Tero Ala-Hulkko

Leikkikatalogi-mobiilisovellus

Opinnäytetyö
Syksy 2017
SeAMK Tekniikka
Tietotekniikan tutkinto-ohjelma

SeAMK
SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

SEINÄJOEN AMMATTIKORKEAKOULU

Opinnäytetyön tiivistelmä

Koulutusyksikkö: Tekniikan yksikkö

Tutkinto-ohjelma: Tietotekniikka

Suuntautumisvaihtoehto: Ohjelmistotekniikka

Tekijä: Tero Ala-Hulkko

Työn nimi: Leikkikatalogi-mobiilisovellus

Ohjaaja: Markku Lahti

Vuosi: 2017 Sivumäärä: 40

Opinnäytetyön tavoitteena oli tehdä Seinäjoen kaupungin varhaiskasvatukselle puhelinsovellus, jota varhaiskasvattaja voi käyttää apuna työpäivän aktiviteettien suunnittelemiseen. Sovelluksella käyttäjä voi selata sekä lisätä leikkejä, ohjelmia ja materiaaleja yhtenäiseen kokoelmaan.

Sovellus koostuu kahdesta osasta, puhelinsovelluksesta sekä palvelimen toiminnoista. Käyttäjä voi tallentaa sovelluksella aktiviteetteja tai materiaaleja. Aktiviteetteihin ja materiaaleihin liitetään kuva sekä opaste. Kuvat ja opasteet tallennetaan sekä palvelimelle että tietokantaan. Käyttäjät voivat tarkastella tietoja puhelinsovelluksen selaa-toiminnolla.

Asiakassovelluksen alustana käytettiin Android-järjestelmää ja palvelimen osalta useita tekniikoita, joista olennaisimpia olivat SQL sekä PHP. Tuloksena työstä on käytännöllinen kannettava ideapankki varhaiskasvattajille. Se auttaa monipuolistamaan lasten päiviä.

Avainsanat: Puhelinsovellus, tietokanta, PHP, Java, SQL

SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Thesis abstract

Faculty: School of Technology

Degree programme: Information Technology

Specialisation: Software Engineering

Author: Tero Ala-Hulkko

Title of thesis: Leikkikatalogi -mobile application

Supervisor: Markku Lahti

Year: 2017 Number of pages:40

The purpose of the thesis was to make a mobile application for the early childhood education department of the city of Seinäjoki. The aim was that the application would assist the childcare providers in planning their workday. The application allows its users to browse the collection as well as to add games, activities and materials to it.

The application consists of two parts: the mobile application and server functions. Through the mobile application, a user can save pictures and instructions to an activity or material. The pictures and instructions are stored both on the server and a database for others to view through the browse function of the mobile application.

An Android operating system was chosen for the client application and several technologies were used on the server side, most relevant of which were SQL and PHP. As the result of the thesis, there is a mobile application that is a handy, portable idea bank for childcare providers, which helps to bring variety to children’s days.

Keywords: mobile app, database, PHP, Java, SQL

SISÄLTÖ

Kuvaluettelo

Käytetyt termit ja lyhenteet

API-taso	Kertoo ohjelman yhteensopivuuden Android-käyttöjärjestelmän versioihin (Android Developers [Viitattu 4.11.2017]).
ASO	**(**App Store Optimization) Sovelluksen näkyvyyden parantaminen sovelluskaupassa (Sefferman 15.3.2016).
BLOB	**(**Binary Targe Object) Tietokannoissa suurten tiedostojen tallentamista kuvaava termi. (Sears, van Ingen & Gray 2006).
Valintaikkuna	Ruudulla esitettävä kenttä, jolla kommunikoidaan käyttäjän kanssa.
EditText	Käyttäjän muokattavissa oleva tekstikenttä.
GIGO	(Garbage In, Garbage Out) Järjestelmän antama tieto on riippuvainen siihen syötetystä tiedosta (Lidwell, Holden & Butler 2010, 16).
ImageView	Kuvien esittämiseen tarkoitettu työkalu.
Jailbreak	Applen käyttöjärjestelmän rajoitusten kiertäminen.
painike, nappi, Button	Painettava ruutu, josta aktivoidaan ohjelman tietty toiminto.
Sekvenssikaavio	Vuorovaikutusta kuvaava kaavio.
Spinner	Avautuva monivalintalista, josta valitaan yksi vaihtoehto.
SQL-injektio-hyökkäys	Hyökkäystyyppi tietokantoja vastaan. Tässä tietokantapalvelimen halutaan toteuttavan toimintoja, joihin sitä ei ole tarkoitettu. (Baars 18.10.2017.)
SQL-lauseke	Tietokantojen tietojen käsittelyyn tarvittava komento.

1 JOHDANTO

1.1 Työn tausta

Varhaiskasvattajat ohjaavat työssään useita erilaisia aktiviteetteja lasten kanssa. Aktiviteeteissa ja muussa toiminnassa käytetään materiaaleja, kuten kirjoja sekä teippauksia lattioilla.

Leikkikatalogi-mobiilisovelluksen taustalla oli varhaiskasvattajien tyytymättömyys saatavilla olevaan tietoon nykyaikaisista materiaaleista, aktiviteeteista ja muusta lasten kanssa tehtävästä toiminnasta. Tiedon helppo saatavuus auttaa monipuolistamaan varhaiskasvatuksen päiväohjelmaa. Nykyään tietoa on saatavilla eri puolilla internetiä ja kirjallisena, mutta tarve oli kehittää helppokäyttöinen puhelinsovellus.

1.2 Työn tavoite

Työn tavoitteena oli tehdä sovellus ideapankista, josta olisi helposti saatavilla tietoa laajasta valikoimasta aktiviteetteja ja materiaaleja. Sovelluksen oli tarkoitus monipuolistaa lasten päiviä päivähoidossa. Aktiviteettien tiedon oli myös oltava nopeasti työpäivän ohessa sisäistettävää ja käyttöönoton oltava helppoa.

Jotta tieto olisi nopeasti sisäistettävää ja aktiviteeteista saisi nopeasti käsityksen, vaati tämä kuvamateriaalia. Seinäjoen varhaiskasvatuksen liikuntalähetit olivat ottaneet ja keränneet kuvia jo ennen työn aloittamista. Kuvia oli tarkoitus käyttää kuvaamaan sovelluksella haettavia aktiviteetteja. Liikuntalähetit kirjoittivat kuviin opasteet, jotka olisivat sovelluksella haettavissa. Lisäksi käyttäjien haluttiin voivan lisätä sovellukseen omia kuviaan ja opasteitaan.

1.3 Työn rakenne

Opinnäytetyön toisessa luvussa käydään läpi käytettävyyden suunnittelua.

Opinnäytetyön kolmannessa luvussa kerrotaan mobiilisovelluksen kehittämiseen liittyvistä seikoista.

Neljännessä luvussa käydään läpi sovelluksen toteuttamisessa käytetyt tekniikat ja ohjelmistot. Lisäksi kerrotaan, missä roolissa ne olivat opinnäytetyön toteuttamisessa.

Viidennessä luvussa tutustutaan ohjelmistokehittämisen elinkaarimalleihin.

Kuudennessa luvussa käydään läpi vaihe vaiheelta asiakassovelluksen toiminta. Luvussa tutustutaan käyttäjien käytössä oleviin yksittäisiin toimintoihin. Lisäksi tarkastellaan käyttöliittymän perusideat, joita hyödynnettiin opinnäytetyössä.

Seitsemännessä luvussa tarkastellaan, mitä vaikutusta luvussa kuusi läpi käydyillä toiminnoilla on palvelimen toimintaan.

Luvussa kahdeksan käydään läpi jatkokehitysmahdollisuuksia.

Viimeisessä luvussa on yhteenveto ja pohditaan opinnäytetyössä onnistumista.

2 KÄYTETTÄVYYS

2.1 Helppokäyttöisyys

Helppokäyttöisyyden periaate on, että tuotteen käyttämiseen ei vaadita erikoistaitoja. Helppokäyttöisyydellä on neljä tunnusmerkkiä: ymmärrettävyys, operoitavuus, yksinkertaisuus ja virheiden minimointi. (Lidwell, Holden & Butler 2010, 16.)

Ymmärrettävyydellä tarkoitetaan tuotteen helppoa tulkittavuutta aistien perusteella. Ymmärrettävyyden parantamisen keinoja ovat mm. tekstin ja kuvien yhdessä käyttäminen. Operoitavuuden tavoite on, että tuote on käytettävissä fyysisistä ominaisuuksista huolimatta. Operoitavuuden parantamisen keinoja on mm. näytön kontrollien helppokäyttöisyys. Yksinkertaisuuden tavoite on, että tuote on kaikkien käytettävissä. Tuote ei saa vaatia aiempaa kokemusta tai korkeaa keskittymisen tasoa. Tuotteen yksinkertaisuuden suunnitteluun kuuluu mm. monimutkaisuuden minimointi, näytön kontrollien selkeys ja vain olennaisen tiedon esittäminen. Virheiden minimointi on virheiden määrän ja vaikutusten vähentämistä. (Lidwell, Holden & Butler 2010, 16.)

2.2 Toimintojen suorittaminen

Käytettävyyttä suunnitellessa käyttäjälle helpoin toimintatapa on syytä suunnitella myös parhaaksi toimintatavaksi. Toimintoja suorittaessaan käyttäjä on vapaa tekemään valintoja. Oletusvalinnat on kuitenkin asetettu siten, että ne ovat mahdollisimman vastuullisia. Käyttäjää voidaan myös opastaa käyttäjälle sopivimman toimintatavan suorittamisessa. Käyttäjälle on pystyttävä luomaan selkeä mielikuva tavoitteesta, jotta tavoitteeseen varmimmin päästään. (Lidwell, Holden & Butler 2010, 170.)

Valinnaisuus. Toimintojen, vaihtoehtojen ja valinnaisuuden paljous heikentää käytettävyyttä. Toimintojen lisäämisellä pystytään vastaamaan useampaan käyttäjän tarpeeseen. Toiminnot ja vaihtoehdot lisäävät myös monimutkaisuutta ja aikaa, joka käyttäjältä menee ohjelman suorittamiseen. Tarpeiden ennakoimisella voidaan saavuttaa korkea käytettävyys ja tarvittavat toiminnallisuudet. (Lidwell, Holden & Butler 2010, 102.)

Tiedon jakaminen osiin. Tiedon asteittainen esittäminen on keino vähentää käyttäjän kuormitusta. Kullakin hetkellä käyttäjälle esitellään vain olennaisin osa tiedosta. Tiedon jakaminen osiin parantaa käyttäjän kykyä käsitellä tietoa ja vähentää turhautumista. Tiedon osiin jakaminen on erityisen tehokas keino parantaa satunnaiskäyttäjän käyttäjäkokemusta. Tiedon osiin jakamisella pystytään myös vähentämään käyttäjien tekemien virheiden määrää. (Lidwell, Holden & Butler 2010, 188.)

2.3 Virheet

Virhe on odottamaton toiminto tai toiminnon puute, joka johtaa odottamattomiin tuloksiin. Suurin osa virheistä tapahtuu puutteellisen suunnittelun vuoksi. Virheet voidaan lajitella kahteen eri tyyppiin: virheellisiin suorituksiin ja virheellisiin toimintoihin. (Lidwell, Holden & Butler 2010, 82.)

Virheellinen suoritus tapahtuu, kun käyttäjä tekee oikean asian väärällä tavalla. Virheellisiä suorituksia voidaan karsia antamalla välitöntä palautetta käyttäjälle virheestä. Palautetta voidaan antaa virhesanomilla, joissa on korjausehdotuksia. Lisäksi näytön kontrollit on aseteltava siten, ettei niitä aktivoida vahingossa. (Lidwell, Holden & Butler 2010, 82.)

Virheellinen toiminto tapahtuu, kun käyttäjä tekee virheen. Virheiden syitä ovat mm. alhainen vireystila tai huomion kiinnittäminen toisaalle. Tehokkaita tapoja välttää virheellisiä toimintoja on asetella oleelliset asiat helposti saataville ja vahvistuspyyntöjen käyttö ennen kriittisiä toimintoja. Lisäksi toimiva virheiden korjaaminen on oleellista virheiden seurauksien minimoimiseksi. (Lidwell, Holden & Butler 2010, 82.)

Virheiden vaikutusten minimointi. Tuotteet, joissa virheillä ei ole suuria vaikutuksia, rohkaisevat tuotteen käyttämiseen. Tietyt virheiden vaikutusten minimoimisen

tekniikat ovat parempia kuin toiset. Mahdollisuuksien rajoittaminen siten, että virhettä ei pääse tapahtumaan on monesti paras keino vähentää virheiden vaikutuksia. Muita hyviä keinoja ovat toiminnon peruminen ja erilaiset turvaverkot, jotka vähentävät virheiden seuraamuksia. Edellä mainitut keinot ovat suotavampia kuin varoitukset, vahvistuspyynnöt ja avun tarjoaminen. Varoituksia, vahvistuspyyntöjä ja avun tarjoamista ei välttämättä tarvita, jos virheiden seuraukset ovat muuten hyvin toteutettu. (Lidwell, Holden & Butler 2010, 104.)

Vahvistuspyynnöt. Vahvistuspyynnöillä pyritään estämään virheitä ja varmistamaan, että suoritettava toiminto on tarkoituksellinen. Vahvistuspyyntöjä on syytä käyttää vain ennen kriittisiä toimintoja, koska ne hidastavat ohjelman suorittamista. Vahvistuspyyntöjä on kahdenlaisia: dialog ja kaksivaiheinen suoritus. Dialog-tyyppinen vahvistuspyyntö esittää käyttäjälle valintaikkunan. Valintaikkunassa kysytään käyttäjältä vahvistus toiminnon suorittamiseen. Useat valintaikkunat voivat johtaa käyttäjän turhautumiseen. Kaksivaiheisessa suorituksessa edellytetään jonkin aiemman asetetun ehdon täyttymistä ennen kuin toiminto voidaan suorittaa. (Lidwell, Holden & Butler 2010, 54.)

GIGO (Garbage In, Garbage Out) on periaate, jossa järjestelmän antama tieto riippuu sille syötetystä tiedosta. Virheiden suodatuksella pyritään välttämään kahden tyyppistä tietoa: väärän tyyppistä tietoa sekä huonolaatuista tietoa. Väärän tyyppisen tiedon pääseminen järjestelmään on vakavampaa, ja voi johtaa odottamattomiin tuloksiin. Väärän tyyppistä tietoa on esim. puhelinnumero luottokortin numerolle tarkoitetussa kentässä. Väärän tyyppinen tieto on helpompi tarkistaa ja suodattaa kuin huonolaatuinen tieto. Huonolaatuista tietoa, kuten kirjoitusvirheitä, on vaikeampi suodattaa, mutta se ei yleensä ole vakavaa. (Lidwell, Holden & Butler 2010, 112.)

2.4 Visuaaliset tekijät

Esteettisyys. Suunnittelussa tuotteen ulkoasu on tärkeä tekijä. Visuaalisesti näyttävä tuote saa nopeammin käyttäjän hyväksynnän. Käyttäjät kokevat visuaalisesti näyttävän tuotteen olevan helppokäyttöisempi, kuin se todellisuudessa on. Visuaalisesti näyttävä tuote vaikuttaa positiivisesti käyttäjän mielialaan, jolloin mahdolliset ongelmat eivät vaikuta yhtä vakavilta. (Lidwell, Holden & Butler 2010, 20.)

Värit. Värejä on suunnittelussa käytettävä harkiten. Värien liiallisella käytöllä tuote vaikuttaa sekavalta. Ihminen huomaa ensivilkaisulla noin 5 väriä, mitä pidetään hyvänä maksimimääränä värien käytölle. Hyvillä väriyhdistelmillä voidaan parantaa tuotteen ulkoasua. Esimerkiksi väripaletin vierekkäisten tai vastakkaisten värien käyttö yhdessä tuottaa esteettisiä yhdistelmiä. Värejä käytetään suunnittelussa mm. huomion kiinnittämiseen, asioiden ryhmittelemiseen ja ulkoasun kohentamiseen. (Lidwell, Holden & Butler 2010, 48.)

Luettavuus. Tekstin luettavuus perustuu tekstin ominaisuuksien lisäksi tekstin ja taustan kontrastiin. Optimaalisena tekstin kokona pidetään fontin kokoa 9-12. Suurempiakin fonttikokoja suositellaan mm. ikäihmisiä varten. Vaaleaa taustaa vasten on käytettävä tummia tekstin värejä. Tekstin luettavuus ei heikkene jos taustan ja tekstin välinen kontrastiero on vähintään 70 %. Tekstin luettavuus pysyy hyvänä myös moniväristä taustaa vasten, jos kontrastiero on vähintään 70 %. (Lidwell, Holden & Butler 2010, 148.)

Korostukset. Korostukset ovat tehokas keino huomion kiinnittämiseksi haluttuun kohtaan. Korostukset menettävät tehokkuuttaan, jos ne peittävät yli 10 % osuuden näkyvästä alasta. Korostamisen keinoja ovat mm. värit, tekstin ominaisuudet ja välkkyminen. (Lidwell, Holden & Butler 2010, 120.)

Alkupiste. Alkupisteellä tarkoitetaan sitä kohtaa, johon käyttäjä kiinnittää huomion ensimmäisenä tuotteeseen tutustuessaan. Alkupisteen hyvällä suunnittelulla pyritään tekemään positiivinen ensivaikutelma. Alkupisteen suunnittelussa tärkeää on mm. mielikuvan muodostaminen tuotteen tarjoamista mahdollisuuksista. (Lidwell, Holden & Butler 2010, 80.)

Elementtien asettelu. Yhdenmuotoiset elementit tulisi asetella linjaan näytön tai muun kiintopisteen mukaisesti. Elementtien linjaan asettelu toisen elementin tai näytön reunojen perusteella edistää mielikuvaa, jossa elementit ovat toisiinsa kytköksissä. Elementtien keskikohdan mukaan linjatut elementit edistävät mielikuvaa, jossa elementit eivät ole toisiinsa kytköksissä. (Lidwell, Holden & Butler 2010, 24.)

Monimuotoiset elementit tulisi asetella linjaan elementtien alan perusteella. Monimuotoisten elementtien asettelu tulisi suorittaa silmämääräisen arvion perusteella. Monimuotoisten elementtien tulisi olla tasapainossa linjassa olevien elementtien muodostaman akselin päällä. (Lidwell, Holden & Butler 2010, 30.)

3 MOBIILISOVELLUKSISTA

Kun kehittämään mobiilisovellusta, kehittäjän on tehtävä useita valintoja liittyen mm. alustan valitsemiseen ja julkaisemiseen.

3.1 Alustan valitseminen

Alustan valitsemisella on suuri rooli kehittämisen alkuvaiheessa. Yksi merkittävä tekijä alustan valitsemisessa on käyttöjärjestelmä. Koska käyttöjärjestelmiä on useita, keskitytään tässä työssä käytetyimpiin vaihtoehtoihin.

Nykyään Googlen Android- ja Applen iOS-käyttöjärjestelmät kattavat yli 90 % älypuhelinmarkkinoista (Netmarketshare 10/2017). Vaikka suurin osa sovelluksista on mahdollista toteuttaa molemmille käyttöjärjestelmille, on kuitenkin valittava, mille käyttöjärjestelmälle sovelluksen kehittää ensimmäisenä.

Markkinaosuudet. Android on tällä hetkellä käytetyin käyttöjärjestelmä mobiililaitteissa. Androidin maailmanlaajuinen markkinaosuus on 78 %, kun iOS-käyttöjärjestelmän markkinaosuus on 18 %. Jos sovellus on Yhdysvaltojen markkinoille erityisesti suunnattu, tilanne muuttuu. Yhdysvalloissa iOS-laitteet hallitsevat 52 % markkinaosuudesta, kun Android-laitteiden markkinaosuus on 42 %. Maailmanlaajuisesti aktiivisia Androidia käyttäviä laitteita on yli kaksi miljardia kappaletta. Applen iOS-laitteiden määrä on n. 800 miljoonaa. (Netmarketshare 10/2017.)

Laite. Toinen huomioitava tekijä käyttöjärjestelmän ohella on laite, jolla sovellusta on suunniteltu käytettävän. Erittäin suosittuja ovat tablettitietokoneet, joissa käytetään edellä mainittuja käyttöjärjestelmiä. Ne tarjoavat tiettyjä etuja älypuhelimiin verrattuna. Merkittävin etu on ruudun koko. On arvioitava, miten paljon sovellus hyötyy ruudun koon kasvamisesta. (Android Developers [Viitattu 4.11.2017].)

Applen iOS-järjestelmä hallitsee tablettitietokoneiden markkinaosuuksia. Tablettitietokoneista yli 60 % käyttää iOS-käyttöjärjestelmää ja Android-käyttöjärjestelmää hieman yli 30 %. (Statcounter 10/2017.).

Asiakaskunta. Alustojen eroavuuksien vuoksi Android- ja iOS-laitteiden asiakaskunnilla on eroja. Android-laitteita on saatavilla jokaisessa hintaluokassa. Android-laitteiden keskimääräiseen asiakaskuntaan kuuluu enemmän teknologista osaamista kuin iOS-laitteiden asiakaskuntaan. (Yarmosh 13.5.2016.)

iOS-laitteiden keskimääräinen hinta on korkeampi kuin Android-laitteiden. Keskimääräinen iOS-käyttäjä on korkeammin koulutettu, varakkaampi ja käyttää enemmän rahaa sovelluskaupassa kuin keskimääräinen Android-käyttäjä. (Yarmosh 13.5.2016.)

Rajoitukset. Joissain erityistapauksissa valintatekijäksi nousee Androidin avoimen lähdekoodin suomat vapaudet. Tietynlaiset sovellukset ovat toteutettavissa vain Android-käyttöjärjestelmällä. (Yarmosh 13.5.2016.)

Apple on tunnettu tiukasta linjastaan iOS-järjestelmän ja ns. jailbreakin suhteen. Jailbreak sallii mm. kolmannen osapuolten sovellusten asentamisen Applen iOS-laitteisiin. Nämä kolmannen osapuolten sovellukset eivät välttämättä noudata Applen sovelluskaupan AppStoren linjauksia. Vaikka jailbreak ei ole laitonta, Apple ei ole antanut sille hyväksyntäänsä. Androidilla muokkausten tekeminen on helpompaa ja sallitumpaa. (Keller 13.2.2017.)

3.2 Julkaiseminen

Android-käyttöjärjestelmän sovellusten julkaisemiseen on useita tapoja. Android-laitteille julkaisun voi toteuttaa Googlen sovelluskaupassa PlayStoressa, sovelluksen kehittäjän verkkosivulla tai sähköpostilla. iOS-laitteille suunnattujen sovellusten julkaisemineen on vain yksi tapa: Applen AppStore. Seuraavassa tarkastellaan julkaisua edeltäviä käytäntöjä.

AppStore. AppStoressa kehittäjän on ensimmäiseksi hakeuduttava Applen sovelluskehittäjille tarkoitettuun ohjelmaan. Seuraavaksi sovelluksen kehittäjän on lisättävä tietueet sovelluksen ominaisuuksista AppStoreen. Tietueiden kuuluu sisältää mm. sovelluksen kuvaus ja kuvankaappaukset. Sovelluksen voi ladata kauppaan tietueiden lisäämisen jälkeen. Sovelluksen lataamisen jälkeen kehittäjällä on mah-

dollisuus beta-testata sovellustaan. Beta-testaaminen ei ole edellytys julkaisemiselle. Sovelluksen voi myös lähettää suoraan julkaisukelpoisuuden tarkistukseen. Julkaisukelpoisuuden tarkistuksessa sovelluksen on täytettävä AppStoren standardit. Jos sovellus todetaan tarkistuksessa julkaisukelpoiseksi, sovelluksen voi julkaista. (Apple [Viitattu 4.11.2017].)

Android. Julkaisutavasta riippumatta Android-sovelluksen on aina täytettävä julkaisukelpoisuuden ehdot. Ohjelmakoodi on siivottava logi-kutsuista ja koodiin on sisällytettävä sovelluksen version tiedot. Edellä mainittujen toimien jälkeen sovellus on julkaisukelpoinen, mutta sen täytyy olla toimiva. Toimivuuden ongelmat saattavat johtaa sovelluksen poistamiseen PlayStoresta. (Android Developers [Viitattu 2.11.2017].)

PlayStore. Ensimmäinen vaihe sovelluksen julkaisemiseen PlayStoressa on markkinointimateriaalin lisääminen. Markkinointimateriaaliin kuuluu mm. kuvankaappaukset, videot, kuvaukset ja mainostekstit. Toinen vaihe on asetusten konfigurointi, esim. maat, joissa sovellus julkaistaan, hinta ja mahdolliset ikärajoitukset. Kun pakolliset asetukset on tehty, sovelluksen voi julkaista. (Android Developers [Viitattu 2.11.2017].)

Muut julkaisutavat. Sovelluksen voi julkaista kehittäjän verkkosivulla tai sovelluksen voi lähettää suoraan asiakkaan sähköpostiin. Koska asennus ei tapahdu PlayStoren kautta, asiakkaan täytyy hyväksyä sovellusten asentaminen tuntemattomista lähteistä. Erityisesti uusimmissa Androidin käyttöjärjestelmäversioissa edellä mainittu voi olla ongelmallista, koska ne eivät salli yhden kerran poikkeuksia asennuksiin tuntemattomista lähteistä. Asiakkaan on itse navigoitava asetuksiin ja sallittava tuntemattomista lähteistä asennettavat sovellukset. (Android Developers [Viitattu 2.11.2017].)

3.3 Tuotteen näkyvyys

Oman sovelluksen saaminen käyttäjälle näkyväksi tuhansien muiden sovellusten joukosta on haastavaa. ASO (App Store Optimization) on näkyvyyden maksimointia

sovelluskaupassa. ASO-menettelyyn kuuluu kehitetyn sovelluksen vahvuuksien tunnistaminen mm. kohderyhmän ja kielen perusteella. (Sefferman 15.3.2016.)

Tehokkaisiin käytäntöihin kuuluu sovelluksen nimen huolellinen valitseminen. Haigin (29.4.2013) tutkimuksen mukaan sovellusta kuvaavan avainsanan sisällyttäminen sovelluksen nimeen lisäsi latausten määrää 10,3 prosentilla. Pitkän nimen valitseminen taas huononsi tulosta, koska sovelluskauppa lyhentää sovelluksen nimen hakutuloksissa tietyn merkkimäärän jälkeen. Avainsanoja on myös syytä käyttää avainsanalistassa. Avainsanalistalle varattu tila tulee käyttää kokonaan, koska se parantaa hakuosumien määrää.

Sovelluksen täytyy olla helposti huomattava myös hakutuloksista koostuvassa listassa. Tärkeimpiä tekijöitä hakutuloksissa erottumiseen ovat ensivilkaisulla houkutteleva esikatselukuva ja tuotteen kuvaus. Sovelluksen kuvausta tulee myös täydentää kuvankaappauksilla sovelluksesta sekä videoilla. (Sefferman 15.3.2016.)

4 KÄYTETYT TEKNIIKAT

Android Studio. Android Studio on Googlen kehittämä IntelliJ IDEA -ympäristöön pohjautuva kehitysympäristö (Android Developers [Viitattu 28.10.2017]). Android Studio on Android-sovellusten kehittämiseen tarkoitettu ohjelma, joten se oli luontainen valinta kehitysympäristöä valitessa.

Android Studio helpottaa joidenkin vaadittujen asetusten tekemistä. Esimerkiksi käyttöjärjestelmän version valintaan oli ohjelmassa opasteet. Opinnäytetyön käyttöjärjestelmäversio asetettiin API-tasolle 25 eli lähes uusimmille käyttöjärjestelmille. Vähimmäisvaatimus asetettiin API-tasolle 15.

Käyttöjärjestelmänä API-taso 15 tarkoittaa vähimmäisversiota 4.0.3, joka kattaa lähes 99 % käytössä olevista laitteista (Android studio [Viitattu 29.10.2017]). Opinnäytetyön käyttöjärjestelmäversio valittiin version 4 mukana tuomien parannusten määrän ja käytössä olevien laitteiden määrän vuoksi. Ohjelman suunnitteluun ja testaamiseen hyödynnettiin käytössä ollutta LG Nexus 5 -puhelinta.

Xampp. Xampp on Apache Friendsin kehittämä paikallisen verkkopalvelimen kehittämistä varten suunniteltu sovellus. Se on ensisijaisesti tarkoitettu testaamista varten, minkä vuoksi siitä on tehty mahdollisimman yksinkertainen. (Apache Friends [Viitattu 4.11.2017].)

Xampp on käyttäjän tarpeisiin muokattava ohjelma. Käyttäjä voi lisätä moduuleja oletuksena asentuvia ominaisuuksia täydentämään. Opinnäytetyöhön riittivät oletuksena asentuvat Apache- ja MySQL-moduulit.

Apache. Apache on Apache Software Foundationin kehittämä avoimen lähdekoodin verkkopalvelinohjelma. Sitä pidetään turvallisena sekä nopeana. Se on käytössä 67 % verkkopalvelimista. (Ellingwood 13.8.2013.)

MySQL/MariaDB. MySQL on Oracle Corporationin kehittämä tietokantojen hallintajärjestelmä. Se on avoimen lähdekoodin SQL-järjestelmä. (MySQL [Viitattu 4.11.2017].)

Xamppin-pakettiin kuuluu MariaDB Fountationin suunnittelema MariaDB. MariaDB Foundation on MySQL-järjestelmän alkuperäisten kehittäjien perustama yritys. MariaDB on käytettävyydeltään lähes yhtäläinen ohjelmoijalle MySQL-järjestelmään verrattuna, mutta sisältää joitain lisäyksiä. (MariaDB [Viitattu 4.11.2017].) Opinnäytetyössä tietokannoilla oli olennainen rooli ohjelman toimivuuden kannalta.

PHP. PHP on avoimen lähdekoodin ohjelmointikieli. PHP soveltuu verkko-ohjelmointiin, koska se toimii palvelimen päässä. (The PHP Group [Viitattu 4.11.2017].) Opinnäytetyössä PHP-kieltä käytettiin toimimaan välittäjänä asiakassovelluksen, palvelimen ja tietokannan välillä.

Java. Java on Sun Microsystemsin kehittämä ohjelmointikieli. Javalla on monia samankaltaisuuksia C- ja C++-kielien kanssa. Se on osin kehitetty niiden pohjalta. (Sun Microsystems, Inc. 1997.).

Java on suunniteltu olemaan yksinkertainen. Mittapuuna yksinkertaiseen on pidetty C++-kieltä, joka on merkittävästi monimutkaisempi, mutta myös samalla jonkin verran monipuolisempi ohjelmointikieli. Java-kieli on suunniteltu luotettavaksi ja toimimaan jokaisella alustalla. Tärkeänä suunnitteluperiaatteena on myös ollut suorituskyky. (Sun Microsystems, Inc. 1997.) Java on Android Studion oletuskieli, mikä teki siitä luontaisen valinnan opinnäytetyön toteuttamiseen.

XML. XML on World Wide Web Consortiumin kehittämä merkitsemiskieli. Se on suunniteltu varastoimaan ja siirtämään tietoa. XML ei varsinaisesti itsessään tee mitään, mutta sillä tehtyjä merkintöjä voidaan käyttää hyödyksi ohjelmoinnissa. (W3Schools [Viitattu 4.11.2017].) XML soveltuu käyttöliittymäsuunnitteluun, johon sitä opinnäytetyössä käytettiin.

Tietojärjestelmä tietokantojen tukena. Tietojärjestelmää hyödyntävässä menetelmässä tietokantoihin tallennetaan polut suuriin tiedostoihin, jotka sijaitsevat tietojärjestelmässä.

Vaihtoehto tiedostojärjestelmän hyödyntämiselle on BLOB-tiedostotyypin käyttö. BLOB (Binary Large Objebt) on tarkoitettu suurien tiedostojen tallentamiseen tietokantaan. BLOB-tyyppisen tiedoston hyöty riippuu käsiteltävien tiedostojen koosta. Alle yhden megatavun kokoiset tiedostot hyötyvät BLOB-tiedostotyypin käytöstä.

Pienikokoisia tiedostoja sisältävä BLOB-tyyppinen tiedosto on nopeampi käsitellä, kuin tietojärjestelmässä sijaitsevia tiedostoja. Ajan mittaan tiedon sirpaloituminen hidastaa BLOB-tiedoston suoritusnopeutta enemmän, kuin tietojärjestelmän suoritusnopeutta. Pitkällä aikavälillä sekä tiedonsiirron nopeus, että sirpaloitumisen vähäisyys ovat parempia tiedostojärjestelmässä. (Sears, van Ingen & Gray. 2006.)

Opinnäytetyössä tiedostojen koot olivat pääasiassa alle yksi megatavua. Suurimmat tiedostot ylittivät yhden megatavun rajan. BLOB-tiedostotyypin käyttö parantaisi suoritusnopeutta lyhyellä aikavälillä. Pitkän aikavälin suoritusnopeus ja toteutuksen mielekkyys olivat päätekijät, joiden vuoksi opinnäytetyössä hyödynnettiin tietokantojen tukena tiedostojärjestelmää BLOB-tiedostotyypin sijaan.

5 SOVELLUKSEN KEHITTÄMINEN

Ohjelmistotuotannossa on käytössä useita malleja, joilla ohjelmiston kehittäminen voidaan jakaa osiin. Malleihin kuuluu mm. vesiputousmalli, spiraalimalli ja nykyään laajasti yritysten käytössä olevat ketterät menetelmät.

Vesiputousmalli on selkeä ja yksinkertainen, mutta harvoin käytännöllinen. Vesiputousmallin periaatteena on kehittää asia kerrallaan. Ongelmaksi muodostuu mm. se, että yhden vaiheen loppuun saattaminen saattaa viedä paljon aikaa koko organisaatiolta, mutta ei kaikilta sen työntekijöiltä. Tällöin syntyy useita toimettomia työntekijöitä. (Tervakari 17.12.2008.)

Ketterät menetelmät on kokoelma malleja, joilla on yhdistävä tekijä. Ne pyrkivät siihen, että ohjelmaa rakennetaan pala palalta, mutta jokaisen vaiheen tuotoksena on julkaisukelpoinen ohjelma. (Tervakari 14.1.2014.)

Spiraalimalli yhdistelee useita seikkoja ketteristä menetelmistä ja vesiputousmallista. Spiraalimallissa kehitys on jatkuvaa ja yksi sovelluksen palanen kehitetään toimivaksi kerrallaan. (Tervakari 17.12.2008.)

Spiraalimallin vaiheet ovat syklisiä. Ensimmäisessä vaiheessa määritellään tavoitteet ja rajoitteet. Toisena arvioidaan kehittämiseen liittyvät vaihtoehdot ja riskit. Kolmantena tapahtuu itse kehittäminen ja seuraavan tason tuotteen määrittely. Neljäntenä suunnitellaan seuraavia vaiheita ja arvioidaan resursseja. Neljännen vaiheen jälkeen alkaa uusi sykli. Syklisyyden vuoksi asiakas on aktiivisesti mukana kehittämisessä. (Tervakari 17.12.2008.)

Opinnäytetyöhön spiraalimalli oli toimivin ratkaisu. Ensimmäisen määrittelyn jälkeen päästiin aloittamaan suunnittelu. Spiraalimallin etuna on, että koko ratkaisun ei tarvitse olla selvillä työn alkaessa. Koska asiakas oli myös yhteistyökumppani, spiraalimallin vaatima asiakkaan aktiivisuus kehitysprosessissa oli luontaista.

6 ASIAKASSOVELLUS

6.1 Käyttöliittymä

Käyttöliittymä on taso, jolla käyttäjä kommunikoi tietokoneen kanssa. Sen tehtävä on sallia tehokas työskentely tietokoneen työkaluilla. Tuotesuunnittelussa on tärkeä ottaa huomioon mm. kohderyhmä, jolle tuote on suunniteltu. Käyttöliittymää ei tule suunnitella liian monimutkaiseksi, jotta kaikilla olisi mahdollisuus käyttää sovellusta tehokkaasti ilman pitkäaikaista opettelua.

Ulkoasussa on pyritty panostamaan selkeyteen. Erilaisten toimintojen määrä on pidetty alhaisena jokaisessa näkymässä, minkä lisäksi jokaiseen vaiheeseen on sanallinen opaste. Painikkeet ovat selkeitä, jotta vältyttäisiin tulkintavirheiltä. Painikkeisiin on asetettu ohjelmallisesti esteet, mikä estää käyttäjää syöttämästä mitään tietokantaa vahingoittavaa tekstiä. Lisäksi painikkeiden aktivointi estetään, eli painikkeet lukitaan, toimintojen suorittamisen ajaksi. Painikkeiden lukitseminen estää käyttäjää esim. lisäämästä samaa aktiviteettia useampaan kertaan. Ruudulle avautuvalla tekstillä on suuri kontrasti taustan kanssa, jotta se olisi paremmin luettavissa. Vaikka sovellus onkin epävirallinen, Seinäjoen värit on esitetty kelta-vaaleanpunaisella taustalla sekä sinisillä painikkeilla. Käyttöliittymän ohjelmointiin käytettiin avuksi XML-kieltä (kuva 1).

```
<TextView
    android:text="Vaihtoehtoisesti kirjoita hakusana. Ei pakollinen"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/textView9"
    android:layout_above="@+id/etHaku"
    android:layout_centerHorizontal="true"
    android:layout_marginBottom="40dp"
    android:textSize="22dp"
    android:textStyle="bold"/>

<Spinner
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/spSijainti"
    android:layout_marginBottom="98dp"
    android:layout_above="@+id/spTyyppi"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true" />
```

Kuva 1. Otos XML-koodista

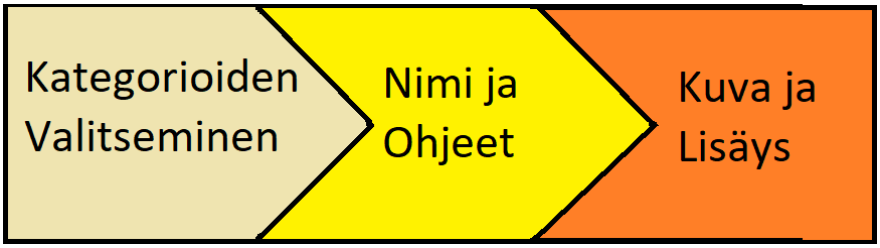
6.2 Päätoiminnot

Ohjelma sisältää kaksi päätoimintoa: aktiviteetin lisääminen sekä olemassa olevien aktiviteettien selaaminen. Kuvassa 2 on ensimmäinen näkymä ohjelmaa avattaessa. Aloitusruutu sisältää kaksi painiketta (kuvan 2 kohdat 1 ja 2), joista pääsee suorittamaan ohjelman päätoimintoja. Näytössä olevista painikkeista avautuu näkymät, jotka opastavat käyttäjää vaihe vaiheelta.

Kuva 2. Aloitusruutu

6.3 Aktiviteetin lisääminen

Ensimmäinen päätoiminto on nimeltään Lisää. Aktiviteetin lisääminen ohjelmaan tapahtuu kolmivaiheisella menettelyllä (kuva 3). Näin on pyritty välttämään informaation liiallista tulvaa yhdellä kerralla. Käyttäjältä pyydetään tietojärjestelmän tarvitsemia tietoja sekä myös tietoja itse aktiviteetista tai materiaalista.

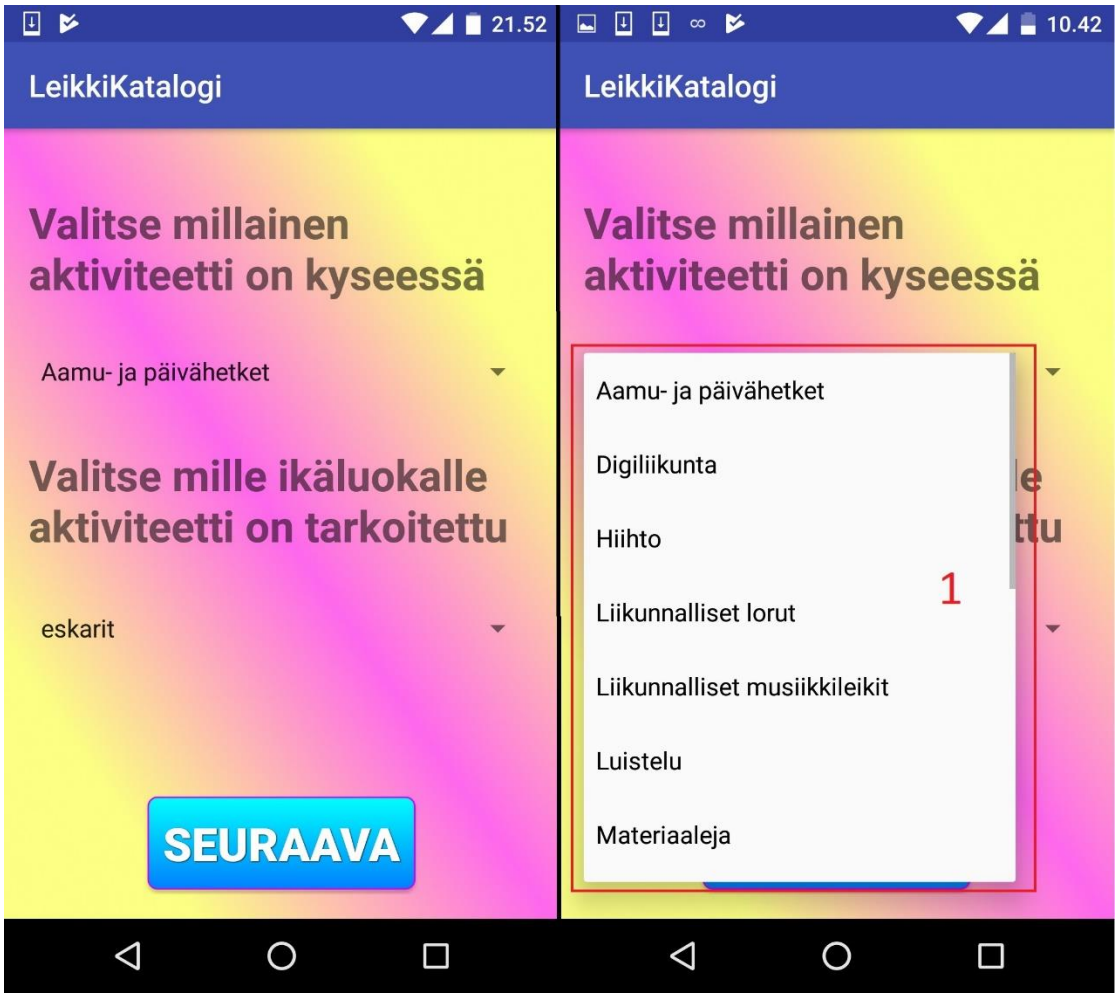


Kuva 3. Aktiviteetin lisäämisen eteneminen

6.3.1 Kategorioiden valitseminen

Aktiviteetin lisäämisen ensimmäisessä vaiheessa ohjelmaan avautuu näkymä kahdesta spinneristä (kuva 4). Käyttöliittymäsuunnittelussa on pyritty välttämään liiallista valinnaisuutta. Käyttäjälle esitellään varhaiskasvatusammattilaisten suunnittelema lista aktiviteeteista, materiaaleista ja opasteista.

Avautuvasta 16 vaihtoehdon listasta (kuva 4, kohta 1) käyttäjä valitsee lisättävää aktiviteettia parhaiten kuvaavan vaihtoehdon. Tarvittaessa ohjelma pyytää myös tietoa, mille ikäluokalle aktiviteetti on tarkoitettu. Materiaaleja lisättäessä ohjelma pyytää materiaalin tyypin esim. kirjallisuus.



Kuva 4. Aktiviteetin lisäämisen ensimmäinen vaihe

6.3.2 Nimi ja opaste

Aktiviteetin lisäämisen toisessa vaiheessa ohjelmaan avautuu näkymä kahdesta EditText-kentästä. Näkymässä käyttäjää opastetaan antamaan aktiviteetille nimi (kuva 5, kohta 1) ja lyhyt opaste (kuva 5, kohta 1).

Opasteet valmiiksi tehtyihin aktiviteetteihin on pyritty pitämään lyhyinä. Opasteiden mitan kohtuullistaminen auttaa aktiviteetin sisäistämisessä esim. kesken työpäivän. Lisäämisvaiheessa kirjoitettavien opasteiden mittaa ei rajoiteta.

Kuva 5. Aktiviteetin lisäämisen toinen vaihe

6.3.3 Kuvan lisääminen

Aktiviteetin lisäämisen kolmannessa ja viimeisessä vaiheessa näytölle avautuu näkymä, jossa on ImageView-ruutu. Kuvan 6 kohdan 1 ruudun painalluksesta avautuu puhelimen kuvagalleria. Kuvagalleriasta käyttäjä voi valita kuvan aktiviteetille tai materiaalille. Kun kuva on valittu, se tulee pienikokoisena näkyviin ImageView-ruudun päälle.

Kuva 6. Aktiviteetin lisäämisen kolmas vaihe

Tiedot lähetetään tietojärjestelmään Lisää katalogiin -painikkeella. Niiden lopullinen sijoituspaikka määräytyy aktiviteetin lisäämisen ensimmäisen vaiheen tietojen perusteella. Lisätty aktiviteetti tulee välittömästi muiden käyttäjien selattavaksi. Kun aktiviteetti on lisätty, käyttäjä palautetaan aloitusruutuun. Lisäksi ruudulle tulee ilmoitus aktiviteetin lisäyksen onnistumisesta.

6.4 Aktiviteettien selaus

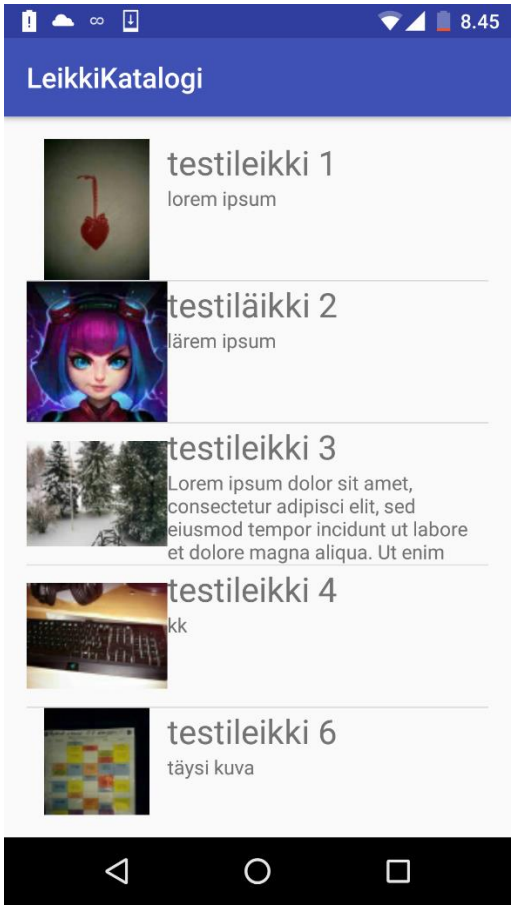
Toinen päätoiminto on nimeltään Selaa. Selaa-toiminnolla haetaan aktiviteetteja kahdella käytössä olevalla tavalla: haulla listaan sekä aktiviteetin nimen perusteella. Hakutavoille on yhteinen näkymä, jossa ruudulla on molempiin opasteet. Kuvassa 7 on aloitusruudun Selaa-painikkeesta avautuva Selaa-toiminnon näkymä.

Kuva 7. Selaa-toiminnon näkymä

6.4.1 Haku listaan

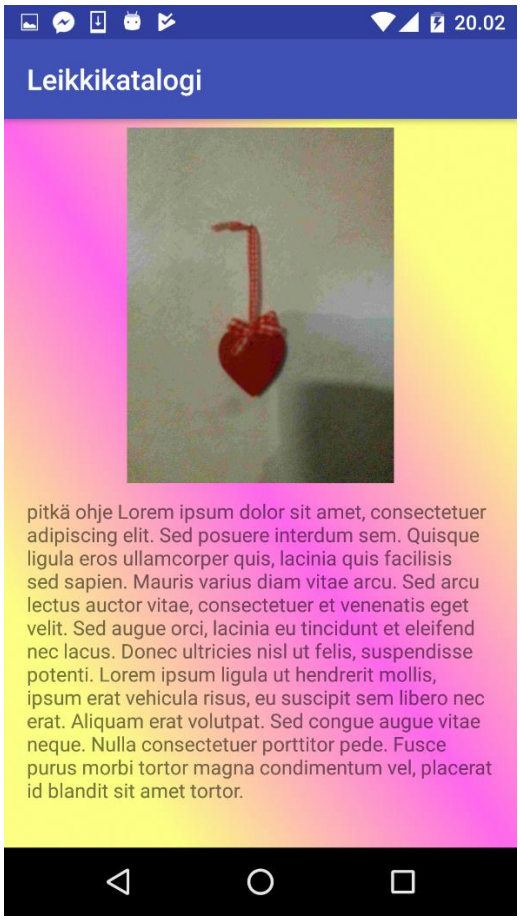
Ensimmäinen tapa hakea aktiviteetteja on haku listaan. Aktiviteettien hakeminen listaan on kätevin tapa selata sisältöä, kun ei ole tiedossa tiettyä aktiviteettia. Käyttöliittymässä on panostettu helppouteen ja nopeuteen sekä pyritty välttämään liian suuria määriä hakutuloksia. Hakutulosten määrää on pyritty kohtuullistamaan useilla kategorioilla, joihin haku kohdistuu.

Selaa-toiminnon näkymässä, kuvassa 7 kohdassa 1, käyttäjä voi valita kategoriat, joihin haku kohdistuu. Näiden perusteella haetaan lista aktiviteeteista, joista nimen lisäksi esitetään myös esikatselukuva sekä lyhyt teksti opasteen alusta (kuva 8).



Kuva 8. Listanäkymä

Mikäli listanäkymä ei luo riittävän informatiivista kuvaa aktiviteetista, voi aktiviteetin valitsemalla avata tarkemman näkymän. Avautuvassa näkymässä valitusta aktiviteetista esitetään suurempi kuva sekä täydelliset opasteet (kuva 9).



Kuva 9. Aktiviteetin täydellinen näkymä

Kuvan 9 näkymässä valitsemalla kuvan voi avata edelleen suurennetun kuvan aktiviteetista.

6.4.2 Haku aktiviteetin nimellä

Toinen tapa hakea aktiviteetteja on hakea aktiviteetti suoraan sen nimen perusteella.

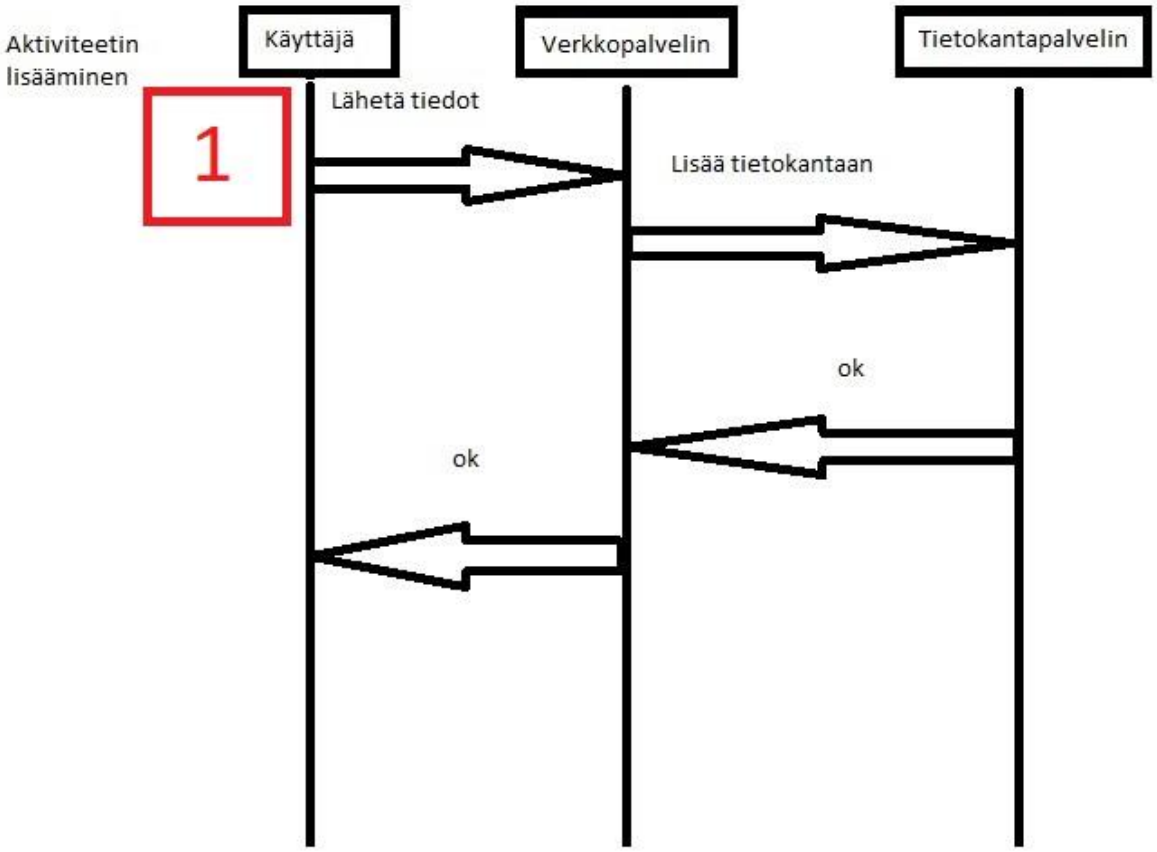
Jos käyttäjä tietää aktiviteetin nimen, voi hän kirjoittaa sen Selaa-toiminnon näkymän hakukenttään (kuva 7, kohta 2). Tällöin Suorita haku -painikkeesta avautuu suoraan kuvan 9 näkymä kyseiselle aktiviteetille.

7 PALVELIMEN TOIMINNOT

Käyttäjäsovelluksen lisäksi ohjelma tarvitsee palvelimen toimintoja. Opinnäytetyössä palvelimen toiminnot koostuvat kahdesta osasta: verkkopalvelimen sekä tietokantapalvelimen toiminnoista.

7.1 Aktiviteetin lisääminen palvelimelle

Esimerkkinä asiakassovelluksen, verkkopalvelimen sekä tietokantapalvelimen keskinäisestä vuorovaikutuksesta on aktiviteetin lisääminen palvelimelle. Apuna tarkasteluun käytetään sekvenssikaaviota (kuva 10).



Kuva 10. Sekvenssikaavio aktiviteetin lisäämisestä

Kuvan 10 kohdassa 1 käyttäjä on painanut Lisää katalogiin -painiketta ja ohjelma on jäänyt taustalle odottamaan palvelimen vastausta. Lisättävän aktiviteetin tiedot lähetetään käyttäjän puhelimesta verkkopalvelimen PHP-tiedostolle (kuva 11).

```
19  $upload_folder ="upload/".$isokategoria . "/".$isoleikkityyppi;
20  $tkpath="http://[redacted]:8012/leikit/upload/".$isokategoria ."/".$isoleikkityyppi."/".$isoname;
21  $conn = new mysqli($servername, $username, $password, $dbname);
22
23  $stmt = $conn->prepare("INSERT INTO leikkitaulu2 (leikki,leikkityyppi,kategoria, path) VALUES (?, ?,?,?)");
24  $stmt->bind_param("ssss", $isoname,$isoleikkityyppi,$isokategoria,$tkpath);
25
26  $stmt->execute();
27  $stmt->close();
28  $conn->close();
29  //kuvat, ohjeet ja pikkukuvat palvelimelle
30  $path="$upload_folder/$isoname.jpeg";
31  $path3="$upload_folder/$isoname". "thumbnail" ".jpeg";
32  $path2="$upload_folder/$isoname.txt";
33  if(file_put_contents($path,$decodedImage) !=false ){
34      echo "upload_success";
35  }
36  else{
37      echo "upload_failed";
38  }
39  if(file_put_contents($path2,$ohjeet) !=false ){
40      echo "upload_success2";
41  }
42  else{
43      echo "upload_failed2";
44  }
```

Kuva 11. PHP-tiedoston palanen

Verkkopalvelimen PHP-tiedoston tehtäviin kuuluu mm. tietokantaan tietojen lähettäminen, tiedostojen sijoittaminen palvelimelle sekä asiakassovellukselle ilmoittaminen tehtävien valmistuttua.

Lisääminen tietokantaan. Ensimmäiseksi käyttäjän lähettämille tiedoille muodostetaan oikeassa muodossa olevat merkinnät tietokantapalvelinta varten. Toisessa vaiheessa tiedot lähetetään tietokantaan. Lähetys tapahtuu esivalmistellulla SQL-lausekkeella, joka yhdessä asiakassovelluksen rajoitteiden kanssa takaa, ettei tiedostojen polut voi mennä väärin.

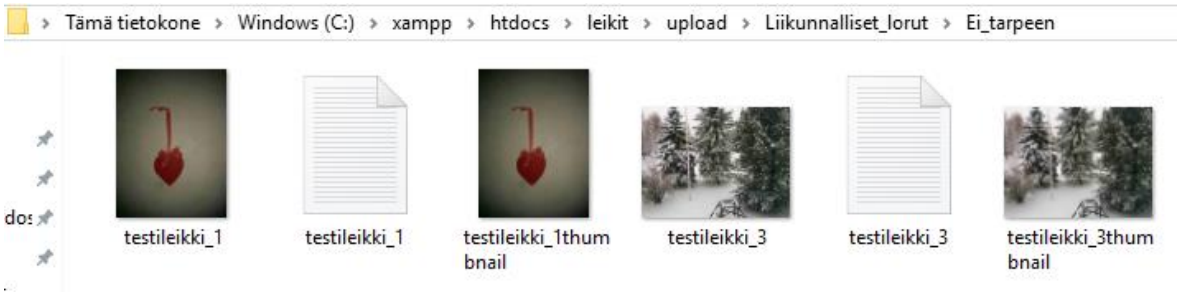
Lisäksi esivalmistellut lausekkeet lisäävät turvallisuutta. Esivalmistellut lausekkeet estävät SQL-injektio-hyökkäyksiä palvelinta vastaan (Baars 18.10.2017).

Tietokantapalvelimen vastaanotettua tiedot ne lajitellaan käyttäjältä ja PHP-tiedostolta saatujen määritelmien mukaan. Tietokantaan tallennetaan hakuehtojen lisäksi polut, joista käyttäjän tallentamat kuvat ja ohjeet voidaan löytää (kuva 12).

leikki	leikkityyppi	kategoria	path
testileikki_1	Ei_tarpeen	Liikunnalliset_lorut	http://[redacted]:8012/leikit/upload/Liikunnal...
testiläikki_2	Ei_tarpeen	Liikunnalliset_lorut	http://[redacted]:8012/leikit/upload/Liikunnal...
testileikki_3	Ei_tarpeen	Liikunnalliset_lorut	http://[redacted]:8012/leikit/upload/Liikunnal...
testileikki_4	Ei_tarpeen	Liikunnalliset_lorut	http://[redacted]:8012/leikit/upload/Liikunnal...
testileikki_6	Ei_tarpeen	Liikunnalliset_lorut	http://[redacted]:8012/leikit/upload/Liikunnal...

Kuva 12. Otos tietokannasta

Lisääminen tietojärjestelmään. PHP-tiedoston toinen tehtävä on lisätä tiedostot tietojärjestelmään (kuva 13). Koska alustava polku on jo muodostettu tietokantaa varten, voidaan samaa polkua käyttää tiedostojen tallentamiseen lisäämällä siihen tiedostopäätteet.



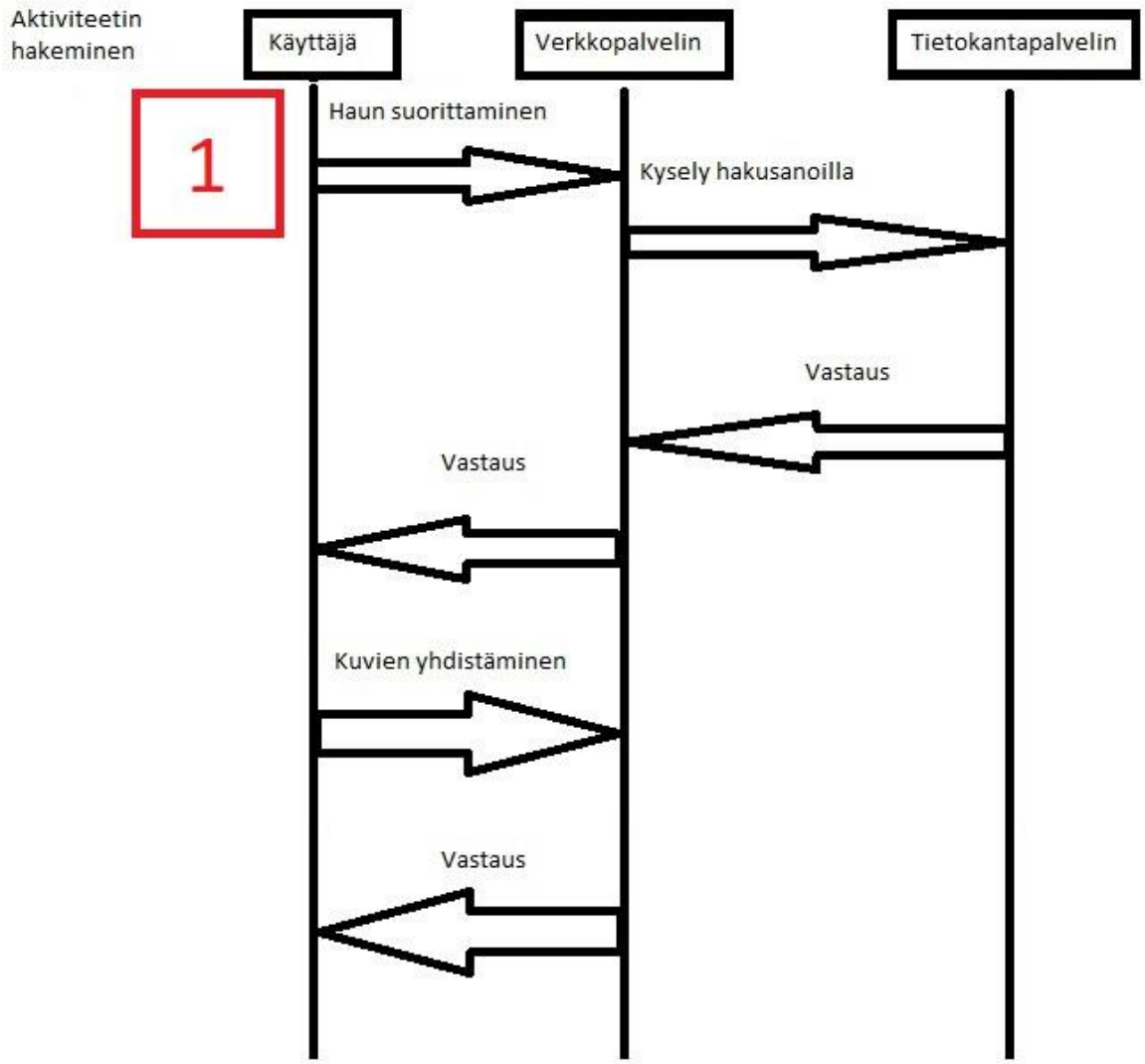
Kuva 13. Otos tietojärjestelmästä

Esikatselukuvien tallentaminen. Viimeinen tehtävä on muodostaa kuvista pienikokoinen esikatselukuva. Esikatselukuvien tarkoitus on parantaa hakutoiminnon suoritusnopeutta vähentämällä ladattavan tiedoston kokoa. Sen sijaan että esikatselukuva muodostettaisiin haun yhteydessä, päädyttiin muodostamaan esikatselukuva aktiviteetin lisäämisen jälkeen. Tämä vähentää hakutoiminnon tehtäviä ja nopeuttaa hakutoiminnon loppuun saattamista. Haittapuolena on, että esikatselukuvan tallentaminen vie muutaman kilotavun tilan verkkopalvelimelta.

Tehtävän suoritettuaan PHP-tiedosto sulkeutuu ja käyttäjän laitteen yhteys verkkopalvelimeen katkaistaan. Yhteyden katkaisun jälkeen aktiviteetin lisääminen on suoritettu.

7.2 Aktiviteettien haku palvelimelta

Toisena esimerkkinä asiakassovelluksen, verkkopalvelimen sekä tietokantapalvelimen keskinäisistä vuorovaikutuksista on aktiviteettien hakeminen palvelimelta. Tapahtumia selvennetään sekvenssikaavion avulla (kuva 14).



Kuva 14. Sekvenssikaavio aktiviteetin hakemisesta

Kuvan 14 kohdassa 1 käyttäjä on Haku-toiminnossa valinnut hakuehdot ja painanut Suorita haku -painiketta. Tällöin asiakassovellus on vaihtanut ruutua ja odottaa palvelimen vastausta. Hakuun liittyvät tiedot lähetetään käyttäjän puhelimesta verkkopalvelimen PHP-tiedostolle.

PHP-tiedoston tehtäviin kuuluu hakukyselyn lähettäminen tietokantapalvelimelle ja hakutulosten toimittaminen asiakassovellukselle.

Ensimmäiseksi käyttäjältä saatujen tietojen avulla muodostetaan esivalmisteltu SQL-kysely. SQL-kysely lähetetään tietokantapalvelimelle, joka etsii tietokannasta hakutietoja vastaavat hakutulokset. Hakutulokset lähetetään PHP-tiedostolle. PHP-tiedosto erottelee hakutulokset merkeillä, jotta asiakassovellus voi ne käsitellä. Tulosten erottelun jälkeen PHP-tiedosto lähettää tulokset asiakassovellukselle.

Asiakassovellus vastaanottaa palvelimelta vastauksena aktiviteettien nimet ja polut opasteisiin sekä kuviin. Asiakassovelluksen on seuraavaksi ladattava palvelimelta kuvat ja opasteet kuvan 7 listaan. Kuvassa 15 on otos Java-koodista, joka jakaa tiedostojen lataamisen viiden aktiviteetin kokoisiin eriin. Koodin kohdassa 1 asiakassovellus lataa viisi aktiviteettia palvelimelta ja lisää ne käyttäjän nähtäville. Kohtaa 1 toistetaan niin kauan, kuin viiden kokoisia eriä on jäljellä. Lopuksi kohdassa 2 asiakassovellus lisää jäljellä olevat aktiviteetit käyttäjän nähtäville.

```
//viiden nippuina näytölle
int tasa=polut.length/5+1;
for(int k=1;k<tasa;k++) {
    polutloput = new String[polut.length - 5*k];
    polutloput2 = new String[polut.length - 5*k];
    for (i = 0; i < 5; i++) {
        polut5[i] = polut[i+((k-1)*5)];
        polut25[i] = polut2[i+((k-1)*5)];
    }
    listBm = getImageBitmap(polut5);
    listOhjeet = getOhjeet(polut25);
    for (i = 0; i < 5; i++) {
        //+((k-1)*5)
        Leikit.add(new Leikki(listOhjeet[i], nimet[i+((k-1)*5)].replaceAll("_", " "), listBm[i]));
    }
Naytolle();
}
//jakojäännös näytölle
for (i=0;i<polut.length-5*(tasa-1);i++){
    polutloput[i]=polut[i+5*(tasa-1)];
    polutloput2[i]=polut2[i+5*(tasa-1)];
}
listBm=getImageBitmap(polutloput);
listOhjeet=getOhjeet(polutloput2);
for (i=0;i<polutloput.length;i++){
    Leikit.add(new Leikki(listOhjeet[i],nimet[i+((tasa-1)*5)].replaceAll("_"," "),listBm[i]));
}
Naytolle();
```

1

2

Kuva 15. Otos Java-koodista

Ratkaisuun, jossa ohjelma hakee kuvat ja ohjeet erikseen verkkopalvelimelta, päädyttiin, koska kymmenien aktiviteettien hakeminen kerralla veisi liikaa aikaa. Ensimmäisen viiden hakutuloksen saaminen käyttäjän ruudulle nopeasti sallii käyttäjän aloittaa selailun aikaisemmin. Vaikka haku olisikin nopeilla yhteyksillä ja kohtalaisen kokoisilla hakutuloksilla nopeaa, tuloksien asteittainen haku palvelimelta auttaa hitaita verkkoyhteyksiä. Esimerkkitapauksessa 30 aktiviteetin haun pilkkominen nopeutti kuvien saamista puhelimen näytölle viidestä sekunnista noin yhteen sekuntiin.

8 JATKOKEHITYS

Ohjelma on valmis julkaistavaksi opinnäytetyön päätteeksi. Joitain lisäkehityksiä on kuitenkin tehtävänä myöhemmissä vaiheissa. Yleisten parannusten lisäksi yksi uusi ominaisuus on kehityksessä.

Suunnitelmissa on lisätä mahdollisuus monivaiheisten aktiviteettien lisäämiseen. Tällä hetkellä ohjelmassa pystyy ainoastaan lisäämään aktiviteettejä, joihin riittää yksi kuva sekä opaste. Monivaiheisessa aktiviteetissa kuvia ja opasteita voisi lisätä useampia. Esimerkkitapaus voisi olla kuntopiiri, jossa jokaiselle piirin liikkeelle olisi oma opasteensa.

9 YHTEENVETO JA POHDINTA

Opinnäytetyön tavoitteena oli toteuttaa mobiilisovellus. Mobiilisovelluksella voi lisätä ja hakea tietoa palvelimen tietojärjestelmästä ja tietokannasta. Työssä tarkasteltiin käytettävyyden suunnittelua ja mobiilialustojen eroavaisuuksia. Työssä käsiteltiin myös sovelluksen toteutus asiakassovelluksen, palvelimen ja käyttöliittymän osilta. Lopuksi kerrottiin ohjelmiston kehittämisestä ja jatkokehityssuunnitelmista.

Opinnäytetyöstä tulokseksi saatu sovellus oli toimivuudeltaan hyvä. Erityisesti intuitiivinen käyttöliittymä oli mieluisa käyttää ja toimiva, joskin hieman itse tehdyn näköinen ulkoasultaan.

Asiakkaan kanssa yhteistyö sujui hyvin. Työn tekijän kokemattomuus ohjelmistokehittäjänä ja useat tahot, joiden kanssa tehtiin yhteistyötä, toi tekemiseen omat haasteensa. Tarpeiden ilmaiseminen olennaisille tahoille onnistuu seuraavalla kerralla paremmin, kun törmää samankaltaisin haasteisiin.

Kaiken kaikkiaan tulos oli onnistunut ja saadun opin määrä suuri. Jokaista onnistunutta tehtyä ratkaisua tukee useat vähemmän onnistuneet, joiden toimimattomuutta selvitettiin perusteellisesti. Mobiilisovellusten ala kasvaa yhä, joten saaduista opeista on hyötyä myös jatkossa.

LÄHTEET

Android Developers. Ei Päiväystä. Publish Your App. [Verkkojulkaisu]. Android Developers. [Viitattu 2.11.2017]. Saatavana: https://developer.android.com/studio/publish/index.html

Android Developers. Ei päiväystä. Dashboards. [Verkkojulkaisu]. Android Developers. [Viitattu 29.10.2017]. Saatavana: https://developer.android.com/about/dashboards/index.html

Android Developers. Ei päiväystä. Meet Android Studio. [Verkkojulkaisu]. [Viitattu 28.10.2017]. Android Developers. Saatavana: https://developer.android.com/studio/intro/index.html

Android Developers. Ei päiväystä. Supporting Tablets and Handsets. [Verkkojulkaisu]. [Viitattu 4.11.2017]. Android Developers. Saatavana: https://developer.android.com/guide/practices/tablets-and-handsets.html

Apache Friends. Ei päiväystä. About. [Verkkojulkaisu]. ApacheFriends.org. [Viitattu 4.11.2017]. Saatavana: https://www.apachefriends.org/about.html

Apple. Ei päiväystä. About App Distribution Workflows. [Verkkojulkaisu]. Apple Inc. [Viitattu 2.11.2017]. Saatavana: https://developer.apple.com/library/content/documentation/IDEs/Conceptual/AppDistributionGuide/Introduction/Introduction.html

Baars, N. 18.10.2017. SQL injection: when a prepared statement is not enough….[Verkkojulkaisu]. [Viitattu 4.11.2017]. Saatavana: https://blog.jdriven.com/2017/10/sql-injection-prepared-statement-not-enough/

Ellingwood, J. 13.8.2013. How To Configure the Apache Web Server on an Ubuntu or Debian VPS. [Verkkojulkaisu].Digital Ocean Inc. . [Viitattu 2.11.2017]. Saatavana: https://www.digitalocean.com/community/tutorials/how-to-configure-the-apache-web-server-on-an-ubuntu-or-debian-vps

Haig, P. 29.4.2013. Have you changed your app title yet?. [Verkkojulkaisu]. Tune. [Viitattu 2.11.2017]. Saatavana: https://www.tune.com/blog/have-you-changed-your-app-title-yet/

Keller, M. 13.2.2017. Geek 101: What Is Jailbreaking?. [Verkkojulkaisu]. PCWorld. [Viitattu 4.11.2017]. Saatavana: https://www.pcworld.com/article/249-091/geek_101_what_is_jailbreaking_.html

Lidwell, W., Holden, K. & Butler, J. 2010. Universal principles of design. [Verkkojulkaisu]. Semantic Scholar. [Viitattu 4.11.2017]. Saatavana: https://pdfs.semanticscholar.org/3527/a9df49cfbcd4ee7d00553b1bfcfd4d79e701.pdf

MariaDB. Ei päiväystä. About MariaDB. [Verkkojulkaisu]. MariaDB Foundation. [Viitattu 4.11.2017]. Saatavana: https://mariadb.org/about/

MySQL. Ei päiväystä. 1.3.1 What is MySQL?. [Verkkojulkaisu]. Oracle Corporation. [Viitattu 4.11.2017]. Saatavana: https://dev.mysql.com/doc/refman/5.7/en/what-is-mysql.html

Netmarketshare. 10/2017. Mobile/Tablet Operating System Market Share. [Verkkojulkaisu]. Net Applications.com. [Viitattu 4.11.2017]. Saatavana: https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=8&qpcustomd=1&qpcd=1300

Sears, R., van Ingen, C. & Gray, J. 2006. To BLOB or Not To BLOB: Large Object Storage in a Database or a Filesystem?. [Verkkojulkaisu]. Redmond: Microsoft Research. [Viitattu 28.10.2017]. Saatavana: https://www.microsoft.com/en-us/research/wp-content/uploads/2006/04/tr-2006-45.pdf

Sefferman, A. 15.3.2016. The App Store Optimization Checklist: Top 10 Tips. [Verkkojulkaisu]. MOZ. [Viitattu 2.11.2017]. Saatavana: https://moz.com/blog/app-store-optimization-checklist

Statcounter. 10/2017. Tablet Operating System Market Share Worldwide. [Verkkojulkaisu]. Statcounter. [Viitattu 4.11.2017]. Saatavana: http://gs.statcounter.com/os-market-share/tablet/worldwide

Sun Microsystems, Inc. 1997. What is The Java Language Environment. [Verkkojulkaisu]. Oracle Corporation. [Viitattu 28.10.2017]. Saatavana: http://www.oracle.com/technetwork/java/intro-141325.html

Tervakari, A-M. 17.12.2008. Ohjelmistotuotannon mallit. [Verkkojulkaisu]. TTY. [Viitattu 2.11.2017]. Saatavana: https://hlab.ee.tut.fi/hmopetus/vpsist-oppimateriaali/4-menetelmia-ja-malleja/4-3-suunnittelumalleja/4-3-1-ohjelmistotuotannon-malli.html

Tervakari, A. 14.1.2014. Ketteriä menetelmiä. [Verkkojulkaisu]. TTY-Piiri. [Viitattu 21.11.2017]. Saatavana: https://iislab.ee.tut.fi/piiri/content/231-ketteri%C3%A4-menetelmi%C3%A4

The PHP Group. Ei päiväystä. What is PHP?. [Verkkojulkaisu]. The PHP Group. [Viitattu 4.11.2017]. Saatavana: http://php.net/manual/en/intro-whatis.php

W3Schools. Ei päiväystä. Introduction to XML. [Verkkojulkaisu]. W3Schools. [Viitattu 4.11.2017]. Saatavana: https://www.w3schools.com/xml/xml_whatis.asp

Yarmosh, K. 6.5.2015. Android vs iOS: Which platform to build for first?. [Verkkojulkaisu]. Savvyapps. [Viitattu 2.11.2017]. Saatavana: https://savvyapps.com/blog/android-vs-ios-which-platform-to-build-for-first