KARELIA-AMMATTIKORKEAKOULU
Tietojenkäsittelyn koulutusohjelma

Teemu Räsänen

ALUSTARIIPPUMATON TEHTÄVIENHALLINTASOVELLUS MOBIILIALUSTALLE

Opinnäytetyö
Joulukuu 2013

Karelia
AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Joulukuu 2013
Tietojenkäsittelyn koulutusohjelma

Karjalankatu 3
80200 JOENSUU
013260600

Tekijä
Teemu Räsänen

Nimeke
Alustariippumaton tehtävienhallintasovellus mobiilialustalle

Toimeksiantaja
Arcusys Oy

Tiivistelmä
Opinnäytetyön tavoitteena oli toteuttaa alustariippumaton tehtävienhallintasovellus mobiilialustalle. Tehtävillä tarkoitetaan Intalio-palvelimella olevia sähköisiä lomakkeita, joita voi avata, muokata, luoda ja selata mobiilisovelluksella. Mobiilisovellus hakee lomaketiedot Intalio-palvelimelta. Työssä kartoitettiin useita eri teknologioita ja työkaluja ja niiden ominaisuuksia. Vertailemalla eri työkalujen ja teknologioiden ominaisuuksia työn vaatimuksiin löydettiin sopivat teknologiat ja työkalut. Mobiilisovellus kehitettiin käyttämällä avoimen lähdekoodin jQuery Mobile -ohjelmistokehystä.

Kehityksessä on pyritty saamaan mobiilisovellus mahdollisimman käyttäjäystävälliseksi yksinkertaisella käyttöliittymällä ja mobiilisovelluksen integroinnilla Intalio-palvelimeen. Käyttäjä näkee vain yhden järjestelmän, vaikka mobiilisovellus keskustelee Intalion ohjelmistorajapinnan kanssa. Käyttöliittymää suunniteltaessa otettiin huomioon erilaiset laitteet ja niiden käyttötavat. Käyttöliittymäkomponenttien asettelussa varmistettiin niiden toimivuus laitteen asennosta riippumatta.

Työn tuloksena kehitettiin avoimen lähdekoodin alustariippumaton mobiilisovellus, joka on testattu ja todettu toimivaksi Android-laitteilla. Toteutetulla mobiilisovelluksella voidaan hallita Intalio-palvelimella olevia sähköisiä lomakkeita.

Kieli
suomi

Sivuja 34

Asiasanat
alustariippumaton, mobiilisovellus, mobiilikehitys, mobiiliteknologiat, verkkosovellus, jQuery Mobile, HTML5, JavaScript, Intalio

Karelia
UNIVERSITY OF APPLIED SCIENCES

THESIS
December 2013
Degree Programme in Business Information Technology
Karjalankatu 3
FI 80200 JOENSUU
FINLAND
013260600

Author
Teemu Räsänen

Title
Cross-Platform Task Manager for a Mobile Platform

Commissioned by
Arcusys Oy

Abstract
The goal of the thesis was to implement a cross-platform task manager application for a mobile platform. Tasks comprise of electronic forms in the Intalio-server, that can be opened, modified, created and browsed with a mobile application. The mobile application retrieves form data from Intalio-server. In this study several different technologies and tools and their features were examined. By comparing the features of different tools and technologies to requirements suitable technologies and tools were found. A mobile application was developed by using open source jQuery Mobile framework.

The aim in the development was to achieve the best possible usability by using simple user interface and integrating the mobile application to Intalio-server. User operates with only one system even if the mobile application communicates with Intalio's application programming interface. In the user interface design different devices and their use were taken into account. In the layout of the user interface the functionality of the interface components were ensured regardless of the position of the device.

As a result of this thesis an open source cross-platform mobile application was developed The mobile application was tested and confirmed to be fully functional on Android devices. It is possible to manage online forms with an implemented mobile application.

Language
Finnish

Pages 34

Keywords
cross-platform, mobile application, mobile development, mobile technologies, web application, jQuery Mobile, HTML5, JavaScript, Intalio.

Sisältö

Lyhenteet ja käsitteet

API	Application Programming Interface eli ohjelmistorajapinta, joka koostuu metodeista. Tarkoittaa siis eräänlaista käyttöliittymää ohjelmoijalle. Sen avulla voi käyttää jonkin laitteen tai sovelluksen ominaisuuksia kuten laite päälle ja pois -metodi.
AWT	Abstract Window Toolkit on Javan alustariippumaton käyttöliittymien kehitystä varten tehty kirjasto.
Dalvik	Googlen prosessivirtuaalikone Androidille, jossa Android-sovellukset ajetaan.
GWT	Google Web Toolkit on avoimen lähdekoodin työkalu selainpohjaisten sovellusten kehitystä varten.
IDE	Integrated Development Environment eli ohjelmointiympäristö, joka tarjoaa tarvittavat työkalut ohjelmointia varten kuten koodieditorin, kääntäjän ja versionhallinnan.
JIT	Just In Time eli ajonaikainen kääntäminen. Sovellus on ladattu muistiin tavukoodina ja ajon aikana tarvittavat osat käännetään konekoodiksi.
Natiivi	Viittaa johonkin alkuperäiseen jo olemassa olevaan ominaisuuteen laitteessa. Esimerkiksi mobiiliympäristössä natiivilla tarkoitetaan alkuperäistä käyttöliittymää, valmiiksi asennettuja sovelluksia ja laitteessa suoritettavaa konekoodia. Sovelluksella voidaan sanoa olevan natiivi tuntuma, jos sen käyttöliittymä ja ulkoasu vastaavat käytetyn käyttöjärjestelmän käyttöliittymää ja ulkoasua.
SDK	Software Development Kit on tyypillisesti joukko ohjelmistokehitystyökaluja, joilla voi luoda sovelluksia tietyille alustoille tai ohjelmistoille kuten käyttöjärjestelmille, peleille tai vastaaville.
SOAP	Simple Object Access Protocol on protokollamääritelmä tiedon siirtämistä varten.
Swing	Javan alustariippumaton käyttöliittymäkirjasto, joka ei käytä käyttöjärjestelmän omia käyttöliittymäkomponentteja kuten AWT.
Token	Sähköinen tunniste, josta järjestelmä tunnistaa tietyn käyttäjän. Käytetään yleensä kirjautumisvaiheessa ja tallennetaan onnistuneen kirjautumisen jälkeen. Jotkin rajapinnat vaativat tokenin käyttämistä palvelupyyntöjen yhteydessä ja jos token on virheellinen tai sitä ei ole, pyyntöä ei käsitellä.

WebKit Selainmoottori on selaimen ydinkomponentti, jonka moniin tehtäviin kuuluu HTML- ja CSS-koodin tulkkaaminen ja verkkosivun piirtäminen näytölle. WebKit on käytössä Chrome-selaimessa.

XML Extensible Markup Language -merkintäkieli, jota voi lukea ihminen tai kone. Sitä käytetään yleensä tiedonsiirrossa tai kuvaamaan sovelluksen asetuksia.

1 Johdanto

Tein työharjoittelujakson Arcusys oy:ssä, josta opinnäytetyön aihe löytyi. Arcusys oy:n tarjoamiin palveluihin kuuluu erilaisten tietojärjestelmien luominen asiakkaiden tarpeiden mukaisesti. Näihin tietojärjestelmiin voi kuulua sähköisiä lomakkeita, joihin tämä opinnäytetyö liittyy. Sähköisen lomakkeen prosessi tyypillisesti käynnistyy, kun asiakas täyttää sähköisen lomakkeen ja lähettää sen eteenpäin. Lähetetyllä lomakkeella on käsittelyvaiheen mukaan eri tiloja ja lomakkeet sisältävät tehtävän kannalta tarpeellisia tietoja, kuten esimerkiksi käsittelijän merkintöjä. Sähköiset lomakkeet ovat Intalio-palvelimella.

Tämän opinnäytetyön tarkoituksena on kehittää mobiilisovellus, joka pystyy listaamaan ja kategorisoimaan sähköiset lomakkeet saapuneisiin ja käsiteltyihin. Mobiilisovelluksen kautta on myös mahdollista täyttää saatavilla olevia lomakkeita ja lähettää ne eteenpäin. Arcusys oy:ssä on jo kehitetty sovellus PC-ympäristöön. Tässä opinnäytetyössä pohditaan erilaisten sovelluskehittimien ja teknologioiden eroja vertailemalla niitä keskenään ja valitaan niistä työn kannalta parhaaksi havaittu teknologia sovelluksen kehitystä varten. Teknologian valinnassa käytetään kriteerejä, jotka nähdään kehityksen kannalta tärkeiksi. Tärkeimpinä valintakriteereinä toimivat seuraavat: valmis ohjelmistokehys ja alustariippumaton mobiilisovellus. Toivottavia ominaisuuksia ovat avoin lähdekoodi ja hyvä dokumentaatio. Työn kehitysvaiheesta kirjataan havaintoja valituista työkaluista ja teknologioista. Työ rajataan mobiilisovelluksen kehitykseen, ja palvelinympäristöt, joiden kanssa mobiilisovellus kommunikoi, eivät ole työn tarkastelun kohteena.

2 Mobiilikäyttöjärjestelmät ja kehitystyökalut

Mobiilisovelluksen kehityksessä, kuten myös työpöytä- tai verkkosovelluksien kehityksessä, on tarjolla useita eri teknologioita ja työkaluja ongelman ratkaisua varten. Tässä luvussa vertaillaan teknologioita, mobiilikäyttöjärjestelmiä ja työ-

kaluja, jotka mahdollistavat mobiilikehityksen. Kiinnostavimpiin vaihtoehtoihin perehdytään perusteellisemmin. Lopuksi tehdään lyhyt yhteenveto ja vertailu kaikista tarkastelluista mobiilikehitystyökaluista.

Työssä kehitettävän mobiilisovelluksen kohdeympäristönä on tablet-laitteet. Android ja iOS ovat suosituimpia mobiilikäyttöjärjestelmiä tableteissa ja älypuhelimissa, joten niihin perehdytään tarkemmin. Työmäärän rajaamiseksi työssä ei keskitytä esimerkiksi Windows- tai Symbian-mobiilikäyttöjärjestelmiin tai niiden kehitystyökaluihin. Tämä ei kuitenkaan tarkoita etteikö kehitettävä mobiilisovellus tukisi Windows-alustalla toimivia laitteita, koska työn tavoitteena on löytää alustariippumaton teknologia. Microsoftin tarjoamat työkalut ja kehitysympäristöt rajataan vain pois. Toiseksi on hyvin todennäköistä, että Microsoft tukee kehitystä vain Windows-alustoille, mikä on ristiriidassa työn tavoitteiden kanssa.

Aikaisempaa kokemusta mobiilikehityksestä minulla ei ollut, joten teknologian valinnassa ensimmäinen tehtäväni oli selvittää, millaisia työkaluja mobiilikehitystä varten on olemassa. Tämän jälkeen oli karsittava suuresta listasta erilaisia työkaluja kiinnostavimmat vaihtoehdot.

2.1 Android-käyttöjärjestelmä ja kehitystyökalut

Android on käyttöjärjestelmä, joka on suunniteltu mobiililaitteita varten, joissa käytettävissä olevat resurssit ovat rajalliset (Wikipedia 2013a). Android-sovellukset suoritetaan eristetyssä hiekkalaatikossa. Hiekkalaatikko eristää yksittäiset sovellukset muista sovelluksista, joten ne eivät pääse käsiksi muihin sovelluksiin tai muokkaamaan niitä. Yleisesti Androidille kirjoitetaan sovelluksia Javalla, mutta ennen suoritusta ne käännetään tavukoodiksi ja suoritetaan Dalvik-virtuaalikoneessa. (Cheng & Buzbee 2010.)

Android ei käytä Javan virtuaalikonetta vaan se käyttää Dalvik-virtuaalikonetta, joka käyttää omaa luokkakirjastoa eli Java SE:n tai Java ME:n luokkakirjastot eivät ole käytettävissä. Esimerkiksi käyttöliittymäluokat AWT tai Swing eivät ole

käytettävissä. Tärkeimmät järjestelmäkirjastot on käännetty tavukoodiksi suorituskyvyn parantamiseksi. Käyttäjän tekemät sovellukset ovat käännettyjä Javan luokkatiedostoja, joista osa muutetaan tavukoodiksi. (Wikipedia 2013b; 2013c.)

Android-käyttöjärjestelmän ydintoimintoihin kuuluu JIT. JIT lataa muistiin ja kääntää osan sovelluksesta tarpeen mukaan suorituksen aikana, joten koko sovellusta ei tarvitse ladata kerralla muistiin käännettynä natiivikoodina. Tämä vähentää muistinkäyttöä. JIT havaitsee sovellusten suorituksen aikana, mikä on raskain suoritettava osa ja kääntää tavukoodin optimoiduksi natiivikoodiksi. Käännettyä koodia ei tarvitse tulkata uudelleen ja sovelluksen suorituskyky kasvaa pian sovelluksen käynnistyksen jälkeen. (Cheng & Buzbee 2010.)

Googlen Android-kehityssivuilta voi ladata ilmaiseksi tarvittavat kehitystyökalut Android-sovelluskehitystä varten yhdessä paketissa. Sivuilta voi ladata SDK-paketin, joka sisältää tarvittavat API-kirjastot Android-kehitystä varten ja IDE:n, joka on rakennettu Eclipsen päälle. IDE sisältää toiminnot Android-sovellusten rakentamiseen, testaamiseen, virheenetsintään ja Android-sovellusten paketoimiseen. Se sisältää myös graafisen käyttöliittymän rakentajan, ja komponentteja voidaan lisätä vedä ja pudota -toiminnolla. IDE mahdollistaa käyttöliittymien rakentamisen myös XML-muodossa. (Google Inc. 2013a; 2013b; 2013c.)

Käyttämällä Googlen tarjoamia työkaluja sovellusten kehittäminen Android-laitteille on helppoa, sillä SDK mm. rakentaa sovellukselle valmiin rungon. Android-kehitys on tapahtumapohjaista ja Android-ydin sisältää useita valmiita tapahtumankäsittelijöitä kuten esimerkiksi onCreaten(), onStopin() jne. Esimerkiksi onCreate()-metodilla kehittäjä voi luoda toiminnallisuutta sovelluksen käynnistyksen yhteydessä. Kun sovellus käynnistyy metodia kutsustaan automaattisesti. Metodi onStop() toimii samalla logiikalla, mutta sitä kutsutaan sovelluksen pysähtyessä. Googlen tarjoama SDK-paketti mahdollistaa kehityksen vain Android-laitteille. (Google Inc. 2013d; 2013e.) Julkaisu Google Storeen maksaa 25 dollaria. Google perii maksullisista sovelluksista 30 % sovelluksen kokonaishinnasta. (Google Inc. 2013f; 2013g.)

2.2 iOS-käyttöjärjestelmä ja kehitystyökalut

iOS on Applen kehittämä käyttöjärjestelmä, joka on käytössä Applen mobiililaitteissa (Wikipedia 2013d; 2013e). Yrityksenä mobiilisovellusten julkaisu App Storeen vaatii iOS-kehittäjän tilin, jonka hallussapito maksaa toistuvan 299 dollarin vuosimaksun. Normaali lisenssi yksittäiselle käyttäjälle maksaa 99 dollaria vuodessa, mutta siihen ei sisälly yrityskäyttö. (Apple Inc. 2013a.)

Applelta on saatavissa täydellinen kehitysympäristö, jonka avulla kehittäjät voivat luoda omia sovelluksiaan iOS:lle. Kehitysympäristö koostuu Xcode IDE:stä ja SDK:sta, jotka sisältävät tarvittavat komponentit sovelluskehitystä varten iOS:lle. Käyttöliittymän voi rakentaa vedä ja pudota -toiminolla (Apple 2013b; 2013c). Yleinen ohjelmointikieli iOS-sovelluksille on Objective-C (Apple 2013d). iOS-kehitystyökaluilla ei ole mahdollista kehittää sovelluksia toisille alustoille kuten Androidille tai Windowsille. Sovelluksia voi myös kirjoittaa verkkosovelluksen tapaan käyttäen HTML:ää, CSS:ää tai JavaScriptiä. Kaikki ydintoiminnot, kuten esimerkiksi tekstiviestien lähettäminen, eivät ole kuitenkaan käytettävissä. Applen tarjoamia työkaluja voi käyttää vain Mac-ympäristössä. Grant (2010) on havainnut, että tarjolla on myös OSx86, jonka avulla kehitystyökaluja voidaan ajaa. Hänen mukaansa Mac-ympäristön voi asentaa myös virtuaalikoneeksi ja asentaa kehitystyökalut virtuaaliseen ympäristöön.

2.3 Vaadin ja TouchKit-ohjelmistokehys

Vaadin on Java-ohjelmistokehys, jolla voi luoda verkkosovelluksia työpöytä- tai mobiiliympäristöön. Vaadin-ympäristöön on ladattavissa TouchKit-lisäosa, jolla voidaan luoda sovelluksia Android- ja iOS-alustoille. TouchKit mahdollistaa sovellusten kirjoittamisen käyttämällä pelkästään Javaa. Oletusteema matkii natiiveja iPhone-sovelluksia, mutta teemoja voi muokata samalla tavalla kuin Vaadin-sovelluksissa. Huono puoli TouchKit-mobiilisovelluksissa on se, että ne vaativat toimivan verkkoyhteyden ja palvelimen asennettuna johonkin tietokoneeseen. ToucKit-sovelluksissa voi käyttää toiminnallisuutta, joka mahdollistaa

sovelluksen toimivuuden verkottomassa tilassa. Verkottoman tilan oletustoiminnallisuuden voi muuttaa kirjoittamalla oman GWT-koodin, joka huolehtii sovellusdatan tallennuksesta selaimen välimuistiin. Yhteyden muodostuttua uudelleen välimuistin data synkronoidaan palvelimen kanssa. Näin tehtäessä kehittäjä hallitsee, milloin ja kuinka dataa synkronoidaan palvelimen kanssa. Vaadin-sivuston mukaan TouchKit on kehitteillä oleva ohjelmistokehys, joka ei Vaadin-wikin mukaan ole vielä valmis tuotantokäyttöön. (Vaadin Ltd. 2013a; 2013b.)

2.4 HTML5-kehitys

HTML5 on sivunkuvauskieli, jonka tarkoituksena on esittää verkkosivun rakenne. HTML5 on opinnäytetyön kirjoitushetkellä HTML-kielen uusin versio, jonka useat ominaisuudet on rakennettu niin, että ne huomioivat myös vähätehoiset laitteet kuten älypuhelimet ja tabletit. Uuden version yleistyessä uusia kehitystyökaluja ja ohjelmistokehyksiä ilmestyy koko ajan lisää. (Wikipedia 2013f; 2013g.)

Yksi HTML5:n uusista ominaisuuksista on verkoton tuki AppCache- ja tietokantatiedon tallennusta varten. Tämä mahdollistaa sen, että mobiilisovellukset voivat tallentaa tietoa paikallisesti. Sovellukset voidaan tehdä niin, että mahdolliset häiriöt yhteydessä eivät riko sovellusta tai kadota tietoja. (Wikipedia 2013f; 2013g.)

2.5 DaVinci

DaVinci on kehitystyökalu HTML5-mobiilisovellusten luomista varten. Siihen kuuluu jQuery-ohjelmistokehys, joka on JavaScript-kirjasto. Sitä voidaan käyttää luomaan mobiililaitteille verkkosovelluksia, joilla on samanlainen käyttäjäkokemus kuin natiiveilla mobiilisovelluksilla. DaVinci perustuu HTML5-, CSS3- ja JavaScript-verkkoteknologioihin. (Wikipedia 2013h.)

DaVinci-kehitystyökalun ominaisuuksiin kuuluu vedä ja pudota -menetelmä, jonka avulla käyttäjä voi luoda sovelluksia vetämällä komponentteja haluttuihin kohtiin käyttöliittymää. DaVinci tukee Model View Controller (MVC) -ohjelmointimallia. DaVincissä kehittäjä voi visuaalisesti yhdistää Model-, View- ja Controller-komponentit toisiinsa. DaVincin mukana tulee myös työkalu, jonka avulla kehittäjä voi automaattisesti asettaa muotoilun ja toiminnot erikokoisille näytöille ja laitteille. (Wikipedia 2013h.)

2.6 jQuery Mobile

jQuery Mobile -webohjelmistokehys tunnetaan myös JavaScript-kirjastona tai mobiiliohjelmistokehyksenä. Kehityksessä on keskitytty yhteensopivan ohjelmistokehyksen luomiseen suurelle joukolle älypuhelimia ja tablet-laitteita. jQuery Mobile on yhteensopiva muiden kehitystyökalujen, kuten esimerkiksi PhoneGap, Worklight ja DaVinci, kanssa. (Wikipedia 2013i.)

jQuery Mobile -sovellus kirjoitetaan kerran, minkä jälkeen käyttöliittymä skaalautuu eri laitteille mobiilista työpöytäympäristöön. jQuery Mobile tukee useita suosittuja mobiilialustoja, kuten Android, iOS, BlackBerry, WindowsPhone ja Symbian. Tarkka lista yhteensopivista laitteista löytyy jQuery Mobilen verkkosivulta. (The jQuery Foundation 2013a.) jQuery Mobile on rakennettu jQuery ytimen päälle, joten sen opettelu on helppoa niille, jotka tuntevat jQueryn ennestään (Wikipedia 2013j).

jQuery Mobilen ominaisuuksiin kuuluu mm. ohjelmistokehys, joka mahdollistaa muokattujen teemojen luonnin. jQuery Mobile on vähätehoisille laitteille optimoitu kevennetty versio jQuerysta. jQuery Mobilen kehityksessä on keskitytty tekemään jQuery Mobile helposti opittavaksi. (Wikipedia 2013j.)

jQuery Mobileen kuuluu joukko kosketustuella olevia käyttöliittymäkomponentteja, kuten painikkeita, dialogeja ja lomake-elementtejä. jQuery Mobilen sivut sisältävät laajan dokumentaation eri komponenteista. (The jQuery Foundation 2013b;2013c.)

2.7 PhoneGap

PhoneGap on ilmainen, avoimen lähdekoodin ohjelmistokehys alustariippumattomien mobiilisovellusten kehittämistä varten. PhoneGapin avulla voi luoda mobiilisovelluksia käyttämällä HTML-, JavaScript- ja CSS-verkkoteknologioita. PhoneGap tukee esimerkiksi Android- ja iOS-alustoja. (Wikipedia 2013k.)

Kehittäjät ja yrittäjät voivat käyttää PhoneGap-ohjelmistokehystä ilmaisten sekä maksullisten sovellusten kehittämiseen ilman lisenssimaksuja. PhoneGap-sivuilta löytyy dokumentaatio ja esimerkkejä. (Adobe Systems Inc 2013a.)

PhoneGap-sovelluksien kääntämiseen on kaksi vaihtoehtoa. Adobella on palvelu, joka mahdollistaa sovelluksen kääntämisen Adobe PhoneGap Build -pilvipalvelulla. Tässä vaihtoehdossa kehittäjän tarvitsee lähettää HTML5-, CSS- ja JavaScript-elementit pilvipalveluun. Pilvipalvelu käyttää automaattisesti uusinta SDK-versiota. Pilvipalvelu mahdollistaa olemassa olevan koodin kääntämisen eri alustoille menettämättä natiivin sovelluksen ominaisuuksia. Pilvipalvelu ei vaadi asentamista toimiakseen. PhoneGap-sovelluksia voi kääntää myös paikallisesti lataamalla haluttujen mobiilialustojen SDK:t. (Adobe Systems Inc 2013a;2013b;2013c.)

PhoneGap tukee suosittuja alustoja kuten Androidia ja iOSia. Sovelluksen kääntämisessä paikallisesti ja pilvessä on eroja eri alustoilla. Esimerkiksi BlackBerry 10-sovelluksen voi kääntää vain paikallisesti. Symbian-sovelluksen kääntäminen toimii vain käyttämällä pilvipalvelua. (Adobe Systems Inc 2013d.)

2.8 Yhteenveto teknologioista

Mobiilisovelluksen kehitykseen ei ole pakko käyttää ohjelmistokehyksiä vaan kaiken voi tehdä manuaalisesti. Työn tuottavuuden ja ylläpidon kannalta ohjelmistokehyksen käyttö on kuitenkin suositeltavaa, koska ohjelmistokehys tarjoaa suuren joukon valmiita ominaisuuksia ja siten helpottaa ja nopeuttaa sovelluksen kehittämistä ja helpottaa ylläpitoa. Taulukossa 1 on esitetty lyhyt yhteenveto teknologioista.

Taulukko 1. Yhteenveto teknologioista.

	Alustariippumaton	Ohjelmistokehys	Mobiiliyhteensopiva	Avoin lähdekoodi
Android	Ei	Kyllä	Kyllä	Kyllä
iOS	Ei	Kyllä	Kyllä	Ei
Vaadin/Touchkit	Kyllä	Kyllä	Kyllä	Kyllä
HTML5	Kyllä	Ei	Kyllä	Kyllä
DaVinci	Kyllä	Kyllä	Kyllä	Kyllä
jQuery mobile	Kyllä	Kyllä	Kyllä	Kyllä
PhoneGap	Kyllä	Kyllä	Kyllä	Kyllä

2.8.1 Yhdelle alustalle tehdyt kehitystyökalut

Virallisilla Androidin ja iOSin kehitystyökaluilla on yhteistä se, että niillä voi tehdä sovelluksia vain yhdelle alustalle. Esimerkiksi Android-SDK:lla ei ole mahdollista tehdä sovelluksia iOS-alustalle vaan ainoastaan Androidille (Google Inc 2013a; 2013b; 2013c). Sama koskee myös iOSin kehitystyökaluja (Apple 2013b; 2013c). Kehitystyökalut, jotka on tarkoitettu vain yhdelle käyttöjärjestelmälle, tarjoavat parhaan tuen natiiveille ominaisuuksille ja ovat monipuolisia toiminnallisuudeltaan. Kehitystyökalut osaavat automaattisesti luoda tarvittavan projektirakenteen ja asetukset. Kehitystyökaluissa voi olla myös valmiiksi asennettuna esimerkiksi pudota ja vedä käyttöliittymän rakennustyökalu. Yleinen vakiotyökalu on virtuaalikone, jolla simuloidaan eri älypuhelimia tai tablet-laitteita.

2.8.2 Alustariippumattomat ohjelmistokehykset

Käytettäessä mahdollisimman vähän tai ei ollenkaan natiiveja käyttöjärjestelmäkutsuja mobiilisovellus toimii pienillä muutoksilla useilla eri alustoilla. Hyvin toteutettu ja standardeja noudattava HTML5-mobiilisovellus näyttää melko samalta useilla eri laitteilla ja mobiiliselaimilla. HTML5 on vielä uusi teknologia ja sen toiminnallisuus ja ulkoasu vaihtelevat eri laitteiden välillä. (Firtman 2013.)

HTML5 mahdollistaa alustariippumattoman kehityksen, mutta natiivisovellusten laitetason ominaisuuksien, kuten esimerkiksi kameran, käyttäminen HTML5-sivunkuvauskielellä voi olla työläämpää kuin laitevalmistajien tarjoamilla työkaluilla (Wikipedia 2013f; 2013g). PhoneGapin merkittävä ominaisuus on Adobe PhoneGap Build-pilvipalvelu, jolla sovelluksen voi kääntää useille alustoille. Sen rajapinta osaa hyödyntää eri alustojen natiiveja ominaisuuksia kuten kameraa (Adobe Systems Inc. 2013a;2013b;2013c). Sekä PhoneGap että jQuery Mobile tukevat HTML:ää, CSS:ää ja JavaScriptiä. jQuery Mobile on kevyempi kuin PhoneGap-ohjelmistokehys siinä mielessä, että se on pelkkä JavaScript-kirjasto, joka ei sisällä muita ominaisuuksia. Sen käyttöönotto on siis erittäin yksinkertaista ja nopeaa.

2.8.3 Teknologian valinta mobiilisovellusta varten

Ohjelmistokehyksen ja teknologian valinnassa pohditaan vastauksia useisiin kysymyksiin. Tukeeko valittu teknologia alustariippumatonta kehitystä? Ovatko dokumentaatio ja tuki riittävä? Täytyykö teknologian tukea mobiililaitteen natiiveja ominaisuuksia? Onko esimerkiksi tarpeellista, että kamera toimii?

Mielenkiintoisia vaihtoehtoja oli useita, mutta valitsin HTML5- ja jQuery Mobile -teknologiat mobiilisovelluksen kehitystä varten, sillä verrattuna muihin teknologioihin niiden käyttöön ottaminen on yksinkertaisempaa ja niiden alustatuki on laajempi. Kaikista tutkituista teknologioista löytyi hyvä dokumentaatio ja paljon esimerkkejä. Valitut teknologiat eivät sisällä tukea natiiviominaisuuksille kuten PhoneGap, mutta halutessaan teknologioita voi käyttää yhdessä.

3 Sovelluksen suunnittelu ja toteutus

3.1 Käytettävyyden ja käyttöliittymän suunnittelu

Sovelluksen suunnittelun ensimmäinen vaihe oli käytettävyyden ja käyttöliittymän suunnittelu. Käytettävyyden suunnittelussa otettiin huomioon erilaiset mobiililaitteet esim. kosketusnäytöllä varustetut laitteet, ja kuinka niitä käytetään. Erikokoiset näytöt, ja se kuinka laitteen asento vaikuttaa käyttöliittymään oli tarkastelun kohteena. Käyttöliittymää suunniteltaessa tehtiin erilaisia käyttöliittymämalleja ja niistä valittiin parhaaksi todettu malli. Käyttöliittymämalli, (Kuva 1) jossa käyttöliittymäkomponentit ovat vaaka-asennossa, on yksi mahdollisista käyttöliittymämalleista, mutta sen ongelmana on tilan riittämättömyys. Pienellä näytöllä varustettu laite ei välttämättä osaa näyttää käyttöliittymää oikein. Laitetta kallistamalla käyttöliittymä kääntyy ja käyttöliittymä voi näyttää hyvältä esim. laitteen ollessa vakaa-asennossa, mutta pystyasennossa käyttöliittymä rikkoontuu, koska napit eivät mahdu koko näytölle.

Kuva 1. Painikkeet vaaka-asennossa.

Toteutukseen valittu käyttöliittymämalli (Kuva 2) sopii hyvin erilaisiin käyttötilanteisiin, se mukautuu hyvin laitteen asennosta huolimatta. Laitetta voidaan pitää joko vaaka- tai pysty -asennossa ja käyttöliittymä näkyy oikein. Tehtävälistan malli noudattaa samaa käyttöliittymämallia kuin painikkeet eli pystyasentoa.

Kuva 2. Painikkeet pystyasennossa.

Kohdejärjestelmänä on kosketusnäytöllä varustetut laitteet, mutta jQuery Mobile sovellukset toimivat myös työpöytäympäristössä. Käytettävyyttä suunniteltaessa keskityttiin kosketusnäytöllä varustettuihin mobiililaitteisiin. Painikkeiden koko on suuri, jotta käyttöliittymää olisi helppo käyttää. Käyttöliittymän ulkoasuksi on valittu yksi jQuery Mobilen oletusteemoista, jota on kuitenkin mahdollista muokata tarpeiden mukaan.

3.2 Toiminnallisuuden tekninen toteutus

Käyttöliittymän suunnittelun jälkeen oli vuorossa toiminnallisuuden luominen. Käyttöliittymän komponenteille luotiin yksilölliset tunnisteet, jotka auttavat tunnistamaan napit ja muut komponentin toisistaan (Kuva 3). Tunnisteiden luomisen jälkeen toiminnallisuuden luominen onnistuu luomalla komponenteille kuuntelijat, jotka voivat kuunnella erilaisia tapahtumia kuten painikkeen painamista. Esim. jos halutaan luoda jokin toiminnallisuus, kun käyttäjä painaa saapuneet tehtävät -nappia, tehdään painiketta varten kuuntelija. Kuuntelijaa luodessa täytyy tietää painikkeen tunniste, joka on tässä tapauksessa receivedTasks (Kuva 3). Seuraava vaihe on tehdä kuuntelija (Kuva 4), joka on asetettu kuuntelemaan haluttua painiketta seuraavalla koodinpätkällä ”$('#receivedTasks').click(...”. Haluttu toiminnallisuus luodaan tekemällä anonyymi funktio (Kuva 4), joka sijoitetaan click()-metodin sisälle. Muu sovelluksen toiminnallisuus kuten kirjautuminen on toteutettu vastaavalla tavalla, asettamalla tunnisteet komponenteille ja luomalla kuuntelija niitä varten.

```
<div data-role="controlgroup" data-type="vertical">
    <a href="#" data-role="button" id="recievedTasks">Saapuneet tehtävät</a>
    <a href="#" data-role="button" id="handledTasks">Käsitellyt tehtävät</a>
    <a href="#" data-role="button" id="createTask">Luo uusi tehtävä</a>
    <a href="#" data-role="button" id="logout">Kirjaudu ulos</a>
</div>
```

Kuva 3. Tehtävälistan koodi.

```
// Click handler for recieved tasks button (decision)
$('#recievedTasks').click(function() {
    clearTaskList();
    // Get tasks
    getAvailableTasks(localStorage.getItem(Strings.token), Strings.activity, Strings.subQueryTaskStateRdy);
});
```

Kuva 4. Saapuneet tehtävät napin toiminnallisuus.

3.3 Taustatietoa Intalion ja mobiilisovelluksen yhteistoiminnasta

Intalio-palvelin sisältää sähköisiä lomakkeita ja käyttöliittymän erilaisia toimintoja varten. Intalio-palvelimen käyttöliittymä sisältää laajan määrän ominaisuuksia, joita ei tuotantokäytössä tarvita. Mobiilisovellus piilottaa Intalion laajan toiminnallisuuden käyttämällä hyväkseen Intalio-palvelimen tarjoamaa rajapintaa. Sovellukseen on rajattu toiminnallisuus, joka on tuotantoympäristön kannalta tarpeellista ja muu toiminnallisuus on karsittu pois. Sovelluksella suoritettavat tehtävät ovat palvelimella olevien, Intalio designerilla luotujen, sähköisten lomakkeiden listaamista, avaamista ja lomakkeella esitetyn datan muokkaamista.

Käyttäjän näkökulmasta katsottuna näyttää, että käytetään vain yhtä sovellusta, mutta avattavat lomakkeet ladataan Intalio-palvelimelta. Sovellus käyttää hyväkseen Intalio-palvelimen tarjoamaa rajapintaa ja kommunikoi lähettämällä ja vastaanottamalla SOAP-viestejä (Simple Object Access Protocol).

Intalio-lomakkeet ovat Ajax-lomakkeita, jotka on suunniteltu toimimaan työpöytäselaimilla ja niiden avaaminen mobiiliselaimella ei välttämättä toimi. Intalio-lomakkeiden toimivuus ja rakenne riippuvat myös käytettävästä Intalio-palvelimen versiosta. Vastaan voi hyvinkin tulla tilanne, jossa mobiilisovellus toimii oikein, mutta tehtävä, eli yleensä lomakkeen avaaminen ei onnistu puuttuvien selainominaisuuksien takia.

3.4 Tietoturva

Sovellus käsittelee luottamuksellisia tietoja, joten työn suunnittelussa ja toteutuksessa tulee ottaa huomioon tiedon käsittelyn turvallisuus. Huomioon on

otettava, kuinka tietoa voidaan pyytää niin, että ulkopuolinen taho ei pääse siihen käsiksi.

Suojatun HTTPS-yhteyden kautta voidaan varmistua, että ulkopuolinen taho ei pääse helposti käsiksi mobiilisovelluksen lähettämään ja vastaanottamaan dataan. Mobiilisovelluksen lähettämän datan suojaaminen edistää tietoturvaa.

3.5 Sovelluksen asennus

Tällä hetkellä mobiilisovellus on asennettava samalle palvelimelle kuin Intalio-palvelin, josta halutaan pyytää tietoa. Tämä johtuu JavaScriptin rajoituksesta, joka liittyy tietoturvaan. Esimerkiksi POST-pyynnöissä on huomioitava saman alkuperän käytäntöä eli domain-nimen, portin ja protokollan on oltava samoja. Välityspalvelinta käytettäessä mobiilisovelluksen voisi asentaa eri palvelimelle. (Mozilla Foundation 2013a.)

3.6 Havaintoja valituista teknologioista

Mobiililaitteiden HTML5- ja JavaScript-tuki paranee koko ajan uusien laitteiden ja sovellusversioiden myötä, joten HTML5:n ja JavaScriptin käyttö mobiilisovelluksen kehityksessä on luonteva vaihtoehto. HTML5 ja JavaScript tekevät mobiilisovelluksesta alustariippumattoman, mikä mahdollistaa olemassa olevan koodin hyödyntämisen myös mobiilialustan ulkopuolella esimerkiksi PC-ympäristössä. Olemassa olevaa koodia ei tarvitse muuttaa paljoa tai lainkaan eri laitteita varten.

3.6.1 HTML5:n hyödyntäminen tiedon tallennuksessa

HTML5-sivunkuvauskielen uusimpiin ominaisuuksiin kuuluu localStorage eli yhteydettömän tilan tallennus. (Pilgrim 2013a). Käytin tätä ominaisuutta mobiilisovelluksessani tallentamaan tokenin, joka tallennetaan käyttäjän kirjautuessa mobiilisovellukseen. Tätä tokenia käytetään palvelukutsuissa Intalio-

palvelimelle. Token identifioi avatun palveluistunnon, joten ilman tokenia käskyä ei käsitellä. localStoragen idea on hyvin samanlainen kuin keksienkin, mutta siihen pystyy tallentamaan enemmän tietoa ja localStoragen sisältämä data on uniikki jokaiselle domainille (Mozilla Developer Network 2013a). Tämä tarkoittaa sitä, että ulkopuoliset sivut eivät pääse käsiksi toisen domainin tallentamaan localStorage-dataan.

3.6.2 jQuery Mobilen hyödyntäminen käyttöliittymän toteutuksessa

jQuery Mobile voi olla helppokäyttöisempi kuin pelkkä JavaScript yksinkertaisten toiminojensa ansioista. jQuery Mobilen kohdalla lause ”kirjoita vähän tee paljon” ei ole pelkkää mainontaa vaan jQuery Mobilen rajapinta piilottaa valtavan määrän toiminnallisuutta, jonka käyttöön ottaminen on yksinkertaista. Esimerkiksi tehtävälistan tyhjentäminen kaikista elementeistä, joka onnistuu yhdellä rivillä koodia ”$('#taskList').find('li').remove();”. Edellä oleva koodi etsii HTML-elementin, jonka id on ”taskList”. Tämä elementti sisältää joukon -elementtejä, joita voidaan kuvitella myös yksittäisinä riveinä eli yksi <li> vastaa yhtä riviä. Näiden -elementtien sisällä on listan elementtejä, esimerkiksi lomakkeen nimi ja pvm. find('li') etsii HTML-elementin sisältä, jonka id on taskList kaikki -elementit ja remove()-metodi poistaa jokaisen -elementin eli tehtävän tehtävälistasta. Ohjelmoijan ei tarvitse miettiä omaa logiikkaa olemassa olevien elementtien poistamista varten. Ilman ohjelmistokehystä koodin määrä tehtävälistan tyhjennystä varten olisi paljon suurempi. Klikkauskuuntelija tietylle elementille on myös helppo toteuttaa $('#recievedTasks').click(function() { Sijoita haluttu toiminnallisuus tähän});.

jQuery Mobile sisältää valmiita käyttöliittymäkomponentteja ja tyylejä komponenteille. Käyttöliittymäkomponentteihin kuuluu esimerkiksi listoja ja nappeja, joilla on jo valmis tyyli eli komponentin muoto, värit ja tekstin tyyli. jQuery Mobilen sivuilta löytyy ThemeRoller-työkalu, jolla on mahdollista luoda vapaasti omia tyylejä, jos valmiit tyylit eivät miellytä. Työkalua voi käyttää verkkosivun kautta ilman asentamista. (The jQuery Foundation 2013d.)

jQuery Mobilea käytettäessä korostui se, kuinka suuri osa ohjelmistokehyksellä voi olla projektissa. Ohjelmistokehykset helpottavat kehitystä ja ne sisältävät valmiiksi toteutettuna suuren joukon toiminnallisuuksia, joita tarvitaan usein projekteissa. Toistuvan koodin kirjoittaminen eri projekteissa vähentyy, koska voidaan käyttää ohjelmistokehyksen valmista toiminnallisuutta. Tämä voi nopeuttaa kehitystä huomattavasti. Esimerkiksi tiedon suodattamista varten on tehty valmis toiminnallisuus, joka voidaan ottaa käyttöön yhdellä rivillä koodia.

Dokumentaatiota ja ohjeita jQuery Mobilesta löytyy kattavasti kotisivuilta ja muista lähteistä. Mobiilikehityksessä jQuery Mobilella kannattaa hyödyntää myös jQuery API-dokumentaatiota, sillä se sisältää toimintoja, joita ei löydy jQuery Mobilen API-dokumentaatiosta.

3.7 GitHubin hyödyntäminen versionhallintana

Työssä käytettiin versionhallintajärjestelmänä GitHubia, jolla voi seurata commit-historiaa. Versionhallinta mahdollistaa tarvittaessa myös palaamisen vanhoihin versioihin. Versionhallinta on kriittinen komponentti ohjelmistokehityksessä ja sen tärkeys näkyy varsinkin suurissa projekteissa, joissa on useita kehittäjiä mukana. Tällaisissa projekteissa voi helposti muodostua ongelmia ja ohjelmakoodi voi rikkoontua. Joskus tällaisessa tilanteessa on helpompaa palata vanhempaan versioon koodista kuin yrittää selvittää mikä sovelluksessa on rikki. Github mahdollistaa usean kehittäjän yhtäaikaisen toiminnan ja koodi on aina saatavilla verkon kautta eikä sitä tarvitse siirtää muistitikulta toiselle. Githubin ominaisuuksiin kuuluu koodin historian selaus. Tämän takia on tärkeää, että julkaisujen kommentit olisivat mahdollisimman kuvaavia varsinkin isoissa projekteissa, joissa on paljon ihmisiä mukana. Opinnäytetyöprojektin kannalta tällä ei ollut suurta merkitystä, koska oli vain yksi kehittäjä.

3.8 Aptana Studio 3 JavaScript-ohjelmointiympäristönä

Aptana Studio 3 on IDE, jolla voi kirjoittaa HTML-, JavaScript-, PHP- ja monia muita kieliä. Käytin mobiilisovelluksen ohjelmoinnissa Aptana Studio 3 IDEä,

koska siinä oli tarvittavat ominaisuudet kuten Git-integraatio, koodin ennustus, jQuery-lisäosa ja koodin väritys. jQuery-lisäosan merkittävä ominaisuus oli jQuery-koodin ennustaminen.

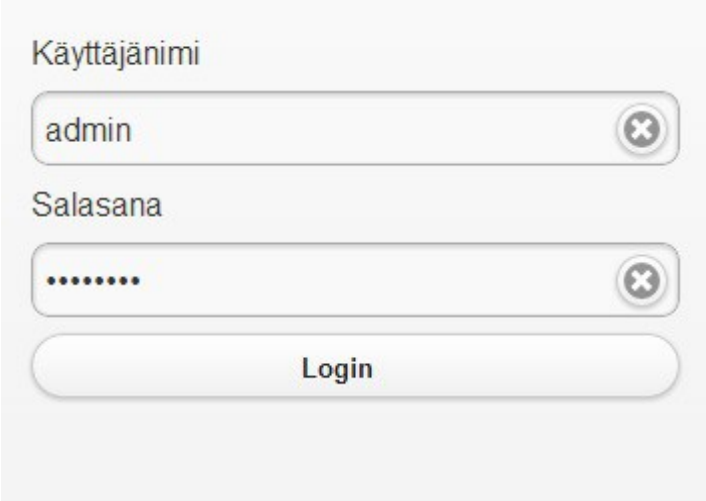
Mobiilisovelluksen projektikansion voi luoda verkkopalvelimen hakemistoon. Tämä mahdollistaa nopean sovelluksen testaamisen, koska aina projektia tallentaessa muutokset näkyvät välittömästi myös sovelluksessa. Testatessa on vain hyvä muistaa tyhjentää välimuisti ja muut väliaikaiset tiedot, koska joskus muutokset eivät välttämättä näy, jos selain hakee tietoa välimuistista.

3.9 Mobiilisovelluksen toiminnallisuuden esittely

Tässä osassa käydään läpi mobiilisovelluksen toiminnallisuutta ja käyttämistä vaihe vaiheelta. Kaikki tässä osassa olevat kuvat on otettu kehitysvaiheessa ja ne eivät välttämättä vastaa täysin valmista tuotetta. Pieniä muutoksia voi tulla toiminnallisuuteen tai tyyleihin.

3.9.1 Kirjautuminen

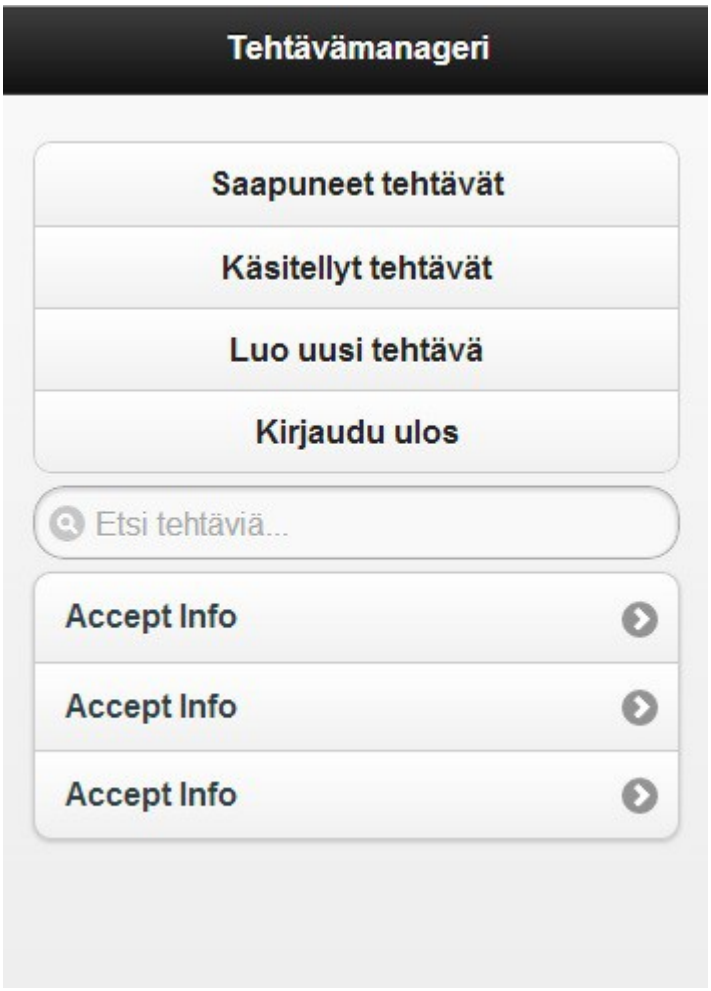
Kirjautumisnäkymä (Kuva 5) aukeaa, jos käyttäjä ei ole kirjautunut sisään tai jos käyttäjä kirjautuu ulos. Käyttäjä voi näpäyttää käyttäjänimi ja salasana -kenttää, joka avaa mobiililaitteella näppäimistön, johon käyttäjä voi kirjoittaa kirjautumistiedot.



Kuva 5. Kirjautumisnäkymä.

3.9.2 Tehtävien selaus

Mobiilisovelluksen oletusnäkymä on saapuneet tehtävät (Kuva 6), jossa käyttäjä näkee hänelle tarkoitetut tehtävät ja voi avata haluamansa tehtävän näpäyttämällä sitä kerran sormella. Nämä tehtävät haetaan Intalio-palvelimelta ja mobiilisovellus ei ota kantaa niiden nimeämiseen. Normaalisti lomakkeilla ja tehtävillä on yksilöllinen nimi, mutta kuvat on otettu kehitysympäristöstä.



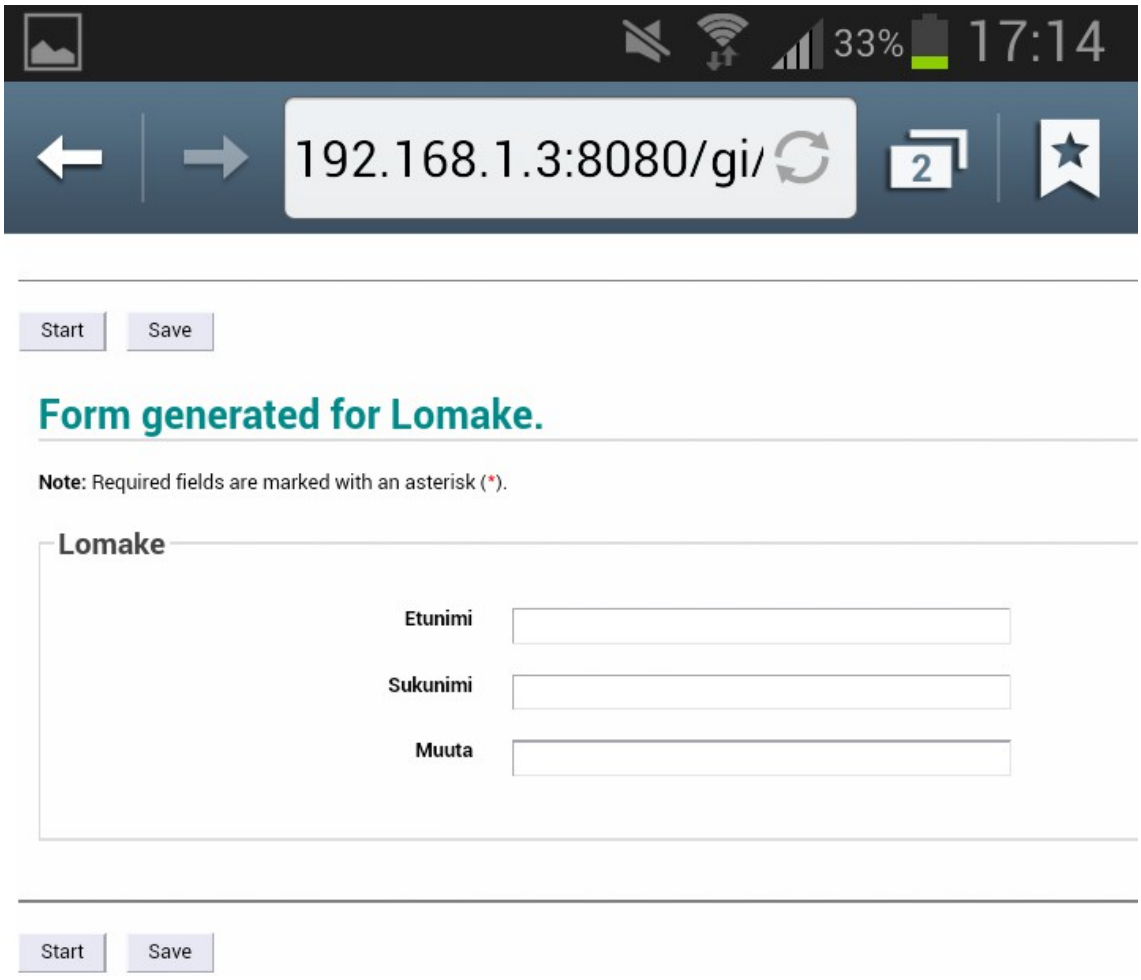
Kuva 6. Selausnäkymä.

Käsitellyt tehtävät -näkymässä käyttäjä näkee käsitellyt tehtävät. Esimerkiksi jos käyttäjä on lähettänyt hakemuksen ja virkailija on käsitellyt sen, tehtävä ilmestyy käsiteltyihin hakemuksiin. Käyttäjä voi halutessaan avata tehtävän näpäyttämällä sitä sormella, minkä jälkeen aukeaa lomake, jossa näkyy alkuperäiset tiedot ja virkailijan päätös.

Luo uusi tehtävä -näkymässä käyttäjä näkee mahdolliset tehtävät, jotka hän voi suorittaa. Näihin tehtäviin voi kuulua esimerkiksi hakemuksia. Käyttäjä voi luoda uuden tehtävän näpäyttämällä tehtävää sormella.

3.9.3 Lomakenäkymä

Lomakenäkymässä mobiilisovellus lataa ja muokkaa Intalio-palvelimelle luotuja lomakkeita. Testatuilla mobiililaitteilla lomake skaalautuu automaattisesti mobiili-näytölle. Luotu esimerkkilomake (Kuva 7), joka on ladattu Intalio-palvelimelta, avautuu koko näytön tilaan. Navigointi takaisin mobiilisovellukseen onnistuu käyttämällä selaimen nuolinäppäimiä tai laitteen takaisin-painiketta.



Kuva 7. Lomakenäkymä.

3.10 Testaus

Kehityksen aikana asensin kotikoneelleni Intalio-palvelimen, jonka Community-Edition -version voi ladata ilmaiseksi. Intalio-palvelimelle voi asentaa lomakkeita ja muita verkkosovelluksia. Käytin Intalio Designeria testilomakkeiden luontiin.

Tein testausta varten yksinkertaisen lomakkeen ja asensin sen palvelimelle. Tein lomakkeesta kolme erilaista versiota, joilla on erilaiset näkyvyydet. Näkyvyyksillä tarkoitetaan sitä, mitä rooleja käyttäjällä on oltava, jotta hän näkee lomakkeen. Kuvitellaan, että lomake on päivähoitohakemus. Ensimmäinen versio lomakkeesta on tarkoitettu päivähoitoa hakevalle ja kaikki kuntalaiset näkevät sen. Kuntalaisen täytettyä hakemuksen lomakkeen tiedot menevät eteenpäin virkailijalle. Päivähoitohakemuksen toinen versio on ”päätösvaihe”, jossa lomake näkyy vain virkailijalle eli päivähoitohakemuksen käsittelijälle. Käsittelijän lomakkeessa voi olla valintalaatikot hyväksy/hylkää, jotka näkyvät vain päätösvaiheessa. Virkailija käsittelee lomakkeen ja se merkataan käsitellyksi. Kolmas vaihe on, kun lomake on käsitelty ja virkailija voi vielä käydä katsomassa yhteenvedon tästä lomakkeesta. Myös kuntalainen näkee käsitellyn lomakkeen, mutta ei voi muokata sitä.

Tehty testausympäristö vastaa tuotantokäytössä olevia järjestelmiä. Päätarkoitus oli luoda yksinkertainen lomake, jonka avulla voi luoda eri vaiheessa olevia lomakkeita, joita sitten voi selata ja avata mobiilisovelluksessa.

Kävin testaamassa mobiilisovellusta myös työpaikalla kehitteillä olevassa tietojärjestelmässä. Mobiilisovellus toimi oikein myös kehitysympäristössä. Testauksen aikana havaittiin käytettävyysongelma, joka voi hankaloittaa sovelluksen käyttämistä. Asiakasympäristössä lomakkeen lähetyksen jälkeen aukeaa kiitossivu ja sieltä pääsee pois vain käyttämällä laitteen edellinen-painiketta. Edellinen painiketta painettaessa aukeaa lomake ja jos halutaan palata selausnäkymään täytyy edellinen painiketta painaa toistamiseen. Mobiilisovellusta ei ole vielä testattu oikeassa tuotantoympäristössä.

HTC Desire (HTC Bravo), Android-versio 2.2.2 avasi mobiilisovelluksen ilman ongelmia, mutta lomakkeen avaaminen päättyi virheilmoitukseen. Tähän ongelmaan harkittiin ratkaisua, jossa olisi haettu tietoa Intalio-lomakkeen rakenteesta Intalio-palvelimelta ja parsittu rakennedata ohjelmallisesti. Tämän jälkeen olisi muodostettu mobiililaitteeseen sopiva versio lomakkeesta. Tähän tarkoitukseen tarvittavaa tietoa ei ollut helposti saatavilla. Intalio-API ei myöskään tue tämän-

tyylistä toimintaa ja Intalio-tietokannan rakenne on erittäin monimutkainen ja laaja kokonaisuus perehtyä. Tämän työn yhteydessä ei ollut mahdollista toteuttaa tämänlaista ratkaisua.

Testauksessa on käytetty kahta erilaista Android-älypuhelinta. Testeissä on myös käytetty PC-ympäristöä selaimilla Chrome, Firefox ja Internet Explorer. Chrome ja Firefox -selaimilla mobiilisovellus toimii moitteettomasti, mutta Internet Explorer-selaimella sovellus ei toimi lainkaan.

3.11 Kohdatut ongelmat

Mobiilisovelluksessa ilmeni kaksi suurta virhettä. Sovellusta ensimmäistä kertaa käytettäessä ilmeni outo ongelma. Mobiilisovelluksen käynnistyessä tarkistetaan, onko käyttäjä kirjautunut ja jos hän ei ole, ladataan kirjautumissivu ja muussa tapauksessa ladataan pääsivu. Ensimmäisellä käyttökerralla latautui pääsivu, joka ehti näkyä vain alle sekunnin. Tämän jälkeen latautui kirjautumissivu niin kuin oli tarkoituskin, mutta jostain syystä pääsivu latautui uudelleen ja kirjautumisikkuna katosi näkyvistä heti, kun se oli ladattu. Ongelma ilmeni WebKit-pohjaisissa selaimissa.

Ongelman syynä oli ”$(document).ready()”, jonka pitäisi suorittaa haluttu koodi vasta sivun latauduttua. Vaikuttaa siltä, että osa selaimista käsittelee tämän käskyn eri tavalla ja tämä aiheutti ongelmia. Löysin GitHubin keskustelufoorumeilta jonkun, jolla oli sama ongelma kuin minulla. Brendan Doms nimimerkillä bdoms oli vastannut tähän viestiketjuun ja hänen mukaansa ongelma esiintyy kaikilla WebKit-pohjaisilla selaimilla. Hän huomasi ongelman korjaantuvan käyttämällä koodinpätkää ”$(window).load()” sivunvaihdon yhteydessä. Hän oli myös selvittänyt, että ongelma vaikuttaisi johtuvan siitä, että ensimmäistä sivua ei lisätä kokonaisuudessaan selaimen historiaan ennen pageChange()-funktion kutsumista. Hänen teoriansa ongelmasta on, että ei-WebKit-selaimet lisäävät sivun välittömästi selainhistoriaan, mutta WebKit-selaimet odottavat, kunnes kaikki sivun resurssit on ladattu. pageChange()-funktion kutsuminen pageinit-tapahtumassa aiheuttaa poikkeavaa käyttäytymistä. Brendan Domsin ehdottama

korjaus toimii samalla tavalla kaikilla testatuilla selaimilla ja ongelma hävisi kokonaan.

Toinen samantyyppinen ongelma ilmeni myös vain tilanteissa, joissa sovellusta käytetään ensimmäistä kertaa tai joissa Intalio-palvelin käynnistetään uudelleen ja mobiilisovellusta yritetään käyttää tämän jälkeen. Ongelmana oli, että kirjautumisen jälkeen pääsivulla ei näkynyt yhtään tehtävää vaikka niitä pitäisi olla. Huomasin, että jostain syystä mobiilisovellus ei aina tee pyyntöä Intalio-palvelimella tai token, jota käytetään pyynnöissä, on tyhjä. Ongelmana oli, että oletin sovelluksen aina ensin tekevän pyynnön palvelimelle, minkä jälkeen se tallentaa tokenin. Näiden toimenpiteiden jälkeen oletin mobiilisovelluksen vaihtavan sivun ja hakevan tehtävät. Olin tietämättäni luonut tokenin tallentamisesta callback- metodin, joka suoritetaan vain tietynlaisessa tapahtumassa. Tämä tarkoittaa, että muu koodi jatkaa suorittamista eikä jää odottamaan, että tokenin tallentavaa metodia kutsustaan. Tästä syystä token oli tyhjä kutsuttaessa metodia, joka hakee tehtävät palvelimelta.

Pohdin, kuinka saan varmistettua, että token on tallennettu ennen kuin yritetään hakea tehtäviä palvelimelta. Löysin artikkelin, joka neuvoi callback-metodien käyttöä. Callback-metodeilla on helppoa varmistaa, että vaihe 1 on suoritettu ennen kuin kutsutaan vaiheen 2 metodia. Tein mobiilisovellukseen muutoksen, jossa mobiilisovellus odottaa, että token on tallennettu ja vasta sitten pyytää tehtävät palvelimelta. (Lazaris 2012.)

4 Tulokset

Tässä luvussa käydään yksitellen läpi jokainen opinnäytetyön vaatimus ja kerrotaan toteutuiko vaatimus. Opinnäytetyön tuloksena syntyi alustariippumaton webpohjainen mobiilisovellus. Tarkoituksena oli löytää teknologia tai teknologiat, jotka mahdollistavat alustariippumattoman mobiilisovelluksen luomisen. Löysin HTML5-, jQuery Mobile- ja JavaScript-teknologiat, jotka täyttävät vaatimuksen. HTML5-sivunkuvauskieli on mobiiliyhteensopiva ja toimii useimmilla mobiililait-

teilla. Tästä syystä se sopii hyvin alustariippumattomaan kehitykseen. jQuery Mobile on puhdasta JavaScript-koodia ja se on tuettu mobiili- sekä PC-alustoilla. JavaScript toimii moderneissa selaimissa, mikä mahdollistaa alustariippumattoman kehityksen.

Kehityksen nopeuttamiseksi ohjelmistokehyksen käyttäminen oli vaatimuksena. Ohjelmistokehykseksi valittiin jQuery Mobile, ja ilman ohjelmistokehystä työn tekeminen olisi ollut paljon haastavampaa. jQuery Mobile huolehtii ulkoasun säilymisestä samanlaisena eri laitteilla. Ilman tämänlaista toiminnallisuutta sovelluksen suunnittelussa tulisi ottaa huomioon, kuinka varmistetaan ulkoasun säilyminen eri alustoilla. Eri selaimet, selainversiot, mobiilialustat ja käyttöjärjestelmäversiot tulisi ottaa huomioon ja suunnitella jonkinlainen logiikka eri versioiden tunnistamista varten. Työn määrä kasvaa sitä enemmän, mitä useampia muuttujia otetaan huomioon.

Toivottavana tavoitteena oli löytää teknologiat, jotka ovat avointa lähdekoodia. jQuery Mobile, jonka suurin tehtävä tässä työssä on käsitellä käyttöliittymän muokkaamista, on avoimen lähdekoodin projekti. HTML5- ja JavaScript-teknologiat ovat avoimia ja vapaasti käytettäviä. Valituilla teknologioilla on yhteistä avoimuus ja niiden vapaa käyttö henkilökohtaiseen tai kaupalliseen käyttöön.

Uuden asian opettelussa dokumentaatio on tärkeä asia ja tästä syystä tavoitteena oli löytää teknologiat, joista löytyy riittävästi dokumentaatiota. HTML5-, jQuery Mobile- ja JavaScript-teknologioista löytyy riittävästi dokumentaatiota eri lähteistä. Kehityksen aikana dokumentaation puutteesta ei tullut ongelmaa. Kehityksen aikana lähteinä käytettiin virallisia ja epävirallisia lähteitä. Asian ymmärtämistä helpottaa, jos siitä etsii tietoa useista lähteistä.

5 Pohdinta

Tämän opinnäytetyön kohdalla mobiilisovelluskehitys ei eroa paljon työpöytäsovellus- tai verkko-ohjelmoinnista. Työkalut ja työmenetelmät ovat samanlaisia tai

muistuttavat toisiaan. Suurin haaste kehityksen aikana oli alkuun pääseminen, johon kuului mobiilikehitykseen kuuluvien teknologioiden ja työkalujen selvittäminen. Mobiilikehitys ei ollut minulle entuudestaan kovin tuttua, joten aikaa meni paljon tiedon etsimiseen ja opiskeluun. Tiedonetsinnän ja teknologian valinnan jälkeen mobiilisovelluksen kehitys alkoi ja toimivan prototyypin tekeminen onnistui yllättävän nopeasti. Pohdin myös vaihtoehtoa, jossa olisin kehittänyt saman mobiilisovelluksen uudelleen käyttämällä eri valmistajien työkaluja, mutta hylkäsin ajatuksen. Erilaisten kehitysympäristöjen tutustumiseen ja asentamiseen menisi helposti liikaa aikaa.

Toimivuutta Windows-alustalle ei testattu eikä se ollut tavoitteena, mutta jQuery Mobilen laitetuki sisältää tuen myös Windows-puhelimille. Yleensä tablet-laitteissa on käytössä sama käyttöjärjestelmä kuin älypuhelimissa, joten teoriassa mobiilisovelluksen pitäisi toimia useissa tablet-laitteissa.

Pohdittavaksi jää, onko Intalio-lomakkeet mahdollista kehittää mobiiliystävällisiksi, mobiilisovelluksen laitetuen kasvattamista varten. Toisenlainen lähestymistapa olisi luoda Intalioon uusi mobiilirajapinta, jota mobiililaitteet voisivat käyttää. Tällainen työ vaatisi kuitenkin erinomaista osaamista Intaliosta ja sen toiminnoista, tietokantarakenteesta ja Intalio-palvelimesta. Täytyisi suunnitella käyttöliittymä, ja kuinka se piirretään näytölle alusta asti. Olemassa olevaa Intalio-koodia voisi mahdollisesti muokata mobiiliystävälliseksi, mutta se täytyisi eristää alkuperäisesti koodista tai muokata toimivaksi mobiili- ja työpöytäalustoilla.

Mobiilisovellus ei sisällä tietoturvallista tapaa kommunikoida Intalio-palvelimen kanssa. Tämänhetkisessä versiossa on mahdollista seurata mobiilisovelluksen lähettämää ja vastaanottamaa tietoa esimerkiksi avoimessa WLAN-verkossa. Käyttäjätunnus ja salasana liikkuu selkokielisessä muodossa ilman minkäänlaista salausta. Kirjautuminen tapahtuu lähettämällä käyttäjätunnus ja salasana Intalio-rajapinnalle, joka havaintojen mukaan hyväksyy tiedon vain salaamattomassa muodossa. Ratkaisuna tähän on käyttää HTTPS-suojausta palvelimella, johon mobiilisovellus asennetaan.

Käytettävyyttä voitaisiin parantaa ratkaisulla, jossa lomakkeen lähetyksen jälkeen selausnäkymään palaaminen olisi tehty helpommaksi. Mobiilisovellukseen voisi esimerkiksi lisätä palaa-painikkeen tai muokata edellinen-painikkeen toimintaa niin, että käyttäjä palaisi suoraan selausnäkymään edellinen-painikkeet painamisen yhteydessä. Joidenkin lomakkeiden toiminnallisuuteen on lisätty kiitossivu, joka latautuu heti lomakkeen lähetyksen jälkeen ja palaa-painiketta on painettava kahdesti, kun halutaan palata mobiilisovelluksen päänäkymään. Mobiilisovellus ei sisällä sivutus-ominaisuutta eli jos saapuneita tehtäviä on satoja tai tuhansia, ne kaikki ladataan kerralla, mikä voi hidastaa sovelluksen toimivuutta huomattavasti tai jopa hidastaa mobiililaitetta. Sivutus-ominaisuuden lisääminen olisi hyvä tulevaisuuden kehitysidea. Mobiilisovelluksessa on hakukenttä, johon voi kirjoittaa haettavan lomakkeen tai tehtävän nimen ja tehtävät suodatetaan sen mukaan. Mobiilisovellus ei sisällä muunlaista suodatusta kuin suodatuksen päivämäärän mukaan. Erilaisten suodatusominaisuuksien lisääminen on kehitysidea, joka voi tulla tarpeeseen. Jos yhteys on hidas tai jos palvelin josta mobiilisovellus pyytää tietoa on kovan rasituksen alla, tiedonhaku voi hidastua huomattavasti ja näyttää siltä ettei mobiilisovellus tee mitään. Jonkinlainen indikaattori, joka ilmaisee, kun mobiilisovellus lataa tietoa tai tekee operaatioita, on kehitysidea tulevaisuutta varten. Nyt mobiilisovellusta käytetään laitteen selaimella ja jos siitä haluttaisiin tehdä enemmän mobiilisovellusta muistuttava, jokaiselle alustalle voitaisiin tehdä mobiilisovellus, joka sisältää webnäkymän, johon opinnäytetyön aikana kehitetty mobiilisovellus ladattaisiin. Näin tekemällä näyttäisi siltä, että käyttäjä käyttää mobiilisovellusta eikä verkkosovellusta.

Lähteet

Adobe Systems Inc. 2013a. FAQs.
http://phonegap.com/about/faq/. 11.6.2013.

Adobe Systems Inc. 2013b. PhoneGap Documentation.
http://docs.phonegap.com/en/2.7.0/index.html. 11.6.2013.

Adobe Systems Inc. 2013c. Adobe® PhoneGap™ Build.
https://build.phonegap.com/. 11.6.2013.

Adobe Systems Inc. 2013d. Adobe® PhoneGap™ Build.
http://docs.phonegap.com/en/edge/guide_support_index.md.html#Platform%2Support. 12.6.2013. 11.6.2013.

Andrew G. How can I develop for iPhone using a Windows development machine?. 11.6.2013.
http://stackoverflow.com/questions/22358/how-can-i-develop-for-iphone-using-a-windows-development-machine. 15.6.2013.

Apple Inc. 2013a. Which Developer Program is for you?
https://developer.apple.com/programs/which-program/. 15.6.2013.

Apple Inc. 2013b. Developer Tools Features.
https://developer.apple.com/technologies/tools/features.html. 15.6.2013.

Apple Inc. 2013c. Setup.
https://developer.apple.com/library/ios/referencelibrary/GettingStarted/RoadMapiOS/index.html. 15.6.2013.

Apple Inc. 2013d. App Development Process.
https://developer.apple.com/library/ios/referencelibrary/GettingStarted/RoadMapiOS/AppDevelopmentProcess.html#//apple_ref/doc/uid/TP40011343-CH4-SW1. 15.6.2013.

Ben C. & Bill B. 2010.
Google I/O 2010 - A JIT Compiler for Android's Dalvik VM.
https://www.youtube.com/watch?v=Ls0tM-c4Vfo. 10.6.2013.

Brendan D. 2011. Redirecting on first load, using changePage, does not work in WebKit browsers.
https://github.com/jquery/jquery-mobile/issues/3190. 1.7.2013.

Google Inc. 2013a. Android.
http://www.android.com/about/. 10.6.2013.

Google Inc. 2013b. Developer Tools.
http://developer.android.com/tools/index.html. 10.6.2013.

Google Inc. 2013c. Get the Android SDK.
http://developer.android.com/sdk/index.html. 10.6.2013.

Google Inc. 2013d. Creating an Android Project.
http://developer.android.com/training/basics/firstapp/creating-project.html. 12.6.2013.

Google Inc. 2013e. Starting an Activity.
http://developer.android.com/training/basics/activity-lifecycle/starting.html. 12.6.2013.
Google Inc. 2013f. Developer Console.
https://play.google.com/apps/publish/signup/. 10.6.2013.
Google Inc. 2013g. Transaction Fees.
https://support.google.com/googleplay/android-developer/answer/112622?hl=en&ref_topic=15867. 10.6.2013.
Louis Lazaris 2012. Callback Functions in JavaScript.
www.impressivewebs.com/callback-functions-JavaScript/. 10.6.2013.
Mark P. 2013a. The Past, Present & Future of Local Storage for Web Applications.
http://diveintohtml5.info/storage.html. 29.5.2013.
Maximiliano Firtman. 2013. MOBILE HTML.
http://mobilehtml5.org/. 12.6.2013.
Mozilla Developer Network. 2013a. DOM Storage guide.
https://developer.mozilla.org/en-US/docs/Web/Guide/API/DOM/Storage?redirectlocale=en-US&redirectslug=Web%2FGuide%2FDOM%2FStorage. 29.5.2013.
Mozilla Foundation. 2013a. Same-origin policy.
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Same_origin_policy_for_Javascript. 18.6.2013.
The jQuery Foundation. 2013a. Mobile Graded Browser Support.
http://jquerymobile.com/gbs/. 14.6.2013.
The jQuery Foundation. 2013b. Demo center for 1.3.1.
http://view.jquerymobile.com/1.3.1/dist/demos/. 14.6.2013.
The jQuery Foundation. 2013c.
jQuery Mobile API Documentation.
http://api.jquerymobile.com/. 15.6.2013.
The jQuery Foundation. 2013d. ThemeRoller.
http://jquerymobile.com/themeroller/. 16.6.2013.
Vaadin Ltd. 2013a. Vaadin TouchKit - Create iPhone applications using Vaadin.
https://vaadin.com/wiki/-/wiki/Main/Vaadin%20TouchKit%20-%20Create%20iPhone%20applications%20using%20Vaadin. 17.6.2013.
Vaadin Ltd. 2013b. Vaadin TouchKit.
https://vaadin.com/add-ons/touchkit. 17.6.2013.
Wikipedia. 2013a. Android.
http://fi.wikipedia.org/wiki/Android. 10.6.2013.
Wikipedia. 2013b. Dalvik-virtuaalikone.
http://fi.wikipedia.org/wiki/Dalvik-virtuaalikone. 10.6.2013.
Wikipedia. 2013c. Dalvik software.
http://en.wikipedia.org/wiki/Dalvik_%28software%29 10.6.2013.
Wikipedia. 2013d. IOS.
http://fi.wikipedia.org/wiki/IOS. 15.6.2013.
Wikipedia. 2013e. IOS.
http://en.wikipedia.org/wiki/IOS. 15.6.2013.
Wikipedia. 2013f. HTML5.
http://en.wikipedia.org/wiki/HTML5. 14.6.2013.
Wikipedia. 2013g. HTML5.
http://fi.wikipedia.org/wiki/HTML5. 14.6.2013.

Wikipedia. 2013h. DaVinci software.
http://en.wikipedia.org/wiki/DaVinci_%28software%29. 14.6.2013.
Wikipedia. 2013i. jQuery Mobile.
http://en.wikipedia.org/wiki/Jquery_Mobile. 14.6.2013.
Wikipedia. 2013j. jQuery.
http://en.wikipedia.org/wiki/JQuery. 14.6.2013.
Wikipedia. 2013k. PhoneGap.
http://en.wikipedia.org/wiki/PhoneGap. 11.6.2013.