Tampere University

Ilari Räisänen

PROGRESSIIVISET WEB-SOVELLUKSET VAIHTOEHTONA NATIIVISOVELLUKSILLE MOBIILILAITTEILLA

Kandidaatintutkielma
Informaatioteknologian ja viestinnän tiedekunta
Tarkastajat: Petri Kannisto
May 2022

ABSTRACT

Ilari Räisänen: Progressiiviset web-sovellukset vaihtoehtona natiivisovelluksille mobiililaitteilla
Kandidaatintutkielma
Tampere University
Informaatioteknologian ja viestinnän tiedekunta
Ohjelmistotekniikka
May 2022

Progressiiviset web-sovellukset ovat web-tekniikoilla toteutettuja sovelluksia, jotka muistuttavat käytössä läheisesti perinteisiä natiivisovelluksia. Natiivisovellukset ovat sovelluksia, jotka kehitetään yksittäiselle alustalle, eivätkä ne sellaisenaan toimi muilla alustoilla ilman lisätyötä vaativia muutoksia. Natiivisovellukset ovat olleet pitkään yleisin sovellusmuoto mobiilikäyttäjille perinteisten verkkosovellusten lisäksi. Viimeisimpien vuosien aikana progressiiviset web-sovellukset ovat yleistyneet toteutusvaihtoehtona mobiilisovelluksille.

Tässä työssä vertaillaan progressiivisia web-sovelluksia natiivisovelluksiin mobiilialustojen näkökulmasta. Vertailua tehdään myös perinteisiin web-sovelluksiin, jotta saadaan parempi kuva progressiivisten web-sovellusten toiminnallisuudesta. Työn tavoitteena on selvittää, millaisissa tilanteissa mobiilisovelluksen voi toteuttaa natiivisovelluksen sijaan progressiivisena web-sovelluk-sena. Työ tehdään kirjallisuuskatsauksena. Työssä tutkitaan ensin natiivisovellusten ominaisuuksia ja seuraavaksi progressiivisten web-sovellusten ominaisuuksia. Vertailussa huomioidaan yleisimmät mobiilialustat, kehitystyö, käyttäjäkokemus ja vertailtavien toteutustapojen ominaisuudet kuten tehokkuus.

Vertailu osoittaa, että natiivisovellusten etuja ovat laitteiston ominaisuuksien monipuoliset käyttömahdollisuudet, usein parempi tehokkuus sekä mahdollisuus julkaista sovellus sovelluskaupoissa. Natiivisovellusten heikkouksina suhteessa progressiivisiin web-sovelluksiin nousee esille erityisesti kalliimpi ja hitaampi kehitysprosessi sekä suurempi tallennustilan käyttö.

Vertailussa progressiivisten web-sovellusten eduksi osoittautuu kehittämisen nopeus ja hinta sekä sovelluksen alustariippumattomuus. Progressiivisten web-sovellusten heikkoudet määritellään natiivisovellusten vahvuuksien kautta. Progressiiviset web-sovellukset ovat saatavilla URL-osoitteen avulla, minkä vuoksi niiden tekeminen maksulliseksi on perinteisiä natiivisovelluksia haastavampaa. Laitteiston ominaisuuksien käyttömahdollisuudet ovat myös rajallisemmat, erityisesti Applen iOS-alustalla.

Tehtyyn tuktimukseen ja vertailuun perustuen tässä työssä annetaan suositus tehdä valinta natiivisovelluksen ja progressiivisen web-sovelluksen välillä perustuen käyttötarkoitukseen ja siitä muodostuviin tarpeisiin. Loppuun on koottu taulukko toteutustapojen ominaisuuksista ja niihin liittyvistä huomioista.

Keywords: Progressiivinen web-sovellus, PWA, natiivisovellus, vertailu, web, mobiiliohjelmointi

CONTENTS

1. Johdanto 1
2. Tutkimusmenetelmä 2
3. Natiivisovellukset 3
3.1 Kehittäminen 3
3.1.1 iOS 3
3.1.2 Android 4
3.2 Sovelluksen julkaiseminen 4
3.3 Käyttäjäkokemus 5
3.4 Sovelluksen löydettävyys 6
4. Progressiiviset web-sovellukset 7
4.1 Kehittäminen 8
4.1.1 Tuki eri verkkoselaimilla ja käyttöjärjestelmillä 10
4.1.2 Haasteet 11
4.2 Käyttäjäkokemus 12
5. Vertailu 13
5.1 Kehittäminen 13
5.1.1 Kehittämiskustannukset 15
5.1.2 Käyttäjäkokemus 15
6. Yhteenveto 17
References 18

LYHENTEET JA MERKINNÄT

API	Ohjelmointirajapinta, Application Programming Interface
Natiivisovellus	Tietylle käyttöjärjestelmälle rakennettu sovellus
PWA	Progressiivinen web-sovellus, Progressive Web Application

1. JOHDANTO

Jo yli puolet internetin liikenteestä tapahtuu mobiilikäyttäjien kautta, ja suhteen ennustetaan suurenevan (“Share of global mobile website traffic 2015-2021”, 2022). Yleensä mobiilikäyttäjien käyttämät palvelut toteutetaan yhdellä kolmesta toteutusvaihtoehdosta:

- Perinteisillä web-sovelluksilla
- Natiivisovelluksilla, tai
- Progressiivisilla web-sovelluksilla.

Progressiivisten web-sovellusten suosio on kasvanut viime vuosina. Progressiiviset web-sovellukset tarjoavat natiivisovelluksen kaltaisen käyttäjäkokemuksen selainpohjaisena. Selainpohjaisuuden vuoksi ne voidaan toteuttaa alustariippumattomasti web-teknologioilla, mikä pienentää niiden kehityskustannuksia ja minkä vuoksi niiden toteuttaminen on nopeampaa suhteessa natiivisovelluksiin.

Progressiiviset web-sovellukset on mahdollista asentaa laitteelle ja ne voivat varastoida resursseja sekä esimerkiksi synkronoida sovelluksen päivityksiä taustalla, minkä ansiosta ne ovat huomattavasti perinteisiä web-sovelluksia nopeampia. Progressiiviset web-sovellukset varaavat vähemmän laitteen tallennustilaa ja ovat löydettävissä esimerkiksi Googlen hakukoneella, minkä ansiosta ne ovat natiivisovelluksia kevyempiä ja löydettävempiä.

Tässä työssä tutkitaan ja vertaillaan natiivisovellusten ja progressiivisten web-sovellusten eroja mobiilisovelluksen toteutustapoina. Tavoite on selvittää, missä ominaisuuksissa prog-ressiiviset web-sovellukset ovat keskimäärin natiivisovelluksia parempi vaihtoehto ja mitä natiivisovellusten ominaisuuksia progressiivset web-sovellukset eivät vielä pysty tarjoamaan natiivisovellusten tarjoamalla tavalla.

Työssä käsitellään ensin molemmat toteutustavat minkä jälkeen niitä vertaillaan. Luvussa 3 käsitellään natiivisovelluksia sekä niiden kehittämistä ja käyttäjäkokemusta. Luvussa 4 käsitellään progressiivisia web-sovelluksia sekä niiden kehittämistä ja käyttäjäkokemusta. Lopuksi luvussa 5 vertaillaan käsiteltyjä natiivisovellusten ja progressiivisten web-sovellusten ominaisuuksia ja luodaan ominaisuuksista taulukko.

2. TUTKIMUSMENETELMÄ

Tutkielma tehdään kirjallisuuskatsauksena. Aineistoa etsitään Andor-, Scopus- ja IEEE-tietokannoista sekä Googlen hakukoneella. Lisäksi tutkimuksessa käytetään käyttötilastoja tilastorekistereistä. Tässä työssä käytetyin tietokanta on Andor. Tietokannoissa ja hakukoneissa käytetyistä hakusanoista keskeisimmät olivat "Progressiiviset web-sovellukset", "web", "mobiilisovellukset", "sovelluskehitys" ja "natiivisovellukset". Lisäksi jokaisesta suomenkielisestä hakusanasta käytettiin myös englanninkielistä käännöstä. Suomenkielisiä ensisijaisia lähteitä ei työhön lopulta päätynyt yhtään.

Tutkimuskysymys, johon työssä pyritään vastaamaan, on

- Millaisessa tilanteessa progressiivinen web-sovellus on natiivisovellusta parempi vaihtoehto mobiilisovelluksen toteutustapana?

Progressiiviset web-sovellukset ovat varsin uusi teknologia, joka kehittyy jatkuvasti. Tämän vuoksi tässä työssä suositaan vuoden 2020 jälkeen julkaistuja lähteitä. Työssä käytettiin myös vanhempia lähteitä, jos uudempaa tietoa ei ollut saatavilla.

Valtaosa yllä esitellyillä hakusanoilla löytyvistä lähteistä tarjoaa optimistisen kuvan progressiivisten web-sovellusten toiminnasta. Tutkimuksessa oli tärkeää etsiä myös kriittisempiä lähteitä, jotta progressiivisista web-sovelluksista muodostuva kokonaiskuva on totuudenmukainen.

3. NATIIVISOVELLUKSET

Natiivisovellukset ovat tietylle alustalle rakennettuja natiivikoodiin perustuvia sovelluksia, jotka eivät sellaisenaan toimi eri alustoilla. Natiivisovelluksia ovat esimerkiksi lähes kaikki Google Play -sovelluskaupasta löytyvistä sovelluksista.

3.1 Kehittäminen

Natiivisovellusten kehittäminen on usein aikaa vievää ja kallista. Kehittäminen käyttää alustojen omia työkaluja ja kieliä, minkä vuoksi sovelluksen kehittäminen usealle alustalle vaatii sovelluksen kehittämistä useaan kertaan. Tämä vaatii enemmän kehittäjiä useammilla eri osaamisprofiileilla. On kuitenkin mahdollista käyttää alustariippumattomia viitekehyksiä, kuten Javan Codename One -viitekehys (“Codename One”, 2021).

3.1.1 iOS

iOS on Applen mobiililaitteileen tuottama käyttöjärjestelmä. Apple on yksi suurimmista mobiililaitteiden valmistajista, sillä se vastasi noin 22 % mobiililaitteiden myynnistä vuoden 2021 viimeisessä kvartaalissa (Chaurasia & Peng, 2022).

Natiivisovellusten kehittämisessä iOS-alustalle käytettävät kielet ovat Swift ja Objective-C. Vuonna 2021 tehtyjen kyselytutkimuksen mukaan näiden kielien suosio on pieni, sillä vain 5,1 % tutkimukseen vastanneista kertoi käyttäneensä Swiftiä ja 2,8 % Objective-C:tä (Vailshery, 2021). Tämä johtuu osittain siitä, että näillä kielillä kehitetään vain Applen käyttöjärjestelmille. On vaikeampaa löytää kehittäjiä Swiftiin ja Objective-C:hen kuin esimerkiksi web-kehittämisessä suosittuun JavaScriptiin, jota samaan kyselytutkimukseen vastanneista noin 65,5 % kertoi käyttäneensä (Vailshery, 2021).

3.1.2 Android

Android on useiden eri valmistajien laitteille suunniteltu mobiilikäyttöjärjestelmä, joka pohjautuu Linux-käyttöjärjestelmän ytimeen (engl. kernel) (“Kernel”, 2020). Android on mobiilikäyttöjärjestelmistä käytetyin, sillä vuoden 2022 tammikuussa Androidin markkinaosuus oli lähes 70 % kaikkien käytössä olevien mobiililaitteiden käyttämistä käyttöjärjestelmistä (Laricchia, 2022).

Android-käyttöjärjestelmälle natiivisovellukset kehitetään yleensä Java-kielellä sekä sihen pohjautuvalla, uudemmalla Kotlin-kielellä. Java-ohjelmointikieltä käyttää kehittäjistä noin 35,4 %, eli huomattavasti Swift- ja Objective-C-kieliä suurempi osa. Kotlin-ohjelmointikieltä käyttää noin 8,3 % kehittäjistä. (Vailshery, 2021)

3.2 Sovelluksen julkaiseminen

Natiivisovelluksen julkaiseminen käyttäjien löydettäväksi poikkeaa web-sovellusten julkaisuprosessista. Natiivisovellusten etsiminen ja lataaminen tapahtuu lähes poikkeuksetta sovelluskaupoissa, joista suurimmat ovat Android-käyttöjärjestelmällä käytössä oleva Google Play ja Applen iOS-käyttöjärjestelmällä käytössä oleva App Store. Sovellukset julkaistaan asennettavina binääritiedostoina, joita kutsutaan Android-käyttöjärjestelmässä APK-paketeiksi (engl. Android Package) (“Application fundamentals”, 2021) ja iOS-käyttöjär-jestelmässä IPA-paketeiksi (iPhone Package) (“What is an IPA file and how can you open one?”, 2022).

Näitä binääritiedostoja voidaan jakaa muuallakin internetissä, ja Androidilla pakettien lataaminen ja asentaminen on yksinkertaista myös sovelluskauppojen ulkopuolelta. Esimerkiksi organisaation sisäiset sovellukset on mahdollista jakaa sähköpostilla. Vastaanottaja avaa paketin, ja Android-järjestelmä tunnistaa paketin ja antaa hänen asentaa sen (“Alternative distribution options”, 2020). iOS-järjestelmällä asentaminen on kuitenkin vaikeampaa. iOS ei salli App Storen ulkopuolelta ladattujen pakettien asentamista kahta poikkeusta lukuun ottamatta. Jos sovellus on tarkoitettu organisaation sisäiseen jakeluun, voi kehittäjä liittyä Apple Developer Enterprise -ohjelmaan, joka mahdollistaa sovelluksen jakelun esimerkiksi omalla verkkosivulla tai palvelimella. Toinen vaihtoehto on Volume Purchase -ohjelma, joka mahdollistaa sovelluksen jakelun URL-osoitteella organisaation hallitsemille laitteille. (“How do I distribute iOS apps without App Store Google play?”, 2018)

Google Play-sovelluskauppaan julkaiseminen vaatii kehittäjältä ensin 25 euron suuruisen aloitusmaksun, minkä jälkeen kehittäjän täytyy odottaa hyväksyntää. Hyväksynnän jälkeen kehittäjä pääsee Google Play-kehittäjäkäyttöliittymään. Sovelluksen julkaiseminen on aloitusmaksun jälkeen ilmaista. Lisäksi sovelluksen tekeminen maksulliseksi on yksinkertaista, ja vaatii vain yhden valinnan kehittäjäkäyttöliittymällä. Julkaistava sovellus lähetetään lopuksi arvioitavaksi. Arviointi perustuu Google Playn voimassa oleviin käyttöehtoihin, ja siinä saattaa mennä jopa päiviä. (Sharma, 2022)

Applen App Store-sovelluskauppaan julkaisemista varten kehittäjän tulee liittyä Applen Kehittäjäohjelmaan, joka maksaa 99 euroa vuodessa. Julkaisuprosessi on samanlainen kuin Google Play-sovelluskaupassa. Maksulliseksi tekeminen on yksinkertaista, ja ennen julkaisua sovellus arvioidaan perustuen sovelluksen sisältöön sekä toiminnallisuuteen. (Ching, 2019)

Vaikka sovellusten lataaminen ja asentaminen sovelluskaupoista on helppoa ja nopeaa, mobiilikäyttäjät käyttävät päivässä keskimäärin vain yhdeksää eri sovellusta. Nämä sovellukset sisältävät suuria sosiaalisen median alustoja, kuten Facebook ja Instagram, pikaviestisovelluksia kuten WhatsApp ja Snapchat, sekä sähköpostisovelluksia kuten Gmail. (“Mobile app Download Statistics”, 2022) Nämä sovellukset poisluettuna yhdeksään keskimäärin päivässä käytettyyn sovellukseen ei mahdu montaa uutta sovellusta.

Mobiilikäyttäjät ovat valikoivia asentamiensa mobiilisovellusten suhteen. Mobiilikäyttäjistä jopa 55 % kertoo asentavansa mobiilisovelluksia harvoin tai ei koskaan, ja vain 16 % kertoo asentavansa mobiilisovelluksia usein. (Schiela, 2020) Natiivisovellusta julkaistessa tulee pohtia, kuinka sovellus herättää mobiilikäyttäjien huomion ja erottuu tuhansista muista päivän aikana julkaistuista sovelluksista, ja erityisesti huomioida käyttäjien valikoiva luonne mobiilisovellusten suhteen.

3.3 Käyttäjäkokemus

Natiivisovellus on ladattava ja asennettava jakelijan kautta. Yleensä natiivisovellusten lataaminen tapahtuu sovelluskauppojen kautta. Sovelluskaupan käyttäminen lisää yhden askeleen suhteessa web-sovelluksiin. Toisaalta yksi suuri sovelluskauppa, josta käyttäjä voi löytää kaikki sovellukset, yksinkertaistaa sovellusten etsimistä.

Turvallisuuden tunne on olennainen osa käyttäjäkokemusta. Tunnetut organisaatiot herättävät luottamusta käyttäjissä. Suurimpien sovelluskauppojen taustalla olevat yritykset, Apple ja Google, sijoittuvat luotetuimpien yritysten joukkoon esimerkiksi Yhdysvalloissa (Gandhi, 2021). Natiivisovellusten jakelu tapahtuu pääasiassa näiden sovelluskauppojen kautta, mikä tuo käyttäjille turvallisuuden tunnetta.

Googlen hakukone indeksoi Google Play-sovelluskauppaa, minkä myötä on mahdollista löytää sovelluksia Android-käyttöjärjestelmälle Googlen kautta. iOS-käyttöjärjestelmälle ei kuitenkaan tarjota App Store-sovelluskaupan indeksointia.

3.4 Sovelluksen löydettävyys

Löydettävyys tarkoittaa sitä, kuinka helposti sovellus löytyy esimerkiksi hakukoneella. (“Avoin Tiede:löydettävyys”, 2022) Natiivisovellusten löydettävyys perustuu suosituksiin ja sovelluskaupoista etsimiseen. 51 % Googlen kyselyyn vastanneista kertoi ladanneensa jonkin sovelluksen, koska heidän tuttavansa käyttivät sitä. 48 % löysi jonkin sovelluksen selailemalla jotain sovelluskauppaa, ja vain 21 % löysi sovelluksen jonkin hakukoneen kautta. (Schiela, 2020)

Applen sovelluskauppaan lisätään päivittäin lähes 1000 sovellusta (Gelzinis, 2021), ja Googlen sovelluskauppaan yli 3700 sovellusta (Sharma, 2022). Näin monen sovelluksen joukossa on vaikea erottua ja saada sovellukselle lataajia ja käyttäjiä, erityisesti kun huomioidaan se, että mobiilikäyttäjät eivät asenna keskimääräisen kuukauden aikana yhtäkään sovellusta. (“Mobile app Download Statistics”, 2022)

4. PROGRESSIIVISET WEB-SOVELLUKSET

Progressiiviset web-sovellukset ovat verkkosivuja, joiden tarkoitus on mahdollistaa natiivinkaltainen kokemus. Ne toimivat verkkoselaimen kautta, tarjoten alustariippumattoman toiminnan. Verkkoselaimen kautta toimiminen mahdollistaa sen, että eri alustoille ei tarvitsee kehittää omia toteutuksiaan.

Progressiiviset web-sovellukset osittain yhdistävät natiivisovellusten sekä perinteisten web-sovellusten etuja. Progressivisen web-sovelluksen voi asentaa laitteelle, mutta toisaalta sitä voi käyttää myös ilman asentamista verkkoselaimessa kuten perinteistä web-sovellusta. Progressiivisen web-sovelluksen voi asentaa ilman sovelluskauppaa suoraan verkkoselaimesta. Progressiivisen web-sovelluksen voi myös saada ladattavaksi sovelluskaupasta, kuten Google Playsta(Google, 2021). Tavallisesti sovelluksen asentamisen tekeminen maksulliseksi sovelluskaupoissa on yksinkertaista, mutta progressiivisen web-sovelluksen asentamista on kuitenkin vaikea tehdä maksulliseksi. Maksullisuuden voi sen sijaan toteuttaa esimerkiksi käyttäjän tunnistamisen avulla.

Richard, S & Lepage, P jakavat progressiivisten web sovellusten perustan kolmeen tärkeimpään ominaisuuteen: Kykenevä, luotettava ja asennettava.

1. Kykeneviä: Progressiivisten web-sovellusten suorituskyky on vastaa lähes natiivisovellusten suorituskykyä. Progressiivisen web-sovelluksen suorituskykyä voidaan mitata samalla tapaa kuin perinteisten web-sovellusten suorituskykyä. Esimerkiksi Lighthouse on avoimen lähdekoodin työkalu, jonka avulla kehittäjät voivat mitata oman web-sovelluksensa suorituskykyä eri mittareilla (“Lighthouse PWA Analysis Tool”, 2019). Näiden mittareiden avulla kehittäjä voi varmistaa, että progressiivisen web-sovelluksen käyttökokemus on natiivinkaltainen.
2. Luotettavia: Progressiiviset web-sovellukset tuntuvat nopeilta ja luotettavilta verkon nopeudesta huolimatta. Nopeat latausajat tarkoittavat sitä, että käyttäjät voivat paitsi käyttää sovellusta myös nauttia siitä. Käyttäjän poistumisen todennäköisyys kasvaa 123 %, kun latausaika muuttuu 1 sekunnista kymmeneen sekuntiin (Richard & LePage, 2020). Lisäksi luotettavuuden tunnetta tuo sovelluksen latautuminen verkosta riippumatta. Tähän riittää sovelluskuori (engl. application shell). Sovelluskuori kertoo verkkoyhteyden puuttumisesta sen sijaan, että lataaminen näennäisesti alkaa ja epäonnistuu. (Behl & Raj, 2018)

3. Asennettavia: Progressiiviset web-sovellukset ovat asennettavissa laitteen aloitusnäyttöön käyttäjälle. Asennettavuus mahdollistaa käyttöjärjestelmän asennetuille sovelluksille tarjoamien palveluiden kuten kansioiden ja hakuominaisuuksien hyödyntämisen.

Lisäksi on olemassa muutamia muita piirteitä, jotka erottuvat progressiivisissa web-sovelluksissa. Nimensä mukaan progressiiviset web-sovellukset ovat progressiivisia – ne toimivat riippumatta alustasta ja käyttöjärjestelmästä. Tässä on kuitenkin muutamia rajoitteita, esimerkiksi verkkoselain Safari ei tue kaikkia samoja ominaisuuksia, kuin esimerkiksi Google Chrome. (Rykov, 2022) Progressiiviset web-sovellukset ovat responsiivisia, eli sama toteutus sovelluksesta mukautuu eri laitteille ja erikokoisille näytöille, myös työpöytälaitteilla. Lisäksi ne ovat helposti löydettävissä, sillä progressiivisen web-sovelluksen voi löytää hakukoneella, URL-osoitteen avulla sekä viedä asennettavaksi sovelluskauppaan. Käyttäjäkokemus progressiivisilla web-sovelluksilla on myös natiivinkaltainen esimerkiksi nopeuden ja navigaation kannalta Toiminta on myös osin riippumatonta verkkoyhteydestä välimuistin hyödyntämisen ansiosta. Päivittäminen ei vaadi käyttäjältä uuden version lataamista ja asentamista. (Muman, 2021)

4.1 Kehittäminen

Progressiivisia web-sovelluksia kehittäessä kehittäjät pääsevät hyödyntämään web-ohjelmoinnin etuja. Sovellukset toimivat samankaltaisesti kaikilla käyttäjillä huolimatta käytetystä selaimesta, mikäli selain tukee progressiivisia web-sovelluksia. Riittää, että sovellus toteutetaan kerran web-teknologioilla. (Gómez-Sierra, 2021) Tämä tekee progressiivisista web-sovelluksista kustannustehokkaita ja toimitusajaltaan nopeita.

Progressiivisissa web-sovelluksissa on kolme pakollista osaa: Manifest-tiedosto, taustasuorittaja (eng. service worker) ja HTTPS-yhteys. Manifest-tiedosto on muodoltaan JSON-tiedosto eli JavaScript-olionotaatiotiedosto, joka sisältää määritystietoja web-sovelluksesta. Taustasuorittajat ovat JavaScript-tiedoston muodossa olevia selaimessa suoritettavia prosesseja. Taustasuorittajat ovat tapahtumalähtöisiä, joka tarkoittaa sitä, että ne kuuntelevat niille määritettyjen tapahtumien varalta ja toimivat niiden tapahtuessa. Kolmas pakollisista osista, HTTPS, on protokolla, joka auttaa siirtämään tietoa verkossa turvallisemmin.

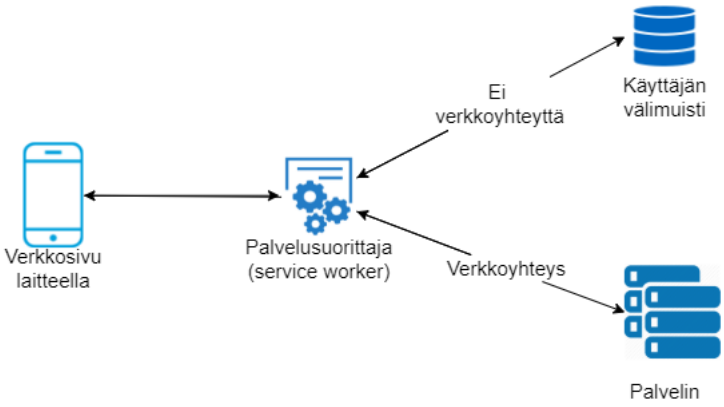

Figure 4.1. *Taustasuorittajan toiminta*

Taustasuorittajat mahdollistavat välimuistin käyttämisen kuvan 4.1. mukaisesti. Taustasuorittaja voi hallita verkkosovellusta, käsitellä pyyntöjä ja tallentaa tietoa välimuistiin. Taustasuorittajilla ei ole pääsyä verkkosovelluksen DOM:iin (Document Object Model, suom. dokumenttioliomalli), minkä vuoksi ne eivät voi suoraan manipuloida verkkosivua. Lisäksi se toimii eri säikeessä kuin itse verkkosovellus. Taustasuorittajat on määritelty vaatimaan HTTPS-yhteyden. HTTPS-yhteys auttaa varmistamaan sivun aitoutta, ja sen puuttuminen altistaisi sovelluksen väliintulohyökkäyksille. (“Service Worker API”, 2022)

Progressiiviset web-sovellukset rakennetaan usein sovelluskuoren (eng. application shell) ympärille, joka mahdollistaa sivuston osittaisen välittömän lataamisen uudelleen sivulla vierailtaessa. Vain muuttuva sisältö on tarpeen ladata (Osmani, 2019). Sovelluskuoren myötä käyttäjäkokemuksesta voi saada hyvän, vaikka verkkoyhteys olisi heikko tai peräti puuttuisi kokonaan. Sovelluskuori mahdollistaa myös progressiivisen web-sovelluksen latausaikojen lyhentymisen perättäisissä vierailuissa. Latausaikojen nopeutumista sovelluskuoren myötä on mitattu viiden eri progressiivisen web-sovelluksen tapauksessa.

Table 4.1. *Sovelluskuoren vaikutus latausaikoihin eri sovelluksissa (Behl & Raj, 2018)*

Sovelluksen nimi	Vierailukerta, latausaika (ms)					
	1.	2.	3.	4.	5.	Paras latausaika (ms)
1. Money Tracker	3.87	1.34	0.95	0.88	0.80	0.80
2. Recipe Book	0.80	0.74	0.71	0.70	0.71	0.70
3. Swiggy	5.25	3.38	1.34	1.27	1.39	1.27
4. Ola Cabs	4.36	3.20	2.95	3.15	2.84	2.95
5. Tinder	2.98	2.36	1.68	0.86	0.76	0.80

Taulukosta 4.1. voidaan nähdä latausaikojen lyhenevän jopa viidesosaan alkuperäisestä toistuvien vierailukertojen myötä. Nopeammat latausajat mahdollistavat enemmän natiivisovellusten kaltaisen käyttäjäkokemuksen.

4.1.1 Tuki eri verkkoselaimilla ja käyttöjärjestelmillä

Kuten luvun alussa mainittiin, progressiiviset web-sovellukset toimivat muidenkin web-sovellusten tapaan selaimen kautta. Eri selainten progressiivisille web-sovelluksille tarjoamassa tuessa on kuitenkin eroja. Käytetyimmät selaimet mobiililaitteilla ovat Google Chrome, jonka markkinaosuus mobiililaitteilla on noin 62 %, sekä Safari, jonka markkinaosuus on vastaavasti noin 26 % (Vailshery, 2022). Taulukossa 4.2. vertaillaan näiden selainten tukea progressiivisille web-sovelluksille. (Rykov, 2022)

Table 4.2. *Mobiiliselainten tarjoama tuki progressiivisille web-sovelluksille (mukaillen (Rykov, 2022))*

Ominaisuus	**Mobiiliselain**	
	Google Chrome	**Safari (iOS)**
Asentaminen	Kyllä	Kyllä
Taustasuorittajan tuki	Kyllä	Kyllä
Maksaminen	-Automaattinen täyttö -Android Pay -Maksupyyntö API (engl. Payment Request API)	-Automaattinen täyttö -Maksupyyntö API (engl. Payment Request API)
Laitteiston ominaisuudet	-Kamera -Mikrofoni -Pääsy tiedostoihin -GPS -Leikepöytä -Gyroskooppi -Kiihtyvyysanturi -NFC	-Pääsy tiedostoihin -Kamera -Mikrofoni -Geolokaatio -Gyroskooppi (osittainen tuki)
Push-ilmoitukset	Kyllä	Betatestauksessa
Offline-toiminnallisuus	Kyllä	Kyllä
Taustasynkronointi	Kyllä	Ei

Kaikkien Applen mobiililaitteiden käyttöjärjestelmänä toimiva iOS (iPad-tableteilla uudelleennimetty iPadOS) rajoittaa progressiivisten web-sovellusten toimintaa Android-käyttö-järjestelmää enemmän. Safari on Applen selain, jota tuetaan vain Applen laitteilla. iOS mahdollistaa progressiivisten web-sovellusten käytön vain Safari-selaimella. Kuten taulukosta 4.2 nähdään, Safari ei tue esimerkiksi taustasynkronointia tai GPS-toimintoja. (Tran, 2021)

Kuten nähdään, progressiivisten web-sovellusten kaikkien ominaisuuksien hyödyntäminen ei tällä hetkellä ole mahdollista iOS-käyttöjärjestelmässä. Apple on kuitenkin tehnyt vuoden 2022 aikana progressiivisten web-sovellusten tuesta kattavampaa. Tammikuussa Apple julkaisi iOS-käyttöjärjestelmälle 15.4-betaversion. Päivityksessä Apple lisäsi esimerkiksi tuen push-ilmoituksille, jotka mahdollistavat natiivisovellusten kaltaisten ilmoitusten lähettämisen käyttäjille. Push-ilmoitukset ovat pitkään puuttuneet iOS-järjestelmän progressiivisille web-sovelluksille tuetuista ominaisuuksista. (Firtman, 2022) Tilanteessa, jossa mahdollistaa progressivisille web-sovelluksille suuremmat toimintamahdollisuudet, saattaa progressiivisten web-sovellusten markkinaosuus kasvaa suhteessa natiivisovelluksiin.

Progressiivinen web-sovellus -termiä käytti ensimmäistä kertaa Googlen työntekijä vuonna 2015 (Rakowski et al., 2019a). Googlen kehittämänä käyttöjärjestelmänä Androidin tuki progressiivisille web-sovelluksille on hyvä, ja sitä kehitetään jatkuvasti. Androidinkin tuessa progressiivisille web-sovelluksille on kuitenkin puutteita, sillä esimerkiksi ympäristön valaistuksen lukemista eikä yhteystietojen lukemista ole vielä virallisesti tuettu. (Bar, 2022)

4.1.2 Haasteet

Uutena teknologiana progressiiviset web-sovellukset tuovat etujen lisäksi myös paljon haasteita. Aiemmin työssä mainittiin natiivisominaisuuksien käyttömahdollisuuksien puutteista. Esimerkiksi kasvojentunnistus, sormenjäljen lukeminen sekä taustasynkronointi eivät ole toistaiseksi tuettuja iOS-käyttöjärjestelmällä. Moni sovellus käyttää sormenjälkeä tunnistautumiseen, ja mikäli tämä toiminnallisuus on toteutettavalle sovellukselle kriittinen, on valittava toinen toteutustapa. (Rakowski et al., 2019b)

Taustasuorittajien kyky tallentaa välimuistiin on keskeinen osa progressiivisia web-sovelluksia. Android-käyttöjärjestelmällä taustasuorittajat voivat hyödyntää laitteen koko tallennustilaa. iOS kuitenkin rajoittaa välimuistiin tallentamisen 50 megatavuun. Tämä rajoitus tekee esimerkiksi videoiden tallentamisen internet-yhteydentöntä tallentamista varten mahdottomaksi, minkä vuoksi esimerkiksi suoratoistopalveluiden internet-yhteydetön toiminnallisuutta ei ole mahdollista toteuttaa progressiiviseen web-sovellukseen. (Rakowski et al., 2019b)

Progressiivisen web-sovelluksen asentaminen on keskeistä, jotta käyttäjät voivat saada natiivinkaltaisen käyttäjäkokemuksen. Haasteeksi muodostuu käyttäjien tietämättömyys asentamismahdollisuudesta. Progressiivisen web-sovelluksen asentaminen selaimen kautta on yksinkertaista, mutta moni kuitenkaan ei tiedä asentamismahdollisuuden olemassaolosta. (Rakowski et al., 2019b)

4.2 Käyttäjäkokemus

Käyttöliittymä on olennainen osa käyttäjäkokemuksen muodostamista. Kuten luvussa aiemmin todetaan, progressiiviset web-sovellukset toimivat yhdellä toteutuksella alustariippumattomasti eri mobiilikäyttöjärjestelmillä. Lisäksi sovellus näyttää samalta sekä suoraan verkkoselaimen kautta avattuna että asennettuna progressiivisena web-sovelluksena, sillä sovellus on sama molemmissa tapauksissa. Yhdenmukaisuuden vuoksi käyttäjien ei tarvitse opetella montaa toisistaan poikkeavaa käyttöliittymää. Tämä tekee progressiivisten web-sovellusten käytöstä eri ympäristöissä vähemmän työlästä verrattuna tilanteeseen, jossa natiivisovelluksen lisäksi on täytynyt toteuttaa erillinen verkkosivu.

Myös nopeat sivuston latausajat parantavat käyttäjäkokemusta. Lähes 70 % kuluttajista myöntää latausajan vaikuttavan ostohalukkuuteen verkkokauppoista. (Baker, 2022) Progressiiviset web-sovellukset lyhentävät latausaikaa palvelusuorittajien avulla. Vuonna 2017 Twitter kehitti sovelluksestaan progressiivisen web-sovelluksen. Twitterillä oli sovelluksen julkaisuhetkellä noin 328 miljoonaa käyttäjää, joista noin 80 % oli mobiilikäyttäjiä. Twitterin tapausesimerkissä progressiivinen web-sovellus lisäsi käyttäjien tekemän selailun määrää noin 65 % vierailua kohden, lisäsi palvelussa lähetettyjä tekstipohjaisia viestejä eli tweettejä noin 75 %, sekä madalsi käyttäjien poistumisprosenttia noin 20 %. Näiden hyötyjen lisäksi sovellus varasi alle 3 % alkuperäisestä natiivisovelluksen varaamasta tallennustilasta, minkä ansiosta käyttäjille jää huomattavasti enemmän tallennustilaa käyttöön esimerkiksi muiden sovellusten asentamista varten. (“Twitter Lite PWA Significantly Increases Engagement and Reduces Data Usage”, 2017)

Myös datan käyttö väheni jopa 70 % taustasuorittajien mahdollistaman välimuistin hyödyntämisen ansiosta. (“Twitter Lite PWA Significantly Increases Engagement and Reduces Data Usage”, 2017) Vähäisempi datan käyttö voi auttaa käyttäjiä säästämään rahaa, erityisesti alueilla, joissa mobiilidata on kallista.

5. VERTAILU

Aiemmissa luvuissa selvisi, että sekä progressiivisilla web-sovelluksilla että natiivisovelluksilla on etuja, joita toinen vaihtoehto ei pysty tarjoamaan nykyteknologian rajoitteista johtuen. Mobiilisovellusta toteutettaessa tulee siis pohtia vaatimuksia sekä prioriteetteja tapauskohtaisesti ja tehdä valinta niiden perusteella. Tässä luvussa vertaillaan natiivisovelluksen ja progressiivisen web-sovelluksen etuja sekä heikkouksia ja esitellään taulukko kummankin ominaisuuksista valintaprosessin tukemiseksi.

5.1 Kehittäminen

Natiivisovellusten ja progressiivisten web-sovellusten erot kehittämisessä liittyvät muutamaa lukuun ottamatta kehittämisessä käytettyjen teknologioiden eroihin. Natiivisovellukset kehitetään kunkin alustan omalla ohjelmointikielellä. Progressiivisten web-sovellusten kehittämisessä ovat käytössä web-teknologiat: HTML, CSS (porrastetut tyyliarkit, engl. cascading style sheets) ja JavaScript.

Selainpohjaisuutensa vuoksi progressiivisten web-sovellusten toiminnalle on keskeistä, että selaimet tarjoavat kattavan tuen niiden vaatimille ominaisuuksille. Kuten luvussa 4 huomattiin, Safari-selaimen tuki progressiivisten web-sovellusten eri ominaisuuksille on Google Chrome-selainta puutteellisempi. Tämä tekee iOSkäyttöjärjes-telmän tuesta prog-ressiivisille web-sovelluksille Android-käyttöjärjestelmää heikomman, sillä Safari on iOS:än ainoa progressiivisia web-sovelluksia tukeva selain.

Table 5.1. *Natiivisovellusten ja progressiivisten web-sovellusten erot eri ominaisuuksissa*

Ominaisuus	**Toteutustapa**	
	Natiivisovellus	**Progressiivinen web-sovellus**
Suorituskyky	-Parempi suorituskyky	-Heikompi suorituskyky
Kehityksen kustannukset	-Keskimäärin kalliimpi	-Keskimäärin halvempi
Toimitusaika	-Keskimäärin hitaampi	-Keskimäärin nopeampi
Tallennustilan käyttö	-Huomattavasti suurempi	-Huomattavasti pienempi
Ylläpito	-Vaikeampi ylläpitää -Sovelluksen päivittäminen on monimutkaisempaa. -Muutokset käyttöjärjestelmässä voivat aiheuttaa ongelmia.	Helpompi ja yksinkertaisempi ylläpitää
Järjestelmä-riippumattomuus	Kalliimpi ja hitaampi toteuttaa, vaatii kaksi erillistä toteutusta.	Vain yksi toteutus riittää.
Löydettävyys	-Vaikeampi löytää -Etsitään pääasissa sovelluskaupoista.	-Helpompi löytää suoraan hakukoneella -Lisättävissä sovelluskauppoihin
Maksut sovelluksessa	-Helpompi tehdä maksulliseksi -Sovelluksen sisäiset maksut on helppo toteuttaa.	-Vaikeampi tehdä maksulliseksi -Sovelluksen sisäiset maksut on helppo toteuttaa.

Taulukosta 5.1. voidaan nähdä natiivisovellusten ja progressiivisten web-sovellusten edut ja heikkoidet suhteessa toisiinsa. Toteutustapaa valitessa taulukosta voidaan valita toteutettavalle sovellukselle keskeisimmät ominaisuudet ja käyttää taulukkoa tukena valinnalle. Esimerkiksi jos hyvä suorituskyky on sovellukselle keskeisin ominaisuus ja kehityksen kustannuksissa ei tarvitse säästää resursseja, voi natiivisovellus olla parempi vaihtoehto. Toisaalta, jos pieni tallennustilan käyttö, halvemmat kehitykustannukset ja yksinkertainen ylläpito ovat tärkeitä kehitettävän sovelluksen ominaisuuksia, progressiivinen web-sovellus täyttää kriteerit paremmin.

Kuten aiemmin mainittiin, progressiivisia web-sovelluksia on mahdollista käyttää sovellusta ilman asentamista. Kuten luvussa 3 mainittiin, mobiilikäyttäjät eivät asenna keskimääräisen kuukauden aikana yhtäkään sovellusta, ja lisäksi käyttävät aktiivisesti vain yhdeksää sovellusta päivän aikana (“Mobile app Download Statistics”, 2022). Toisaalta keskimääräinen käyttäjä tekee päivässä 3-4 hakua Googlen hakukoneella. (Mohsin, 2022). Tässä progressiivisten web-sovellusten etu on siis se, että niitä voidaan hakea hakukoneella ja käyttää perinteisenä verkkosivuna.

5.1.1 Kehittämiskustannukset

Kustannukset ovat keskeinen huomioitava asia tehdessä valintaa natiiviovelluksen ja progressivisen web-sovelluksen välillä. Taulukon 5.1. mukaisesti progressiivinen web-sovelluksen kehityskustannukset ovat keskimäärin natiivisovellusta pienemmät. Mataliin kustannuksiin vaikuttaa moni luvussa 4 mainittu progressiivisten web-sovellusten kehittämisetu. Progressiivisten web-sovellusten kohdalla yksi toteutus riittää alustariippumattomaan toteutukseen, ja ylläpito on yksinkertaisempaa, sillä päivityksiä ei tarvitse viedä erikseen sovelluskaupan hyväksyntään. Lisäksi käyttöjärjestelmän päivitykset eivät aiheuta ongelmia sovelluksen toiminnassa, sillä riittää, että selaimet toimivat päivityksen jälkeen.

Yksittäiselle harrastuksena ohjelmoivalle kehittäjälle Applen App Storen vuosittainen 99 euron maksu sovelluksen julkaisuoikeuksista voi olla suuri määrä rahaa. Web-sovelluksen tapauksessa on mahdollista selvitä täysin ilman kuluja, jos kehittäjä luo itse oman sovelluksensa ja julkaisee sen ilmaista isännöintiä tarjoavassa palvelussa, kuten Firebasessa. Toisaalta web-sovelluksen tapauksessa sovelluksen isännöinti sekä verkkotunnuksen ostaminen voivat tuoda suurenkin määrän kustannuksia, jos sovelluksella on paljon käyttäjiä ja verkkotunnus on kallis.

5.1.2 Käyttäjäkokemus

Käyttäjäkokemus natiivisovellusten ja progressiivisten web-sovellusten välillä on samankaltainen, sillä progressiivisten web-sovellusten on tarkoitus muistuttaa toiminnaltaan natiivisovelluksia. Eroavaisuudet toteutustapojen välillä voivat kuitenkin aiheuttaa toisistaan poikkeavan käyttäjäkokemuksen.

Sovelluksen varaama tallennustilan suuri määrä voi olla este sovelluksen asentamiselle. Kuten taulukossa 5.1. mainitaan, progressiiviset web-sovellukset vievät huomattavasti natiivisovelluksia vähemmän tallennustilaa. Käyttäjät saattavat kyllästyä ja luovuttaa sovel-

luksen etsimisen, jos sovelluksen löytäminen on vaikeaa. Taulukon 5.1. mukaisesti progressiivisten web-sovellusten löytäminen on pääsääntöisesti helpompaa, jos kehittäjät ovat mahdollistaneet niiden löytämisen sekä sovelluskaupoista että hakukoneiden avulla. Toisaalta natiivisovellusten käyttäjäkokemus voi olla parempi offline-toiminnallisuuden kannalta. iOS-järjestelmä mahdollistaa progressiivisille web-sovelluksille vain 50 megatavun verran välimuistin käyttöä, mikä rajoittaa suurten progressiivisten web-sovellusten toimintaa ilman internet-yhteyttä.

6. YHTEENVETO

Tutkielman tutkimuskysymys oli, että millaisissa tilanteissa mobiilisovellus on natiivisovelluksen sijaan kannattavaa toteuttaa progressiivisena web-sovelluksena. Kannattavuutta tutkittiin usean eri ominaisuuden kautta. Vertailussa huomioitiin esimerkiksi suorituskyky, kustannustehokkuus ja toimituksen nopeus.

Tutkielmassa selvisi, että progressiivinen web-sovellus on vaihtoehdoista edullisempi, erityisesti usealle käyttöjärjestelmälle toteutettaessa. Lisäksi progressiivinen web-sovellus on nopeampi toimittaa ja sen käyttämille teknologoille löytyy enemmän kehittäjiä sekä kirjastoja.

Natiivisovelluksella on kuitenkin yhä mahdollista saada parempi suorituskyky sekä hyödyntää mobiililaitteen natiiviominaisuuksia laajemmin. Progressiiviset web-sovellukset eivät esimerkiksi pysty hyödyntämään GPS:ää (maailmanlaajuinen paikallistamisjärjestelmä, engl. global positioning system) iOS-käyttöjärjestelmässä. Ennen toteutustavan valintaa on siis tärkeä miettiä, mitä laitteen ominaisuuksia sovelluksen tarvitsee voida käyttää.

Mobiililaitteiden laitteiston ominaisuuksien käytöstä progressiivisilla web-sovelluksilla on kuitenkin tehty viimeisimpien vuosien aikana huomattavasti helpompaa. Web-ohjelmointirajapinnat mahdollistavat lähes kaikkien laitteiston natiiviominaisuuksien hyödyntämisen Android-käyttö-järjestelmällä, ja iOS:in natiiviominaisuuksien hyödyntämisestä tehdään tasaisesti helpompaa käyttöjärjestelmän päivitysten myötä.

Jos nopeat latausajat, tehokkuus sekä puhelimen natiiviominaisuuksien laaja hyödyntäminen ovat suunnitellussa sovelluksessa tärkeitä, saattaa natiivisovellus olla parempi vaihtoehto. Esimerkiksi raskasta prosessointia vaativat pelit kannattaa yleensä toteuttaa natiivisovelluksina. Mikäli sovelluksen kehityksen edullisuus, joustavuus ja nopeus ovat tärkeitä, eikä suorituskyvyn tarvitse olla optimaalinen, progressiivinen web-sovellus voi olla parempi vaihtoehto sovellukselle.

Toteutustapaa harkitessa on kuitenkin tärkeää ajatella myös sovelluksen tulevaisuutta. Jos tulevaisuudessa sovellus tulee vaatimaan raskaampaa prosessointia tai esimerkiksi verkkoyhteydetöntä suoratoistoa, on natiivisovellus nykyteknologian rajoitteissa ainoa vaihtoehto tässä tutkimuksessa käsitellyistä toteutustavoista.

REFERENCES

Alternative distribution options. (2020). Retrieved April 19, 2022, from https://developer.android.com/distribute/marketing-tools/alternative-distribution

Application fundamentals. (2021). Retrieved April 24, 2022, from https://developer.android.com/guide/components/fundamentals

Avoin tiede:löydettävyys. (2022). Retrieved April 8, 2022, from https://tieteentermipankki.fi/wiki/Avoin_tiede:l%C3%B6ydett%C3%A4vyys

Baker, K. (2022). 11 website page load time statistics you need [+ how to increase conversion rate]. Retrieved April 23, 2022, from https://blog.hubspot.com/marketing/page-load-time-conversion-rates

Bar, A. (2022). What web can do today. Retrieved April 24, 2022, from https://whatwebcando.today/

Behl, K., & Raj, G. (2018). Architectural pattern of progressive web and background synchronization. *2018 International Conference on Advances in Computing and Communication Engineering (ICACCE)*, 366–371. https://doi.org/10.1109/ICACCE.2018.8441701

Chaurasia, S., & Peng, N. (2022). Canalys: Apple retakes top spot in global smartphone market in q4 2021. Retrieved February 23, 2022, from https://www.canalys.com/newsroom/canalys-global-smartphone-market-Q4-2021

Ching, C. (2019). How to submit your app to the app store in 2019. Retrieved April 9, 2022, from https://codewithchris.com/submit-your-app-to-the-app-store/

Codename one. (2021). Retrieved February 20, 2022, from https://www.codenameone.com/

Firtman, M. (2022). Push notifications, webxr, and better pwa support coming to ios-firt.dev. Retrieved April 23, 2022, from https://firt.dev/ios-15.4b

Gandhi, M. (2021). Do americans trust tech giants? Retrieved May 6, 2022, from https://www.seoclarity.net/blog/americans-and-trusting-tech

Gelzinis, G. (2021). Almost 1,000 new applications are published on apple app store daily. Retrieved May 6, 2022, from https://esomar.org/newsroom/almost-1-000-new-applications-are-published-on-apple-app-store-daily

Gómez-Sierra, C. J. (2021). Design and development of a PWA - progressive web application, to consult the diary and programming of a technological event. *IOP Conference Series: Materials Science and Engineering*, *1154*(1), 012047. https://doi.org/10.1088/1757-899x/1154/1/012047

Google. (2021). Adding your progressive web app to google play. Retrieved April 23, 2022, from https://developers.google.com/codelabs/pwa-in-play#0

How do i distribute ios apps without app store google play? (2018). Retrieved April 19, 2022, from https://www.3dissue.com/help/knowledge-base/submission/how-do-i-distribute-apps-outside-of-the-app-store-google-play/

Kernel. (2020). Retrieved April 19, 2022, from https://source.android.com/devices/architecture/kernel

Laricchia, F. (2022). Mobile os market share 2021. Retrieved March 2, 2022, from https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/

Lighthouse pwa analysis tool. (2019). Retrieved February 25, 2022, from https://developers.google.com/web/ilt/pwa/lighthouse-pwa-analysis-tool

Mobile app download statistics. (2022). Retrieved April 13, 2022, from https://buildfire.com/app-statistics/

Mohsin, M. (2022). 10 google search statistics you need to know in 2022. Retrieved April 13, 2022, from https://www.oberlo.com/blog/google-search-statistics

Muman, J. (2021). Progressive web apps: An optimistic approach to traditional application development.

Osmani, A. (2019). *The app shell model*. Retrieved February 20, 2022, from https://developers.google.com/web/fundamentals/architecture/app-shell

Rakowski, F., Grzybowska, K., & Karwatka, P. (2019a). The history of pwa development: The pwa book. Retrieved April 9, 2022, from https://www.divante.com/pwabook/chapter/02-the-history-of-pwas

Rakowski, F., Grzybowska, K., & Karwatka, P. (2019b). The history of pwa development: The pwa book. Retrieved April 23, 2022, from https://www.divante.com/pwabook/chapter/02-the-history-of-pwas

Richard, S., & LePage, P. (2020). What are progressive web apps? Retrieved February 20, 2022, from https://web.dev/what-are-pwas/

Rykov, S. (2022). Progressive web app development: How to cook pwa in 2022. Retrieved April 12, 2022, from https://mobidev.biz/blog/why-when-use-progressive-web-app-pwa-development

Schiela, J. (2020). The state of app discovery. Retrieved April 8, 2022, from https://www.businessofapps.com/insights/the-state-of-app-discovery/

Service worker api. (2022). Retrieved February 20, 2022, from https://developer.mozilla.org/en-US/docs/Web/API/Service_Worker_API

Share of global mobile website traffic 2015-2021. (2022). Retrieved February 20, 2022, from https://www.statista.com/statistics/277125/share-of-website-traffic-coming-from-mobile-device/

Sharma, A. (2022). Step-by-step process to upload app to google play store. Retrieved April 9, 2022, from https://appinventiv.com/blog/how-to-submit-app-to-google-play-store/

Tran, C. (2021). Pwas on ios 15: Improvements? - simicart. Retrieved April 23, 2022, from https://www.simicart.com/blog/pwa-ios-15/

Twitter lite pwa significantly increases engagement and reduces data usage. (2017). Retrieved February 25, 2022, from https://developers.google.com/web/showcase/2017/twitter

Vailshery, L. S. (2021). Most used languages among software developers globally 2021. Retrieved March 2, 2022, from https://www.statista.com/statistics/793628/worldwide-developer-survey-most-used-languages/

Vailshery, L. S. (2022). Mobile browser market share worldwide 2012-2021. Retrieved April 19, 2022, from https://www.statista.com/statistics/263517/market-share-held-by-mobile-internet-browsers-worldwide/

What is an ipa file and how can you open one? (2022). Retrieved May 6, 2022, from https://www.appmysite.com/blog/what-is-an-ipa-file-and-how-can-you-open-one/