. Suuremman hyödyn lisäosasta saa silloin, kun projektia on kehittämässä useampi henkilö, jolloin versionhallinnasta tulee lähes välttämätön. (Subversive SVN.)

2.5 Navicat Data Modeler

SQLite-tietokannan suunnittelua varten täytyi etsiä sovellus, jolla voidaan luoda SQLite ER -malleja. Tähän tehtävään olisi ehkä riittänyt myös ilmainen MySQL Workbench, mutta sillä ei olisi saanut SQLite-syntaksin mukaisia taulujen luontilauseita. Lopulta löytyi maksullinen Navicat Data Modeler -ohjelma, josta oli saatavilla 30 päivän ilmainen kokeiluversio. Kokeiluversion avulla pystyin suunnittelemaan tietokannan rakenteen ja luomaan SQLite-syntaksin mukaiset luontilauseet.

Navicat on graafinen tietokantojen hallinta- ja kehitysympäristö. Navicat-tuoteperheen on kehittänyt PremiumSoft CyberTech. Tuoteperheeseen kuuluu hallinta- ja kehitysympäristöt MySQL, PostgreSQL, Oracle, SQL Server sekä SQLite -tietokantoja varten. Navicat Data Modelerilla voidaan suunnitella ja kehittää tietokantoja kaikkiin edellä mainittuihin tietokantaympäristöihin. Luoduista ER-malleista voidaan myös luoda tietokannan luontilauseet. (Navicat Data Modeler.)

2.6 SQLite

SQLite on relaatiotietokanta, joka sisältyy erittäin pieneen (~350 KB) C-kielellä kirjoitettuun kirjastoon. SQLitessä ei ole erillisiä prosesseja, joiden kanssa sovellus kommunikoisi, vaan SQLite-kirjasto on linkitettynä sovellukseen, jolloin tietokannasta tulee osa sovellusta. SQLite tallentaa koko tietokannan ohjelmaa pyörittävän koneen muistiin yhteen laitteistoriippumattomaan tiedostoon. Tietokantaan kirjoitettaessa SQLite lukitsee koko tietokantatiedoston. (SQLite.)

SQLite sisältää suurimman osan SQL-92-standardista SQL-kielelle, mutta siitä puuttuu joitakin toimintoja, kuten vajaa tuki herättimille (triggers) ja kykenemättömyys kirjoittaa näkymiä (views). Vaikka SQLite tukee monimutkaisiakin kyselyjä (queries), on sillä heikko taulun muokkaus (ALTER TABLE) -tuki. Se ei voi esimerkiksi muokata tai poistaa sarakkeita. (SQLite.)

SQLite-tietokantoja käytetään muun muassa Mozilla Firefox, Mozilla Thunderbird, Google Chrome, Opera sekä Skype -ohjelmistoissa. Pienen kokonsa vuoksi SQLite käy hyvin sulautettuihin järjestelmiin ja se sisältyy muun muassa Microsoft Windows Phone 8, Apple, Symbian, Maemo, Android ja BlackBerry -käyttöjärjestelmiin. (SQLite.)

3 PROJEKTIN MÄÄRITTELY, SUUNNITTELU JA TOTEUTUS

Opinnäytetyö aloitettiin maaliskuussa 2013. Työ pystyttiin tekemään omalla ajalla, mikä oli hyvä, koska opinnäytetyötä voitiin tehdä muun työn ohella. Työn aikana pidettiin parin viikon välein palavereita, joihin osallistuivat lisäkseni ohjaavat opettajat sekä työn tilaaja.

3.1 Määrittely

Koska vastaavanlaista sovellusta on yritetty kehittää aiemminkin Savonia AMK:ssa, olivat sovelluksen vaatimat toiminnot hyvin selvillä. Aloituspalaverissa keskusteltiin lisäksi sovelluksen muista vaatimuksista. Vaatimuksiin kuului, että sovelluksen pitää pystyä synkronoimaan paikallinen SQLite-tietokanta ulkoisen MySQL-tietokannan kanssa. Paikallinen tietokanta toimisi vierastietokantana ja ulkoinen puolestaan isäntätietokantana. Osa toiminnoista siirrettäisiin web-käyttöliittymällä suoritettaviksi, kuten pelaajien ja joukkueiden hallinta. Itse Android-sovelluksella voitaisiin pääasiassa tallentaa ja selata tilastoja. Tilastojen tallentaminen vaatii, että MySQL-tietokantaan on lisätty joukkue ja joukkueeseen on lisätty pelaajia ja että tietokantojen välillä on suoritettu synkronointi.

3.2 Suunnittelu

Tärkeimpiä suunnittelun osa-alueita olivat tietokanta, tehokas koodi, projektin jatkokehityksen mahdollistaminen sekä käyttöliittymä.

Tietokanta on ohjelman tärkein osa, sillä ilman toimivaa tietokantaa ei voi olla toimivaa ohjelmaa. Tietokannan suunnitteluun kului noin 2 - 3 viikkoa. Tavoitteena oli saada tietokannasta niin kattava, että se riittää helposti täyttämään sovelluksen vaatimukset.

Toinen tavoite oli tehokkaan ja selkeän koodin kirjoittaminen. Aiempien ryhmien tekemistä sovelluksista oli erittäin vaikea saada selvää, koska koodia ei ollut kommentoitu juuri millään tavalla. Tämän vuoksi pyrittiin kommentoimaan suurimman osan tekemästäni koodista, jotta tarvittaessa projektin kehittäminen olisi helpompaa. Koodin selkeyttämiseen kuului myös sovelluksen luokkakirjastojen suunnittelu. Esimerkiksi tietokantaan liittyvät toiminnot ovat kokonaan omassa kirjastossaan. Näin projektista saatiin selkeä ja siihen tutustuminen helpommaksi.

Käyttöliittymän vaatimuksina olivat selkeys ja helppo käytettävyys. Lisäksi toiveena oli sovelluksen toimivuus 7 tuuman tabletilla 10-tuumaisen lisäksi. Asiaan perehdyttyäni löysin tavan, jolla näkymistä tehdään kopioita, joita voidaan muokata erikokoisille näytöille. Päätelaite huolehtii tämän jälkeen oikeankokoisen näkymän valitsemisesta. Pyrin pitämään kaiken mahdollisimman yksinkertaisena ja helppokäyttöisenä, joten käyttöliittymän suunnittelu oli siltä osin helppoa. Uskon saaneeni aikaiseksi sovelluksen, jota on nopea, helppo ja miellyttävä käyttää.

3.3 Toteutus

Työ toteutettiin omalla ajalla keväällä 2013. Käytettävissä oli oma tietokone ja koululta lainaan saatu tablet tietokone.

Työ aloitettiin suunnittelemalla tietokanta kokonaan uusiksi. Tietokannan suunnittelemiseen käytettiin suhteellisen paljon aikaa ja sitä katselmoitiin useaan otteeseen ennen varsinaisen sovelluskehityksen aloittamista. Kun tietokanta oli saatu suunniteltua valmiiksi, aloitettiin sovelluskehitys. Ensimmäiseksi suunniteltu tietokanta toteutettiin ja toteutus integroitiin sovellukseen. Tietokantaa testattiin syöttämällä sinne testidataa ja hakemalla syötettyä dataa.

Tietokannan testaamisen jälkeen sovellukselle luotiin käyttöliittymä. Jokaisen näkymän (Activityn) toiminnallisuuden ohjelmoiminen aloitettiin käyttöliittymän luomisen jälkeen. Näkymien toiminnot pyrittiin ohjelmoimaan mahdollisimman valmiiksi ennen seuraavan näkymän toimintojen ohjelmoimista. Käyttöliittymää ja sovellusta testattiin aina, kun uusia toimintoja saatiin toteutettua.

Käytettävissä oleva aika oli todella lyhyt. Siksi sovelluskehityksen aikana koodia kommentoitiin mahdollisimman paljon, että mahdollisesti keskeneräiseksi jäänyttä sovellusta voitaisiin jatkaa helposti myöhemmin.

Opinnäytetyön kirjallinen osuus kirjoitettiin pääasiassa sovelluksen POC-version valmistuttua toukokuussa 2013. Sovelluskehitystä jatketaan kesällä 2013.

4 TIETOKANTA

4.1 Tietokannan suunnitteleminen

Tietokannan suunnittelu on salaista tietoa, eikä sitä lisätä opinnäytetyön julkiseen versioon.

4.2 Tietokannan toteutus

Android-sovelluksessa on eri tapoja luoda sovellukselle tietokanta. Yksi versio on luoda SQLite-tietokantatiedosto, jollakin kolmannen osapuolen sovelluksella ja lisätä tiedosto Android-projektiin. Sitten sovellus kopioi tiedoston omaan käyttöönsä. Tätä tapaa oli käytetty aiemmissa toteutuksissa, joten käytin samaa tapaa aluksi itsekin. Pian kuitenkin huomasin, että tämä tapa on erittäin herkkä virheille ja sovellukseen tuli suuria käytön estäviä virheitä. Päätin etsiä toisenlaisen tavan luoda tietokannan. Löysinkin tavan, jossa tietokanta luodaan sovelluksessa SQL-lauseita suorittamalla. Yhden luokan (DBAdapter) avulla pystyin hallitsemaan tietokannan luonnin, tietojen lisäämisen, päivittämisen ja poistamisen yksillä yleispätevillä-funktioilla.

Esimerkki kaikille tauluille käyvästä poista-funktiosta löytyy alla olevasta kuvasta (KUVA 1). Tämä funktio löytyy DBAdapter-luokasta ja sitä voidaan kutsua taulut määrittelevistä luokista.

```
/*
 * delete record from any table. define deletable record with id.
 */
public boolean deleteRecord(String TABLE_NAME, String IDENTIFY_COLUMN, long ROW_ID){
    return db.delete(TABLE_NAME, IDENTIFY_COLUMN + "=" + ROW_ID, null) > 0;
}
```

KUVA 1. Esimerkki tiedon poistamisesta

Tietokannan tauluja varten luodaan luokkia. Yksi luokka määrittelee yhden taulun. Alla olevassa kuvassa (KUVA 2) on näyte ketju-taulun määrittelystä. Ns. taulu-luokka sisältää merkkijonomuuttujat taulun ja sarakkeitten nimiä sekä taulun luontilausetta varten. Taulu luodaan kutsumalla taulu-luokan create-functiota, mikä palauttaa taulun luontilauseen merkkijonona, mikä suoritetaan SQL-lausekkeena.

```
public class DBLines {

        //define table name
        private static final String TABLE_NAME = "lines";
    //define tables column names
        private static final String KEY_ID = "_id";
        private static final String KEY_NAME = "name";
    //define create statement for all your tables. Create new file (class) for each table.
        private static final String CREATE_STATEMENT = "create table if not exists " + TABLE_NAME + "
```

KUVA 2. Jokainen luokka määrittää yhden taulun

Alla oleva kuva (KUVA 3) näyttää kuinka lines-tauluun lisätään uusi rivi. Tietokantatoimintoja käytettäessä on ehdottoman tärkeätä muistaa avata ja sulkea tietokantayhteys.

```
//Creating a new instance of the DBAdapter at this point makes sure that we have a working database.
db = new DBAdapter(this);
DBLines li = new DBLines(); //Defines database table
db.open();//open database for the first time. if database doesn't exist it is created.
li.insert("1. Ketju", db); //DBLines calls DBAdapter insertRecord function
db.close();//close database and go to the main activity.
```

KUVA 3. Esimerkki tiedon lisäämisestä

Kuvassa (KUVA 3) kutsutaan alla olevan kuvan (KUVA 4) insert-funktiota. Jokaisella taulu-luokalla on omat funktionsa, sillä kaikki taulut eivät ole samanlaisia ja ne vaativat eri määrän tietoa. Jokaisen taulu-luokan insert-funktio kutsuu kuitenkin yhtä ja samaa DBAdapter-luokan insertRecord-funktiota. Parametreina insertRecord-funktio vaatii esimerkiksi kuvassa (KUVA 2) määritellyn taulun nimen (TABLE_NAME) sekä lähtöarvot, kuten ketjun nimi (name) kuvassa (kuva DBLines luokan insert-funktio). Lopuksi insertRecord-funktio palauttaa uuden lisätyn rivin id:n.

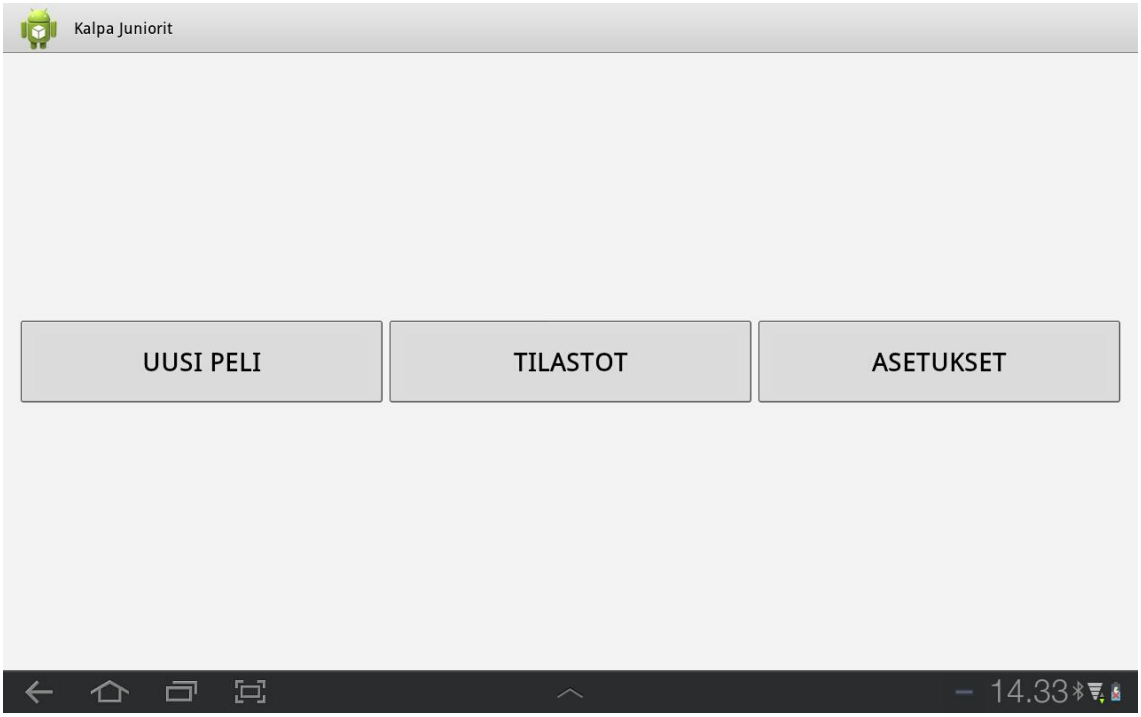
```
public long insert(String name, DBAdapter db){

    ContentValues initialValues = new ContentValues();
    initialValues.put(KEY_NAME, name);

    return db.insertRecord(TABLE_NAME, initialValues);

}
```

KUVA 4. DBLines luokan insert-funktio

5 JÄÄKIEKKOTILASTOJEN TILASTOINTISOVELLUS

5.1 Aloitusnäkymä

Sovelluksen käynnistyessä aukeaa sovelluksen aloitusnäkymä. Vaikka sovelluksessa on paljon monimutkaisia toimintoja, on sen käyttäminen pyritty tekemään yksinkertaiseksi niin, että sen käyttäminen onnistuu jopa ilman ohjeita. Aloitusnäkymässä (KUVA 5) on vain kolme yksiselitteistä nappia. Jo napin nimi kertoo käyttäjälle, mitä tapahtuu, jos nappia painaa. Samaa periaatetta on käytetty kaikissa näkymissä.



KUVA 5. Aloitusnäkymä

5.2 Sovellukseen sisäänkirjautuminen

Ensimmäisellä sovelluksen käynnistyskerralla käyttäjän täytyy syöttää omat tunnuksensa. Tunnukset täytyy olla luotuna erillisen Web-palvelun kautta ennen sovellukseen sisäänkirjautumista. Tunnukset syötetään sovelluksen Asetukset-valikossa (KUVA 6) ja tallennetaan laitteen muistiin. Käyttäjä voi kirjautua myöhemmin myös toisilla tunnuksilla, jolloin laitteeseen ladataan ko. tunnusten tiedot ulkoisesta tietokannasta.

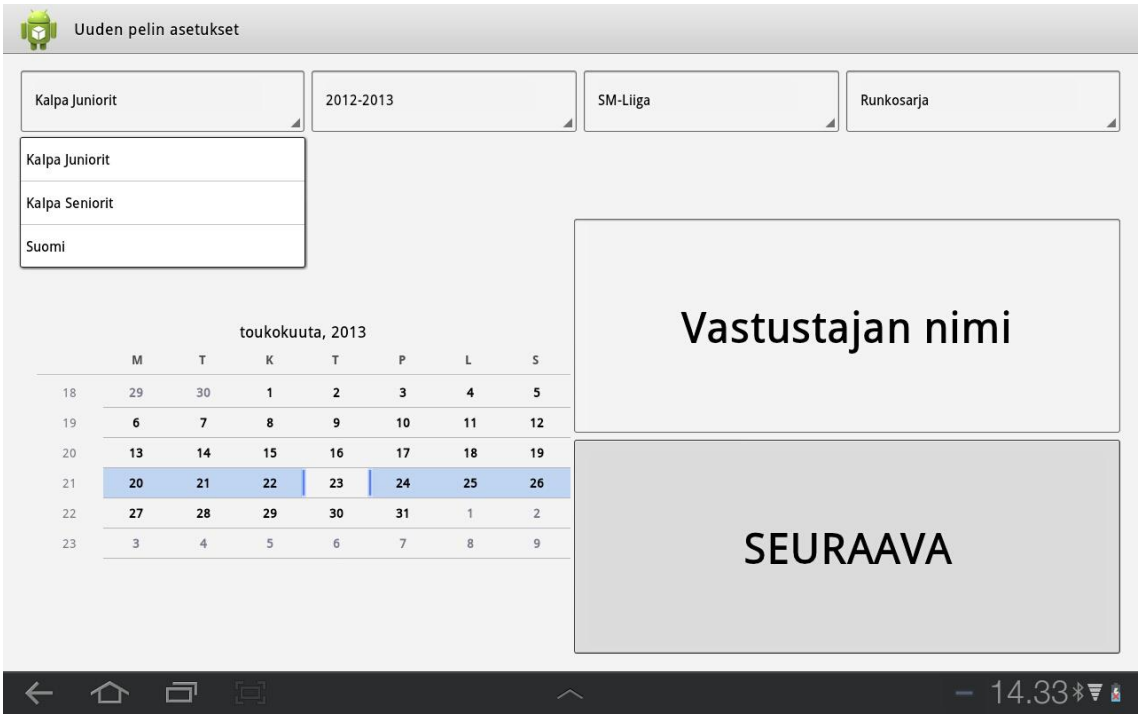
KUVA 6. Asetukset-valikko

5.3 Tietokannan alustaminen ja synkronointi

Tabletin tietokannan alustaminen ja synkronoiminen vaatii aina, että käyttäjä on rekisteröitynyt palveluun ja tallentanut kirjautumistietonsa sovelluksen muistiin. Lisäksi vaaditaan toimiva internetyhteys. Kun nämä vaatimukset on täytetty, voidaan tiedonsiirtäminen päätelaitteen ja Web-palvelun tietokannan välillä aloittaa. Ensin mahdolliset päätelaitteen tiedot siirretään Web-palveluun, minkä jälkeen päätelaitteen tietokanta alustetaan tyhjäksi ja tyhjään tietokantaan ladataan viimeisimmät tiedot käyttäjä-, joukkue-, kausi-, liiga- ja pelaajatiedot.

5.4 Uuden pelin asetukset

Kun käyttäjä painaa avausnäkymän ”Uusi peli” -painiketta, avautuu uuden pelin asetukset näkymä (KUVA 7). Tässä näkymässä käyttäjä valitsee yhden hallitsemistaan joukkueista, kauden jolloin ottelu pelataan, liigan, sarjatason ja ottelun päivämäärän sekä syöttää vastustajan nimen. Nyt syötettyjä tietoja ei tallenneta, vaan ne lähetetään seuraavalle activitylle. Kun kaikki tiedot on syötetty, käyttäjä siirtyy rosterin hallinta-activityyn klikkaamalla ”Seuraava”-painiketta.

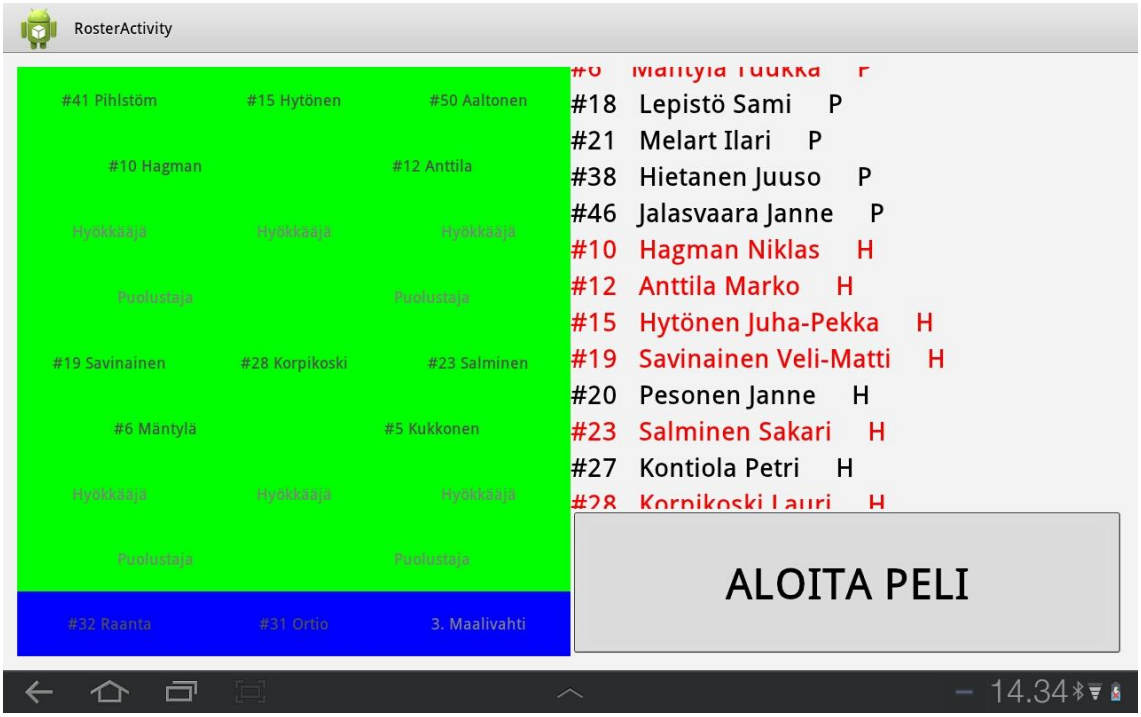


KUVA 7. Uuden pelin asetukset

5.5 Rosterin muokkaaminen

Jokaista peliä varten luodaan oma pelaajakokoonpano käytettävissä olevista pelaajista. Kokoonpano määritetään rosterin muokkaus-näkymässä (KUVA 8). Käytettävissä olevat pelaajat listataan. Listasta käyttäjä voi valita pelaajan ja vetää tämän halutulle pelipaikalle vedä ja pudota -toiminnolla (drag & drop). Kun pelaaja on viety listasta halutulle pelipaikalle, ei pelaajaa voida enää valita listalta. Tällöin pelaaja näytetään listalla punaisena. Pelipaikalla olevaa pelaajaa voidaan siirtää pelipaikalta toiselle tai pelaaja voidaan vetää takaisin listaan. Listaan vietäessä pelaajan tiedot häviävät pelipaikalta ja pelaaja voidaan jälleen valita listalta. Kun käyttäjä on saanut kokoonpanon valmiiksi ja

painaa ”Aloita peli” -painiketta, tallentuvat niin pelin kuin kokoonpanonkin tiedot. Tämän jälkeen käyttäjälle avautuu tilastojen tallennus-activity.



KUVA 8. Rosterin muokkaaminen

5.6 Ottelun tilastojen reaaliaikainen tallentaminen

Tilastojen tallennus-activityssä (KUVA 9) käyttäjä voi reaaliajassa tallentaa ottelun tapahtumia. Osa toiminnoista on selkeästi näkyvillä ja käyttäjän hallittavissa, osa puolestaan on automatisoitu. Automatisoituja toimintoja ovat mm. voittomaalintekijän tai maalivahdin nollapelin tarkistaminen ottelun päätyttyä ja vaihtojen pituuksien tilastoiminen. Otteluun valituille pelaajille lisätään automaattisesti yksi pelattu ottelu lisää pelin alkaessa. Pelin päättyessä pelaajille lisätään automaattisesti voitto tai tappio ja merkintä siitä, tuliko ratkaisu normaalilla peliajalla, jatkoajalla vai voittolaukauskisassa.

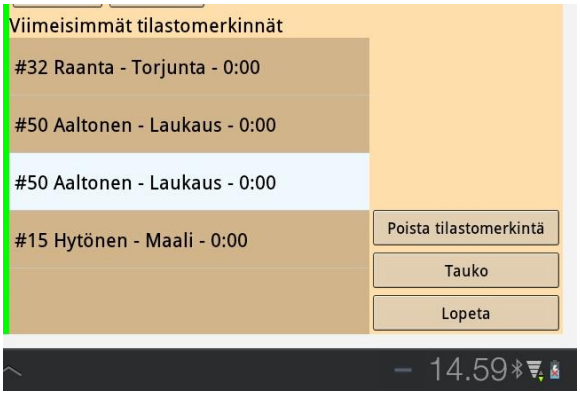
Pelin alkaessa käyttäjä valitsee pelaajat kentälle, joko yksitellen klikkaamalla pelaajan nimeä vihreällä pohjalla, tai koko kentän kerrallaan klikkaamalla ”x. Kenttä” -painiketta. Tällöin pelaajien tiedot ilmestyvät valkoiselle pohjalle. Joka kerta, kun kentällä oleva pelaaja vaihtuu, kentälle tulleelle pelaajalle alkaa juosta oma kello, joka mittaa pelaajan vaihdon pituutta. Peliaikaa mitataan vain silloin kun pelikello on käynnissä. Kun pelaaja vaihtuu uudelleen, edellisen pelaajan pelikello pysäytetään ja vaihdon pituus tallennetaan tietokantaan. Käyttäjän täytyy huolehtia, että pelikello käy oikeaan aikaan ja valittuna on oikea erä. Käyttäjä voi myös poistaa pelaajia kentältä painamalla pelaajan nimeä pitkään.

KUVA 9. Tilastojen tallentaminen

Kun pelaajalle halutaan lisätä tilastomerkintä, tulee käyttäjän valita oikea pelaaja kentällä olevista pelaajista ja klikata haluttua tilastomerkintä-painiketta. Esimerkiksi maalin syntyessä ensin klikataan maalin tehnyttä pelaajaa ja sitten ”Maali” -painiketta, jolloin käyttäjältä kysytään tarkempia tietoja maalista (KUVA 10).

KUVA 10. Maalin lisätiedot

Käyttäjän lisäämät tilastomerkinnät, kuten maalit ja jäähyt, näytetään käyttäjälle listassa. Tältä listalta käyttäjä voi halutessaan poistaa haluamiaan tilastomerkintöjä (KUVA 1 11). Tilastomerkinnästä näytetään sen omistajan pelinumero ja nimi, tilastotyyppi ja pelikellon aika, jolloin tilasto syntyi.

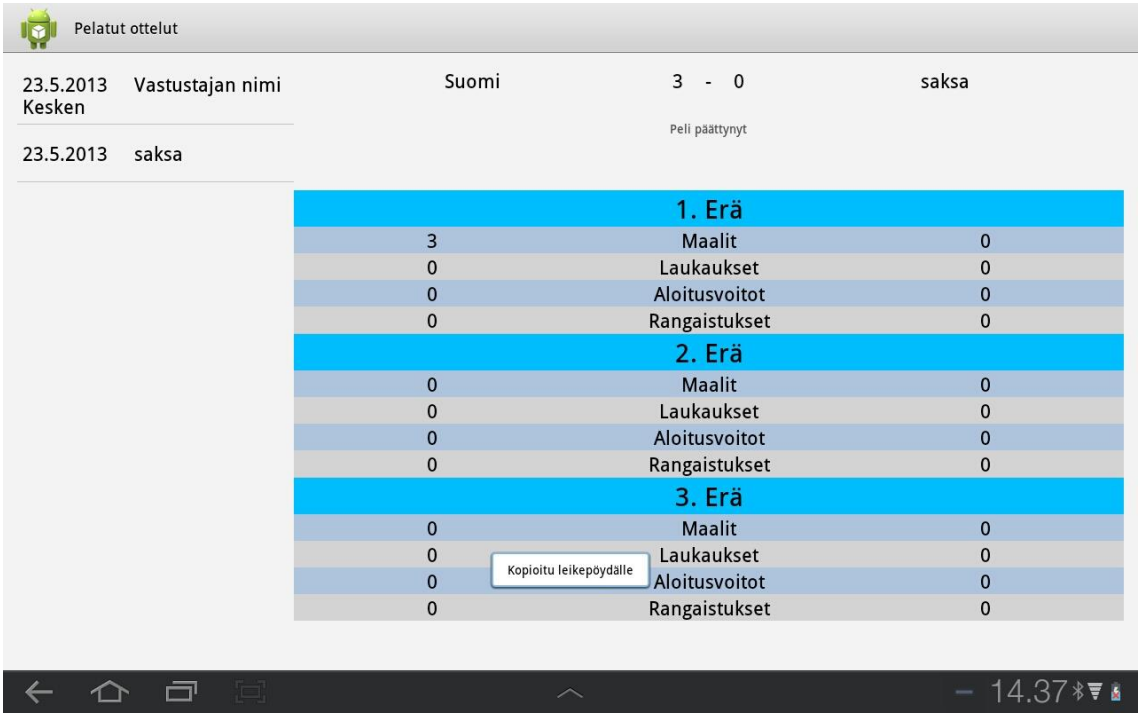


KUVA 11. Tilastomerkinnän poistaminen

Loukkaantumisten tai pelillisten syiden takia jääkiekko-ottelussa voidaan tehdä muutoksia ketjukoostumuksiin. Tämäkin on otettu huomioon. Pelaajia voi siirrellä ketjun sisäisesti tai ketjujen välillä valitsemalla vihreällä pohjalla oleva pelaaja ja vetämällä tämän uudelle pelipaikalle. Jos uudella pelipaikalla oli jo entuudestaan pelaaja, vaihtavat nämä pelaajat paikkaa päikseen.

5.7 Tilastojen katseleminen

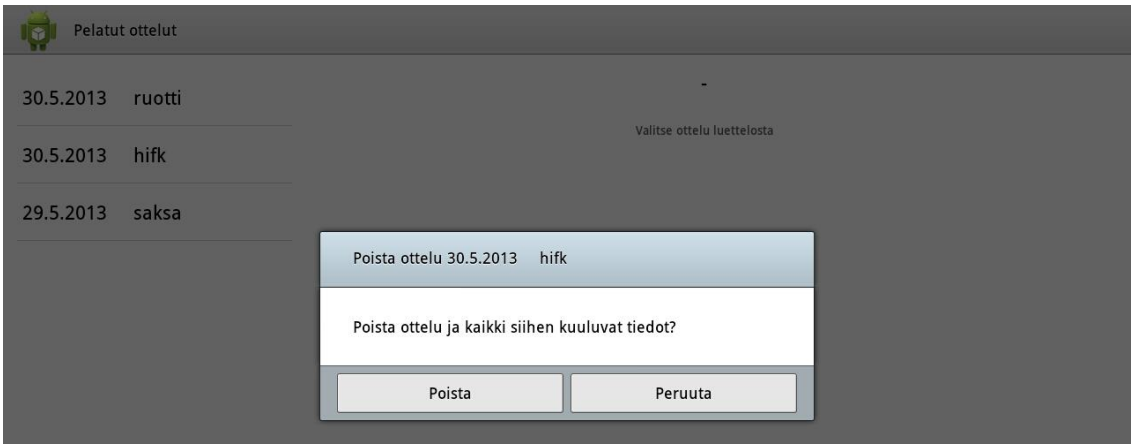
Päätelaitteella käyttäjä voi tarkastella pelattujen otteluiden tilastomerkintöjä joukkue ja erä kohtaisesti (KUVA 12). Tilastoista esitetään vain maalit, laukaukset, aloitusvoitot ja rangaistusminuutit. Pelaajittain tilastoja pääsee tarkastelemaan Web-sovelluksen kautta. Ottelut ja niihin kuuluvat tilastot poistetaan päätelaitteesta, kun käyttäjä on onnistuneesti ladannut tiedot Web-palveluun. Tämän jälkeen tilastoihin pääsee käsiksi vain Web-palvelun kautta.



KUVA 12. Otteluiden tilastot

Pelatut ottelut listataan ja käyttäjä voi valita listalta, minkä ottelun tilastoja tämä haluaa tarkastella. Listalla olevia otteluita on mahdollista poistaa painamalla pitkään listalla olevaa ottelua, jonka haluaa poistaa (KUVA 13. Poista ottelu 13). Tällöin käyttäjältä varmistetaan haluaako tämä todella poistaa

ottelun ja kaikki siihen liittyvät tiedot. Jos käyttäjä poistaa ottelun tilastot, niin tilastot häviävät lopullisesti, eikä niitä voi ladata Web-palveluun.



KUVA 13. Poista ottelu

5.8 Sovelluksen asetukset

Sovelluksen asetuksiin pystyy tällä hetkellä tallentamaan vain käyttäjätunnuksen ja salasanan, joilla käyttäjä tunnistetaan, kun tämä synkronoi päätelaitteen ja Web-palvelun tietokantoja keskenään. Tarvittaessa sovelluksen asetuksiin voidaan lisätä erilaisia asetuksia, mutta niitä ei ole vielä määritelty.

5.9 Web-sovellus

Tilastointisovellusta varten luodaan erillinen Web-sovellus, minkä avulla käyttäjät voivat rekisteröityä palveluun, hallita tietojaan, luoda joukkueita ja pelaajia sekä lisätä kausia ja pelattavia liigoja. Web-sovelluksen kautta, sinne ladattuja tilastoja, voidaan tarkastella paljon tarkemmin ja yksityiskohtaisemmin, mitä päätelaitteella. Web-sovelluksen toteutuksesta vastaa toinen projektiryhmä. Web-sovelluksen määrittely on osa tätä opinnäytetyötä ja määrittely käydään läpi tässä kappaleessa. Määrittely on jaettu kolmeen pääosaan: sovelluksen käytettävyyteen, sovelluksen toimintojen kuvaamiseen sekä sovelluksen tietokantaratkaisun kuvaamiseen.

5.9.1 Käyttöliittymä ja käytettävyys

Web-sovelluksen käyttöliittymän ulkoasun määrittely jätetään sovelluksen kehittäjälle. Tässä kappaleessa käydään läpi lähinnä sovelluksen vaatimuksia käytettävyyden kannalta.

Sovelluksen käyttöliittymän tulee olla yksinkertainen ja helppo käyttää. Sovellusta pitää pystyä käyttämään vaivatta ilman tarkkoja ohjeita, myös sellaisten henkilöiden, jotka eivät sovellusta ole aiemmin käyttäneet. Kaikkien linkkien ja nappien teksien pitää olla tarpeeksi informatiivisia, jotta käyttäjä tietää tai osaa ainakin olettaa, mitä nappia painamalla tapahtuu.

Käyttöliittymässä tulee olemaan erilaisia lomakkeita, joihin kaikkiin pitää tulla tarpeelliset tarkistukset, että käyttäjä ei saa aikaiseksi virhetilanteita.

5.9.2 Toiminnot

5.9.2.1 Käyttäjien hallinta ja oikeudet

Sovelluksen avulla yksittäisen käyttäjän pitää pystyä rekisteröitymään sovellukseen. Rekisteröitymistä varten käyttäjältä vaaditaan käyttäjätunnus, salasana ja sähköpostiosoite. Vapaaehtoisina tietoina käyttäjä voi antaa oman etunimen ja sukunimen. Käyttäjätilejä voi olla kolmenlaisia: admin, pääkäyttäjä ja normaalikäyttäjä. Admin-tunnuksia käyttävät palvelun ylläpitäjät. Admin-tunnuksilla päästään käsiksi käyttäjien hallintaan eli käyttäjiä voidaan muokata tai poistaa. Jos admin poistaa pääkäyttäjätilin, niin kaikki tiliin liitetyt tiedot poistetaan tietokannasta. Normaalikäyttäjän poistaminen ei vaikuta muihin tietoihin. Lisäksi admineilla on oikeus nähdä kaikkien käyttäjätilien tilastotiedot. Pääkäyttäjällä on oikeus lisätä normaalikäyttäjiä. Normaalikäyttäjä liitetään pääkäyttäjän tiliin. Tällöin normaalikäyttäjillä voi olla vain yksi pääkäyttäjä, mutta pääkäyttäjällä voi olla monta normaalikäyttäjää. Pääkäyttäjä voi lisätä, muokata ja poistaa tiliin liitettyjä normaalikäyttäjiä, joukkueita, kausia, pelaajia, otteluita ja tilastotietoja. Normaalikäyttäjä voi tablet-sovelluksella ladata otteluiden tilastoja Web-sovelluksen tietokantaan sekä katsella tilastoja Web-sovelluksen kautta.

5.9.2.2 Joukkueiden hallinta

Pääkäyttäjä voi lisätä, muokata ja poistaa joukkueita. Joukkueita varten pääkäyttäjän pitää myös pystyä lisäämään, muokkaamaan ja poistamaan, sarjoja, kausia ja kausityyppejä.

5.9.2.3 Pelaajien hallinta

Pääkäyttäjä voi lisätä, muokata ja poistaa pelaajia. Jos pelaaja, jolla on tilastomerkintöjä, aiotaan poistaa, niin pelaajaa ei oikeasti poisteta. Vain pelaajan aktiivisuustila muutetaan ei-aktiiviseksi. Näin ollen pelaajan tilastot säilyvät ja niitä voidaan edelleen tarkastella.

5.9.2.4 Tilastojen hallinta

Sovelluksella pitää pystyä tekemään muutoksia tilastomerkintöihin, jos jälkeenpäin huomataan, että tilastoissa on virheitä. Tilastoja pitää pystyä selaamaan helposti joukkueittain, otteluittain, kenttäpelaajittain ja maalivahdeittain. Tilastoja pitää pystyä rajaamaan, sarjoittain, kausittain, kausityyppien mukaan sekä pelaajien aktiivisuuden mukaan. Tilastoja pitää pystyä myös järjestämään suuruusjärjestykseen yksittäisten tilastomerkintätyyppien mukaan. Esimerkiksi pelaajat täytyy pystyä järjestämään pienuus-/suuruusjärjestykseen vaikkapa tehtyjen maalien osalta.

5.9.2.5 Päätelaitteen ja Web-sovelluksen tietokantojen synkronointi-palvelu

Web-sovelluksella täytyy olla rajapinta, minkä avulla tietoa voidaan siirtää päätelaitteesta Web-sovellukseen ja toisinpäin. Tämä suoritetaan käyttämällä JSON-tiedonsiirtomuotoa. Palvelua kutsutaan aina päätelaitteesta. (verkkosivu. JSON).

5.9.3 Tietokanta

Tietokantana käytetään MySQL-tietokantaa. Tietokanta on rakenteeltaan samanlainen kuin aiemmin tässä dokumentissa esitelty päätelaitteen tietokanta. (verkkosivu. MySql).

6 TESTAUS

Projektia varten ei laadittu testaussuunnitelmaa. Projektia on testattu sitä mukaa, kun uusia ominaisuuksia on tullut lisää. Jatkuvalla yksikkötestaamisella on pyritty heti karsimaan suurimmat ohjelmointivirheet. Pienempiä ohjelmointivirheitä on korjailtu, kun niitä on löytynyt ja jos ne ovat oleellisesti haitanneet sovelluksen testaamista. Kaikkein pienimpiä virheitä ei ole vielä ryhdytty korjaamaan, vaan löytyessä niitä on listattu, jotta ne voidaan myöhemmin korjata viimeiseen versioon.

Vakavien ja epäselvien ohjelmistovirheiden paikantamiseen on käytetty Eclipsen debuggeria, jolla koodia voidaan suorittaa rivi kerrallaan. Debuggerin käyttö vaati aluksi hieman opettelua, mutta sen käytön oppi lopulta melko nopeasti.

Lopullinen testaaminen suoritetaan, kun sovellus täyttää kaikki vaaditut toiminnot. Tällöin testaamisesta laaditaan yksinkertainen testaussuunnitelma ja löydetyt virheet kirjataan. Erillistä testausraporttia testaamisesta ei kuitenkaan kirjoiteta. Testaamisen jälkeen löytyneet virheet korjataan kerralla, minkä jälkeen sovellus testataan vielä kertaalleen. Mikäli uusia virheitä ei löydy, katsotaan projektin olevan valmis.

Sovelluksesta ei ole tarkoitus tehdä kaupallista, vaan sen tehtävänä on toimia POC-versiona, jonka avulla projektille voitaisiin hankkia rahoitusta ja aloittaa jatkokehitys. Mahdollista jatkokehitystä varten testaamisen aikana pyritään miettimään uusia ominaisuuksia, joita sovellukseen voisi jatkossa lisätä. Näitä ominaisuuksia ei kuitenkaan toteuteta tämän opinnäytetyöprojektin aikana.

7 TYÖN ARVIOINTI

Työn aihe oli erittäin mielenkiintoinen. Jääkiekko on aina ollut lähellä sydäntä, mikä auttoi suuresti työtä suunnitellessa ja tehdessä. Projektin alussa sovelluksen suunnitteluun käytettiin paljon aikaa, mikä helpotti ja nopeutti sovelluksen kehittämistä. Myös projektia edeltäneet Android-ohjelmointia sisältäneet opintojaksot osoittautuivat hyödyllisiksi. Erityisesti kehitysympäristön asentaminen ja käyttöönottaminen sekä Androidin dokumentaation käyttäminen olivat tulleet tutuiksi opintojaksojen aikana.

Aluksi tarkoitus oli pyrkiä käyttämään hyödyksi aiempina vuosina tehtyjen sovelluksien lähdekoodeja. Lähes heti kuitenkin huomattiin ja päätettiin, että on parempi tuottaa itse kaikki koodi kuin käyttää valmiita koodeja. Uskon, että tämä nopeutti koodaamista huomattavasti sekä vähensi mahdollisten ohjelmointivirheiden syntymistä. Varsinkin aiempien sovellusten tietokantaratkaisu ei tuntunut kovin hyvältä.

Heti sovelluskehityksen alkuvaihella halusin, että sovellusta on helppo jatkokehittää. Tämän vuoksi pyrin kommentoimaan koodiani mahdollisimman paljon ja selkeästi. Lisäksi pyrin nimeämään muuttujia järkevästi, niin että asiaan tutustumaton ohjelmoija ymmärtää lähdekoodia helposti. Yritin myös tehostaa ja tiivistää koodia mahdollisimman paljon. Eli kun huomasin, että jokin asia toistuu monissa kohdissa, tein siitä oman metodin. Näin vältin paljon turhaa koodin toistamista ja mielestäni onnistuin kirjoittamaan tehokasta koodia.

Projektin haastavuus ja kiinnostavuus vastasivat odotuksiani täydellisesti. Tiesin, että projekti olisi haastava, mikä lisäsi mielenkiintoa itse sovelluskehitystä kohtaan. Dokumenttien ja opinnäytetyön kirjoittaminen puolestaan eivät olleet mielenkiintoisia, mikä aiheutti ongelmia opinnäytetyön loppuvaiheessa.

Olen tyytyväinen projektin lopputulokseen. Projektin aikana opin käyttämään paljon uusia tekniikoita ja työkaluja. Ja vaikka projektia ei saatu täysin valmiiksi tämän opinnäytetyön aikana, työn tilaaja sekä ohjaava opettaja ovat olleet tyytyväisiä projektin tuloksiin. Projektia on tarkoitus jatkaa vielä kesän yli valmistumisen jälkeenkin, sillä saan siitä itselleni hyvän käyntikortin työnhakuun tulevaisuudessa.

8 POHDINTA

Olen tyytyväinen, kuinka opettavainen opinnäytetyöprojekti on ollut. Vaikka työ onnistui mielestäni hyvin, tekisin kuitenkin joitakin asioita toisin. Keskittyisin enemmän ohjelmiston suunnittelun ja määrittelyn dokumentoimiseen. Saisin selkeämmät linjat siitä, millä aikataululla seuraava asia pitää toteuttaa. Välttäisin myös lisäominaisuuksien keksimistä ohjelmointivaiheessa. Laatisin myös paljon tarkemman projektiaikataulun ja yrittäisin pitää siitä kiinni paremmin. Oman haasteensa toi työssäkäyminen projektin ohella. Tämän vuoksi projektia piti työstää vapaa-ajalla, iltaisin ja öisin. Nukkuminen jäi vähälle moneen otteeseen, mikä nosti stressitasoa ajoittain hyvinkin paljon. Erityisesti harmittamaan jäi se, etten saanut aihetta yhtään aikaisemmin. Halusin valmistua normaalin aikataulun mukaisesti, mikä aiheutti kiirettä opinäytetyön tekemiseen.

Opinnäytetyön kirjoittaminen jäi myös viime tippaan. Jos aloittaisin saman projektin uudelleen, varaisin opinnäytetyön kirjoittamiseen enemmän aikaa pidemmältä aikaväliltä ja pyrkisin kirjoittamaan opinnäytetyötä useammissa lyhyemmissä jaksoissa. Varsinkin kirjoittamisen aloittaminen oli aina erittäin hankalaa, joten minun täytyisi kehittää itsekuriani.

Ohjelmistokehityksessä pyrkisin jaottelemaan sovelluksen vielä paremmin loogisempiin osiin. Loogiset osat pyrkisin koodaamaan valmiiksi ennen uuden osan koodaamisen aloittamista. Lisäksi koodin kommentointini ei ollut hyvällä tasolla koko ohjelmistokehityksen aikana, vaan se oli ajoittain hieman ailahtelevaista ja sekavaa.

Opinnäytetyön tekeminen oli ajoittain henkisesti todella raskasta, mutta työn tilaajan ja ohjaavan opettajani sekä ystävieni ja tuttujen kannustuksella jaksoin loppuun asti.

LÄHTEET

Android SDK. [verkkosivu].[viitattu 15.4.2013]. Saatavissa:
http://developer.android.com/sdk/index.html

ADT-lisäosa. [verkkosivu].[viitattu 15.4.2013]. Saatavissa:
http://developer.android.com/tools/sdk/eclipse-adt.html

Eclipse IDE. [verkkosivu].[viitattu 15.4.2013]. Saatavissa:
http://www.eclipse.org/

SQLite. [verkkosivu].[viitattu 22.4.2013]. Saatavissa:
http://www.sqlite.org/

Navicat Data Modeler. [verkkosivu].[viitattu 26.3.2013]. Saatavissa:
http://www.navicat.com/products/navicat-data-modeler

Subversive SVN. [verkkosivu].[viitattu 4.5.2013]. Saatavissa:
http://www.eclipse.org/subversive/

JSON. [verkkosivu].[viitattu 18.5.2013]. Saatavissa:
http://www.w3schools.com/json/default.asp

MySql. [verkkosivu].[viitattu 19.5.2013]. Saatavissa:
http://www.mysql.com/

Silander S., Ollikainen V., Peltomäki J., Kosonen P. 2010. Java. Docendo Oy; Jyväskylä.

LIITE 1

PROJEKTISUUNNITELMA

Projektisuunnitelma

Projektin nimi: Kalpa Juniorit

Suunnitelman laatimispäivämäärä: 3.4.2013

Suunnitelman laatijat: Matti Ilvonen

Projektin lyhyt kuvaus: Projekti tehdään opinnäytetyönä tmi Jussi Koistiselle. Projektin tavoitteena on saada aikaiseksi sovellus, jolla voidaan helposti tallentaa jääkiekko-ottelun aikana syntyneitä tilastomerkintöjä. Sovelluksella pystyy myös tarkastelemaan syntyneitä tilastoja. Syntyneet tilastot voidaan myös viedä ulkoiseen tietokantaan, josta niitä voidaan käsitellä tietokoneella erillisen sovelluksen avulla.

Versio: 1.0

VERSIOHISTORIA

0.1	13.4.2013	Dokumentti luotu	Matti Ilvonen
0.2	24.4.2013	Opettajan ehdottamat korjaukset tehty	Matti Ilvonen
1.0	4.6.2013	Viimeiset korjaukset ja liittäminen opinnäytetyöhön	Matti Ilvonen

SISÄLLYSLUETTELO

1. JOHDANTO

1.1. Tuote

Opinnäytetyön tavoitteena on saada valmis ja toimiva sovellus jääkiekkotilastojen käsittelyyn. Sovelluksella pitää pystyä tallentamaan tilastotietoja pelin aikana sekä muokkaamaan tilastoja vielä pelin jälkeen jos niihin on tullut virheellisiä tietoja. Android sovelluksella ei voida luoda joukkueita eikä lisätä pelaajia joukkueisiin. Nämä toiminnot toteutetaan web pohjaisella työkalulla. Sitten Android sovelluksen tietokanta synkronoidaan ulkoisen tietokannan kanssa, jolloin joukkueet ja pelaajat latautuvat tabletille.

1.2. Yleiskuvaus

Projekti toteutetaan opinnäytetyönä tmi Jussi Koistiselle. Opinnäytetyön tekijänä toimii Matti Ilvonen. Ohjaavina opettajina toimivat Sami Lahti ja Keijo Kuosmanen. Opinnäytetyön tavoitteena on saada toimiva sovellus jääkiekkotilastojen tallentamista varten Android tabletti-tietokoneille. Projekti tullaan suunnittelemaan ja dokumentoimaan hyvin mahdollista jatkokehitystä silmällä pitäen. Projektia varten on olemassa paljon valmista lähdekoodia, josta osaa voidaan käyttää tätä projektia varten. Mutta esimerkiksi sovelluksen tietokanta pitää suunnitella kokonaan uudelleen.

1.3. Suunnitelman ylläpito

Tätä projektisuunnitelmaa päivitetään aina tarvittaessa projektin edetessä.

2. PROJEKTIN ORGANISOINTI

2.1. Projektin vaiheistus

Aluksi projektissa pidetään opinnäytetyön aloituspalaveri. Tähän palaveriin osallistuu opinnäytetyön tilaaja, ohjaavat opettajat sekä opinnäytetyön tekijä. Vastaavia palavereita pyritään pitämään tasaisin väliajoin, jotta projektin tavoitteet ja aikataulu pysyisivät kurissa.

Aloituspalaverin jälkeen aloitetaan tietokannan suunnittelu. Kyseessä on monimutkainen sovellus, joka vaatii monimutkaisen tietokannan. Tietokannan pitää olla selkeä ja hyvin sovelluksessa tehtäviin muutoksiin mukautuva. Tietokannasta pitää pystyä hakemaan paljon erilaisia historiatietoja, kuten aiemmin pelattujen pelien kokoonpanot ja tilastot. Tietokantaan pitää pystyä tallentamaan tilastotiedot mahdollisimman tarkasti. Ennen sovelluksen varsinaisen kehittämisen aloittamista projektista laaditaan projektisuunnitelma sekä yksinkertainen määrittelydokumentti.

Sovelluksen kehittäminen aloitetaan alustavien dokumenttien valmistuttua. Sovelluskehityksen rinnalla aloitetaan kirjoittamaan myös opinnäytetyön kirjallista osuutta, että se ehtii kieltenopettajien tarkistukseen toukokuun alkuun mennessä.

Sovellusta testataan sitä mukaa kun uusia ominaisuuksia tulee lisää. Sovellus pyritään saamaan varsinaiseen kenttätestaukseen vuoden 2013 jääkiekon MM-kisoihin mennessä. Sovellus pyritään saamaan kokonaan valmiiksi vuoden 2013 kesäkuuhun mennessä. Opinnäytetyö pyritään saamaan valmiiksi samaan aikaan.

2.2. Vastuuhenkilöt

Taulukossa (Taulukko 1 Vastuuhenkilöt) esitetään projektiin osallistuvat henkilöt, heidän yhteystiedot sekä rooli projektissa.

Taulukko 1 Vastuuhenkilöt

Nimi	**Titteli**	**Sähköposti ja puhelinnumero**	**Rooli**
Matti Ilvonen	Opiskelija	matti.a.ilvonen@edu.savonia.fi 044 350 8288	Opinnäytetyöntekijä
Sami Lahti	Lehtori	sami.lahti@savonia.fi 044 785 6337	Opinnäytetyön 1. ohjaaja
Keijo Kuosmanen	Lehtori	keijo.kuosmanen@savonia.fi 044 785 6371	Opinnäytetyön 2. ohjaaja
Jussi Koistinen	Lehtori	jussi.koistinen@savonia.fi 044 785 5512	Opinnäytetyön tilaaja

3. PROJEKTIN OHJAAMINEN

3.1. Tavoitteet ja priorisointi

Tavoitteena on saada sovellus sekä siihen liittyvä opinnäytetyö valmiiksi kesään 2013 mennessä. Tärkeimpänä osa-alueena on itse opinnäytetyö. Opinnäytetyöstä pyritään saamaan mahdollisimman hyvä arvosana. Siksi siihen tullaan panostamaan paljon resursseja varsinkin projektin loppupuolella.

Seuraavaksi tärkein asia on itse sovellus. Sovelluksesta pyritään saamaan valmis ja toimiva kokonaisuus. Sovelluksen kehitykseen lasketaan mukaan tietokannan suunnittelu ja toteutus. Tietokannan suunnittelun tärkeys on suuri, sillä se luo pohjan koko sovelluksen toimivuudelle. Jos näyttää siltä, että aika loppuu kesken, eikä sovellusta saada kokonaan valmiiksi määräaikaan mennessä, pyritään sovellus dokumentoimaan ja lähdekoodi kommentoimaan erittäin tarkasti jatkokehitystä varten.

3.2. Riskien hallinta

Alla olevassa taulukossa (Taulukko 2 Mahdollisia riskejä) esitellään mahdollisia projektissa esiintyviä riskejä.

Taulukko 2 Mahdollisia riskejä

Riski	Todennäköisyys	Varautuminen	Vaikutus
Ajan loppuminen kesken	Pieni	Sovelluskehitystä jatketaan haluttuun pisteeseen asti.	Tuotetta ei saada valmiiksi
Tietotaidon riittämättömyys	Pieni	Projektissa vaaditaan paljon erilaisia tietoja ja taitoja. Joitakin asioita joudutaan opettelemaan tarvittaessa. Ohjaavilta opettajilta pyydetään neuvoja.	Aikaa kuluu asian opiskeluun.
Laitteen ja ulkoisen tietokannan synkronointi	Kohtalainen	Sovellus suunnitellaan siten, että sovellusta voidaan käyttää myös ilman ulkoista tietokantaa.	Kaikkia haluttuja ominaisuuksia ei saada toteutettua kunnolla.

3.3. Seuranta ja ohjaus

Opinnäytetyön tekijä Matti Ilvonen kutsuu asiakkaan ja ohjaavat opettajat palaveriin parin viikon välein. Tällöin voidaan käydä läpi esille tulleita ongelmia ja yrittää löytää niihin ratkaisuja. Joka kerta selvitetään mitä on saatu aikaiseksi ja mitä seuraavaksi tehdään. Täten seurataan projektin aikataulua ja laajuutta. Aikataulun pitäisi pysyä kurissa, kun projektin laajuus rajataan selvästi. Projektiin voi tulla kuitenkin pieniä muutoksia joka palaverissa.

Ongelmien esiintyessä ohjausta voidaan pyytää opettajilta myös sähköpostitse.

4. TEKNIIKKA

4.1. Menetelmät ja työkalut

Opinnäytetyö ja sovellusprojektin dokumentit laaditaan Microsoft Office:n tarjoamilla työkaluilla. Opinnäytetyö ja muut dokumentit kirjoitetaan suomeksi.

Sovelluksen kehittämisessä käytetään Android SDK:ta, joka sisältää Eclipse sovelluskehittimen sekä emulaattorityökalut sovelluksen testaamiseen. Tietokanta suunnitellaan SQLite ja MySQL tietokantaympäristöihin. SQLite tietokanta suunnitellaan käyttämällä Navicat Data Modeler ohjelman 30-päivän ilmaisversiota. SQLite tietokantaa voidaan muokata ja testata käyttämällä ilmaista SQLite Database Browser ohjelmaa. MySQL tietokanta suunnitellaan ilmaisella MySQL Workbench ohjelmalla ja se testataan käyttämällä ilmaista phpMyAdmin sovellusta. Sovelluskehityksessä kielenä käytetään englantia.

4.2. Dokumentointi

Opinnäytetyön käytännön osuudesta tullaan laatimaan projektisuunnitelma ja määrittelydokumentti. Kaikista palavereista tullaan kirjoittamaan lyhyt palaverimuistio. Sovelluksen lähdekoodit tullaan kommentoimaan huolellisesti. Tietokannan kuvaus dokumentoidaan tarkasti ja selkeästi. Sovelluksen dokumentointi on tärkeää sen jatkokehityksen kannalta ja siksi se pyritään tekemään huolella. Dokumentointi tapahtuu kuitenkin vasta kun opinnäytetyön kirjallinen osuus sekä sovellus itse ovat valmiita. Lähdekoodi pyritään kommentoimaan heti koodaamisvaiheessa.

5. AIKATAULUT

5.1. Sovelluskehityksen aikataulu

Määrittelydokumentti toimii sovelluskehityksen suuntaa-antavana aikatauluna. Sovelluskehitykselle ei laadita tarkkaa aikataulua.

5.2. Opinnäytetyön aikataulu

Alla olevassa taulukossa (Taulukko 3 Aikataulu) esitetään projektin aikataulu.

Taulukko 3 Aikataulu

Päivämäärä	Tapahtuma	Valmis
14.3.2013	Opinnäytetyön aloituspalaveri	14.3.2013
27.3.2013	Palaveri, aihe: Tietokannan suunnittelu	27.3.2013
11.4.2013	Palaveri, aihe: Missä mennään, mitä seuraavaksi, ongelmia?	11.4.2013
24.4.2013	Palaveri, aihe: Missä mennään, mitä seuraavaksi, ongelmia?	26.4.2013
26.4.2013	Opinnäytetyö seminaari	26.4.2013
3.5.2013	Opinnäytetyön 1. versio kieltenopettajille tarkistukseen.	8.5.2013
8.5.2013	Palaveri, aihe: Käydään läpi testituloksia.	8.5.2013
22.5.2013	Palaveri, aihe: Käydään läpi testituloksia sekä viimeiset muutosehdotukset.	30.5.2013
29.5.2013	Palaveri, tarvittaessa	elokuu 2013
31.5.2013	Tuotteen luovutus asiakkaalle	elokuu 2013