ARTIKKELI Lumen 1/2019

Digitaalista etäisyyksien hallintaa: Kokemuksia mobiilisovellusten kehittämisestä Lapin AMK:n projekteissa

Tuomas Valtanen, insinööri (ylempi AMK), projektipäällikkö, Arktiset luonnonvarat ja talous, Digitaaliset ratkaisut, ohjelmistotekniikan laboratorio pLAB, Lapin ammattikorkeakoulu

Juhani Kuru, insinööri (AMK), projekti-insinööri, Arktiset luonnonvarat ja talous, Digitaaliset ratkaisut, ohjelmistotekniikan laboratorio pLAB, Lapin ammattikorkeakoulu

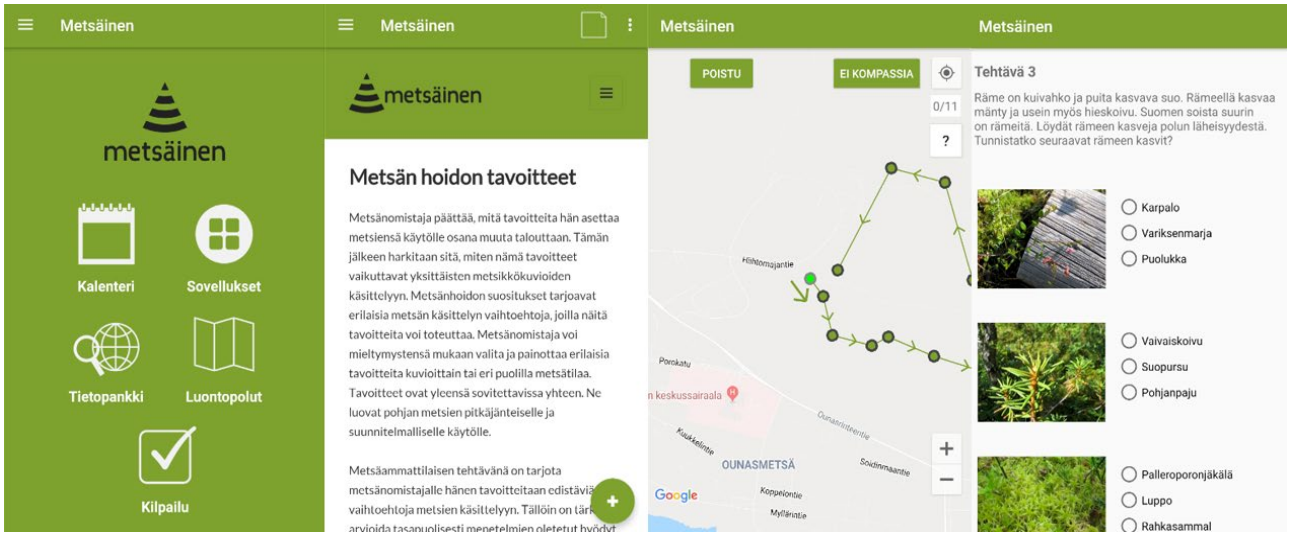
Mikko Pallas, insinööri (AMK), projekti-insinööri, Arktiset luonnonvarat ja talous, Digitaaliset ratkaisut, ohjelmistotekniikan laboratorio pLAB, Lapin ammattikorkeakoulu

Mikko Pajula, tieto- ja viestintätekniikan opiskelija, projektityöntekijä, Arktiset luonnonvarat ja talous, Digitaaliset ratkaisut, ohjelmistotekniikan laboratorio pLAB, Lapin ammattikorkeakoulu

Asiasanat: tieto- ja viestintätekniikka, ohjelmistotekniikka, mobiilisovellukset, web-sovellukset, projektisuunnittelu, projektityö

Mobiilisovellusten määrä ja kysyntä on ollut jatkuvassa kasvussa jo useiden vuosien ajan (MindSea 2018). Samasta syystä niiden kysyntä on kasvanut tasaisesti myös Lapin AMK:ssa ohjelmistotekniikan laboratorio pLABin projekteissa. Tällaisia projekteja ovat olleet esimerkiksi Metsäinen Innostaa, MTI Match, LuontoRovaniemi, Hilumi 3D, ArkTori, Dwell sekä Yritystä pelissä -hankkeet. Kehitetyt mobiilisovellukset ovat sisältäneet mm. mobiililuontopolkuja, osaamisen kehittämisen työkaluja, yhteisöllisyyttä edistäviä ominaisuuksia sekä visualisointeja. Yleensä lopputuloksena on Android-sovellus, koska Androidin markkinaosuus älylaitteissa on yli 85% (IDC 2018). Joissain tapauksissa vastaava mobiilisovellus on tehty myös iOS:lle (iPhone/iPad).

Mobiilisovellustarpeiden lisääntyessä on hyvä tietää, miten mobiilisovelluksen kehittäminen eroaa tyypillisen web-sovelluksen kehittämisestä. Mobiilisovelluksen budjetti suunnitellaan projekteissa usein samalla kaavalla kuin tavanomaisissa web-sovelluksissa, mikä voi tuoda suuria haasteita hankkeen toteutuksessa. Tässä artikkelissa viitataan pääsääntöisesti Mobiilisti metsään- ja Metsäinen innostaa–hankkeissa kehitettyyn Metsäinen–mobiilisovellukseen, sekä sen kehitystyössä esille nousseisiin haasteisiin. Pohdimme samalla myös sitä, mitä tulisi ottaa huomioon uusia mobiilisovellusprojekteja suunnitellessa.



Kuva 1. Kuvakaappauksia Metsäinen –mobiilisovelluksesta (Android)

Vertailukohteena web-sovellukset

Lapin AMK:n projekteissa mobiilisovelluksen rooli on usein toimia eräänlaisena modernina vaihtoehtona web-sovellukselle. Mobiilisovelluksen etu on sen mobiiliudessa – älypuhelin on lähes aina käyttäjän mukana, ja esimerkiksi käyttäjän sijaintia ja kameraa voidaan hyödyntää mobiilisovelluksessa luovemmin kuin perinteisessä web-sovelluksessa (Summerfield 2019). Vaikka käyttötarkoitus voi molemmissa olla sama, on mobiilikehitys monelta osin erilaista web-kehitykseen verrattuna.

Web-sovellusten taustalla olevat tekniikat ovat vanhempia ja kypsempiä kuin mobiilisovelluksissa. Web-sovellusten nousukautena voitaneen pitää 90-luvun loppupuolta, kun taas modernit älypuhelimet ovat yleistyneet vasta 2010-luvulla. Mm. tämän vuoksi mobiilitekniikat kehittyvät tällä hetkellä nopeammin kuin web-tekniikat. Osaamisen näkökulmasta tämä on haastavaa, sillä mobiiliosaaminen vanhenee nopeasti ja valmiita mobiilisovelluksia pitää yleensä säännöllisesti korjata. Web-sovelluksetkin vaativat säännöllistä ylläpitoa, mutta ne voivat pysyä käyttökunnossa vuosiakin vain pienellä ylläpidolla. (DDI Development 2015; Bouwkamp 2016; de Looper 2018; Khan 2018a; Saeed 2018; Costello 2019; Goel 2019.)

Lapin AMK:n projekteissa toteutetaan yleensä joko perinteinen tietokoneella käytettävä web-sovellus tai erillinen mobiilisovellus. Toisaalta projektien web-sovelluksissa on vain harvoin erityisvaatimuksena mobiiliyhteensopiva käyttöliittymä. Mobiiliyhteensopivan web-sovelluksen toteuttaminen ei tyypillisissä projekteissa vaadi yleensä suurta lisäresurssia, vaan se voi olla tehtävissä jopa reilun viikon työpanoksella. Jos projektissa on tarkoituksena ensisijaisesti vain jakaa kätevästi tietoa, voi mobiiliyhteensopiva web-sovellus olla kustannustehokkaampi vaihtoehto mobiilisovellukselle. Poikkeuksena tähän on mobiili- ja työpöytäsovellusta matkivat web-sovellukset, joiden kehittäminen vaatii yleensä suuremman työpanoksen verrattuna pelkkään mobiiliyhteensopivuuteen. (LinkAssistant 2019.)

Mobiilisovellusten kehittäminen projekteissa

Viimeaikaisten projektien perusteella mobiilisovellusten kehittäminen vaatii enemmän työaikaa kuin ominaisuuksiltaan vastaava web-sovellus. Työaikaa kuluu erityisesti kehitettävän mobiilisovelluksen testaamiseen. Esimerkiksi Androidissa on erittäin suuri ja vaihteleva aktiivinen laitekanta, jossa eri valmistajia, malleja sekä ohjelmistoversioita on käytössä useita erilaisia. Erot eri laitteiden välillä voi olla suuriakin, joista osa käsittelee puhelimen kameraa, antureita, virrankäyttöä sekä käyttöliittymää eri tavoilla. Pahimmissa tapauksissa osa mobiilisovelluksen ominaisuuksista pitää ohjelmoida ja testata useaan kertaan, jotta toimivuus eri puhelimissa voidaan taata. (ks. Gilmore 2018)

iOS–sovelluksissa laitekanta on pieni, koska niiden ainut valmistaja on Apple. Toisaalta pieni markkinaosuus tuo mukanaan myös sen, että erikoisemmat ominaisuudet ja visuaaliset efektit voivat olla työläitä toteuttaa, koska niihin ei löydy aina valmiita ratkaisuja. Androidissa valmiin avun löytyminen on todennäköisempää suuremman markkinaosuuden ja kehittäjäyhteisön vuoksi. (ks. Averyanov 2017; MindSea 2018.)

Mitä vanhempia mobiililaitteita halutaan tukea, sitä enemmän ohjelmoitavaa ja testattavaa kehitysvaiheessa on. Jopa pelkkä näytön kääntäminen mobiilisovelluksessa voi aiheuttaa korjaustarpeita johtuen tavasta, jolla älylaitteet käsittelevät sovellusten tuottamaa kuormitusta. Erityisesti puhelimen sijaintia (GPS) hyödyntävät sovellukset ovat työläitä toteuttaa, koska niiden testaaminen pitää suurilta osin tehdä ulkotiloissa, mikä kuluttaa työresurssia ongelmatilanteissa runsaasti. Pahimmassa tapauksessa korjaamattomat ongelmat aiheuttavat sen, että mobiilisovellus hidastaa jatkuvasti puhelinta käyttäjän tietämättä. Kaiken kaikkiaan

mobiilisovelluksen tarvitsema testaustyöpanos on yleensä moninkertainen verrattuna web-sovellukseen. (ks. Gilmore 2018; Roy 2019).

Toinen suuri haaste mobiilisovellusten kehittämisessä on mobiilitekniikoiden jatkuva kehittyminen. Esimerkiksi Metsäinen-mobiilisovellus julkaistiin kesällä 2017, jonka jälkeen jo syksyllä 2018 piti toteuttaa ensimmäinen suuri muutospäivitys. Ilman tätä päivitystä Metsäinen olisi automaattisesti poistettu Google Play-palvelusta muuttuneiden standardien vuoksi (ks. Google Developers 2019a). Nämä muutospäivitykset ovat sellaisia, joita ei pystytä hankesuunnittelussa ennakoimaan. Tästä syystä mobiilisovellusten jatkuvaan ylläpitoon pitää käyttää enemmän työaikaa kuin vastaavaan web-sovellukseen.

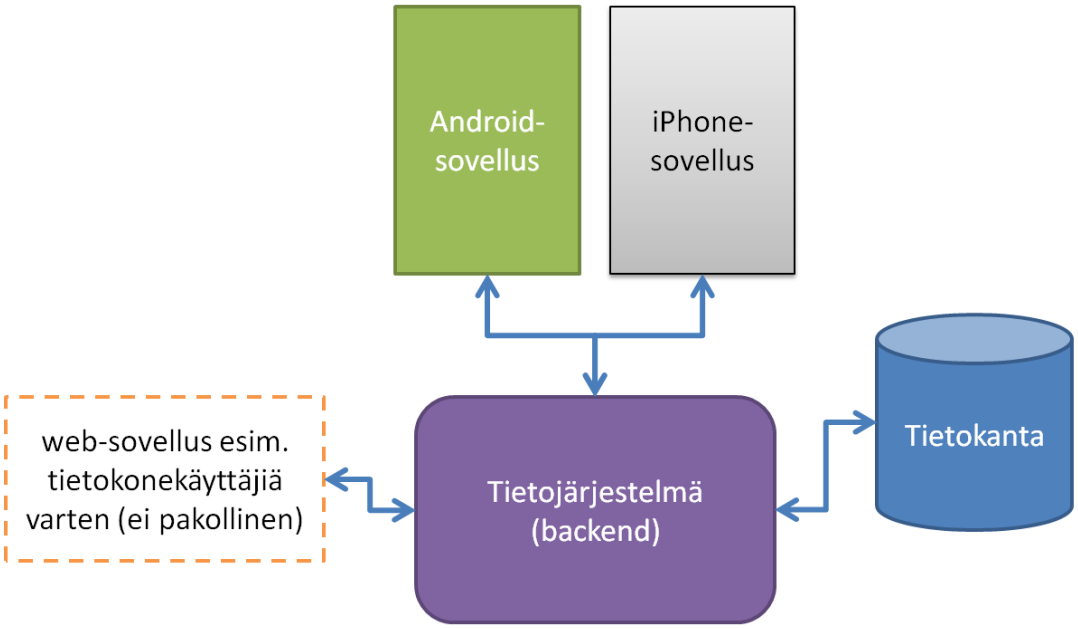
Mobiilisovelluksen päivitystarpeeseen vaikuttaa oleellisesti myös sen sisältämät ominaisuudet. Tämä johtuu siitä, että mobiililaitteiden akkukestoa pyritään jatkuvasti kehittämään, mikä vaikuttaa mm. sijainti- ja ilmoituspalvelun hyödyntämiseen mobiilisovelluksessa. Mikäli mobiilisovellus uudemmissa Android-puhelimissa tarvitsee reaaliaikaisia viestejä vaikkapa sijainnista, vaatii se huomattavan paljon lisäohjelmointia (Google Developers 2019b ; Google Developers 2019c.)

Mobiilisovelluksen julkaisemiseen pitää myös varata tarpeeksi aikaa. Jos julkaistava mobiilisovellus ei läpäise Google Play:n ja Apple App Storen julkaisutestejä, pitää sovellusta korjata niin kauan, kunnes sovellus kelpaa julkaistavaksi. Varsinkin App Store on tarkka siitä, että sovellus noudattaa tiettyjä käytäntöjä ennen julkaisua. (Google Developers 2019d; Apple Developer 2019; Aleksandrova 2018.)

Myös käyttöliittymän rakentaminen mobiilisovelluksissa vie enemmän aikaa kuin web-sovelluksissa. Eri kokoisille näytöille pitää pahimmassa tapauksessa tehdä Android-sovelluksissa useita lisäkäyttöliitymiä. iOS-sovelluksissa sen sijaan käyttöliittymän ohjelmointia varten on vähemmän työkaluja, minkä vuoksi erikoisemmat käyttöliittymät ovat hitaita toteuttaa.

Jotkut ominaisuudet ovat taas työläitä toteuttaa siksi, että Android- ja iOS-laitteet haluavat mobiilisovellusten toimivan tietyllä tavalla. Tämä näkyy esimerkiksi monikielisissä sovelluksissa, joissa Android haluaa oletuksena käyttää puhelimen käyttämää kieltä. Jos kieli halutaan vaihtaa ohjelmasta käsin painikkeella, vaatii se huomattavasti enemmän lisäohjelmointia. Joitain vaikeita ominaisuuksia voidaan tehdä epävirallisten liitännäisten avulla, mutta niiden toimivuus tulevaisuudessa voi olla haaste. (ks. Berezanskyi 2017; Neuhaus 2017.)

Koska mobiilisovellus toimii usein eräänlaisena web-sovelluksen korvikkeena, tarvitsee se usein myös taustalleen erillisen tietojärjestelmäsovelluksen, jossa mobiilisovelluksen tietoja varastoidaan ja käsitellään. Tämä tietovarasto on pääsääntöisesti oma erillinen taustalla toimiva web-sovelluksensa, jota mobiilisovellus hyödyntää. Tämä kasvattaa omalta osaltaan mobiilisovelluksen kehityskustannuksia. (Tea 2017.)



Kuva 2. Mobiilisovellukset ja tyypilliset taustajärjestelmät

Vaihtoehtona monialustatekniikat

Yksi vaihtoehto mobiilisovellusten kehittämiseen on käyttää monialustatekniikoita. Tällaisia monialustatekniikoita ovat mm. PhoneGap, Xamarin ja Flutter. Monialustatekniikoiden perusajatus on mahdollistaa sekä Android- että iOS-sovelluksen kehittäminen samalla ohjelmakoodilla mahdollisimman pitkälle. Toisaalta mobiilisovelluksen ulkoasuun liittyvät ominaisuudet pitää silti yleensä ohjelmoida molemmille alustoille erikseen. Jos ohjelmassa on monimutkainen käyttöliittymä, on monialustatekniikasta tuleva hyöty yleensä pienempi. Lisäksi jotkut ominaisuudet voivat olla haastavia toteuttaa, kuten karttapalvelut ja kolmannen osapuolen tekniikat. Monialustatekniikat myös päivittyvät hitaammin, koska ne joutuvat odottamaan mobiilialustojen päivittymistä. (ks. Barnett 2017; Redbytes 2017; SteelKiwi Inc. 2018.)

Mobiiliyhteensopivien verkkosivujen rinnalle on noussut vielä toistaiseksi uusi tekniikka PWA (progressive web apps), joka vapaasti suomennettuna tarkoittaa edistyksellisiä web-sovelluksia. Käytännössä PWA-sovellukset ovat tavanomaisia web-sovelluksia, mutta ne esiintyvät käyttäjän älypuhelimessa mobiilisovellusten kaltaisesti. Tällaisia sovelluksia ovat esimerkiksi VR:n Junat kartalla -palvelu sekä HSL:n reittiopaspalvelu. Kun käyttäjä siirtyy mobiililaitteellaan ensimmäistä kertaa em. verkkosivuille, ehdottaa puhelin web-palvelun lisäämistä puhelimen sovellusvalikkoon. Tämä ominaisuus tosin toimii vain uusimmissa älylaitteissa, minkä vuoksi sillä ei vielä tällä hetkellä voi korvata mobiilisovelluksia. PWA:lla ei myöskään välttämättä pysty aina käyttämään kaikkia samoja ominaisuuksia joita varsinaisissa mobiilisovelluksissa on mahdollista käyttää (Khan 2018b).

Pohdinta

Mobiilisovelluksen kehittäminen ei ole projektitoiminnassa aivan niin suoraviivaista kuin web-sovelluksen kehittäminen. Karkeasti arvioituna esimerkiksi Android-mobiilisovelluksen kehittämiseen pitää varata melkein kaksinkertainen työresurssi web-sovellukseen verrattuna. Toisaalta mobiilipalvelut ovat se tapa, millä palveluita halutaan tänä päivänä käyttää, oli kyse sitten mobiilisovelluksesta tai mobiiliverkkosivusta (MindSea 2018). Tästä syystä mobiilisovelluksen kehittäminen on perusteltua silloin, kun tuotettavan palvelun käytettävyys on tärkeää.

Web-sovelluksen ja mobiilisovelluksen ero työajassa voidaan suuntaa antavasti arvioida seuraavalla taulukolla:

Työn tarve (karkea arvio)	**Mobiiliyhteensopiva web-sovellus**	**Android- ja iOS-sovellukset**	**Monialustasovellus (Android ja iOS)**
Sovelluksen kehittäminen	3-4kk	5kk+4kk (voi tarvita lisäksi web-tietojärjestelmän, +1kk)	6-8kk (oletus, että monialustatekniikka soveltuu projektiin (voi tarvita lisäksi web-tietojärjestelmän, +1kk)
Ylläpidon tarve / vuosi	0-3 työpäivää	3-6 työpäivää	2-4 työpäivää

Taulukko 1. Suuntaa antava työmääräarvio keskimääräisistä mobiilisti käytettävien sovellusten kehittämisestä ja ylläpidosta. Sovelluksen ominaisuudet ja laajuus vaikuttaa oleellisesti todelliseen työmääräarvioon.

Kaikki tämä ei kuitenkaan tarkoita sitä, etteikö mobiilisovellusten kehittäminen olisi projektitoiminnassa järkevä vaihtoehto. Kyse on enemmänkin siitä, miten projektien osapuolet sitoutuvat mobiilisovelluksen jatkokehittämiseen, ja millaisia teknologisia ratkaisuja projektissa kannattaa toteutuksen valossa suosia. Mobiilisovelluksen kehittämiseen tulee myös varata realistinen budjetti, ettei projektin toteutus vaarannu. Mobiilisovelluksia on mahdollista kehittää myös kustannustehokkaasti, kunhan pahimmat sudenkuopat tiedostetaan, ja projektin mobiilisovellus kartoitetaan jo suunnitteluvaiheessa mahdollisimman tarkasti.

Metsäinen on kehitetty Metsäkeskuksen ja Lapin AMK:n yhteistyönä. Metsäinen Innostaa –hankkeen on rahoittanut Maaseuturahasto.

Metsäinen –mobiilisovelluksen voi ladata Google Play –palvelusta! https://play.google.com/store/apps/details?id=com.laplanduas.plab.metsainen

Lapin AMK:n hankkeita, joissa on joko tuotettu tai kirjoitushetkellä tuotetaan mobiilisovelluksia:

ArkTori:

https://www.lapinamk.fi/fi/Yrityksille-ja-yhteisoille/Lapin-AMKin-hankkeet?RepoProject=521710

Dwell:

https://www.lapinamk.fi/fi/Yrityksille-ja-yhteisoille/Lapin-AMKin-hankkeet?RepoProject=521834

LuontoRovaniemi:

https://www.lapinamk.fi/fi/Yrityksille-ja-yhteisoille/Lapin-AMKin-hankkeet?RepoProject=521707

Metsäinno:

https://www.lapinamk.fi/fi/Yrityksille-ja-yhteisoille/Lapin-AMKin-hankkeet?RepoProject=521709

Yritystä Pelissä:

https://www.lapinamk.fi/fi/Yrityksille-ja-yhteisoille/Lapin-AMKin-hankkeet?RepoProject=521711

Lähteet

Aleksandrova, M. 2018. How to Publish Your App on App Store and Google Play? A Comprehensive Go-to-Market Guide. Eastern Peak 3.1.2018. Viitattu 10.4.2019 https://easternpeak.com/blog/how-to-publish-your-app-on-app-store-and-google-play-comprehensive-go-to-market-guide/.

Apple Developer 2019. Submit your apps today. Viitattu 10.4.2019 https://developer.apple.com/app-store/submissions/.

Averyanov, K. 2017. Creating custom UI components in iOS with Swift. Medium 16.5.2017. Viitattu 10.4.2019 https://medium.com/@Kirillzzy/create-a-custom-button-for-ios-e81026bd6148.

Barnett, J. 2017. Can PhoneGap Satisfy Your App Development Needs? Iflexion 10.6.2017. Viitattu 10.4.2019 https://www.iflexion.com/blog/can-phonegap-satisfy-app-development-needs.

Berezanskyi, Y. 2017. How to change the language on Android at runtime and don't go mad. ProAndroidDev 6.8.2017. Viitattu 10.4.2019. https://proandroiddev.com/change-language-programmatically-at-runtime-on-android-5e6bc15c758.

Bouwkamp, K. 2016. The 9 Most In-Demand Programming Languages of 2016. Coding Dojo 27.1.2016. Viitattu 10.4.2019 https://www.codingdojo.com/blog/9-most-in-demand-programming-languages-of-2016

Chandra, S. 2017. Top 6 Leading PHP Frameworks For 2017. SAMWebStudio 21.9.2017. Viitattu 10.4.2019 https://www.samwebstudio.com/blog/post/top-6-leading-php-frameworks-for-2017.

Costello, S. 2019. The History of iOS, from Version 1.0 to 12.0. Lifewire 25.2.2019. Viitattu 10.4.2019 https://www.lifewire.com/ios-versions-4147730.

de Looper, C. 2018. From Android 1.0 to Android 9.0, here's how Google's OS evolved over a decade. Digital Trends 23.10.2018. Viitattu 10.4.2019 https://www.digitaltrends.com/mobile/android-version-history/.

DDI Development 2015. Top 10 the best web development frameworks. Viitattu 10.4.2019 http://ddi-dev.com/blog/programming/top-10-best-web-development-frameworks/.

Gilmore, L. 2018. Your step-by-step mobile application testing process. Testlio 17.9.2018. Viitattu 10.4.2019 https://testlio.com/blog/step-step-mobile-application-testing-process/.

Goel, A. 2019. Top 10 Web Development Frameworks in 2019. hackr.io 19.3.2019. Viitattu 10.4.2019 https://hackr.io/blog/top-10-web-development-frameworks-in-2019.

Google Developers 2019a. Meet Google Play's target API level requirement. Viitattu 10.4.2019 https://developer.android.com/distribute/best-practices/develop/target-sdk.

Google Developers 2019b. Android 8.0 Behavior Changes. Viitattu 10.4.2019 https://developer.android.com/about/versions/oreo/android-8.0-changes.

Google Developers 2019c. Create a Notification. Viitattu 10.4.2019 https://developer.android.com/training/notify-user/build-notification.html.

Google Developers 2019d. Publish your app. Viitattu 10.4.2019 https://developer.android.com/studio/publish.

IDC 2018. Smartphone Rankings Shaken Up Once Again as Huawei Surpasses Apple, Moving into Second Position While Overall Market Declined 1.8% in Q2 2018, According to IDC. Viitattu 10.4.2019 https://www.idc.com/getdoc.jsp?containerId=prUS44188018.

Ionos 2019. Web programming languages: the best languages for web development. Ionos 11.3.2019. Viitattu 10.4.2019 https://www.ionos.com/digitalguide/websites/web-development/web-programming-languages/.

Khan, A. 2018a. Top 10 Web Development Frameworks for 2018-19. Appy Pie 25.8.2018. Viitattu 10.4.2019 https://www.appypie.com/top-10-web-development-frameworks-for-2018-19.

Khan, U. 2018b. The Pros and Cons of Progressive Web Apps. Clutch 26.6.2018. Viitattu 10.4.2019 https://clutch.co/app-developers/resources/pros-cons-progressive-web-apps.

LinkAssistant 2019. How to make your website mobile friendly? A 5-step Google mobile friendly optimization plan for your website! Viitattu 10.4.2019 https://www.link-assistant.com/news/mobile-seo.html.

MindSea 2018. 25 Mobile App Usage Statistics To Know in 2019. Viitattu 10.4.2019 https://mindsea.com/app-stats/.

Neuhaus, J. 2017. 9 Steps: Choosing a tech stack for your web application. Medium 22.8.2017. Viitattu 10.4.2019 https://medium.com/unicorn-supplies/9-steps-how-to-choose-a-technology-stack-for-your-web-application-a6e302398e55.

Redbytes 2017. Pros and Cons of Xamarin App Development. Viitattu 10.4.2019 https://www.redbytes.in/pros-cons-xamarin-app-development/.

Roy, A. 2019. What you need to know about Android app memory leaks. TechBeacon 2019. Viitattu 10.4.2019 https://techbeacon.com/app-dev-testing/what-you-need-know-about-android-app-memory-leaks.

Saeed, A. 2018. Here Are The Ten Best Programming Languages to learn in 2019. Coding Infinite 22.12.2018. Viitattu 10.4.2019 https://codinginfinite.com/best-programming-languages-to-learn-2019/.

SteelKiwi Inc. 2018. Flutter: Pros and Cons for Seamless Cross Platform Development. Hacker Noon 11.9.2018. Viitattu 10.4.2019 https://hackernoon.com/flutter-pros-and-cons-for-seamless-cross-platform-development-c81bde5a4083.

Summerfield, J. 2019. Mobile Website vs. Mobile App: Which is Best for Your Organization? Human Service Solutions 2019. Viitattu 10.4.2019 https://www.hswsolutions.com/services/mobile-web-development/mobile-website-vs-apps/.

Tea, M. 2017. A Massive Guide to Building a RESTful API for Your Mobile App. Savvy Apps 19.6.2017. Viitattu 10.4.2019 https://savvyapps.com/blog/how-to-build-restful-api-mobile-app.