HAAGA-HELIA
ammattikorkeakoulu

Matchart viinivalitsijan jatkokehitys

Sebastian Nikkonen

Opinnäytetyö
Tietojenkäsittelyn koulutusohjelma
2014

HAAGA-HELIA
ammattikorkeakoulu

Tiivistelmä

17.11.2014

Tietojenkäsittelyn koulutusohjelma

Tekijä tai tekijät Sebastian Nikkonen	**Ryhmätunnus tai aloitusvuosi** 2011
Raportin nimi Matchart viinivalitsijan jatkokehitys	**Sivu- ja liitesivumäärä** 26 + 3
Opettajat tai ohjaajat Teemu Patala	

Tämä opinnäytetyö käsittelee Matchart viinivalitsijan jatkokehitystä. Ainutlaatuinen konsepti yhdistää oikeanlaiset viinit valitulle ruoalle hyödyttäen asiakasta. Työn päätavoite oli kehittää sovellusta eteenpäin, lisätä uusia ominaisuuksia ja tuottaa lisähyötyä Matchart Oy:lle. Sovelluksen ensimmäisen version valmistuttua vuonna 2012 on ilmennyt tarvetta jatkokehitykselle. Lopputyön tarkoituksena on välittää kokonaisvaltainen kuva sovelluksen kehittämisestä alkaen siitä kun ensimmäistä kertaa istuttiin suunnittelupöytään keskustelemaan eri vaihtoehdoista projektiryhmän kanssa.

Opinnäytetyö sisältää teoriaosuuden missä käsitellään teknologiaa ja mahdollisuuksia mitä PhoneGap sovelluskehys tarjoaa ja sitä mitä responsiivisella suunnittelulla tarkoitetaan. Tutkimustyö pohjautuu määrällisiin ja laadullisiin menetelmiin. Tietoperusta pohjautuu useaan eri lähteeseen, tarkistettuihin tietoihin ja kirjoittajan omaan tuntemukseen.

Sovelluksen eri toiminnallisuudet määritellään, koska se nähdään tehokkaana tapana parantaa lopullista tuotetta. Kehitys osiossa kuvaillaan kehityksen eri vaiheita, kuten olemassa olevan ohjelmakoodin tehostamista ja uuden kehittämistä. Testaus osiossa käsitellään tiedon siirtämistä sovelluksen eri osien välillä. Testauksen tarkoituksena oli havaita ja korjata mahdolliset virheet, puutteet ja häiriöt. Opinnäytetyön lopputuloksena annetaan kattava yleiskuva siitä mitä ohjelmistokehitys on kaikissa sen eri vaiheissa.

Lopullinen työ paransi nykyistä ohjelmakoodia. lisäsi uusia ominaisuuksia ja tuottaa lisäarvoa Matchart Oy:lle. Seuraava jatkotoimenpide on tehdä integraatio Tastingbookin tietokantaan. Matchart sisältää jo valmiit ominaisuudet tätä tarkoitusta varten.

Asiasanat
Älypuhelinsovellus, määrittely, kehitys, testaus, tuote, integraatio

HAAGA-HELIA
University of Applied Sciences

Abstract

17.11.2014

Degree programme in information technology

Authors Sebastian Nikkonen	**Group or year of entry** 2011
The title of thesis Further development of the Matchart wine selector	**Number of report pages and attachment pages** 26 + 3
Advisor(s) Teemu Patala	

This thesis deals with the further development of the Matchart wine selector and its smartphone application. The unique concept matches right wines and the food to customers' advantage. The main goal was to develop an application forward, add new features and provide added value to Matchart Oy. The first version of the application completed in 2012 has shown the need for further development. Purpose of this Bachelor's thesis is to provide a comprehensive picture of the development of the application from the first day when the project team sat down to discuss different options.

The thesis includes a theoretical part which deals with the technology and possibilities that PhoneGap framework offers and what responsive design means. The research is based on quantitative and qualitative methods. The theoretical foundation is based on several different sources, verified data and the author's own knowledge.

Definition of application functionalities were made, because it's seen as an effective way to improve the final product. Development section describes the different stages of development, such as increasing the efficiency of already existing code and creating the new. The test section deals with the data transfer between different parts of the application. The study has resulted in a comprehensive overview of what software development is at its different stages.

The final work improved the already existing program code, added new features and produces added value to Matchart Oy. The following procedure is to make the integration to Tastingbook's database. Matchart already contains ready-made features for this purpose.

Key words
Smartphone application, definition, development, testing, product, integration

Sisällys

1 Johdanto

Matchart viinivalitsijan käyttäjän täytyy tietää vain ruoka mitä hän haluaa syödä. Sovellus suosittelee parhaiten sopivat puna- ja valkoviinit valitulle ruoalle. Käyttäjän ei tarvitse tietää viineistä mitään, koska pohjoismaiden parhaiden viiniasiantuntijoiden tuntemus on aina mukana älypuhelinsovelluksen muodossa. Työssä tutustutaan kehityksen eri vaiheisiin, kuten määrittelyyn, ohjelmointiin ja testaukseen.

1.1 Projektin tausta

Alkuperäinen Matchart aloitettiin maaliskuussa 2012 Haaga-Helian rahoittamana projektina. Projektiin palkattiin kaksi tietoejenkäsittelyn opiskelijaa ja yksi ravintola-alan opiskelija Haaga-Helia ammattikorkeakoulusta. Opiskelijat kehittivät alkuperäisen sovelluksen kesällä 2012. Yksi opiskelijoista teki vuoden 2013 aikana lopputyön, jossa hän käänsi ohjelman älypuhelinsovellukseksi Android, iOS ja Windows Phone käyttöjärjestelmille PhoneGap viitekehystä hyödyntäen. Syksyn 2013 aikana allekirjoittanut on tehnyt ohjelman eri versioihin uusia ominaisuuksia, kuten haun Alkon sivuille ja lukuisia parannuksia hallintatyökaluihin.

Projekti käynnistettiin, koska Matchart viinivalitsijan jatkokehityksessä nähdään potentiaalia ja sitä halutaan kehittää lisää. Sovellukseen pystytään tekemään ominaisuuksia, joilla ohjelma saadaan yhä useampien ihmisten tietoisuuteen. Parhaassa tapauksessa sovellusta pystytään hyödyntämään kaupallisesti.

Vuonna 2013 perustettiin Matchart Oy, minkä alle projekti sijoitettiin. Eri sidosryhmien kanssa on käyty neuvotteluja erilaisista yhteistyömalleista ja ohjelman kehittämisestä eteenpäin. On päädytty malliin, missä olemassa olevaa sovellusta monistetaan eri versioiksi yhteistyökumppaneiden käyttöön. Taloustaidon kanssa on tehty yhteistyötä ja sovelluksesta on oma versio heidän käytössään. Tastingbookin versiossa on tarkoitus hyödyntää heiltä löytyvää tietokantaa, jolloin asiakkaalle pystytään tarjoamaan tarkempia kuvauksia valituista viineistä. Sovellukseen halutaan lisätä mahdollisuus jakaa viinisuosituksia sosiaalisessa mediassa, mitä se ei ole aikaisemmin mahdollistanut täysin.

1.2 Projektin tavoite ja lopputulokset

Projektin tavoitteena on tehdä olemassa olevaan Matchart viinivalitsijaan uusia ominaisuuksia, jonka avulla sovellus voidaan kaupallistaa. Tavoitteet koskevat internetissä toimivaa Matchart viinivalitsijaa sekä älypuhelinsovellusta, mistä on omat versionsa Android, iOS ja Windows Phone käyttöjärjestelmille. Aikaisemmassa versiossa Matchart viinivalitsija antoi suositukset viinin tyypeille eikä omaa tuotekorttia ollut olemassa. Sovellukseen halutaan oma sivu tuloksille, mikä hakee tiedot suoraan Tastingbookin tietokannasta viinityypin perusteella, mikä saadaan Matchartin tietokannasta.

Matchart viinivalitsijan polku halutaan jakaa facebookissa. Sovelluksen suosituksiin toteutetaan ominaisuudet tykkäämiselle ja jakamiselle. Sosiaalisessa mediassa jaetaan polku, jolla haluttuun tulokseen on päästy ja viinit, joita ohjelma suosittelee käyttäjälle. Tämä mahdollistaa uusien käyttäjien saamisen sosiaalisesta mediasta.

1.3 Projektin eri vaiheet

Taulukko 1. Pääkohdat joihin projekti jakaantuu.

Vaihe	Kuvaus
Teoriatausta	Teoriataustassa tutustutaan käytettäviin teknologioihin ja menetelmiin. Löytyy kohdasta 2. Teoriatausta.
Vaatimus-määrittely	Vaatimusmäärittelyssä kuvataan sovellukseen tehtäviä ominaisuuksia ja niiden toimintaa eri tilanteissa. Löytyy kohdasta 3. Vaatimusmäärittely. Lisäksi on liite numero 1. Käyttötapauskaaviot.
Suunnittelu ja kehitys	Suunnittelu tapahtuu yksilötyöskentelynä ja projektitiimin kanssa. Sovelluksen kehitys jakaantuu kahteen eri vaiheeseen. Rakennetaan sovellukseen mahdollisuus siirtyä tuotekorttiin, mikä on toinen vaihe ja sitä varten tehdään oma tietokanta ja sivu. Löytyy kohdasta 4. Suunnittelu ja kehitys.
Testaus	Testauksella varmistetaan että ohjelmoitu sovellus toimii halutulla tavalla ja ei tuota ei toivottuja virheitä, puutteita tai häiriöitä. Löytyy kohdasta 5. Testaus.

1.4 Projektin rajaus

Projektin tarkoitus ei ole rakentaa uutta sovellustaa, vaan laajentaa jo olemassa olevan ominaisuuksia ja koodipohjaa. Some jakamisella tarkoitetaan facebook liitännäisten hyödyntämistä sovelluksessa.

1.5 Projektin organisaatio

Projektin organisaatioon kuuluu seuraavat henkilöt:

- Projektipäällikön ja sihteerin tehtävissä toimii Sebastian Nikkonen
- Opinnäytetyön ohjaajana toimii Teemu Patala
- Toimeksiantajan edustajina toimii Jussi Koivusaari ja Risto Karmavuo

1.6 Käsitteet

Käsitteissä käydään läpi eri teknologioihin liittyviä termejä ja avataan niiden tarkoitusta lukijalle kertomalla lyhyesti ja yksinkertaisesti kuvaukseen liittyvistä tekijöistä.

AJAX: Asynchronous JavaScript And XML on tekniikka, millä web-sovelluksista tehdään vuotovaikuittesempia. AJAX kutsuja käytetään ottamaan yhteyttä sivuston palvelimelle ilman että sivua täytyisi ladata uudestaan.

Back-end: Tarkoittaa osaa joka ei ole näkyvillä ohjelmassa, mutta johon käyttäjä voi olla yhteydessä front-endin kautta epäsuorasti. Kutsutaan myös palvelinpuoleksi.

Cordova / PhoneGap: Sovelluskehys mikä tarjoaa sovelluskehittäjille mahdollisuuden käyttää HTML5, CSS3 ja JavaScript kieliä älypuhelinsovelluksissa. Kehitysympäristön rajapinta keskustelee älypuhelimen natiivin kielen kanssa. PhoneGapin kehitti Nitobi, jonka Adobe Systems osti. Myöhemmässä vaiheessa Adobe siirsi PhoneGapin Apache lisenssin alaisuuteen ja nimeksi vaihdettiin Cordova.

Plug-in: Liitännäinen on tietokoneohjelma, mikä toimii isäntäsovelluksen kanssa vuorovaikutuksessa tarjotakseen uusia ominaisuuksia käyttäjälle. Tyypilliset web-

kehityksessä käytettävät liitännäiset toimivat JavaScriptillä. Sosiaalisen median liitännäisiin viitataan usein liitännäisenä.

Some: Sosiaalinen media. Näihin lukeutuu mm. Facebook, Twitter ja Google+.

API: Application programming interface on ohjelmointirajapinnan määritelmä, minkä avulla ohjelmat voivat keskustella keskenään. Hyvä esimerkki tästä on PhoneGapin rajapinta, mitä ohjelmoidaan JavaScriptillä ja se on yhteydessä puhelimen paikalliseen rajapintaan.

POST muuttuja: Käytetään useasti datan siirtämiseen tiedostoa tai lomaketta lähetettäessä. Post muuttuja ei jää välimuistiin, ei säily selaimen sivuhistoriassa, sen pituudessa ei ole rajoituksia. Ei voida laittaa selaimen suosikkeihin, koska se ei näy osoiterivillä.

GET muuttuja: Käytetään datan hakemiseen osoiteriviltä. Voidaan tallentaa välimuistiin, sivuhistoriaan ja suosikkeihin. Sen pituudessa on rajoituksia ja sitä ei suositella käytettäväksi arkaluonteisen datan kanssa, koska se on näkyvillä osoiterivillä.

Framework: Viitekehys, jossa on määritelty valmiiksi useasti käytettyjä käyttötapauksia.

Front-end: Tarkoittaa käyttöliittymän näkyvää osaa, minkä kanssa käyttäjä on tekemisissä.

Rivinsisäiset tyylit: CSS tyyli, joka on upotettu HTML tunnisteen sisään.

Ruudukko: Viitekehys, jossa sivusto on jaettu pienempiin ruudukkoihin, mitkä käyttäytyvät eri tavalla eri kokoisissa näytöissä ja resoluutioissa.

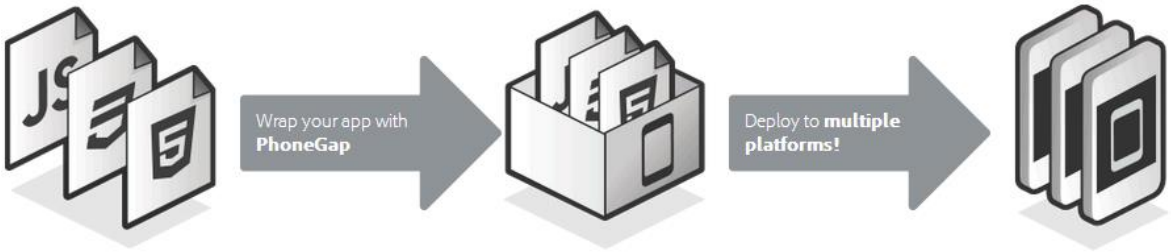
URL: Uniform Recourse Locator. Käytetään osoittamaan WWW-sivuja.

2 Teoriatausta

2.1 PhoneGap

PhoneGapin kehityksen kantavana ideana oli yksinkertaistaa älypuhelinsovellusten kehitystä mahdollistamalla web-kielien käyttö sovelluskehityksessä. PhoneGapin koodiin vaikutti osaltaan Apache Software Foundation (ASF), jonka Apache lisenssin alle PhoneGap lisensoitiin. PhoneGapin haluttiin olevan aina vapaata lähdekoodia.

PhoneGap käyttää HTML, CSS ja JavaScript kieliä. JavaScriptille on tehty lukuisia eri kirjastoja, mitä pystytään hyödyntämään. Ideana on ohjelmoida sovellus olemassa olevilla kielillä, minkä jälkeen se voidaan kääntää eri alustoille. Tämän tyyppisiä älypuhelinsovelluksia kutsutaan hybridi sovelluksiksi.



Kuva 1. PhoneGapin kantava idea (Adobe Systems Inc. 2014)

PhoneGap on vapaata lähdekoodia hyödyntävä sovelluskehys, mikä antaa suoraan yhteyden ohjelmoitavan laitteen eri ominaisuuksiin. Älypuhelimen ominaisuuksiin lukeutuu mm. kiihtyvyysanturi, kamera, kompassi, yhteystietiedot, paikallinen tiedostojärjestelmä, verkot, erilaiset ilmoitukset, median toistaminen ja tallentaminen. (PhoneGap 2014.) Kaikkien näiden toimintaa voidaan ohjelmoida JavaScriptillä mikä kutsuu laitteen syntyperäistä kieltä, mitä ei tarvitse ohjelmoida riviäkään. (Trice Andrew 2012.)

Kehittäjän toteuttaessa PhoneGap ohjelmointirajapinnalla ominaisuuden kutsuu sovellus rajapintaa käyttämällä JavaScriptiä. Silloin erikoistaso ohjelman sisällä kääntää PhoneGap rajapintakutsun oikeanlaiseksi syntyperäiseksi koodiksi, jota toiminnallisuus käyttää. (Trice Andrew 2012.) Eri puhelinmallit voivat käyttää esim. kameraa eri tavalla, mutta PhoneGapin rajapinnalla kehittäjä voi toteuttaa yhden käyttöliittymän, joka käännetään aina kulissien takana oikeanlaiseksi syntyperäiseksi koodiksi, mitä laite käyt-

tää. Kaikki PhoneGapin käyttämät ohjelmointirajapinnat eivät ole saatavilla jokaiselle alustalle, minkä vuoksi syntyperäinen koodi tarjoaa useasti laajemman tuen ohjelmoinnille. Vastaavasti hyötyihin voidaan lukea sovellusten helppo kääntäminen usealle eri alustalle yhdestä ohjelmakoodista.

2.2 Sivuston responsiivinen suunnittelu

Eri laitekantoja ja resoluutioita on nykyään niin paljon että web-sivua ei voida luoda erikseen jokaiselle versiolle. Sama HTML tarjotaan kaikille laitteille, jolloin CSS muuttaa sivun ulkomuotoa. Sen sijaan että luotaisiin eri koodipohja kaikille maailman eri näytöille ja laitteille voi yksi koodipohja palvella kaikkia eri näyttöjä käyttäviä käyttäjiä. Responsiivinen suunnittelu on lähestymistapa web-kehitykseen, mikä luo dynaamisia muutoksia sivun ulkoasuun riippuen katsottavan laitteen näytön koosta, resoluutiosta ja suuntautumisesta. (Smashing magazine 2011.) On olemassa kolme responsiivisen web-suunnittelun muodostavaa ominaisuutta:

- Joustavat kuvat ja media
- Joustava ruudukko-pohjainen ulkoasu, joka käyttää suhteellista mitoitusta
- Media queryjen järkevä käyttö

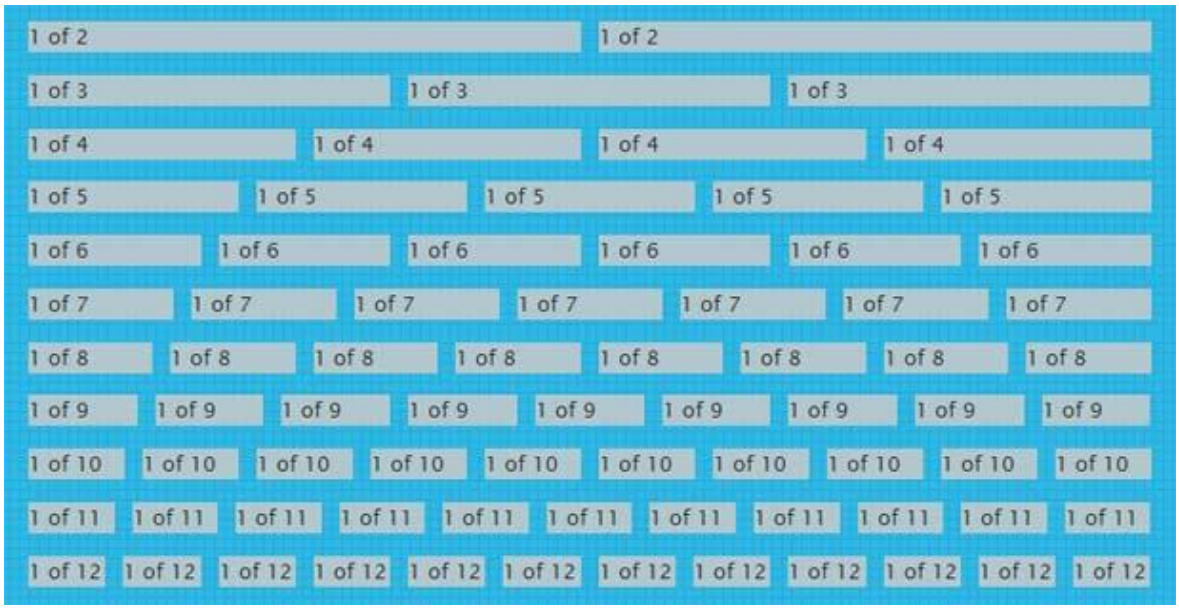
Käyttäjän vaihtaessa tietokoneen tablettiin suhteuttaa sivu automaattisesti kokonsa käytettävälle resoluutiolle, mikä eliminoi tarpeen tehdä erillistä versiota jokaiselle eri päätelaitteelle. (Smashing magazine 2011.) Pienet näytöt eivät ole vain zoomattuja suuria näyttöjä. Sen sijaan sisältö ja tyylit täytyy räätälöidä usealle laitteelle. Suositeltavaa on suunnitella pienelle näytölle ja lähteä askel kerrallaan laajentamaan suurempiin näyttöihin. Tätä lähestymistapaa kutsutaan nimellä mobile first. (Techopedia 2014.)

2.2.1 Joustavat kuvat

Jokainen kuva latautuu täydessä koossaan ja sille annetaan suurimmaksi leveydeksi sata prosenttia. Ylemmän elementin koon muuttuessa selain kaventaa tai laajentaa kuvaa siten että se istuu parhaiten sivulle (MSDN Magazine 2011.) Kyseessä on helppo ja hyvä tapa muuttaa kuvien kokoa luonnollisesti. Ideana on että ei määritellä erikseen leveyttä tai korkeutta vaan annetaan selaimen tehdä se.

2.2.2 Joustava ruudukko

Joustava ruudukkopohjainen sijoittelu on yksi responsiivisen suunnittelun kulmakivistä. Valmiita viitekehyksiä erilaisille ruudukoille on paljon saatavilla. Ruudukko rakennelma koostuu useasti tarkoin määrätystä määrästä sarakkeita, kuten kuusi tai kaksitoista. (Creative blog 2012.) Sarakkeiden sisään voidaan asettaa elementtejä, mitkä vievät sarakkeen antaman tilan. Sisältöalue voi esim. viedä yhdeksän saraketta kahdestatoista sarakkeesta, kun taas sivupalkki vie kolme saraketta kahdestatoista sarakkeesta. Rakennelmat sisältävät useasti ulkoisen sisältöalueen, missä on kiinteä leveys ja välitykset sarakkeiden välillä. Sivun pienentyessä ruudukon eri sarakkeet osaavat automaattisesti asettua allekkain media queryjen avulla, mikä tekee tästä loistavan ratkaisun älypuhelimille ja tableteille.



Kuva 2. Responsiivisen ruudukon tarjoamat mahdollisuudet missä sarakkeet voivat viedä kaiken tarvitsemansa tilan (Creative blog 2012.)

2.2.3 Media queryt

Media queryt ovat yksinkertaisia suodattimia, joita voidaan asettaa CSS tyyleihin. Niiden avulla muokataan tyylejä riippuen laitteen näytöstä, leveydestä, korkeudesta, suuntauksesta tai resoluutiosta. Useasti käytettyjä suodattimia ovat esim. min-width ja max-width, joista ensimmäisen avulla voidaan asettaa minimileveys, missä annetut CSS tyylit toimivat haluttuihin elementteihin, luokkiin tai tunnisteisiin. Max-width tekee päinvas-

taisen ja asettaa suurimman leveyden, missä annetut tyylit toimivat. Sivuston leveyden täyttäessä annetut reunaehdot käytetään annettuja tyylejä.

3 Vaatimusmäärittely

Vaatimusmäärittelyn tarkoituksena on määritellä riittävällä tavalla sovelluksen avainkohdat, minkä avulla sovellusta voidaan kehittää eteenpäin halutun suuntauksen mukaisesti. Riittävän määritelmänä on sovelluksen kattava kuvaus ja käyttötapauskaaviot, joista käy ilmi sovelluksen toiminta eri vaiheissa.

3.1 Yleiskuvaus sovelluksen toiminnasta

Matchart viinivalitsija antaa suosituksia viinin tyyppiin perustuen. Sovelluksen web- ja älypuhelinversiolle toteutetaan yhteinen sivu tuloksille. Sivu avautuu uuteen ikkunaan kun käyttäjä painaa viinisuositusta sovelluksessa. Tiedot lähetetään lomakkeella sovelluksesta tuotekortti-sivulle POST muuttujissa. Facebookista sivulle saavuttaessa muuttujat ovat GET muuttujissa, jonka avulla tarvittava tieto haetaan tietokannasta.

Viinisuosituksesta voidaan tykätä facebookissa ja se voidaan jakaa omalla seinällä. Liitännäiset käyttävät osoiterivillä olevaa tietoa, mihin on määritelty tykätty tai jaettu viini. Sama tieto määrittää myös polun ja sivulle tulostettavan viinin.

Ohjelman lopullisessa versiossa tulokset haetaan Tastingbookin tietokannasta ja tulostetaan tuotekortti-sivulle. Tastingbookin mallissa sivuston front-end puoli ottaa palvelimelle yhteyttä AJAX-kutsulla, mistä tiedot päivitetään reaaliaikaisesti. Tulokset haetaan viinin tyypin perusteella mikä saadaan Matchartin omasta tietokannasta. Tastingbookin järjestelmään tehdään tuki vasta myöhempänä ajankohtana. Tässä opinnäytetyössä Matchartin omaa tietokantaa laajennetaan tukemaan viinihakuja, mitkä tulostetaan tuotekortti-sivulle.

3.2 Käyttöympäristö

Matchart viinivalitsijan käytetään tietokoneen web-selaimen tai älypuhelinsovelluksen kautta.

3.2.1 Tietokone

Sovellus on saatavilla osoitteesta www.matchart.fi ja rekisteröityneet käyttäjät saavat käyttää sitä ilmaiseksi. Sovelluksesta on olemassa muita versioita, mitkä ovat saatavilla erillisten yhteistyökumppanien kautta. Kaikki eri sovelluksen versiot käyttävät omaa tietokantaansa, mutta ovat tekniikaltaan pääpiirteittäin identtisiä.

3.2.2 Älypuhelin

Sovellus on ladattavissa Anrdoid, iOS ja Windows Phone käyttöjärjestelmille. Tekniikaltaan sovellus toimii pääpiirteiltään samalla tavalla kuin työpöytäversiossakin, mutta skaalautuvuus on tehty älypuhelimia silmällä pitäen ja toiminta PhoneGap sovelluskehystä hyödyntäen.

3.3 Käyttötapaukset

Käyttötapausten kuvauksissa kuvataan eri toimintojen riippuvaisuutta toisistaan. Toiminnallisuudet jaetaan toiminnallisuuksiin ja tapahtumiin, mitä sovelluksessa tapahtuu sitä käytettäessä. Käyttötapaukset kattavat sovelluksen ja tuotekortin ja niitä käsitellään liitteessä numero 1.

3.4 Versioiden eroavaisuudet

Älypuhelimen versio toimii PhoneGapilla. Internetissä oleva versio toimii suoraan palvelimelta. Molemmat versiot ovat AJAX:n avulla yhteydessä samalle palvelimelle. Alun perin älypuhelinversion tulokset näytettiin kun kaikki valinnat oli tehty. Tämä on kuitenkin myöhemmin muutettu toimimaan samalla tavalla kuin internetin versio, mikä näyttää tulokset silloin kuin kaikki vaadittavat hakuehdot ovat täyttyneet.

4 Suunnittelu ja kehitys

4.1 Projektin kulku

Opinnäytetyö suunniteltiin alkujaan tehtäväksi n. neljässä kuukaudessa 27.1. – 31.5.2014. Ensimmäiset kolme kuukautta varattiin tehokkaalle työskentelylle. Neljäs kuukausi laskettiin ylimääräiseksi ajaksi, mikäli sovelluksen kehitys tulisi venymään.

Projektia ennen oli keskusteltu valmiiksi Tastingbookin edustajien kanssa siitä miten heidän tietokantaansa pystyttäisiin hyödyntämään Matchart viinivalitsijassa. Matchart viinivalitsijaan rakennettu logiikka tarjoaa suosituksia viinin tyypin tarkkuudella. Puutteena on kattava viinitietokanta moninaisine tietoineen. Tätä puutetta lähdettiin paikkaamaan oikeanlaisen yhteistyökumppanin muodossa.

Opinnäytetyön tekeminen alkoi aloituskokouksella, missä käytiin läpi jo valmiiksi tiedetyt linjaukset ja listattiin valmiilta sovellukselta vaadittavia ominaisuuksia. Alkuperäisessä suunnitelmassa sovelluksen tulokset ajateltiin vain sovelluksen web-versioon. Sovelluksen älypuhelinsovelluksella oli jo helmikuussa 2014 yli 2500 latausta minkä vuoksi samat ominaisuudet haluttiin tehdä tähän versioon.

Tuloksien näyttämistä ajateltiin ensin sovelluksen sisällä näytettäviksi tiedoiksi, koska se koettiin luonnolliseksi ja helposti toteutettavaksi ratkaisuksi. Älypuhelinsovelluksen haasteeksi kuitenkin tuli sosiaalinen media. Käyttäjän jakaessa sivun facebookissa sovelluksella ei linkkiä painavaa henkilöä voida suoraan ohjata sovellukseen, mikä avaisi käyttäjälle jaetut tiedot. Käyttäjälle olisi pystytty tarjoamaan sovelluksen lataus, mutta se ei olisi ratkaissut ongelmaa. Ongelma täytyi ratkaista web-tekniikkaa käyttämällä, jonka vuoksi ratkaisuksi esitettiin käyttäjän ohjaamista erilliseen tuotekorttiin. Sovelluksen web-versio ja älypuhelinsovellus käyttävät samaa pohjaa, mistä käyttäjä ohjataan viinin tyyppiä painaessa molemmassa versiossa yhteiseen erilliseen tuotekorttiin. Tämän sivun eri osista voidaan tykätä tai jakaa ne. Yhteinen sivu tuloksille asetti haasteita myös sivun ulkoasulle, koska sivun täytyi olla yhteensopiva tietokoneen ruudulle ja älypuhelimen pienemmälle näytölle. Ainoa mahdollisus tämän ratkaisemiseksi oli responsiivinen suunnittelu.

Responsiivisen suunnittelun pystyy tekemään monella eri tavalla: Yksi tapa on tunnistaa laite, jolla sivulle saavutaan ja tehdä laitekohtaiset asetukset. Tapa on kuitenkin hyvin työläs, koska jokainen asetus täytyy tehdä erikseen. Toinen nykyään useammin käytetty tapa on tunnistaa laitteen näytön koko ja määrittää asetukset eri resoluutioille CSS tiedostossa. Tätä varten on olemassa erilaisia ruudukkoja, mitkä käyttäytyvät eri tavalla näytön koosta ja resoluutiosta riiippuen. Sivuston front-end on kuitenkin sen verta yksinkertainen rakenteeltaan että ruudukon käyttäminen ei ollut tarpeellista. Elementit saadaan skaalautumaan hyvin tekemällä CSS asetukset huolellisesti.

Ensimmäisessä luonnoksessa tiedot vastaanotettiin tuotekortissa osoiterivillä olevista GET muuttujista. Ideaa ajateltiin liitännäisiä silmällä pitäen, koska URL on helppo jakaa facebookissa. Lopulta päädyttiin käyttämään POST muuttujia, mitkä ovat käyttäjän näkymättömissä. Facebookille annettava URL päädyttiin generoimaan minkä seurauksena facebookin linkistä sivulle saavuttaessa sivu saa GET muuttujasta tiedon sivulla näytettävästä datasta, mikä haetaan tietokannasta.

Noin puolessa välissä projektia maaliskuun lopulla selvisi että Tastingbook ei kerkeä tehdä tukea omaan järjestelmäänsä opinnäytetyön aikarajan puitteissa. Tämä asia oli kuitenkin otettu huomioon riskienhallinnassa ja siihen osattiin varautua. Integraatio Tastingbookin järjestelmään päätettiin siirtää myöhempään ajankohtaan. Kysymykseksi tulikin se että miten korvaava ominaisuus tehdään. Esiehtona ominaisuudella oli se että sen täytyy hyödyttää ohjelman toimintaa jollakin tavalla. Samaan aikaan oli myös palava tarve saada sisältöä tuotekorttiin.

Asiaa pohdittiin jonkin aikaa ja käytiin läpi vaihtoehtoja. Yhtenä mielenkiintoisena ideana oli että sovellukselle haettaisiin testidataa tekstitiedostosta. Idea kuitenkin hylättiin nopeasti, koska se ei mitenkään tukenut ohjelman valittua suuntausta. Lopulta päädyttiin siihen että laajennetaan Matchartin tietokantaa tukemaan tarkempia tietoja viineistä. Ominaisuus ei tule vielä käyttöön, koska tarvitaan myös viiniasiantuntijan sisällönsyöttö, mihin ohjelmoijat eivät kykene. Ominaisuus katsottiin kuitenkin hyväksi ratkaisuksi, jotta sivusto saadaan toimintaan ja sen eri ominaisuuksia voidaan testata.

Asiasta keskusteltiin ohjauskokouksessa huhtikuun alussa ja päädyttiin siihen että se hyväksyttiin. Sovellukseen tulee yksi hyödyllinen ominaisuus lisää, mitä voidaan käyttää tarvittaessa. Tämän jälkeen pystytään hakemaan näytettävää dataa ja ohjelman toiminnallisuutta voidaan samaan aikaan testata.

Projekti venyi alkujaan toukokuun yli, koska allekirjoittaneella alkoi 5.5.2014 työt Accenturen palveluksessa samaan aikaan kun opinnäytetyötä olisi vielä pitänyt tehdä. Työtä riitti niin runsaasti että ei jäänyt aikaa lopputyön viimeistelemiselle keväällä.

4.2 Sovelluksen jatkokehitys

Tekniikan toteutus lähti olemassaolevasta Matchart viinivalitsijasta, jonka tulokset täytyi ohjata erilliselle sivulle. Matchart viinivalitsija ottaa yhteyttä back-end puoleen AJAX-kutsulla ja palauttaa paljon erilaista dataa jota hyödynnetään sovelluksessa. Sovelluksen front-end puoli rakentuu n. 600 rivistä koodia, joka on koodattu noudattaen huonoja koodauskäytäntöjä.

Koodi on rakennettu yhteen putkeen eikä sitä ole hajotettu erillisiin funktioihin, joka haittaa luettavuutta. Hyvä koodauskäytäntö sanoo että yksi funktio on tehty yhdelle ominaisuudelle (W3.org 2013). Tätä kutsutaan modulaariseksi ohjelmoinniksi, missä osat on jaettu pienempiin kokonaisuuksiin. (W3.org 2013). Ratkaisuna koodi pilkottiin pieniksi palasiksi ja ohjelman kaikki eri toiminnot jaettiin pienempiin funktioihin, joita koodissa kutsutaan. Ratkaisumalli mahdollistaa funktioiden käytön eri kohdissa ohjelmakoodia ja tarvittavia muutoksia koodiin ei tarvitse tehdä kuin yhteen paikkaan. Koodia pilkottiin useampaan tiedostoon riippuen käytettävästä kielestä.

Toinen miinus oli rivinsisäiset tyylit, mitkä on määritelty suoraan HTML:n yhteyteen. Rivinsisäiset tyylit vaikuttavat vain valittuun tunnisteeseen ja aiheuttavat päänsärkyä kehittäjälle, koska tyylit eivät automaattisesti kohdistu muihin elementteihin. Hyvä käytäntö on määrittää kaikki tyylit CSS tiedostoon (About.com 2014), missä voidaan viitata elementteihin, id tai class kenttiin. Ratkaisuna rivinsisäiset tyylit poistettiin kokonaan ja muutokset määritettiin CSS tiedostoon. Koodin tilantarve väheni huomattavasti ja muutos helpotti kehittäjän työtä.

Jokaiselle viinityypin suositukselle on sovelluksessa oma lomake, missä on lähetettävät tiedot. Käyttäjän painaessa valittua suositusta tiedot lähetetään POST muuttujissa uudelle sivulle avautuvaan tuotekorttiin.

4.3 Tuotekortti-sivun kehitys

Tuotekortit sivulle saavuttaessa käyttäjä painaa valittua viinityyppiä ja sovellus lähettää lomakkeen ja POST muuttujat. Tuotekortti vastaanottaa muuttujat ja osaa niiden avulla hakea tietokannasta käyttäjälle esitettävän datan. Sovelluksesta sivulle saavuttaessa tuotekortti korostaa aina ensimmäistä viinisuositusta.

Facebookin linkki rakennetaan sivun URL:n avulla, mihin lisätään muuttujien tiedot ja facebookista saavuttaessa tykätyn / jaetun viinin nimi. Käyttäjän tykätessä tai jakaessa sivun tieto menee facebookkiin. Toisen käyttäjän painaessa linkkiä sama sisältö aukeaa kun hän saapuu sivulle. Facebookista saavuttaessa korostetaan ensimmäisenä aina sitä viiniä, mistä on tykätty tai jota on jaettu.

Käyttäjän saapuessa tuotekortti-sivulle on POST tai GET muuttujissa aina tarvittava data, mitä sovellus käyttää tietokantahakua varten. Sivustolle saavuttaessa ilman asianmukaista dataa muuttujissa ohjataan käyttäjä suoraan sovellukseen.

Tuotekortti-sivun ulkoasu kehitettiin normaalia kauemmin, koska sivusta ehdittiin tehdä useampi graafinen ohjeistus ennen kun toimeksiantaja antoi täyden hyväksyntänsä. Ohjeistuksena oli tehdä mahdollisimman yksinkertainen ja sovelluksen ulkoasua mukaileva tuotekortti. Ensimmäinen esitetty graafinen ohjeistus sisälsi viinin otsikon lisäksi tyyppitietoja, kuvauksen ja kuvan. Siinä oli kuitenkin liian paljon materiaalia eikä se saanut hyväksyntää.

Seuraavalla kierroksella tätä ulkoasua hiottiin eteenpäin ja yksityiskohtiin panostettiin enemmän. Lopulta tuli tieto että kuvaa ei haluta käyttää tuloksissa, minkä jälkeen graafinen ohjeistus suunniteltiin täysin alusta. Lopullinen leiska on hyvin yhdenmukainen itse sovelluksen kanssa: Oikeassa yläkulmassa näytetään logo, tämän alla on vasemmalla

ruoan polku. Viinisuositukset näytetään kukin omalla rivillään, vasemmalta löytyy puna- tai valkoviinilasin logo, otsikko ja tekstisisältö. Tämän alla on sosiaalisen median liitännäiset. Kun jotain viiniä painaa avautuvat sen tiedot animaation avulla ja muut menevät piiloon. Näkyvän viinin otsikkoa painettaessa se menee piiloon.

Tuotekortti-sivu on paljon sovellusta staattisempi, koska moneen asiaan ei voi vaikuttaa. Sivun interaktiiviset elementit ovat oikeassa yläkulmassa oleva linkki, josta sovellus avautuu uuteen ikkunaan. Viinin otsikko, mistä joko piilotetaan tai näytetään viinin tiedot sekä sosiaalisen median liitännäiset.

4.4 Tekniikat ja menetelmät

Matchart ensimmäinen versio toteutettiin käyttäen web-tekniikoita. Älypuhelinsovelluksessa haluttiin pystyä hyödyntämään jo olemassa olevaa tekniikkaa. Päädyttiin käyttämään PhoneGap kehitysympäristöä. Lopputuloksena sivusto pystyttiin tehokkaasti kääntämään älypuhelinsovellukseksi. Sovelluksen palvelinpuoli toimii PHP ja MySQL kielien avulla ja siihen otetaan monessa käyttötapauksessa yhteyttä AJAX-kutsulla. Front-end puolella hyödynnetään JavaScriptin jQuery kirjastoa. Sovelluksen kehityksessä on alusta asti käytetty samoja menetelmiä eikä niiden vaihtamiselle ole ollut missään vaiheessa tarvetta.

Kehitystä on tehty useammalla eri koneella, joissa on käytetty Windows 7 ja Linux Ubuntu käyttöjärjestelmiä. Dokumentointiin käytettiin Microsoft Word ja Excel ohjelmia Windowsilla. Linuxilla vastaavat ohjelmat ovat LibreOffice Writer ja Calc. Prosessikaaviot on tehty Microsoft Visio mallinnusohjelmalla.

Grafiikan luomiseen ja muokkaamiseen käytettiin Photoshop kuvankäsittelyohjelmaa Windowsilla ja Gimp kuvankäsittelyohjelmaa Linuxilla. Ohjelmakoodia on muokattu Eclipsellä ja Sublime Text 3 tekstieditorilla. Palvelinpuoleen oltiin yhteydessä SSH ja SCP yhteyksien avulla Linuxin komentoriviltä.

4.5 Liittymät muihin järjestelmiin

4.5.1 Facebook

Sivustolle liitetään facebookin kehittäjäsivulta hankittu JavaScript koodi, mikä ottaa reaaliaikaisesti yhteyttä facebookin järjestelmään. Tämän lisäksi sivustolle täytyy lisätä koodi siihen kohtaan, jonne haluttujen nappuloiden halutaan ilmestyvän. Tämän koodin sisään laitetaan kyseisen sivun osoite, jonka facebook indeksoi omaan järjestelmäänsä. Tähän osoitteeseen pystytään lisäämään mm. tunnistetietoja, jotka selain pystyy lukemaan osoiteriviltä sivustolle saavuttaessa GET muuttujista. Sovelluksen tuloksissa käytetään tätä ominaisuutta valitun viinin korostamiseen ja muiden viinien tietojen piilottamiseen ruudulta. Ratkaisuun päädyttiin, koska osoiterivi on ainoa tapa sisällyttää tarvittava tieto linkkiin mistä saavutaan sivustolle.

Osoiterivillä olevan tiedon avulla sovelluksen eri sivuille kerätään tykkäyksiä ja jakoja. Facebook pystyy palauttamaan erilaista tietoa sivulle, kuten esim. sen paljonko sivusta on tykätty ja montako kertaa ihmiset ovat jakaneet sen omalla aikajanallansa.

5 Testaus

Testaaminen rajoittuu vahvasti kahteen eri osaan: Matchart viinivalitsija sovellukseen ja tuotekortti-sivuun, jossa tulokset näytetään loppukäyttäjälle. Sovelluksesta valitaan viinipolku jo edellisistä versioista tutulla tavalla. Erona on käsiteltävä data ja ohjaus tuotekortti-sivulle, minkä eheys ja toiminta halutaan testata. Tuotekortti-sivulla testattavia ominaisuuksia ovat käyttöliittymän toiminta, ulkoasu ja sosiaalisen median liitännäisten toiminta.

Älypuhelinsovellus on luonteensa puolesta laajemmin testattavissa kuin syntyperäisellä kielellä kirjoitettu sovellus. Tämä on kiinni sovelluksessa käytettävistä web-kielistä ja siitä että versiota on useammalle eri laitealustalle. Sovellusta testataan paikalliselta tietokoneelta ja puhelimelle ladatusta sovelluksesta. Paikallinen kone antaa luotettavia tuloksia, koska sovelluksen tässä osassa ei käytetä liitännäisiä, jotka vaativat julkista palvelinta toimiakseen. Sovelluksen muut osat on testattu toimiviksi jo aikaisemmassa kehitysvaiheessa. Tuotekortti toimii aina samasta paikasta palvelimelta, minne molemmat sovellukset yhdistävät. Testauksessa korostuu suuresti visuaalinen puoli eri päätelaitteilla sekä käyttöliittymä ja liitännäisten oikeaoppinen toiminta. Yksikkötestauksessa varmistetaan sovelluksen palvelinpuolen oikeanlainen toiminta ja datan kulkeminen sovelluksen eri osien välillä.

5.1 Yksikkötestaus

Yksikkötestauksen tarkoitus on varmistaa että annettuun syötteeseen saadaan halutunlainen vastaus. (Microsoft 2014.) Yksikkötestauksessa syvennytään sovelluksen eri osien tiedonvälitykseen. On tärkeätä varmistaa että tieto lähetetään sovelluksesta oikein vastaanottavalle tuotekortille. Lisäksi varmistetaan että facebook saa oikeat tiedot ja osaa ohjata halutulla tavalla takaisin tuotekorttiin.

5.1.1 Datan lähettäminen sovelluksesta

Estämällä sovelluksesta lomakkeen lähettäminen ja rakentamalla konsoliin kirjoittava funktio voidaan haluttu data tulostaa ulos halutussa kohdassa ja varmistaa että siinä ei ole mitään virheitä, puutteita tai häiriöitä.

```
function unitTesting(itemArray, color, number) {
                console.log(itemArray, color, number);
}
```

Funktiota kutsutaan koodin seasta ja sille annetaan parametreiksi muuttujat, jotka tulostetaan konsoliin kun käyttäjä painaa viiniä sovelluksesta.

```
["Game", "Leg", "Venison", "Braised", "Medium", "Gravy", "Root_Vegetables", "white",
"Californian Fumé-Blanc, USA"] "white" 1
```

Ulos tulevasta datasta on luettavissa sovelluksesta valittu polku. Tiedon ollessa tyhjä tarkoittaa se sitä että kyseisessä polussa ei ollut kyseistä valintaa. Viimeisellä järjestysnumerolla kuvataan tietoa monesko puna tai -valkoviini on kyseessä.

5.1.2 Datan vastaanottaminen tuotekorttiin sovelluksesta

Sovelluksesta tuotekorttiin saavuttaessa data otetaan vastaan POST muuttujassa, joka ei ole käyttäjälle näkyvissä. Seuraava koodi muuttaa PHP objektin JavaScript objektiksi ja tulostaa sen sisällön konsoliin.

```
var object = JSON.parse('<?php echo json_encode($itemArray) ?>');
console.log(object);
```

Seuraavan perusteella pystymme toteamaan että kaikki data tulee sovellukselle asti.

```
Object
1. animal: "Beef"
2. degree: "Medium-rare"
3. ingredient: "Fillet"
4. method: "Fried"
5. result: "Argentinian Malbec"
6. sauce: "Gravy"
7. sidedish: "Boiled Sidedish"
8. species: ""
9. type: "red"
```

5.1.3 Datan vastaanottaminen tuotekorttiin facebookista

Facebookin linkistä tuotekorttiin saavuttaessa tiedot ovat GET muuttujassa osoiterivillä käyttäjän nähtävissä. Tätä dataa tarkastelemalla voidaan päätellä onko kaikki tieto tullut oikein läpi.

Kuva 3. Osoiterivillä oleva tieto on GET muuttujissa

Datan lähettäminen facebookiin tuotekortista tapahtuu siten että kyseisen osoitteen perään lisätään muuttujat. Tämä data syötetään liitännäisille, mitkä muodostavat siitä tykkäys ja jaa napit. Tämän testaamista ei erikseen käsitellä tässä lopputyössä.

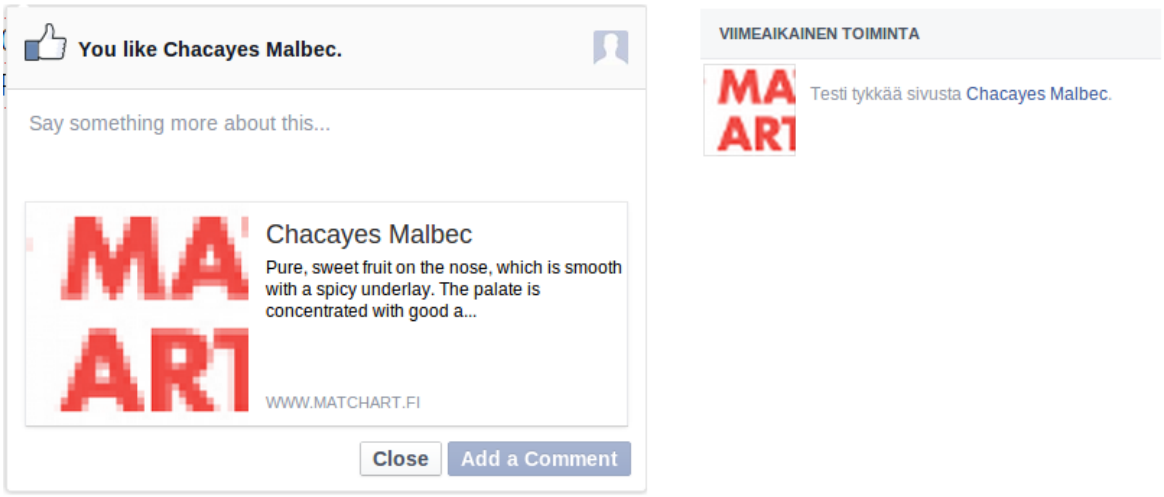
5.2 Käyttöliittymätestaus

Käyttöliittymätestauksessa on tärkeätä analysoida ne komponentit, mitä halutaan testata. (Android Developers 2014.) Käyttöliittymätestaus jakautuu kahteen eri osa-alueeseen tuotekorttissa: JavaScript toiminnallisuudet. Tällä varmistetaan että kaikki sivulla olevat napit ja toiminnallisuudet toimiva. Näihin lukeutuu liitännäisten luomat napit, valintojen korostaminen painamalla nimeä, sivun tyylittely ja responsiivisuus. Näillä testeillä varmistetaan ulkoasun perusrakenteet ja skaalautuminen halutunlaisesti niin tietokoneen selaimelle kuin älypuhelimen näytöllekin.

Liitännäisiä pystytään testaamaan käyttämällä niiden tarjoamia toiminnallisuuksia. Koodissa annetaan parametrit liitännäisille. Parametrien ollessa virheellisiä ei liitännäinen toimi. Testitapauksissa tykätään viineistä ja jaetaan niitä. Läpimenemisehtona on se että liitännäinen osaa hakea tarvittavat tiedot, kuten otsikon ja tekstiosan toiminnallisuutta käytettäessä. Tämän jälkeen haluttu tieto täytyy vielä pystyä löytämään facebokin sivuilta.

5.2.1 Tykkää toiminnallisuuden testaaminen

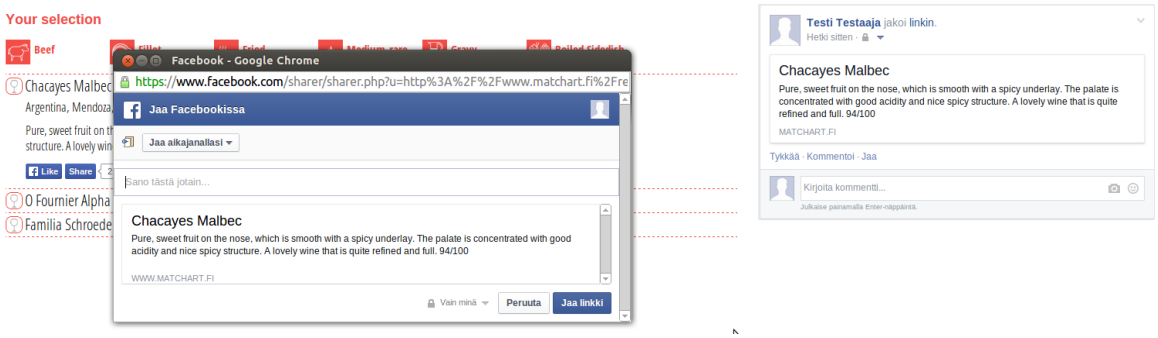
Ensimmäisenä testitapauksessa testataan tykkää napin toiminta. Viinistä tykätään aluksi tuotekortit sivulla, minkä jälkeen mennään facebookkiin katsomaan viimeaikaista toimintaa ja etsitään kyseinen tapahtuma. Lopulta painetaan linkkiä ja siirrytään takaisin tykätylle sivulle.



Kuva 4. Viinistä tykkääminen tuotekortissa ja facebookissa

5.2.2 Jaa toiminnallisuuden testaaminen

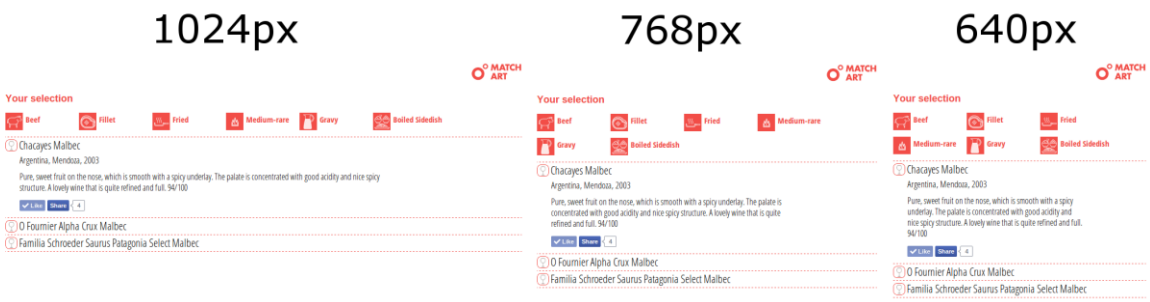
Toisessa testitapauksessa testataan jaa napin toiminta. Painamalla jaa nappia avautuu ikkuna, missä voidaan kertoa jotain jaettavasta sisällöstä. Facebookin liitännäinen hakee tiedot sivulta ja siirtymällä facebookkiin löytyvät samat tiedot omalta aikajanalta. Linkkiä painamalla pystytään jälleen siirtymään takaisin tuotekortti-sivulle.



Kuva 5. Viinin jakaminen tuotekortissa ja facebookissa

5.2.3 Responsiivisuuden testaaminen

Kolmannessa testitapauksessa halutaan varmistaa tuotekortti-sivun oikeaoppinen toiminta erikokoisilla näytöillä. Tätä testataan muuttamalla selaimen kokoa manuaalisesti ja käyttämällä selaimen tarjoamia emulointityökaluja eri tableteille ja älypuhelimille.



Kuva 6. Tuotekortti-sivun responsiivisuus tietokoneen selaimella

Emulointi ei kuitenkaan aina täysin vastaa oikeata laitetta, jonka vuoksi samat testit tehtiin myös älypuhelimella pysty- ja vaakatasossa. Näiden testien myötä pystyttiin varmistamaan että sovelluksen pääosat toimivat halutulla tavalla.

6 Pohdinta

Pohdinnassa tutustutaan työn eri vaiheisiin ja lopputulokseen.

6.1 Tulosten tarkastelu ja jatkokehittämisehdotukset

Alkuperäiseen tavoitteeseen ei päästy, koska Tastingbookin integraatio ei valmistunut ajallansa keväällä 2014. Myöhemmin Tastingbookin järjestelmän valmistuessa ei opinnäytetyön suuntaa enää pystynyt muuttamaan ajankäytöllisistä syistä, koska olin jo täysipäiväisessä työssä. Keväällä integraatiota ei ollut mahdollista tehdä ja syksyllä resurssit eivät olisi riittäneet sen ohjelmointiin, opinnäytetyön kirjoittamiseen ja täysipäiväiseen työhön. Seurauksena integraatio jätettiin tekemättä tämän opinnäytetyön aikana. Korvaavaksi ominaisuudeksi tehtiin tuotekortti, missä näytetään tietokannasta haettuja tuloksia ja myöhemmässä vaiheessa integraation valmistuttua tietoa Tastingbookin tietokannasta. Alun perin opinnäytetyö myöhästyi kesäkuun loppuun asetetusta rajasta, koska aloitin täysipäiväiset työt jo Toukokuun alussa, mitä en pystynyt vielä suunnitteluvaiheessa tietämään tai arvioimaan riskienhallintaan.

Opinnäytetyön tuloksena syntyi täysin uusia ominaisuuksia. Sovellusta ei voida hyödyntää vielä täysin, koska yhteys Tastingbookin tietokantaan puuttuu. Lopullista integraatiota edistäviä edistysaskelia on syntynyt paljon. Prosessin aikana olemassa olevaa koodia on optimoitu entistä selkeämmäksi ja paremmin toimivaksi. Liitännäisten lisääminen oli nykyajan trendien mukainen vaatimus ohjelmaan. Seuraava vaihe on tehdä integraatio Tastingbookin tietokantaan, mikä jäi aikatauluksen takia vielä tästä versiosta tekemättä.

6.2 Pohdintaa työn eri vaiheista

Tuotekortin kehitys aloitettiin käyttämällä osoiterivillä näkyviä GET muuttujia. Nopeasti päädyttiin kuitenkin vaihtamaan menetelmä POST muuttujiin, jotta tieto ei olisi näkyvillä. Vanhassa ratkaisussa tuotekorttiin saavuttaessa sivu tunnistaa että POST muuttujissa on dataa ja ottaa tiedot talteen muuttujiin. Saatu data käsitellään, ja tallennetaan taulukkoon. Datasta generoidaan yksilöllinen koodi, mikä on tallennettu sovelluksessa tietokantaan. Käyttäjän saapuessa tuotekorttiin käyttää sovellus osoiterivillä

olevaa yksilöllistä koodia tietokantahakua varten. Koodi palauttaa kaiken tarvittavan datan käyttäjälle, mihin linkissä on viitattu. Osoiterivillä on tieto valitusta viinistä, jota käytetään korostamaan kyseistä viiniä sivustolla. Seurauksena muut viinit menevät piiloon otsikoita ja liitännäisiä lukuun ottamatta.

Melko nopeasti syntyi ajatus siitä että tehdään tuloksille oma sivu. Tämä kuitenkin laittoi asiat uuteen perspektiiviin ja täytyi miettiä miten jatketaan tästä eteenpäin. Eniten aikaa meni siihen miten sovelluksesta ja facebookista siirrytään tuotekorttiin:

1. Ensimmäisessä versiossa päädyin ratkaisuun, missä tiedot tallennetaan GET muuttujaan selaimen osoiteriville ja ne ovat suoraan käyttäjän nähtävillä sovelluksesta tai facebookista tuotekorttiin saavuttaessa.
2. Toisessa iteraatiossa päädyin rakentamaan logiikan sovellukseen, missä jokaiselle erityyppiselle haulle tallennetaan yksilöllinen tunniste. Tieto tallennetaan kun käyttäjä painaa valitsemaansa viiniä ja lähetetään piilotetussa POST muuttujassa eteenpäin. Sain tämän ratkaisun toimimaan sovelluksesta ja facebookista tuotekorttiin saavuttaessa. Lopulta päädyin hylkäämään ratkaisun liian monimutkaisena. Halusin tiivistää logiikan niin yksinkertaiseksi, toimivaksi ja pomminvarmaksi kuin se vain on mahdollista tehdä.
3. Kolmannessa iteraatiossa en nähnyt enää tarvetta liian monimutkaiselle käsittelylle, minkä takia muokkasin ensimmäisestä versiosta nykyisen version. Tallensin tiedot POST muuttujaan sovelluksesta tuotekorttiin saavuttaessa. Liitännäisille annoin GET muuttujat, jonka avulla ohjaus facebookista onnistuu takaisin sivulle. Yksilöllistä tunnistetta ei enää käytetty tässä versiossa vaan idea hylättiin.

Kolmas versio on ensimmäisen ja toisen version summa, mihin vain parhaat puolet on säästetty pitäen samalla ohjelman logiikka tarpeeksi yksinkertaisena. Näiden työstämiseen meni paljon aikaa ja tätä työvaihetta olisi pystynyt tehostamaan tehokkaammalla suunnittelulla.

6.3 Opinnäytetyöprosessin ja oman oppimisen arviointi

Suurimmaksi haasteeksi koin oman kokemattomuuteni, koska oli paljon erilaisia suuntia joihin olisi voinut lähteä kehitystä viemään. Asioita tuli tehtyä kokeile ja erehdy ta-

van kautta, koska ei ollut tarpeeksi selkeätä näkemystä olemassa alusta asti. Näkemys tarkentui sitä mukaa kun sovelluksen kehitys eteni ja uusia lähestymistapoja syntyi prosessin aikana. Prosessia olisi helpottanut tarkempi suunnitteleminen ja useamman suunnittelukokouksen järjestäminen projektitiimin kautta

Koen enemmän syventäneeni olemassa olevia taitoja kuin oppineeni täysin uusia asioita. Tämä johtuu siitä että Matchart viinivalitsija oli jo valmiiksi projektina minulle hyvin tuttu ja siinä oli käytetty sellaisia kieliä mitä käytän jokapäiväisessä työssäni. Ohjelmoinnista oppii jatkuvasti uutta ja ei ole päivää ettei uusia ratkaistavia ongelmatilanteita tulisi vastaan. Opinnäytetyön rajaus oli hyvä vaikka projektin puolessa välissä tulikin suuria haasteita Tastingbookin tuen myöhästyessä odotetusta aikamääreestä. Tämä aiheutti ison muutoksen, koska täytyi miettiä tarkkaan mihin suuntaan ohjelmaa kehitetään ja mitkä nyt nousevat sovelluksen tärkeimmiksi tavoitteiksi.

Ohjelmistokehittäminen kiinnostaa itseäni paljon ja saatan helposti käyttää paljonkin vapaa-aikaani siihen. Tämän tuloksena voi hyvin syntyä todella hyvä tai huonoa ohjelmakoodia. Tämä selittää myös useat eri versiot Matchart viinivalitsijasta. Ohjelmoinnissa samaan tulokseen voidaan päätyä usealla eri tavalla ja perfektionistin luonteeseen kuuluu että sieltä ei mennä mistä aita on matalin. Pysähtymällä hetkeksi ja miettimään asioita voidaan helposti välttää useampi huonompi ratkaisua ja siirtyä suoraan parhaaseen ratkaisuun. Toimintatapoja tarkastelemalla ja kehittämällä itseä jatkuvasti voidaan saada näkyviä tuloksia aikaan.

7 Lähteet

About.com 2014. Avoid Inline Styles for CSS. Luettavissa: http://webdesign.about.com/od/css/a/aa073106.htm. Luettu: 10.10.2013.

Adobe Systems Inc. 2014. PhoneGapin kotisivu. Luettavissa: http://phonegap.com. Luettu: 18.10.2014.

Android Developers 2014. UI Testing. Luettavissa: http://developer.android.com/tools/testing/testing_ui.html . Luettu: 19.10.2014.

Creative blog 2012. Design sites using the Responsive Grid System. Luettavissa: http://www.creativebloq.com/css3/design-sites-using-responsive-grid-system-9122926. Luettu 25.10.2014.

Microsoft 2014. Testing Concepts and Phases. Luettavissa: http://msdn.microsoft.com/en-us/library/ff798502.aspx. Luettu: 19.10.2014.

MSDN Magazine 2011. Responsive Web Design. Luettavissa: http://msdn.microsoft.com/en-us/magazine/hh653584.aspx. Luettu 25.10.2014.

Smashing magazine 2011. Responsive Web Design: What It Is and How To Use It. Luettavissa: http://www.smashingmagazine.com/2011/01/12/guidelines-for-responsive-web-design/ Luettu 25.10.2014.

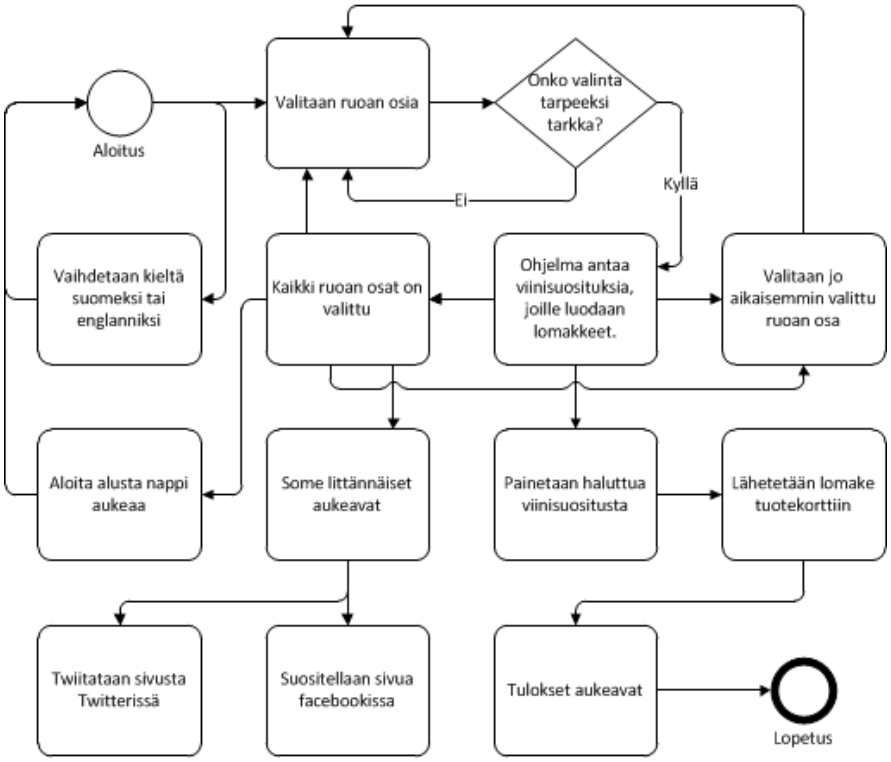
Techopedia 2014. Mobile First Strategy. Luettavissa: http://www.techopedia.com/definition/29153/mobile-first-strategy. Luettu 15.11.2014.

Trice Andrew 2012. Building PhoneGap applications. Luettavissa: http://www.adobe.com/devnet/phonegap/articles/phonegap-apps-powered-by-developercom.html. Luettu: 18.10.2014.

W3.org 2013. JavaScript best practices. Luettavissa:
http://www.w3.org/wiki/JavaScript_best_practices. Luettu: 24.4.2014

8 Liitteet

Liite 1. Käyttötapauskaaviot

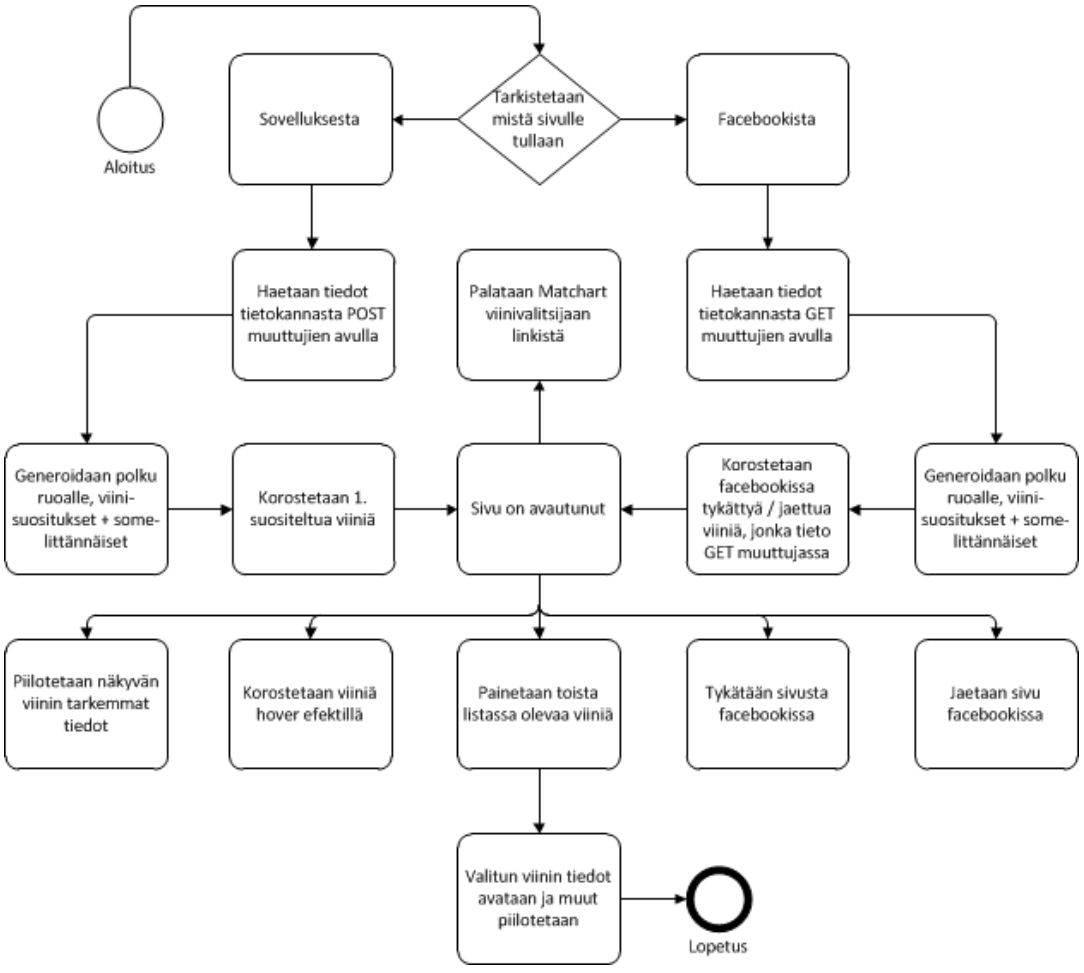


Kuva 1. Käyttötapauskaavio Matchart viininvalitsijan toiminnasta. Pätee Matchartin internetissä toimivaan versioon.

Käyttötapausten kuvaus sovelluksessa

Käyttötapaus	**Esiehto**	**Lopputulos**
Valitaan ruoan osia.	Sovellukseen on saavuttu ja tiedot on tulostettu ruudulle.	Siirrytään seuraavaan ruoan osaan, mikäli kyseessä ei ole viimeinen ruoan osa.
Onko valinta tarpeeksi tarkka?	Käyttäjä on painanut vähintään yhtä valintaa.	Sovellus tutkii onko valinnasta saatu data riittävän tarkka.
Ohjelma antaa viinisuosituksia, joille luodaan lomakkeet	Ruoan osien valinta on riittävän tarkka, eli tarpeeksi osia on valittu.	Tulostetaan tulokset sivulle. Viinisuosituksille luodaan piilossa olevat lomakkeet.
Valitaan jo aikaisemmin valittu ruoan osa.	Vähintään yksi ruoan osa on valittuna.	Valitaan lisää ruoan osia.
Vaihdetaan kieltä suomeksi tai englanniksi.	Sovellukseen on saavuttu ja tiedot on tulostettu ruudulle.	Sovelluksen toiminta alkaa alusta
Aloita alusta nappi aukeaa.	Kaikki ruoan osat on valittu.	Tulostetaan sivun oikeaan

		alalaitaan nappi, jota painamalla pääsee alkuun.
Some liitännäiset aukeavat.	Kaikki ruoan osat on valittu.	Tulostetaan sivun vasempaan alalaitaan some-littännäiset.
Suositellaan sivua facebookissa.	Kaikki ruoan osat on valittu ja some-liitännäiset on tulostettu esille.	Käyttäjä suosittelee Matchart viinivalitsijaa facebook seinällänsä.
Twiitataan sivusta Twitterissä.	Kaikki ruoan osat on valittu ja some-liitännäiset on tulostettu esille.	Käyttäjä twiittaa Matchart viinivalitsijasta Twitterissä.
Painetaan haluttua viinisuositusta.	Vähintään yksi viinisuositus on tulostettu esille.	Tallennetaan tiedot tietokantaan uniikilla koodilla ja lähetetään tiedot tulos-sivulle, joka esittää ne.
Lähetetään lomake tuotekorttiin.	Käyttäjä on tehnyt valintansa.	Lähetetään tiedot.
Tulokset aukeavat.	Viinisuositusta on painettu.	Avataan uuteen ikkunaan sivu tuloksille.



Kuva 2. Käyttötapauskaavio sivusta, joka näyttää tulokset käyttäjälle. Pätee Matchartin internetissä toimivaan versioon.

Käyttötapausten kuvaus tuotekortti-sivulla

Käyttötapaus	**Esiehto**	**Lopputulos**
Tarkistetaan mistä sivulle tullaan.	Sivulle on saavuttu.	Sivulle voidaan saapua sovelluksesta tai facebookin linkistä.
Haetaan tiedot tietokannasta POST muuttujien avulla.	POST muuttuja on tarkistettu ja siinä on tietoa.	Tiedot on haettu tietokannasta POST muuttujan avulla, parsittu ja tallennettu arrayhin
Haetaan tiedot tietokannasta GET muuttujien avulla.	GET muuttuja on tarkistettu ja siinä on tietoa.	Tiedot on haettu tietokannasta GET muuttujan avulla, parsittu ja tallennettu arrayhin
Generoidaan polku ruoalle, viinisuositukset + some-liitännäiset.	Kaikki tarvittava data on pystytty hakemaan, mitä tähän tarvitaan.	Tulostetaan ruoan polku, viinit ja some-littännäiset sivulle.
Korostetaan 1. suositeltua viiniä.	Juoman tiedot on pystytty hakemaan POST muuttujasta.	Korostetaan 1. viinisuositus, ja kaikki muut vaihtoehdot pysyvät piilossa.
Korostetaan facebookissa tykättyä / jaettua viiniä.	GET muuttujassa on tieto tykätystä / jaetusta viinistä, joka haetaan.	Korostetaan valittu viini ja kaikki muut vaihtoehdot pysyvät piilossa.
Painetaan toista listassa olevaa viiniä.	Viinisuosituksia on useampi kuin yksi.	Painettu viini tuodaan esiin ja painamista ennen esillä ollut viini piilotetaan.
Piilotetaan näkyvän viinin tarkemmat tiedot.	Viinisuosituksia on vähintään yksi.	Valittu viini piilotetaan. Vain otsikko ja some-liitännäiset näkyvät.
Korostetaan viiniä hover efektillä.	Hiiri on tuotu viinin otsikon päälle.	Hover efekti näkyy niin kauan kun käyttäjä pitää hiirtä valinnan päällä.
Tykätään sivusta facebookissa.	Käyttäjä on painanut facebookin like nappia.	Käyttäjä tykkää sivusta facebook seinällänsä.
Jaetaan sivu facebookissa.	Käyttäjä on painanut facebookin share nappia.	Käyttäjä jakaa viinisuosituksen facebook seinällänsä.
Palataan Matchart viinivalitsijaan linkistä	Sivu on avautunut käyttäjälle ja käyttäjä on painanut logoa	Matchart viinivalitsija avataan toiseen ikkunaan
Valitun viinin tiedot avataan ja muut piilotetaan	Käyttäjä on painanut toista viiniä	Painettu viini tuodaan esiin ja painamista ennen esillä ollut viini piilotetaan.