Henri Willman

PROGRESSIIVISET VERKKOSOVELLUKSET JA NIIDEN SOVELLUSALUEET

JYVÄSKYLÄN YLIOPISTO
INFORMAATIOTEKNOLOGIAN TIEDEKUNTA
2022

TIIVISTELMÄ

Willman, Henri
Progressiiviset verkkosovellukset ja niiden sovellusalueet
Jyväskylä: Jyväskylän yliopisto, 2022, 26 s.
Tietojärjestelmätiede, kandidaatintutkielma
Ohjaaja: Lampi, Anna

Progressiiviset verkkosovellukset ovat selaimen kautta toimivia sovelluksia, jotka pyrkivät yhdistämään perinteisten natiivisovellusten sekä verkkosivujen hyödyllisimpiä ominaisuuksia. Vaikka aihe on relevantti, ja se on saanut tieteellistä huomiota, optimaalisia progressiivisten verkkosovelluksien sovellusalueita ei ole juurikaan tutkittu. Tällä tutkielmalla pyritään paikkaamaan tätä puutetta kartoittamalla ensin progressiivisten verkkosovellusten etuja ja haasteita, ja sen jälkeen selvittämällä mahdollisia sovellusalueita näihin perustuen.

Tutkielmassa havaitaan useita relevantteja hyötyjä koskien esimerkiksi tietoturvaa, käytettävyyttä ja jakelumahdollisuuksia, mutta samalla todetaan, että progressiivisilla verkkosovelluksilla on myös muutamia haasteita. Nämä haasteet linkittyvät havaittuihin etuihin, tai johtuvat suoraan niiden tarjoamista ominaisuuksista. Kartoitettujen etujen ja haasteiden perusteella todetaan, että ei voida eksplisiittisesti sanoa, mitkä ovat progressiivisten verkkosovellusten optimaalisia sovellusalueita, vaan sopivia kohteita tulee tarkastella sovelluksen vaatimukset ja käyttäjät edellä. Tästä huolimatta tutkielmassa esitellään kaksi potentiaalista kohdetta progressiivisen toteutustavan hyödyntämiselle. Tätä kontribuutiota voivat yritykset ja yksityishenkilöt käyttää lähes sellaisenaan sovellustyyppiä valitessa.

Avainsanat: progressiivinen verkkosovellus, PWA, web-sovelluskehitys, service worker

ABSTRACT

Willman, Henri
Progressive web-applications and their main applications
Jyväskylä: University of Jyväskylä, 2022, 26 pp.
Information Systems, Bachelor's Thesis
Supervisor: Lampi, Anna

Progressive web-applications are browser-based applications, which aim to combine the most useful and desired features of both native, and web-applications. Even though the subject of progressive web-applications is relevant, and it has gained academic attention, most optimal use-cases for them are yet to be researched. This study attempts to address this need, by focusing on the advantages and disadvantages of progressive web-applications and combining them to find out possible use-cases. This study unveils many relevant advantages concerning e.g., cybersecurity, usability, and distribution. Some disadvantages are also found, but the most interesting part is, that these disadvantages have a direct link to the advantages found earlier. By analysing the advantages and disadvantages, it can be said, that we cannot explicitly state which use-cases are optimal for progressive web-applications. Rather, when figuring out whether progressive web-application-standard is suitable for a certain use-case, the needs and users of the applications must be considered. In this study, however, we present two potential use-cases in which progressive web-application-standard could be used. This contribution can be used by both companies and private persons when choosing the type of application.

Keywords: progressive web-application, PWA, web-development, service worker

KUVIOT

TAULUKOT

SISÄLLYS

KÄSITEHAKEMISTO

API-kutsu	Sovelluksen tai selaimen lähettämä tietopyyntö ohjelmistorajapinnalle (API).
Cache	Välimuisti, jonne voidaan tallentaa muun muassa ohjelmistorajapinnan lähettämiä vastauksia API-kutsuille. Mahdollistaa offline-käytön.
GET-pyyntö	Sovelluksen tai selaimen lähettämä tietopyyntö ohjelmistorajapinnalle (API). Esimerkiksi *hae www.google.com*
HTTPS	*HyperText Transfer Protocol.* HTTP- ja TLS/SSL-protokollien yhdistelmä. Käytetään suojaamaan tietoa käyttäjän ja palvelimen välillä.
Natiivisovellus	Tietylle laitteelle rakennettu sovellus. Ei siis lähtökohtaisesti toimi muilla laitteilla.
Push-ilmoitus	*engl. push notification* Ilmoitus, jolla sovellus viestii käyttäjilleen.
PWA	*engl. progressive web-application.* Selainpohjainen toteutustapa sovellukselle. Sisältää ominaisuuksia, joita on tavallisesti ollut vain natiivisovelluksissa.
Responsiivinen web-suunnittelu	*engl. responsive web-design* Suunnittelutapa, jossa web-sovellus skaalautuu käyttäjän laitteeseen sopivaksi.
Service worker	Välityspalvelin (proxy) sovelluksen, selaimen ja verkkoyhteyden välillä. Mahdollistaa API-kutsujen tallentamisen.

TLS/SSL	*Transport Layer Security / Secure Sockets Layer*. Salausprotokolla, jolla tietoliikennettä suojataan.

1 JOHDANTO

Progressiiviset verkkosovellukset (PWA) ovat suhteellisen uusi aihe sovelluskehityksessä, ja ne tarjoavat käyttäjälleen ominaisuuksia kuten alustariippumattomuuden, natiivisovelluksen tuntuman, sekä offline-käytön mahdollisuuden (Khan, Al-Babi, Al-Kindi, 2019; Loreto, Braga, Peixoto, Machado & Abelha, 2018). Käytännössä tämä tarkoittaa sitä, että progressiiviset verkkosovellukset tarjoavat natiivisovelluksen kaltaisen käyttökokemuksen - joko mobiilissa tai tietokoneella – vaikka kaikki toiminta tapahtuukin selaimen sisällä (Lee, Kim, Park, Shin & Son, 2018).

Progressiiviset verkkosovellukset tarjoavat useita hyödyllisiä ominaisuuksia. PWA-toteutus saa sovelluksen esimerkiksi toimimaan hieman nopeammin kuin suoraan selaimelta käytettäessä (Khan ym., 2019), eikä käyttäjän tarvitse navigoida itseään sovelluksen sivuille, vaan se löytyy suoraan hänen työpöydältään tai kotinäytöltään.

Vaikka progressiiviset verkkosovellukset toimivat selainperäisesti, tarjoavat ne silti natiivisovellusten tapaan esimerkiksi push-ilmoituksia ja monipuolisia kosketusominaisuuksia (Khan ym., 2019). Lisäksi ne toimivat tehokkaammin huonolla nettiyhteydellä sovellusten selainversioihin verrattuna, tarjoten käyttömahdollisuuden myös alueille, joissa internet-yhteyden kanssa on ongelmia (Khan ym., 2019). Lisäksi progressiiviset verkkosovellukset ovat olleet viime aikoina relevantteja, sillä niiden hyödyntäminen säästää kehittäjiltä aikaa ja rahaa (Magomadov, 2020). Sovellusta ei PWA-toteutustavan avulla tarvitse toteuttaa useampaan kertaan – esimerkiksi Android- sekä iOS-käyttöjärjestelmille - vaan sama sovellus taipuu lähes jokaiseen eri laitteeseen. Ne eivät myöskään vie käyttäjiltään juurikaan tallennustilaa selainperäisen toiminnan takia. Vaikka käyttökelpoisia ominaisuuksia on runsaasti, on kuitenkin syytä selvittää, sopiiko PWA-toteutus todellisuudessa kaikenlaisiin sovelluksiin, vai toimiiko se paremmin vain tietynlaisten sovellusten parissa. Vaikka progressiivisia verkkosovelluksia on yleisesti tutkittu runsaasti, optimaalisista sovellusalueista on vain vähän tieteellistä tutkimusta. Tämän tutkielman tarkoitus on siis koota tätä tutkimusta yhteen, ja tarkastella sitä kautta mahdollisia sovellusalueita.

Tässä tutkielmassa tarkastellaan progressiivisten verkkosovellusten etuja ja haasteita, ja pyritään niiden avulla selvittämään mahdollisia käyttökohteita. Keskeisiä tutkimuskysymyksiä ovat:

- Millaisia etuja PWA:n hyödyntämisellä on?
- Millaisia haasteita PWA:n hyödyntämisessä on?
- Mitkä voisivat olla PWA:n sovellusalueita hyötyjen ja haittojen perusteella?

Pelkkä progressiivisten verkkosovellusten hyötyjen tutkiminen voi olla tutkimuksen aihealueena liian rajattu, ja se voi myös aiheuttaa tutkimuksen tekijälle liian puolueellisen näkökulman progressiivisten verkkosovellusten mahdollisesta paremmuudesta. Tästä syystä on hyvä tarkastella myös progressiivisten verkkosovellusten haasteita. Luonnollinen jatkumo hyötyjen ja haasteisten tarkastelulle ovat mahdolliset sovellusalueet. Tutkimuksen perimmäinen tavoite on pyrkiä tuottamaan aito kontribuutio aiheesta, jonka avulla voidaan paremmin kartoittaa, missä tilanteissa progressiivisia verkkosovelluksia olisi syytä hyödyntää. Tutkielman avulla pyritään myös siihen, että progressiivista toteutustapaa ei kartettaisi siitä syystä, ettei aiheesta löydy tarpeeksi tutkimusta.

Tutkimusmenetelmä tässä työssä on kuvaileva kirjallisuuskatsaus, ja siihen liittyvää lähdekirjallisuutta haettiin pääosin Google Scholar- sekä JYKDOK-tietokannoista. Lisäksi oli tarpeellista tukeutua myös aiheen viralliseen dokumentaatioon (MDN Web Docs), jotta tietyistä faktoista, kuten ominaisuuksien yhteensopivuuksista, saataisiin mahdollisimman ajankohtaista tietoa. Dokumentaation ulkopuolinen lähdekirjallisuus on kaikki julkaistu korkeatasoisilla tieteellisillä foorumeilla, joiden julkaisufoorumi-luokka on vähintään perustasoa (1). Hakusanoina lähdekirjallisuuden etsinnässä käytettiin esimerkiksi seuraavia sanoja: pwa*, progressive web app*, progressive web-app*, native app*, native-app*, web app*, web-app* ja service worker*. Lisäksi hakusanojen lisänä käytettiin erilaisia hakuoperaattoreita, kuten AND ja OR, joiden avulla yhdisteltiin relevantteja termejä toisiinsa. Aikaisempaa tutkimusta aiheesta löytyi runsaasti, mutta vastaavaa konseptia, jossa etuja, haasteita sekä sovellusaiheita kartoitettiin, eivät tutkimukset pitäneet sisällään.

Tutkielma etenee seuraavasti: ensimmäiseksi käydään läpi progressiivisten verkkosovelluksiin taustaa ja käsitteitä, sekä niiden etuja että haasteita. Seuraavassa luvussa pohditaan mahdollisia sovellusalueita kartoitettujen etujen ja haasteiden perusteella. Lopuksi yhteenvedossa keskustellaan saavutetuista tuloksista sekä niiden merkityksestä, ja ehdotetaan relevantteja jatkotutkimusaiheita. Tutkimuksessa käytetään runsaasti teknisiä termejä, joten sen lukua ja ymmärtämistä helpottamaan on liitetty käsitehakemisto, joka löytyy sisällysluettelon jälkeisiltä sivuilta.

2 PROGRESSIIVISET VERKKOSOVELLUKSET

Tässä luvussa käydään läpi progressiivisten verkkosovellusten etuja ja haasteita. Lisäksi avataan aihealueeseen liittyviä käsitteitä, jotta tutkielmaa, sen tavoitteita ja tuloksia on helpompi tulkita.

2.1 Tausta

Vuonna 2017 yhteisö- ja mikroblogipalvelu Twitter halusi kehittää mobiilikäyttäjien käyttäjäkokemusta, jotta sovelluksen käyttö olisi nopeampaa, luotettavampaa sekä puoleensa vetävämpää (Web.dev, 2022a). Ratkaisuna tähän Twitter päätti luoda uuden sovelluksensa, Twitter Liten, progressiivista toteutustapaa hyödyntäen, joka oli kyseisellä ajanhetkellä vielä suhteellisen vähän käytetty teknologia. Sovelluksen avulla saavutetut tulokset ylittivät odotukset, sillä lähetettyjen twiittien määrässä nähtiin kuuden kuukauden aikana jopa 50 prosentin nousu (Shah, 2017).

Mobiilisovellusten suuri suosio ja kysyntä on viime aikoina luonut yrityksillä painetta luoda jokaiselle käyttöjärjestelmälle sopiva sovellus (Fortunato & Bernardino, 2018). Menetelmänä on tyypillisesti ollut kaksi vaihtoehtoa: (i) sovellus kehitetään täysin selainpohjaiseksi käyttämällä hyväksi responsiivista websuunnittelua, jossa sovellus skaalautuu käyttäjän laitteen mukaan (Baturay & Birtane, 2013), tai (ii) sovelluksesta rakennetaan natiiviversiot jokaiselle tarvittavalle käyttöjärjestelmälle.

Kummastakin menetelmästä on kuitenkin esitetty paljon kritiikkiä. Que, Guo ja Zhu (2016) muun muassa kritisoivat työn määrää, joka natiivisovelluksen laite- ja käyttöjärjestelmäyhteensopivuuden eteen täytyy tehdä. Selainpohjaisia sovelluksia on vuorostaan kritisoitu esimerkiksi ansaintamallin vaikeudesta (Jobe, 2013). Natiivisovellukset voidaan julkaista eri käyttöjärjestelmien sovelluskauppoihin ja veloittaa niiden lataamisesta tietty hinta (Huynh, Ghimire & Truong, 2017; Martin, Sarro, Jia, Zhang & Harman, 2016). Selainpohjaisissa sovelluksissa vastaava ansaintamalli täytyy kuitenkin rakentaa sovelluksen

sisäiseksi, hyödyntämällä esimerkiksi mikrotransaktioita tai kuukausiveloitusta (Jobe, 2013).

Progressiiviset verkkosovellukset pyrkivät minimoimaan näiden kahden standardin – natiivi- ja selainpohjaisen – kehityksen ongelmia yhdistämällä kummankin etuja. Progressiivisuudella tarkoitetaan tässä tapauksessa kehitysfilosofiaa, jolla pyritään saavuttamaan riittävä käytettävyys mahdollisimman monelle käyttäjälle, mutta samalla tarjoamaan paras mahdollinen kokemus niille käyttäjille, joilla on käytössään uusimmat selainversiot (Wells & Draganova, 2007).

Käytännössä tämä tarkoittaa sitä, että progressiivisen verkkosovelluksen tulisi toimia käyttäjillä, joiden käyttämä laitteisto ja ohjelmisto ei ole nykypäivän standardien tasolla. Näillä käyttäjillä progressiivinen verkkosovellus ei välttämättä tarjoa parasta mahdollista käyttäjäkokemusta, sillä esimerkiksi vanha selainversio ei välttämättä tue kaikkia sovelluksen tarjoamia ominaisuuksia. Uudempien laitteistojen ja ohjelmistojen omistajille progressiivisten verkkosovellusten tulisi kuitenkin tarjota paras mahdollinen käyttökokemus.

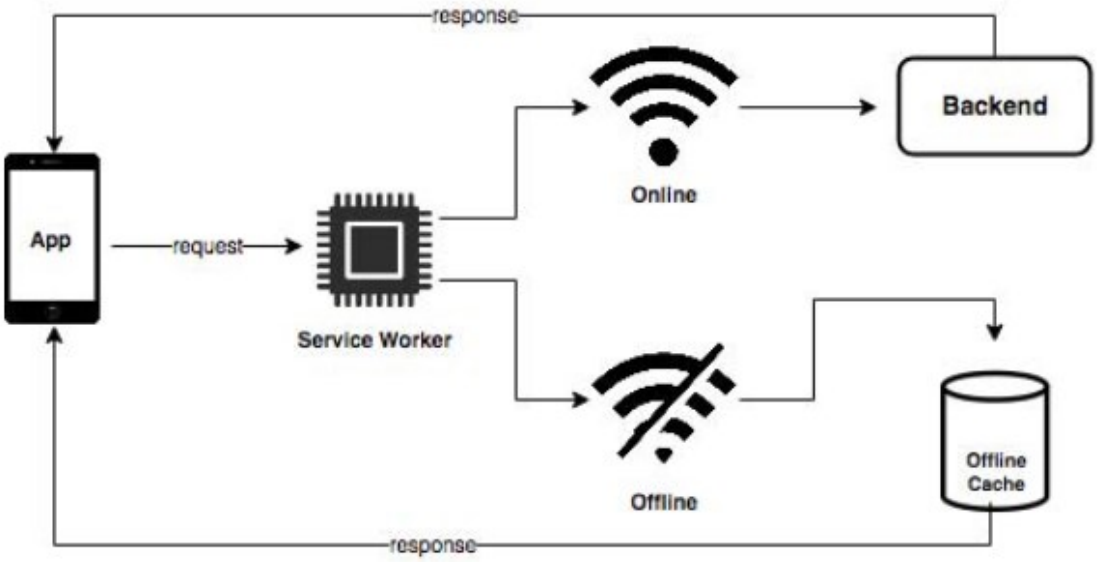
2.2 PWA-sovellusten edut

Tässä luvussa esitellään progressiivisten verkkosovellusten tyypillisiä hyötyjä verrattuna sekä natiivi- että web-sovelluksiin. Niitä ovat muun muassa käytettävyys ilman verkkoyhteyttä, mahdollisuus vähentää sovelluskehityksen kuluja, sekä laajat jakelu- ja asentamismahdollisuudet.

2.2.1 Käytettävyys ilman verkkoyhteyttä

Jotta web-sovelluksia voidaan käyttää, tarvitaan lähes poikkeuksetta yhteys internetiin. Progressiiviset verkkosovellukset pyrkivät kuitenkin kiertämään tämän haasteen hyödyntämällä niin sanottua service workeria, joka mahdollistaa sovelluksen käytön offline-tilassa tai huonolla verkkoyhteydellä (Pande, Somani, Samal, Kakkirala, 2016).

Service worker on eräänlainen skripti, jota suoritetaan uudenlaisessa web-ympäristössä perinteisen ympäristön rinnalla (Chinprutthiwong, Vardhan, Yang, Zhang & Gu, 2021). Web-sovellus voi rekisteröidä tämän service worker -skriptin käyttöönsä, joka mahdollistaa muun muassa offline-käytön. Tämä offline-käyttö toimii siten, että Service Worker nauhoittaa sovelluksen tekemiä API-kutsuja, eli tässä tapauksessa GET-pyyntöjä, ja tallentaa niihin saadut vastaukset cacheen, eli välimuistiin (Pande ym., 2016). Cache koostuu kahdesta eri osa-alueesta, joita ovat ”pre-cache assets” sekä ”fallback for offline access”. ”Pre-cache assets” pitää sisällään sovelluksen staattista sisältöä, kuten Html-kieltä, CSS-tyylejä, kuvia sekä JavaScriptiä, jotka latautuvat heti ensimmäisellä käyttökerralla. Nimensä mukaisesti ”fallback for offline access” puolestaan tallentaa API-kutsujen vastauksia offline-käyttöä varten. Kun vastaukset API-kutsuihin ovat tallessa, voidaan sovelluksen vastaukset kutsuihin hakea suoraan välimuistista, ilman verkkoyhteyttä. Seuraava kuvio kuvailee service workerin toimintaa (kuvio 1).



KUVIO 1 Service workerin toiminta (Gambhir & Raj, 2018 s. 295)

Sovellus lähettää service workerille tietokyselyn, johon se vastaa riippuen verkkoyhteyden saatavuudesta. Mikäli verkkoyhteys on saatavilla, haetaan vastaus pyyntöön suoraan palvelimelta, jolloin saadaan mahdollisimman ajankohtaista tietoa. Mikäli verkkoyhteyttä ei kuitenkaan ole saatavilla, service worker hakee vastauksen pyyntöön välimuistista, johon vastaus on tallennettu siltä kerralta, kun sovellus oli viime kerran yhteydessä internettiin. (Gambhir ym., 2018.)

Suurimmalla osalla ihmisistä on nykypäivänä mahdollisuus verkkoyhteyteen. Esimerkiksi Euroopassa keskimäärin 90 prosenttia ihmisistä pääsee halutessaan internettiin (van Kessel, Wong, Rubinić, O'Nuallain & Czabanowska, 2022), mutta on toisaalta olemassa alueita, joilla internet-yhteys on erityisen hidas, tai se puuttuu kokonaan. Esimerkiksi Meksikon kehitys verkkoyhteyksien suhteen on ollut varsin hidasta, ja nopeudet ovatkin pysyneet matalina vielä tähänkin päivään saakka (Martínez-Domínguez & Mora-Rivera, 2020). Näillä alueilla progressiivisten verkkosovellusten tarjoama käytettävyys voisi siis olla erityisen toivottavaa.

2.2.2 Sovelluskehityksen kulujen minimointi

Sovelluskehitys voi olla nykypäivänä erittäinkin kallista, mikä johtuu sovellusten laajuudesta, mutta myös siitä, että sama sovellus täytyy usein kehittää useaan eri käyttöympäristöön (Charland & Leroux, 2011). Näitä käyttöympäristöjä ovat esimerkiksi tietokoneiden käyttöjärjestelmät kuten Windows sekä macOS. Lisäksi mobiililaitteet tuovat sekaan useita eri variaatioita eri käyttöjärjestelmistä sekä resoluutioista.

Koska progressiiviset verkkosovellukset ovat selainpohjaisuutensa ansioista alustariippumattomia (Huber, Demetz & Felderer, 2021), voivat ne helpottaa tätä urakkaa tuomalla alustariippumattoman sovelluskehityksen

konkreettisemmaksi keinoksi luoda uusi sovellus. Käytännössä tämä tarkoittaa sitä, että sovellus täytyy kehittää progressiivista toteutustapaa hyödyntämällä vain kerran. Voidaan esimerkiksi kuvitella tilanne, jossa sovelluksen toteutustavaksi on valittu natiivisovellus. Sovellus täytyy kuitenkin käytännössä kehittää kolmeen kertaan, mahdollisesti jopa kolmella eri ohjelmointikielellä, sillä haluttuja alustoja on kolme. Mikäli projekti on kuitenkin toteutettavissa progressiivista toteutustapaa hyödyntäen, tarvittavaa työtä olisi käytännössä vain kolmasosa tästä.

2.2.3 Mukautuvuus sovelluksen jakelun ja asentamisen kannalta

Tavallisesti sovellukset asennetaan tietyn kanavan kautta. Tämä kanava voi olla vaikkapa puhelimen sovelluskauppa tai web-sivulta ladattava asennustiedosto. Progressiiviset verkkosovellukset tuovat kuitenkin mukanaan aivan uudenlaisen tavan asentaa sovelluksia, mikä tarjoaa sekä kehittäjille, että käyttäjille huomattavasti enemmän vapauksia. Sovellusta ei välttämättä tarvitse julkaista sovelluskauppaan, eikä sen tarvitse tarjota asennustiedostoa, vaan sovellus voidaan asentaa klikkaamalla selaimessa olevaa asennuspainiketta (Steiner, 2018). Tämä keino toki vaatii sellaisen selaimen käyttämistä, joka tukee progressiivisia verkkosovelluksia. Nykyisin käytännössä kaikki suosituimmat selaimet kuitenkin tukevat tätä ominaisuutta (Can I use, 2022).

Vaikka progressiiviset verkkosovellukset tarjoavat tämän asennusmahdollisuuden, voidaan ne silti julkaista myös sovelluskauppaan (Renzulli & LePage, 2020), mikä tekeekin niistä erityisen joustavia. Kehittäjä voi itse valita mielestään tehokkaimman keinon jakeluun, eikä ole sidottu ainoastaan tiettyyn vaihtoehtoon. Tämä tuo mukanaan muutamia hyötyä, joista ensimmäinen on sovelluksen jakelun maksimointi: kun sovellus julkaistaan usean eri kanavan kautta, voidaan sen jakelu maksimoida. Jakelun maksimointi voi myös tuoda yritykselle lisää asiakkaita (Rosenbloom, 2007), millä on suora vaikutus yrityksen talouteen ja kannattavuuteen. Toisena etuna on mahdollisten sovelluskaupan kulujen välttäminen, sillä sovelluksen julkaiseminen sovelluskauppaan voi olla hyvinkin kallista. Google Play -sovelluskauppa veloittaa esimerkiksi jopa 15–30 prosenttia sovelluksen avulla saaduista voitoista (Google, 2022). Vastaava prosenttiosuus App Store -sovelluskaupassa on 15 prosenttia, mutta julkaisijan täytyy myös ostaa 99 euroa vuodessa maksava lisenssi (Apple, 2022).

2.2.4 Käyttäjäperäisten tietoturvariskien torjuminen

Internetin käytön lisääntyessä lisääntyvät myös riskit joutua kyberhyökkäyksen kohteeksi (Zwilling, Klien, Lesjak & Wiechetek, 2022). Vaikka käyttäjällä olisikin käytössään asialliset työkalut näiden uhkien torjumiseen, Tam, Glassman ja Vandenwauver (2010) mukaan ihmiset ovat kuitenkin kyberturvallisuuden heikoin lenkki. Yksi monista riskeistä käyttäjän osalta on esimerkiksi sovelluksen päivittämättä jättäminen.

Usein ajatellaan, että sovelluksia päivitetään pääasiassa uusien ominaisuuksien vuoksi, vaikka tosiasiassa suuri osa päivityksistä pitää sisällään turvallisuusuhkien paikkauksia tai muita tietoturvapäivityksiä. Satunnaiset käyttäjät arvioivatkin päivittämättömien sovellusten tietoturvariskit merkittävästi vähäisemmiksi kuin tietoturva-asiantuntijat (Creese, Hodger, Jamison-Powell & Whitty, 2013). Vaikka sovelluksen käyttäminen vanhassa versiossaan ei siis välttämättä tunnu suurelle uhalle suurimalle osalle väestöä, on asia kuitenkin päinvastoin.

Selainpohjaiset sovellukset toimivat kuitenkin hieman eri tavalla. Sovelluksen suorittamiseen tarvittavia tiedostoja ei ladata sitä käyttävälle laitteelle natiivisovellusten tapaan, vaan kaikki sovelluslogiikka on sisällytetty esimerkiksi JavaScript -koodiin, jonka sovellus hakee joka kerta uudelleen sovellussivua ladattaessa. Sovellus siis ”päivitetään” lataamalla sivu uudelleen, ja sen avulla sovellus on siis aina uusimmassa versiossaan (Jobe, 2013). Koska progressiiviset verkkosovellukset toimivat selainpohjaisesti, voivat ne vähentää käyttäjäperäisiä riskejä. Tästä voidaan päätellä, että riski joutua kyberhyökkäyksen kohteeksi on progressiivisia verkkosovelluksia käyttämällä pienempi.

2.2.5 Suojattu kommunikaatio sovelluksen ja palvelimen välillä

Lähes kaikki verkkoliikenne on nykypäivänä suojattu HTTPS-protokollan (HyperText Transfer Protocol Secure) kautta (Durumeric, Kasten, Bailey & Halderman, 2013). Käytännössä tämä tarkoittaa sitä, että perinteisen HTTP-liikenteen kanssa hyödynnetään SSL- tai TLS-salausprotokollaa, joka tekee yhteydestä turvallisemman esimerkiksi pankkipalveluita hyödyntäessä (Naylor ym., 2014). Sivusto, joka ei suojaa yhteyttään käyttämällä HTTPS-protokollaa, tarjoaa tunkeilijoille mahdollisuuden kuunnella verkkoliikennettä sivuston ja käyttäjän välillä, ja tätä kautta esimerkiksi tallentaa keskusteluja, käyttäjänimiä, tai muita henkilökohtaisia tietoja (Web.dev, 2020b).

Tästä huolimatta protokollan käyttö ei ole kuitenkaan vaatimus julkisen web-sovelluksen julkaisemiseksi (Callegati, Cerroni & Ramilli, 2009). Tämä tarkoittaa, että web-sovelluksiin, joissa protokolla ei ole käytössä, kohdistuu kyberhyökkäyksen uhka, joka voi esimerkiksi maksukortteja käsittelevän sivun osalta johtaa rahallisiin menetyksiin, tai jopa identiteettivarkauteen. Progressiiviset verkkosovellukset kuitenkin vaativat HTTPS-protokollan käyttöä (Loreto ym., 2018), mikä tarkoittaa, että nämä verkkosovellukset ovat lähtökohtaisesti turvallisempia.

2.2.6 Muut edut

Edellä mainittujen etujen ja vahvuuksien lisäksi on myös pienempiä, mutta hyödyllistä ja käytännöllisiä etuja, joita progressiiviset verkkosovellukset tarjoavat.

Web-sovelluksista poiketen progressiiviseen verkkosovellukseen saa sisällytettyä ikonin, aivan kuten natiivisovelluksissa (Behl, Kashish & Gaurav Raj,

2018). Tämä lisää sovelluksen houkuttelevuutta, sillä hyvin suunniteltu ikoni houkuttelee käyttäjää klikkaamaan, ja tätä kautta myös käyttämään sovellusta (Jylhä & Hamari, 2019).

Progressiivisten verkkosovellusten ulkoasu on jokaisella laitteella lähes identtinen yhteisen koodikannan takia. Tämä luo yhtenäisyyttä eri laitteilta käytettynä, siinä missä esimerkiksi natiivisovelluksien eri laiteversioissa voi olla suuriakin eroja esimerkiksi laitteiden tukemien ominaisuuksien vuoksi.

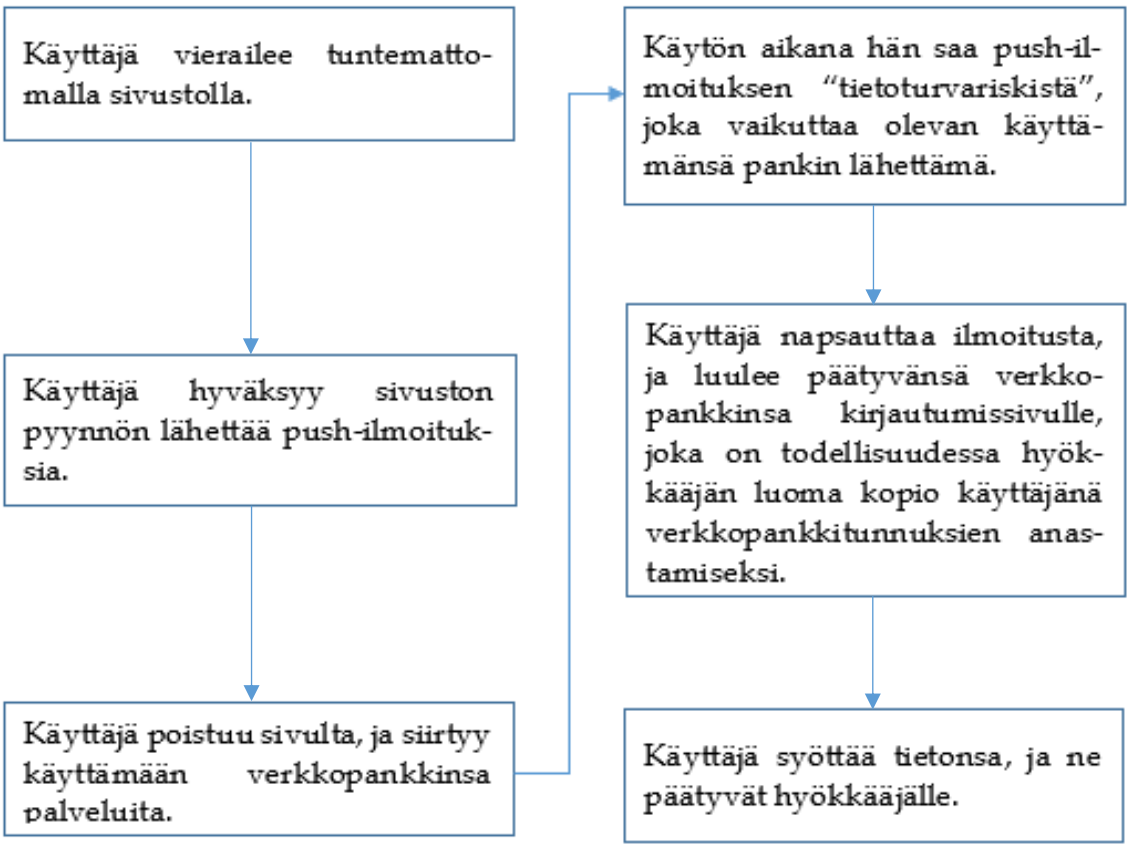
2.3 PWA-sovellusten haasteet

Vaikka progressiivisilla verkkosovelluksilla on suuri määrä erilaisia etuja, on niillä myös haasteensa, joita käsitellään tässä luvussa.

2.3.1 Tietoturvariskit push-ilmoituksien kautta

Progressiivisten verkkosovellusten push-ilmoitukset toimivat hieman eri tavalla verrattuna natiivisovellusten lähettämiin push-ilmoituksiin. Natiivisovelluksien tapauksissa sovellukset hallitsevat itsessään näitä ilmoituksia, mutta progressiivisten verkkosovellusten push-ilmoituksia hallitsevat puolestaan web-instanssit sekä selaimet (Lee ym., 2018). Käyttäjät joutuvat myös hyväksymään selaimen pyynnön lähettää ilmoituksia (Behl ym., 2018).

Koska push-ilmoituksia hallitaan web-instanssin kautta, mahdollistaa se riskin tietojenkalasteluun (ks. kuvio 2). Kun käyttäjä hyväksyy sivuston pyynnön lähettää push-ilmoituksia, voi mahdollinen tietojenkalastelija lähettää ilmoituksia myös silloin, kun käyttäjä ei ole kyseisellä sivulla. Ilmoituksia voidaan myös muokata näyttämään vaikkapa käyttäjän käyttämän pankin tai muun organisaation ilmoitukselta. Ennalta arvaamaton käyttäjä voikin joutua näiden ilmoituksien kautta kyberhyökkäyksen uhriksi.



KUVIO 2 Esimerkki tietojenkalastelusta PWA-sovelluksen kautta

2.3.2 Ominaisuuksien yhteensopimattomuus

Nykypäivänä web-sovellukset eivät enää ole pelkästään staattisia html-sivuja, joissa lomakkeella keretään tietoja käyttäjältä. Ne voivat olla hyvinkin monimutkaisia (Kreps & Kimppa, 2015), ja sisältää ominaisuuksia, joita on aikaisemmin ollut tarjolla ainoastaan natiivisovelluksissa. Näitä ominaisuuksia ovat esimerkiksi web-bluetooth ja push-ilmoitukset.

Progressiiviset verkkosovellukset eroavat tavallisista verkkosovelluksista esimerkiksi hyödyntämällä näitä ominaisuuksia, mutta kaikki näistä ominaisuuksista eivät välttämättä ole saatavilla kaikilla alustoilla (ks. taulukko 1). Varsinkin Apple on ollut varsin varovainen PWA-sovellusten suhteen, ja tarjoaakin tuen iOS- ja iPadOS-käyttöjärjestelmilleen ainoastaan Safari-selaimen kautta (Web.dev, 2022c). Safari ei kuitenkaan tue kaikkia progressiivisten verkkosovellusten ominaisuuksia, kuten push-ilmoituksia tai web-bluetoothia (MDN Web Docs, 2022a; MDN Web Docs, 2022b). Käytännössä tämä tarkoittaa sitä, että sovellus ei välttämättä tule toimimaan halutulla tavalla. Eri selaimet ovat kuitenkin nousevissa määrin lisänneet tukeansa eri progressiivisten verkkosovellusten tukemille ominaisuuksille, mutta ei voida varmuudella sanoa, tulevatko tulevaisuuden selainversiot silti tukemaan jokaista saatavilla olevaa ominaisuutta. Seuraavassa taulukossa (taulukko 1) esitetään muutamia yleisimpiä sekä

halutuimpia progressiivisten verkkosovellusten ominaisuuksia sekä niiden yhteensopivuuksia selaimittain. Taulukkoa seuraa kuvaukset kustakin ominaisuudesta.

TAULUKKO 1 PWA-ominaisuuksien yhteensopivuudet selaimittain

											
	Chrome	Edge	Firefox	Opera	Safari	Chrome Android	Firefox for Android	Opera Android	Safari on iOS	Samsung Internet	WebView Android
Push-ilmoitukset (MDN Web Docs, 2022a)	✓	✓	✓	✓	✓	✓	✓	✓	✗	✓	✗
Web bluetooth (MDN Web Docs, 2022b)	✓	✓	✗	✓	✗	✓	✗	✓	✗	✓	✗
Web NFC (MDN Web Docs, 2022c)	✗	✗	✗	✗	✗	✓	✗	✓	✗	✓	✓
Background Synchronization (MDN Web Docs, 2022d)	✓	✓	✗	✗	✗	✓	✗	✗	✗	✓	✓
Web Audio (MDN Web Docs, 2022e)	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Battery API (MDN Web Docs, 2022f)	✓	✓	✗	✓	✗	✓	✗	✓	✗	✓	✓

Push-ilmoitukset ovat usein tärkeä osa tiedon levittämisen kannalta (Pan, Liang, Zhou, Ge & Zhao, 2015). Niiden avulla sovellus voi viestiä käyttäjälle vaikkapa kiinnostavasta tarjouksesta. Tuki selaimittain on kattavaa, mutta iOS käyttäjille niitä ei ole tarjolla (21.10.2022). Bluetooth -teknologiaa käytetään esimerkiksi yhdistämään laitteita keskenään (Bisdikian, 2001). Web bluetooth mahdollistaa tämän myös progressiivisissa verkkosovelluksissa, mutta vielä varsin rajatusti. Web NFC puolestaan sallii datan jakamisen NFC-teknologian avulla (MDN Web Docs, 2022c). Progressiiviset verkkosovellukset voivat hyödyntää tätä ominaisuutta esimerkiksi lähimaksamisessa. Ominaisuus ei ole vielä kovinkaan tuettu, ja suosituimmista selaimista ainoastaan Chrome Android tukee sitä. Background Synchronization on progressiivisten verkkosovellusten kannalta varsin

relevantti ominaisuus: se sallii työskentelyn jatkumisen, vaikka verkkoyhteys häiriintyisi, ja verkkoyhteyden palautuessa ominaisuus synkronisoi offline-tilassa tehdyt API-kutsut (Behl ym., 2018). Vaikka ominaisuus onkin selkeästi hyödyllinen varsinkin alueilla, joissa verkkoyhteys ei ole stabiili, on sen tuki varsin huonoa. Natiivisovelluksissa äänet ovat usein isossa roolissa (Korhonen, Holm & Heikkinen, 2007). Niillä voidaan indikoida vaikkapa klikkauksia, onnistunutta toimintoa, tai virhetilannetta. Progressiiviset verkkosovellukset pääsevät myös hyödyntämään näitä ominaisuuksia Web Audion avulla, joka onkin varsin hyvin tuettu. Battery API sallii progressiivisen verkkosovelluksen pääsyn laitteen virtatietoihin, kuten mobiililaitteen akun varaukseen (Mandyam & Ehsan, 2012). Käyttö voi olla hyödyllistä esimerkiksi tilanteessa, jossa sovellus kuluttaa paljon virtaa, ja on koko näytön tilassa, milloin käyttäjä ei näe akun kuntoa. Tuki ominaisuudelle näyttäisi olevan varsin laajaa.

3 SOVELTUVUUS ERI KÄYTTÖKOHTEISIIN

Kun progressiivisten verkkosovellusten etuja ja haasteita on kartoitettu, voidaan siirtyä tarkastelemaan mahdollisia sovellusalueita. Tässä kappaleessa esitetyt käyttökohteet perustuvatkin siis aikaisemmissa kappaleissa esitettyihin tuloksiin.

3.1 Alueet heikolla internet-yhteydellä

Maailmassa on tällä hetkellä useita maita, joissa internet-yhteys ei ole itsestäänselvyys: yhteyteen ei välttämättä ole varaa, nopeus on nykystandardeihin verrattuna hidas, tai sitä ei yksinkertaisesti ole saatavilla (Martínez-Domínguez ym., 2020).

Kuten luvussa 2 huomattiin, progressiiviset verkkosovellukset pystyvät selainpohjaisesta toiminnastaan huolimatta toimimaan heikolla internet-yhteydellä, tai jopa ilman yhteyttä (Pande ym., 2016). Tästä syystä progressiiviset verkkosovellukset voisivat tuoda ratkaisun pitkille sivustojen latausajoille, ja samalla pienentää teknologiakuilua ihmisten välillä. Näissä tilanteissa riittää, että käyttäjä pääsee käyttämään verkkoyhteyttä silloin tällöin, vaikkapa yhdistämällä julkiseen langattomaan verkkoon. Käyttäjän pitää tietenkin myös käyttää sitä progressiivista verkkosovellusta, jonka hän haluaa toimivan myös heikolla verkkoyhteydellä. Tällä tavoin sovelluksen service worker tallentaa sivukyselyt, ja mahdollistaa myöhemmän käytön (Chinprutthiwong ym., 2021; Pande ym., 2016).

Täytyy kuitenkin muistaa, että progressiivisten verkkosovellusten käyttö vaatii kuitenkin verkkoyhteyteen yhdistämistä ainakin kertaalleen, sillä service workerin täytyy tallentaa vastaus API-kutsuihin ainakin kerran. Käytännössä tämä tarkoittaa, että sovellusta on mahdotonta käyttää alueilla, joissa verkkoyhteyteen ei ole lainkaan pääsyä. Sovellus ei myöskään pääse päivittymään ilman verkkoyhteyteen silloin tällöin yhdistämistä (Jobe, 2013), eli vaikkapa uutissovelluksen toteuttamisesta progressiivisten verkkosovellusten kautta ei kannata harkita, mikäli sen motivaationa on saavuttaa henkilöitä, jotka kärsivät huonosta verkkoyhteydestä.

3.2 Suurien massojen saavuttaminen

Kun sovellusta, vaikkapa sosiaalisen median alustaa, suunnitellaan suurelle yleisölle, halutaan sovelluksen olevan hyvin saatavilla. Toisin sanoen yrityksen tarjoaman sovelluksen tulisi olla käytettävissä mahdollisimman monella eri laitteella, käyttöjärjestelmällä sekä resoluutiolla. Progressiivisten verkkosovellusten etuja tarkastellessa tunnistettiin vahvuus usealla eri alustalle kehitystyötä tehdessä: riittää, että sovellus kehitetään ainoastaan kerran, ja tämä sovellus sopii lähes jokaiseen käyttökohteeseen. Lisäksi huomattiin, että progressiivisten verkkosovellusten jakelumahdollisuudet ovat natiivi- ja web-sovelluksia paremmat: sovellukset voidaan julkaista sovelluskauppaan, mutta latauslinkin voi sisällyttää myös sovellukseen. (Huber ym., 2021; Renzulli ym., 2020.)

Näiden kahden edun vuoksi progressiiviset verkkosovellukset ovat ideaalisia laajaan jakeluun: ne tarjoavat helpon tavan luoda sovellus mahdollisimman usean henkilön saataville. Suurille massoille sovellusta kehittävät yritykset tai yksityishenkilöt voisivatkin hyötyä tästä edusta niin rahallisesti kuin ajallisesti (Rosenbloom, 2007). Lisäksi sovelluksen ylläpitokustannukset luonnollisesti vähenevät ylläpidettävien sovellusten määrän laskiessa.

Progressiivisten verkkosovellusten haasteet mielessä pitäen, on toteutustapaa valitessa kyseiseen käyttökohteeseen kuitenkin pidettävä mielessä sovellukselta tarvittavat ominaisuudet. Kuten luvussa 2 todettiin, selainpohjaiset sovellukset eivät aina tue kaikkia sovelluksen vaatimuksia, kuten web-bluetoothia tai push-ilmoituksia (MDN Web Docs, 2022b; MDN Web Docs, 2022a). Pelkällä pintapuolisella tarkastelulla ei voida siis sanoa, että progressiiviset verkkosovellukset sopisivat aina suurien massojen saavuttamiseen, vaan valinta vaatii tarkemman tarkastelun sekä vaatimusmäärittelyn. Tällaisia sovelluksia tehtäessä olisi kuitenkin hyödyllistä ainakin harkita progressiivista toteutusta.

4 YHTEENVETO JA JATKOTUTKIMUSAIHEET

Sopivan teknologian valinta sovellusprojektia varten voi olla iso ja vaikea päätös, jolla voi kuitenkin olla suuri merkitys projektin onnistuvuuden kannalta. Oikea valinta voi parhaimmassa tapauksessa säästää yritykseltä rahaa, aikaa ja hermoja. Tässä tutkimuksessa pyrittiin selvittämään nimenomaan progressiivisten verkkosovellusten etuja, haasteita sekä niiden sovellusalueita. Tarkoituksena oli helpottaa yritysten ja yksityishenkilöiden päätöksentekoa teknologiaratkaisua valitessa ja etenkin progressiivista toteutustapaa harkitessa. Tutkimusta tehdessä havaittiin, että tutkimusta aiheesta on runsaasti, mutta aikaisemmin sovellusalueet eivät ole juuri saaneet akateemista huomiota.

Tutkimuksessa huomattiin, että progressiivisilla verkkosovelluksilla on useita etuja koskien käytettävyyttä, tietoturvaa sekä saavutettavuutta. Näitä ovat käytettävyyden kannalta toimintamahdollisuus heikolla tai olemattomalla verkkoyhteydellä, tietoturvan kannalta käyttäjäperäisten uhkien minimoiminen ja saavutettavuudessa laajat jakelumahdollisuudet. Toisaalta todettiin myös, että progressiiviset verkkosovellukset tuovat mukanaan myös uudenlaisia tietoturvariskejä liittyen lähinnä käyttäjän varomattomuuteen, sekä yhteensopivuusongelmia muun muassa push-ilmoitusten ja web-bluetoothin suhteen.

Kartoitettujen hyötyjen ja haasteiden perusteella löydettiin myös muutamia sovellusalueita, mutta samalla todettiin, etteivät progressiiviset verkkosovellukset eksplisiittisesti sovellu mihinkään selkeään käyttökohteeseen huolimatta useista eduistaan, vaan yrityksen tai yksityishenkilön täytyy ottaa huomioon muun muassa kohderyhmä ja sovellukselta vaadittavat ominaisuudet. Toisaalta progressiiviset verkkosovellukset voivat tuoda useaan tarkoitukseen – kuten sosiaalisen median alustaksi - sopivan ratkaisun, joka hyödyttää sekä kehittäjää että käyttäjiä, mikäli kohderyhmä ja vaadittavat ominaisuudet on otettu hyvin huomioon. Vastaavanlaisen onnistumisen on saavuttanut esimerkiksi Twitter, joka rakensi uuden sovelluksensa progressiivista toteutustapaa käyttäen, ja lisäsi lähetettyjen twiittien määrää jopa 50 prosenttia (Shah, 2022).

Tutkielmassa on myös rajoitteita. Varsin huomattavaa on, että tämän tutkielman ulkopuolelle jäi progressiivisten verkkosovellusten etujen ja haasteiden tutkiminen multi-origin arkkitehtuurissa. Lisäksi virallinen dokumentaatio (MDN Web Docs) pohjautuu vapaaehtoisten henkilöiden kontribuutioihin

aiheesta, eikä sitä sen vuoksi välttämättä päivitetä täsmällisesti. Sivuston ylläpitäjät käyvät kuitenkin jokaisen kontribuution läpi ennen julkaisua.

Tässä tutkielmassa esitettiin sovellusalue-ehdotuksia progressiivisille verkkosovelluksille. Näiden ehdotusten pohjalta voisi olla hyödyllistä tutkia kyseisiä sovellusalueita tarkemmin, eli selvittää, käytetäänkö progressiivisia verkkosovelluksia näissä tapauksissa, ja onko se todellisuudessa tarjonnut kartoitettuja etuja. Mikäli kartoitettuja etuja on saatu, vahvistaisi se tämän tutkimuksen legitimiteettiä.

Progressiivisten verkkosovellusten ollessa relevantteja ja kiinnostavia, olisi myös hyödyllistä kartoittaa, onko niiden opettamisesta tehty ehdotuksia tai suosituksia. Aiheesta voisi tehdä esimerkiksi empiirisen tutkimuksen, jossa tutkittaisiin, mitä aiheesta opetetaan, ja mitä sovelluskehityksessä tarvitaan.

LÄHTEET

Apple. (08.12.2022). Choosing a Membership. https://developer.apple.com/support/compare-memberships/

Behl, K., & Raj, G. (2018). Architectural pattern of progressive web and background synchronization. In 2018 International Conference on Advances in Computing and Communication Engineering (ICACCE) (pp. 366-371). IEEE.

Bisdikian, C. (2001). An overview of the Bluetooth wireless technology. IEEE Communications magazine, 39(12), 86-94.

Baturay, M. H., & Birtane, M. (2013). Responsive web design: a new type of design for web-based instructional content. *Procedia-Social and Behavioral Sciences*, *106*, 2275–2279.

Callegati, F., Cerroni, W., & Ramilli, M. (2009). Man-in-the-Middle Attack to the HTTPS Protocol. IEEE Security & Privacy, 7(1), 78-81.

Can I Use. (08.12.2022). Add to home screen (A2HS). https://caniuse.com/web-app-manifest

Charland, A., Leroux, B. (2011). Mobile application development: web vs. native. Commun. ACM 54, 5 (May 2011), 49–53. https://doi-org.ezproxy.jyu.fi/10.1145/1941487.1941504

Chinprutthiwong, P., Vardhan, R., Yang, G., Zhang, Y., & Gu, G. (2021). The service worker hiding in your browser: The next web attack target?. In 24th International Symposium on Research in Attacks, Intrusions and Defenses (pp. 312-323).

Creese, S., Hodges, D., Jamison-Powell, S., & Whitty, M. (2013). Relationships between password choices, perceptions of risk and security expertise. In *International Conference on Human Aspects of Information Security, Privacy, and Trust* (pp. 80-89). Springer, Berlin, Heidelberg.

Durumeric, Z., Kasten, J., Bailey, M., & Halderman, J. A. (2013). Analysis of the HTTPS certificate ecosystem. In Proceedings of the 2013 conference on Internet measurement conference (pp. 291-304).

Fortunato, D., & Bernardino, J. (2018). Progressive web apps: An alternative to the native mobile Apps. In *2018 13th Iberian Conference on Information Systems and Technologies (CISTI)* (pp. 1–6). IEEE.

Gambhir, A., & Raj, G. (2018). Analysis of cache in service worker and performance scoring of progressive web application. In 2018 International Conference on Advances in Computing and Communication Engineering (ICACCE) (pp. 294-299). IEEE.

Google. (08.12.2022). Service fees. https://support.google.com/googleplay/android-developer/answer/112622?hl=en

Huber, S., Demetz, L., & Felderer, M. (2021). Pwa vs the others: A comparative study on the ui energy-efficiency of progressive web apps. In International Conference on Web Engineering (pp. 464-479). Springer, Cham.

Huynh, M. Q., Ghimire, P., & Truong, D. (2017). Hybrid app approach: could it mark the end of native app domination?. Issues in Informing Science and Information Technology, 14, 049-065.

Jobe, W. (2013). Native Apps vs. Mobile Web Apps. *International Journal of Interactive Mobile Technologies*, *7*(4).

Jylhä, H., & Hamari, J. (2019). An icon that everyone wants to click: How perceived aesthetic qualities predict app icon successfulness. International Journal of Human-Computer Studies, 130, 73-85.

Khan, A. I., Al-Badi, A. & Al-Kindi, M. (2019). Progressive Web Application Assessment Using AHP. *Procedia computer science, 155*, 289–294. https://doi.org/10.1016/j.procs.2019.08.041

Korhonen, H., Holm, J., & Heikkinen, M. (2007). Utilizing sound effects in mobile user interface design. In IFIP Conference on Human-Computer Interaction (pp. 283-296). Springer, Berlin, Heidelberg.

Kreps, D., & Kimppa, K. (2015). Theorising Web 3.0: ICTs in a changing society. Information Technology & People.

Lee, J., Kim, H., Park, J., Shin, I., & Son, S. (2018). Pride and prejudice in progressive web apps: Abusing native app-like features in web applications. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security* (pp. 1731–1746).

Loreto, P., Braga, J., Peixoto, H., Machado, J., & Abelha, A. (2018). Step towards progressive web development in obstetrics. Procedia Computer Science, 141, 525-530.

Magomadov, V. S. (2020). Exploring the role of progressive web applications in modern web development. In Journal of Physics: Conference Series (Vol. 1679, No. 2, p. 022043). IOP Publishing.

Mandyam, G. D., & Ehsan, N. (2012). Html5 connectivity methods and mobile power consumption.

Martin, W., Sarro, F., Jia, Y., Zhang, Y., & Harman, M. (2016). A survey of app store analysis for software engineering. IEEE transactions on software engineering, 43(9), 817-847.

Martínez-Domínguez, M., & Mora-Rivera, J. (2020). Internet adoption and usage patterns in rural Mexico. *Technology in society*, *60*, 101226.

MDN Web Docs. (08.12.2022d). Background Synchronization API. https://developer.mozilla.org/en-US/docs/Web/API/Background_Synchronization_API

MDN Web Docs. (08.12.2022f). Battery Status API. https://developer.mozilla.org/en-US/docs/Web/API/Battery_Status_API

MDN Web Docs. (08.12.2022a). Push API. https://developer.mozilla.org/en-US/docs/Web/API/Push_API

MDN Web Docs. (08.12.2022e). Web Audio API. https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API

MDN Web Docs. (08.12.2022b). Web Bluetooth API. https://developer.mozilla.org/en-US/docs/Web/API/Web_Bluetooth_API

MDN Web Docs. (08.12.2022c). Web NFC Api. https://developer.mozilla.org/en-US/docs/Web/API/Web_NFC_API

Naylor, D., Finamore, A., Leontiadis, I., Grunenberger, Y., Mellia, M., Munafò, M., ... & Steenkiste, P. (2014). The cost of the" s" in https. In Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies (pp. 133-140).

Pan, Z., Liang, X., Zhou, Y. C., Ge, Y., & Zhao, G. T. (2015). Intelligent push notification for converged mobile computing and internet of things. In 2015 IEEE International Conference on Web Services (pp. 655-662). IEEE.

Pande, N., Somani, A., Samal, S. P., & Kakkirala, V. (2018). Enhanced web application and browsing performance through service-worker infusion framework. In 2018 IEEE International Conference on Web Services (ICWS) (pp. 195-202). IEEE.

Que, P., Guo, X., & Zhu, M. (2016). A comprehensive comparison between hybrid and native app paradigms. In *2016 8th International Conference on Computational Intelligence and Communication Networks (CICN)* (pp. 611–614). IEEE.

Renzulli, D., & LePage, P. (08.12.2020). How to define your install strategy. Web.Dev. https://web.dev/define-install-strategy

Rosenbloom, B. (2007). Multi-channel strategy in business-to-business markets: Prospects and problems. *Industrial marketing management*, *36*(1), 4-9.

Shah J. (08.12.2017). Twitter Lite in the Google Play Store in 24 more countries. https://blog.twitter.com/en_us/topics/product/2017/twitter-lite-in-the-google-play-store-in-24-more-countries

Steiner, T. (2018). What is in a web view: An analysis of progressive web app features when the means of web access is not a web browser. In Companion Proceedings of the The Web Conference 2018 (pp. 789-796).

Tam, L., Glassman, M., & Vandenwauver, M. (2010). The psychology of password management: a tradeoff between security and convenience. *Behaviour & Information Technology*, *29*(3), 233-244

van Kessel, R., Wong, B. L. H., Rubinić, I., O'Nuallain, E., & Czabanowska, K. (2022). Is Europe prepared to go digital? Making the case for developing digital capacity: an exploratory analysis of Eurostat survey data. *PLOS Digital Health*, *1*(2), e0000013.

Web.dev. (08.12.2022a). Twitter Lite PWA Significantly Increases Engagement and Reduces Data Usage. https://web.dev/twitter/

Web.dev. (08.12.2022c). Progressive Web Apps. https://web.dev/learn/pwa/progressive-web-apps/

Web.dev. (08.12.2020b). Why HTTPS matters. https://web.dev/why-https-matters/

Wells, J., & Draganova, C. (2007). Progressive enhancement in the real world. In Proceedings of the eighteenth conference on Hypertext and hypermedia (pp. 55-56).

Zwilling, M., Klien, G., Lesjak, D., Wiechetek, Ł., Cetin, F., & Basim, H. N. (2022). Cyber security awareness, knowledge and behavior: a comparative study. Journal of Computer Information Systems, 62(1), 82-97.