jamk

Selvitys erilaisista vaihtoehdoista muuntaa PWA-sovellus APK-paketiksi

Tomi Naukkarinen

Opinnäytetyö
Toukokuu 2022
Liiketalouden ala
Tradenomi (AMK), tietojenkäsittelyn tutkinto-ohjelma

jamk | Jyväskylän ammattikorkeakoulu
University of Applied Sciences

jamk

Kuvailulehti

Naukkarinen, Tomi

Selvitys erilaisista vaihtoehdoista muuntaa PWA-sovellus APK-paketiksi.

Jyväskylä: Jyväskylän ammattikorkeakoulu. Toukokuu 2022, 37 sivua.

Liiketalouden ala. Tietojenkäsittelyn tutkinto-ohjelma. Opinnäytetyö AMK.

Julkaisun kieli: suomi

Julkaisulupa avoimessa verkossa: kyllä

Tiivistelmä

Mobiiliselainten käyttö on ollut suuressa kasvussa viime vuosina. Tämän seurauksena erilaiset verkkosovellukset, kuten esimerkiksi PWA-sovellukset (Progressive Web App) ovat yleistyneet huomattavasti. Koska PWA-sovellukset toimivat suoraan verkkoselaimessa, se mahdollistaa sovelluksen toimimisen laitteessa käytännössä sen käyttöjärjestelmästä riippumatta.

Monesti pelkkä PWA-sovellus ei kuitenkaan riitä, vaan siitä halutaan versio myös sovelluskauppaan. Sovelluksen saaminen Google Play-kauppaan voi olla iso tekijä näkyvyyden ja saatavuuden kannalta varsinkin pienempien yritysten tai yksityishenkilöiden sovelluksille. Ongelmana kuitenkin on, että PWA-sovellusta ei pysty suoraan julkaisemaan sovelluskaupassa. Tästä syystä PWA-sovelluksen muuntaminen APK-paketiksi on hyödyllinen vaihtoehto, ettei sovelluskauppaan tarvitse tehdä erikseen toista natiivia Android-sovellusta.

Tutkimuksen tavoitteena oli tehdä tutkimus, jossa selvitetään erilaisia vaihtoehtoja muuntaa PWA-sovellus APK-paketiksi, eli sovelluskauppaan julkaistavaksi Android-sovellukseksi. Tutkimuksessa selvitetään, millaisilla erilaisilla tavoilla kyseinen muunnos on mahdollista tehdä, sekä tutkitaan ja tehdään vertailua niiden välillä. Tutkimus toteutettiin kvalitatiivisena, eli laadullisena tutkimuksena.

Tutkimuksessa luotiin yksinkertainen PWA-sovellus Angular-ohjelmistokehyksellä, jota muutettiin erilaisilla internetistä löytyvillä toimintatavoilla APK-paketiksi. Muunnoksen vaiheet ja niiden aikana ilmenneet havainnot raportointiin ja niistä nostettiin esiin menetelmien hyötyjä, haittoja ja eroavaisuuksia.

Tuloksena saatiin selvitys erilaisista muuntamismenetelmistä ja selvitykset niiden hyödyistä, haitoista ja eroavuuksista. Havainnoista ja tuloksista luotiin vertailua, josta saatiin tehtyä johtopäätöksiä. Tulokset vastaavat suoraan tutkimuskysymyksiin.

Muuntamismenetelmiä löytyi neljä, joilla kolmesta onnistuttiin muuntamaan PWA-sovellus APK-paketiksi. Muuntamismenetelmistä nopeimpia muunnoksessa olivat ne, jotka toimivat pilvipalveluna verkossa. Pilvipalveluna toimivat olivat myös muuntamismenetelmistä helpoimmat toteuttaa.

Avainsanat (asiasanat)

Progressiivinen verkkosovellus, APK-paketti, verkkosovellus, mobiilisovellukset

Muut tiedot (salassa pidettävät liitteet)

jamk

Description

Naukkarinen, Tomi

Explanation of different options for converting a PWA application to an APK package

Jyväskylä: JAMK University of Applied Sciences, April 2022, 37 pages.

Business Information Technology. Degree Programme in Business Information Technology. Bachelor's thesis.

Permission for open access publication: Yes

Language of publication: Finnish

Abstract

The use of mobile browsers has been on the rise recently. As a result, various web applications, such as PWA (Progressive Web App) applications, have become significantly more common. Because PWA applications run directly in the web browser, it allows the application to run on the device regardless of its operating system.

In many cases, however, a PWA application alone is not enough, and a version of it is also desired in the application store. Getting the app to the Google Play Store can be a big factor in visibility and accessibility for applications from very small businesses or individuals. The problem, however, is that the PWA application cannot be published directly in the app store. For this reason, converting a PWA app to an APK package is a useful option without having to make another native Android app separately for the app store.

The aim of the study was to investigate different options for converting a PWA application to an APK package, an Android application to be published in an application store. The study examines the different ways in which this conversion can be made and examines and compares them. The study was carried out as qualitative research.

A simple PWA application was created with the Angular software framework, which was transformed into an APK package using various modes available on the Internet. The steps of the transformation and the findings that emerged during them were reported and highlighted the advantages, disadvantages, and differences of the methods.

The result was an explanation of the different conversion methods and their advantages, disadvantages, and differences. A comparison was made of the findings and results, from which conclusions were drawn. The results directly answer the research questions.

Four conversion methods were found, three of which succeeded in converting a PWA application to an APK package. Of the conversion methods, the fastest in conversion were those that operated as a cloud service online. The methods used as a cloud service were also the easiest to implement.

Keywords/tags (subjects)

Progressive Web app, APK-package, web application, mobile applications

Miscellaneous (Confidential information)

Sisältö

1 Johdanto 6
2 Tutkimusasetelma 7
2.1 Tutkimuksen tarkoitus ja tavoitteet 7
2.2 Tutkimuksen rajaukset 7
2.3 Tutkimuskysymykset 7
2.4 Tutkimusmenetelmät 8
3 Sovellusalustat 8
3.1 Web- ja PWA-sovellukset 8
3.2 Android 10
4 Muuntamismenetelmät 11
4.1 PWA2APK 11
4.2 Bubblewrap 11
4.3 PWA Builder 11
4.4 PwaToTwa 12
5 Toteutus 12
5.1 Muuntaminen PWA2APK-työkalulla 12
5.2 Muuntaminen PWABuilder-työkalulla 17
5.3 Muuntaminen Bubblewrap-työkalulla 22
5.4 Muuntaminen PwaToTwa-työkalulla 27
6 Tulokset 28
6.1 PWA2APK-työkalun tulokset 28
6.2 PWABuilder-työkalun tulokset 29
6.3 Bubblewrap-työkalun tulokset 30
6.4 PwaToTwa-työkalun tulokset 31
6.5 Tulosten taulukointi 31
7 Johtopäätökset 32
8 Pohdinta 34
8.1 Tutkimuksen onnistuminen 34
8.2 Luotettavuus 35
Lähteet 36

Kuviot

Kuvio 1. PWA-sovelluksen URL-osoite syötetty sille tarkoitettuun kenttään 13
Kuvio 2. Sovelluksen nimen ja kuvakkeen määritys 14
Kuvio 3. Sovelluksen generointi 14
Kuvio 4. Assetlinks.json-tiedoston asettaminen oikeaan polkuun 15
Kuvio 5. Tiedostot valmiina lataamiseen 16
Kuvio 6. Mahdollinen tartunta 17
Kuvio 7. PWA-sovelluksen URL-osoite syötetty PWABuilderiin 18
Kuvio 8. Analyysin tulokset. Sovellus ei kaipaa parannusta 19
Kuvio 9. Microsoft-, Google Play- ja Ios App-kauppojen paketit 20
Kuvio 10. Viimeiset asetukset ja "Generate"-painike 21
Kuvio 11. Ladattu ja purettu ZIP-tiedosto, joka sisältää APK ja AAB-tiedostot 22
Kuvio 12. Bubblewrap CLI asennettu 23
Kuvio 13. JDK 11 polku määritetty ja Android SDK asennettu 24
Kuvio 14. Domain ja URL-polku määritetty 24
Kuvio 15. Tietoja, jotka vaikuttavat siihen miltä sovellus näyttää laitteella 25
Kuvio 16. Sovelluksen käynnistyskuvakkeen määrittäminen 25
Kuvio 17. Uuden Keystoren luonti 27

Taulukot

Taulukko 1. Muuntamisessa onnistuneiden muuntamismenetelmien vertailu 32

1 Johdanto

Vuonna 2020 Perficient Inc:n tekemän tutkimuksen mukaan maailmanlaajuisesti 68,1 % kaikille verkkosivuille tulleista käynneistä tulivat mobiiliselaimien kautta (Enge 2021). Tutkimus osoittaa, että mobiiliselaimen käyttö on erittäin suurta ja tulee varmasti kasvamaan lähivuosina entisestään. Koska mobiiliselaimen käyttö on selkeästi kasvamassa, se on varmasti osasyynä sille, että PWA-sovellukset (Progressive Web Application) ovat yleistyneet viime aikoina. Syy tälle on se, että PWA-sovellukset on rakennettu niin, että ne toimivat suoraan verkkoselaimessa. Sovelluksen toimiminen verkkoselaimessa mahdollistaa sovelluksen toimimisen laitteessa käytännössä sen käyttöjärjestelmästä riippumatta ja se toimii kuin tavallinen mobiilisovellus, vaikka se on verkkosovellus. PWA-sovellus on mahdollista myös ladata selaimen kautta käyttäjän laitteelle. (Richard & LePage 2020.)

Vaikka PWA-sovellukset ovat joustavia käyttöjärjestelmän ja laitteiden suhteen, joskus pelkkä PWA-sovellus ei kuitenkaan riitä, vaan sovellus halutaan laittaa myös sovelluskauppoihin ladattavaksi. Syy tälle on se, että sovelluksen saaminen Google Play-kauppaan voi olla iso tekijä näkyvyyden ja saatavuuden kannalta varsinkin pienempien yritysten tai yksityishenkilöiden sovelluksille. Usein sovellukset tai osa niistä halutaan asettaa maksulliseksi. Tästä syystä sovelluskauppaan julkaiseminen on myös tämän vuoksi järkevää, koska sovelluskaupan kautta sen saa asetettua helposti maksulliseksi. Ongelmana kuitenkin on se, että PWA-sovellusta ei pysty suoraan julkaisemaan Google Play-kauppaan, koska se ei ole natiivi sovellus. Näistä syistä PWA-sovelluksen muuntaminen APK-paketiksi, eli Android sovellukseksi on varmasti tarpeellinen vaihtoehto, jotta koko sovelluksesta ei tarvitsisi kehittää erikseen sovelluskauppaan kelpaavaa natiivia Android-sovellusta.

Tämän tutkimuksen tuloksena syntyy selvitykset toimintatavoista, joilla muuntaminen PWA-sovelluksesta APK-paketiksi onnistuu. Tutkimuksen tuloksia pystyy hyödyntämään ihmiset, joilla on tarve muuntaa PWA-sovellus APK-paketiksi. Tällaisen tarpeen omaava pystyy tuloksista vertailemaan erilaisia toimintatapoja ja katsomaan hänelle sopivimman tavan tehdä kyseinen muunnos.

2 Tutkimusasetelma

Tässä luvussa käydään läpi tutkimuksen tarkoitus, tavoitteet, rajaukset sekä tutkimuskysymykset. Viimeisessä kappaleessa kerrotaan tutkimusmenetelmät, joiden avulla tutkimuskysymyksiin pysytään vastaamaan.

2.1 Tutkimuksen tarkoitus ja tavoitteet

Tämän opinnäytetyön tavoitteena on tehdä tutkimus, jossa selvitetään erilaisia vaihtoehtoja muuttaa PWA-sovellus APK-paketiksi, eli Android-sovellukseksi. Tutkimuksessa selvitetään, millaisilla erilaisilla tavoilla kyseinen muunnos on mahdollista tehdä, sekä tutkitaan ja tehdään vertailua niiden välillä. Tutkimuksessa luodaan yksinkertainen PWA-sovellus Angular-ohjelmistokehyksellä, jota muutetaan erilaisilla internetistä löytyvillä toimintatavoilla APK-paketiksi. Tutkimus toteutetaan kvalitatiivisena, eli laadullisena tutkimuksena.

2.2 Tutkimuksen rajaukset

Tähän opinnäytetyöhön on kerätty internetistä tehdyillä hauilla neljä erilaista menetelmää, joilla PWA-sovelluksen pystyy muuntamaan APK-paketiksi. APK-paketiksi muuntaminen rajaa jo itsessään tutkimuksen tarkoituksen sellaisille menetelmille, jotka pystyvät muuntamaan PWA-sovelluksen muotoon, joka on mahdollista julkaista vain Google Play-kaupassa.

Tutkimuksen toteutusvaiheessa olevia muuntamisvälineitä tullaan käyttämään Microsoft Windows-käyttöjärjestelmällä, joten tutkimuksesta ei selviä, kuinka muuntamismenetelmät toimivat muilla käyttöjärjestelmillä.

2.3 Tutkimuskysymykset

Tutkimukseen valitut tutkimuskysymykset on valittu niillä perustein, jotka selvittävät ja tutkivat tutkimuksen aihetta mahdollisimman selkeästi ja kattavasti.

Tutkimuskysymykset ovat:

- Mitä erilaisia vaihtoehtoja on olemassa, joilla voi muuntaa PWA-sovelluksen APK-paketiksi?

- Mitkä ovat menetelmien hyödyt, haitat ja eroavaisuudet?

2.4 Tutkimusmenetelmät

Opinnäytetyön päätarkoituksena on selvittää ja tutkia erilaisia menetelmiä muuntaa PWA-sovellus APK-paketiksi. Selvitys tulee toteutumaan keräämällä aineistoa internetistä löytyvistä artikkeleista, dokumenteista ja eri muuntamismenetelmien omista dokumentaatioista. Aineistoa saadaan myös havainnoimalla, koska kyseisiä muuntamismenetelmiä tullaan kokeilemaan ja tutkimaan käytännössä. Näistä syistä tähän tutkimukseen valikoitui tutkimusmenetelmiksi kvalitatiivinen eli laadullinen tutkimus, havainnointi sekä saatujen tulosten analysoiminen.

Jorma Kananen kertoo, että laadullinen tutkimus soveltuu parhaiten tilanteisiin, joissa muun muassa ilmiöstä ei ole vielä tutkimusta, teoriaa tai tietoja, halutaan saada perustavanlaatuinen näkemys ilmiöstä, luodaan erilaisia hypoteeseja ja teorioita tai halutaan saada kattava kuvaus ilmiöstä (Kananen 2017, 33). Näihin tilanteisiin pohjautuen kvalitatiivinen eli laadullinen tutkimus tulee olemaan tutkimusmenetelmänä tässä opinnäytetyössä, koska tässä tutkimuksesta ei ole olemassa vielä toista tutkimusta sekä tutkimuksen aiheesta halutaan saada kattava kuvaus ja näkemys.

Kirsi Juhila kirjoittaa tekstissään kvalitatiivisesta tutkimuksesta, että sen aineistona käytetään empiiristä aineistoa, kuten esimerkiksi tekstejä, keskusteluja, haastatteluja ja havainnointipäiväkirjoja (Juhila n.d). Tästä syystä aineistonkeruumenetelmiä ovat tekstit, dokumentaatiot ja omat kerätyt havainnot menetelmien kokeilusta käytännössä. Dokumentaatiot ja havainnot muuntamismenetelmien kokeilemisesta antavat kattavan kuvan muuntamismenetelmästä, joita kuka tahansa voi hyödyntää muuntaessaan PWA-sovellusta APK-paketiksi.

3 Sovellusalustat

3.1 Web- ja PWA-sovellukset

PWA-sovellus eli Progressive Web Application on verkkoselaimessa toimiva verkkosivu tai sovellus. Koska se on verkkoselaimeen kehitetty sovellus, se mahdollistaa sen käytön käyttöjärjestelmästä riippumatta. Se käytännössä toimii millä tahansa mobiililaitteella, jossa on asennettuna

verkkoselain. PWA-sovellukset toimivat kuin tavalliset natiivit mobiilisovellukset, ne tarjoavat samat ominaisuudet ja käyttäjäkokemukset, vaikka ovatkin verkkosovelluksia. PWA-sovellukset on mahdollista ladata myös käyttäjän omalle laitteelle. (Richard & LePage 2020.)

Service Worker on JavaScript-resurssi. Ne toimivat välityspalvelimina verkkopalvelimen ja verkkoselaimen välillä. Ne parantavat luotettavuutta tarjoamalla sivustolle tai sovellukselle offline-toimintoja sekä ne tehostavat sivun suorituskykyä. Esimerkiksi ne tallentavat välimuistiin tietoja, hakevat tietoja välimuistista ja toimittavat Push-viestejä (Introduction to Service Worker 2019.)

Web App Manifest on JSON-tiedosto, joka kertoo verkkoselaimelle, kuinka PWA-sovelluksen pitäisi näyttää ja käyttäytyä, kun se on asennettuna laitteelle. Esimerkiksi se sisältää tiedot URL-osoitteesta, joka avataan, kun sovellus käynnistetään. Se sisältää myös sovelluksen nimen ja ikonit, joita sovellus käyttää. (LePage, P., Beaufort, F. & Steiner, T. 2021.)

Trusted Web Activities (TWA) eli luotetut verkkotoiminnot ovat Google Chromen mukautettujen välilehtien toimintatila, jossa ei ole verkkoselaimen käyttöliittymää ja jotka ovat käytettävissä verkkosisällössä (Feature: Trusted Web Activities 2021). Se toimii siis kokonäytön-tilassa. Sen sisältö on luotettua, koska sovellus ja sen avaama sivusto on peräisin samalta kehittäjältä. Se varmistetaan Digital Asset Linksillä. Käyttäjälle tämä paketti näyttää ja toimii kuin natiivisovellus. TWA:t voidaan julkaista Google Play Kaupassa. (LePage & Bandarra 2020.)

Google Lighthouse on Googlen luoma avoimeen lähdekoodiin perustuva automatisoitu työkalu, joka on suunniteltu parantamaan verkkosivujen laatua. Työkalun voi ajaa millä tahansa verkkosivulla ja se tarkastelee verkkosivun suorituskykyä, saatavuutta, progressiivisen verkkosovelluksen kriteerejä, SEO-määrityksiä ja parhaita käytäntöjä. Se antaa raportin sivun toimivuudesta ja kertoo, mitä asioita verkkosivulla täytyy korjata, jotta edellä mainituista tarkastelun kohteista saisi parhaimmat tulokset. (Lighthouse 2021.) Mitä parempi tulos, sitä paremmin verkkosivusto toimii.

3.2 Android

APK-paketti eli Android Package Kit on tiedostomuoto, jota Android käyttää jakaakseen ja asentaakseen sovelluksia. APK-paketti sisältää kaikki tarvittavat elementit, joita tarvitaan asennettaessa sovellusta oikein laitteelle. Ne ovat kuin .ZIP- tai .RAR-tiedostot, eli ne ovat pakattuja tiedostoja, mutta sisältävät kuitenkin lisätietoja toimiakseen oikein APK-tiedostona. (Stegner 2021.)

Android App Bundle on julkaisumuoto Google Play-sovelluksille. Se sisältää kaiken sovelluksen kootun koodin ja resurssit. Google Play käyttää sovelluspakettia optimoitujen APK-tiedostojen luomiseen ja jakamiseen eri laitekokoonpanoille. Elokuusta 2021 lähtien, uudet sovellukset täytyvät olla Android App Bundle eli AAB-tiedostoja, jotta ne voi julkaista Google Play-kauppaan. (About Android App Bundles 2022.)

Digital Asset Links on todennusmenetelmä. Digital Asset Links-protokolla ja API mahdollistavat sen, että verkkosivusto tai sovellus voi tehdä todennettavia julkisia lausuntoja muista sovelluksista tai verkkosivustoista. Sen avulla verkkosivusto voi esimerkiksi ilmoittaa, että se kuuluu tiettyyn Android-sovellukseen tai että se jakaa käyttäjätunnukset toisen verkkosivuston kanssa. (Digital Asset Links Overview 2022.) Sillä voidaan myös todentaa, että sovellus ja verkkosivusto ovat saman kehittäjän omaisuutta (Lepage & Bandarra 2020).

Keystore-tiedosto on binääritiedosto, joka toimii varmenteena ja yksityisen avainten arkistona. **(**Sign your app 2022). Tiedostoa tarvitaan, kun ensimerkiksi rakentaa sovelluksesta AAB- tai APK-tiedoston Android Studiolla.

Natiivisovellus on ohjelmisto, joka on suunniteltu toimimaan tietyssä laitteessa ja sen käyttöjärjestelmässä (Gillis n.d). Kuten esimerkiksi Google Play kaupasta löytyvät sovellukset ovat suunniteltu toimimaan laitteilla, joissa on Android-käyttöjärjestelmä tai vastaavasti kuten Applen App Storesta löytyvät sovellukset ovat suunniteltu toimimaan iOS-käyttöjärjestelmällä toimivilla laitteilla. Natiivisovellukset asennettaan suoraan laitteen omaan muistiin.

4 Muuntamismenetelmät

Tällä hetkellä neljällä seuraavalla menetelmällä on mahdollista muuntaa PWA-sovellus APK-paketiksi. Muuntamismenetelmät on löydetty internetistä tehdyillä hauilla. Hakusanoina käytettiin muun muassa: ”PWA to APK”,” How to convert PWA to APK”, “convert PWA-application to APK-package” ja “how to publish PWA-app in google play store”.

4.1 PWA2APK

PWA2APK on verkossa toimiva työkalu, jolla pystyy muuntamaan PWA-sovelluksen APK-paketiksi. Se kuuluu Coffye Innovations Private Limitedin tarjoamaan palveluun nimeltä Appmaker.xyz. Tämä työkalu lupaa muuntaa minkä tahansa PWA-sovelluksen APK-paketiksi, jotta siitä saisi valmiin sovelluksen Google Play kauppaan. Tällä työkalulla on mahdollista muuntaa PWA-sovellus APK-paketiksi syöttämällä työkaluun pelkästään PWA-sovelluksen URL-osoitteen. Tämän työkalun käyttö tapahtuu pilvipalvelun avulla verkosta, joten sitä ei tarvitse tehdä paikallisesti. Työkalun käyttö on ilmaista, mutta työkalusta on olemassa myös maksullinen versio, joka puolestaan mahdollistaa kustomointimahdollisuuden nimetä paketin haluamakseen. (Upload PWA to Playstore n.d.)

4.2 Bubblewrap

Bubblewrap on Google-tiimin kehittämä komentorivikäyttöliittymä (CLI), jonka avulla voi luoda projektin Android-sovellukselle, joka käynnistää jo olemassa olevan PWA-sovelluksen käyttämällä Trusted Web Activity -toimintoa. Bubblewrap on ladattavissa GitHubista ja työkalua käytetään paikallisesti. Tällä työkalulla voi muuntamaa PWA-sovelluksen APK-paketiksi ja se antaa suoraan myös Google Play-kauppaan sopivan Android App Bundlen eli AAB-tiedoston. Vaikka Googlella on muuntamisesta tällä työkalulla oma tutoriaali, se ei kuitenkaan ole virallisesti tuettu Google-tuote. (GoogleChromeLabs 2022.)

4.3 PWA Builder

PWA Builder on Microsoftin perustama avoimen lähdekoodin projekti, joka perustettiin edistämään PWA-sovellusten käyttöönottoa. PWA Builder on verkossa toimiva työkalu, jonka toiminta tapahtuu työkalun verkkosivulla siten, että PWA-sovelluksen URL-osoite syötetään verkkosivulle

siihen tarkoitettuun kenttään. PWA Builder analysoi PWA-sovelluksen, jonka jälkeen se kertoo täyttääkö PWA-sovellus tarvittavat kriteerit. Mikäli se ei täytä, PWA Builder kertoo vinkkejä niiden korjaamiseksi. Kun kriteerit täyttyvät, pääsee PWA-sovelluksen muuntamaan sovelluskauppakelpoiseksi. (Converting your web app… 2021.)

4.4 PwaToTwa

PwaToTwa on Dominik Chrásteckýn kehittämä työkalu GitHubissa, jolla voi muuntaa PWA-sovelluksen Android sovellukseksi käyttäen luotettuja verkkotoimintoja (TWA). PwaToTwa perustuu Googlen viralliseen esimerkkiin ja korvaa kerätyt arvot ja kuvat PWA-sovelluksesta. Kun työkalua on käytetty, projektia pitäisi pystyä muokkaamaan Android Studiolla ja tekemään sillä siitä APK-paketin. (Chrástecký 2020.)

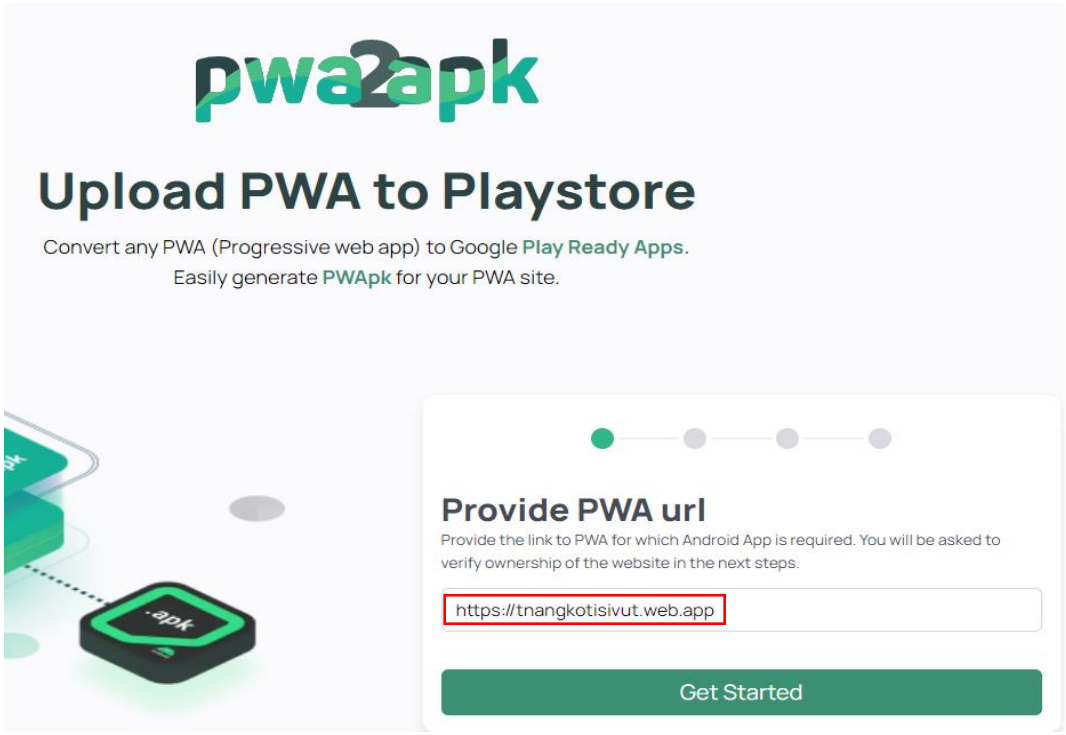
5 Toteutus

Tässä luvussa muunnetaan PWA-sovellus APK-paketeiksi jokaisella muuntamismenetelmällä erikseen. Muuntamismenetelmiä tutkitaan ja niistä kirjataan niiden vaiheet ja tulleet havainnot ylös. Muuntamisessa käytetään Angular-ohjelmistokehyksellä itse tehtyä PWA-sovellusta, joka läpäisee Google Lighthousen PWA-sovelluskriteerit.

5.1 Muuntaminen PWA2APK-työkalulla

PWA2APK-työkalun käyttö tapahtuu suoraan työkalun omalta verkkosivulta. Työkalu toimii pilvipalveluna, eli kaikki vaiheet pitäisi pystyä tekemään verkossa. Lopputuloksena pitäisi saada ladattavaksi valmis APK-paketti, jonka pystyy julkaisemaan Google Play-kaupassa. Työkalun käytön vaatimuksena ei siis ole mitään muuta kuin PWA-sovellus, joka on julkisesti saatavilla internetissä.

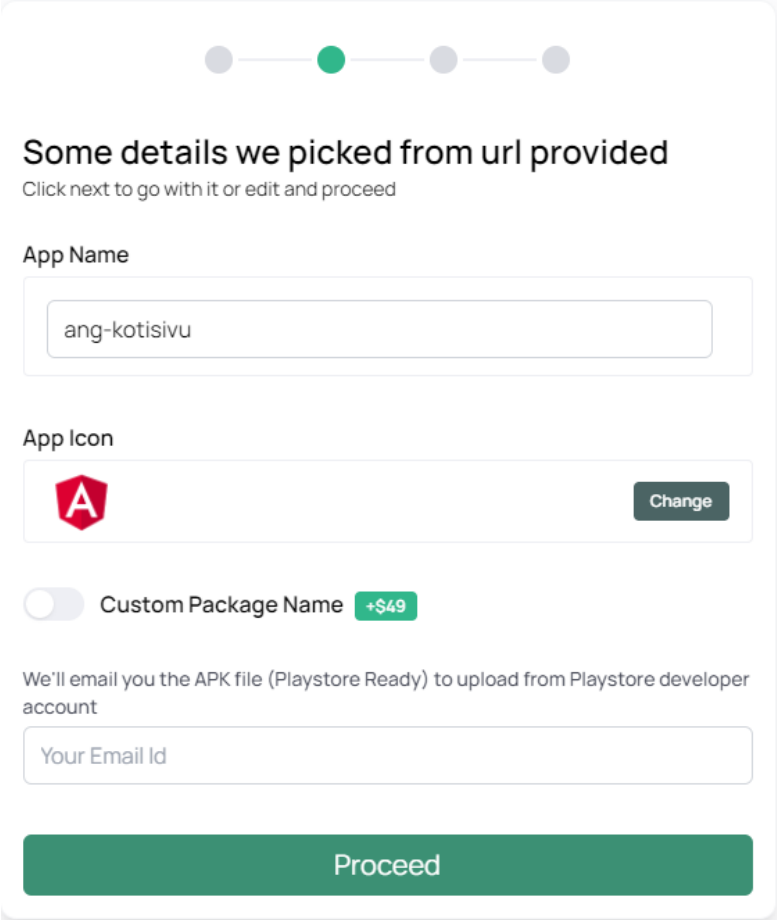
Työkalun käyttö alkaa menemällä sen verkkosivuille: ” https://appmaker.xyz/pwa-to-apk”. Verkkosivujen etusivulta pääsee heti työkaluun käsiksi. PWA2APK-työkaluun syötetään PWA-sovelluksen URL-osoite (ks. Kuvio 1).



Kuvio 1. PWA-sovelluksen URL-osoite syötetty sille tarkoitettuun kenttään

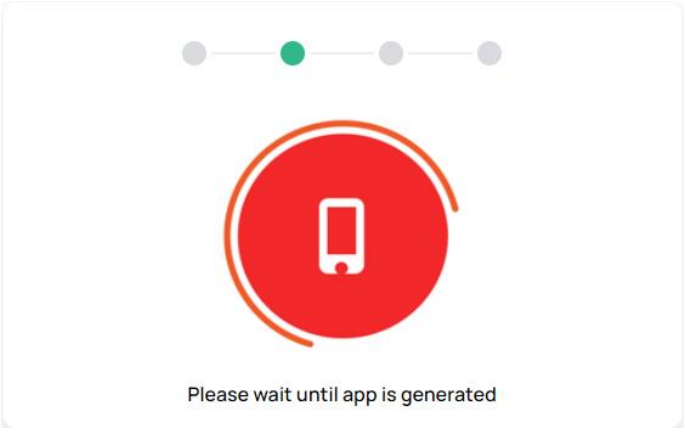
Kun PWA-sovelluksen URL-osoite on syötetty, jatketaan eteenpäin painamalla ”Get Started”-painiketta. Tämän jälkeen aukeaa näkymä, jossa voi määrittää sovelluksen nimen ja sovelluksen kuvakkeen. PWA2APK hakee nimen ja kuvakkeen automaattisesti, jos ne on jo aiemmin määritetty PWA-sovellukselle. Jos ei ole, voi ne manuaalisesti lisätä tässä vaiheessa tai vaihtaa halutessaan (ks. Kuvio 2). Tässä kohdassa oleva kuvake tulee oletetusti näkyviin käyttäjän laitteella, kun sovellus on asennettu. Itse työkalussa tätä tietoa ei varmisteta, joten se on oma oletus.

Tässä vaiheessa on mahdollista myös luoda maksua vastaan sovelluksen paketille kustomoitu nimi. Kustomoinnin hinta on 49 dollaria. Kustomointi ei ole pakollista ja vaihtoehtona PWA2APK-työkalu tulee luomaan sellaisen. Paketin nimi ei vaikuta sovelluksen nimeen. Se on nimi, jota sovelluskauppa käyttää tunnistamiseen. Käyttäjälle se näkyy ainoastaan verkkoselaimen URL-kentässä. Tässä opinnäytetyössä kustomointia ei tehdä. Lopuksi annetaan sähköpostiosoite, jonne lähetetään APK-paketti, joka on valmis Google Play-kauppaan julkaistavaksi.



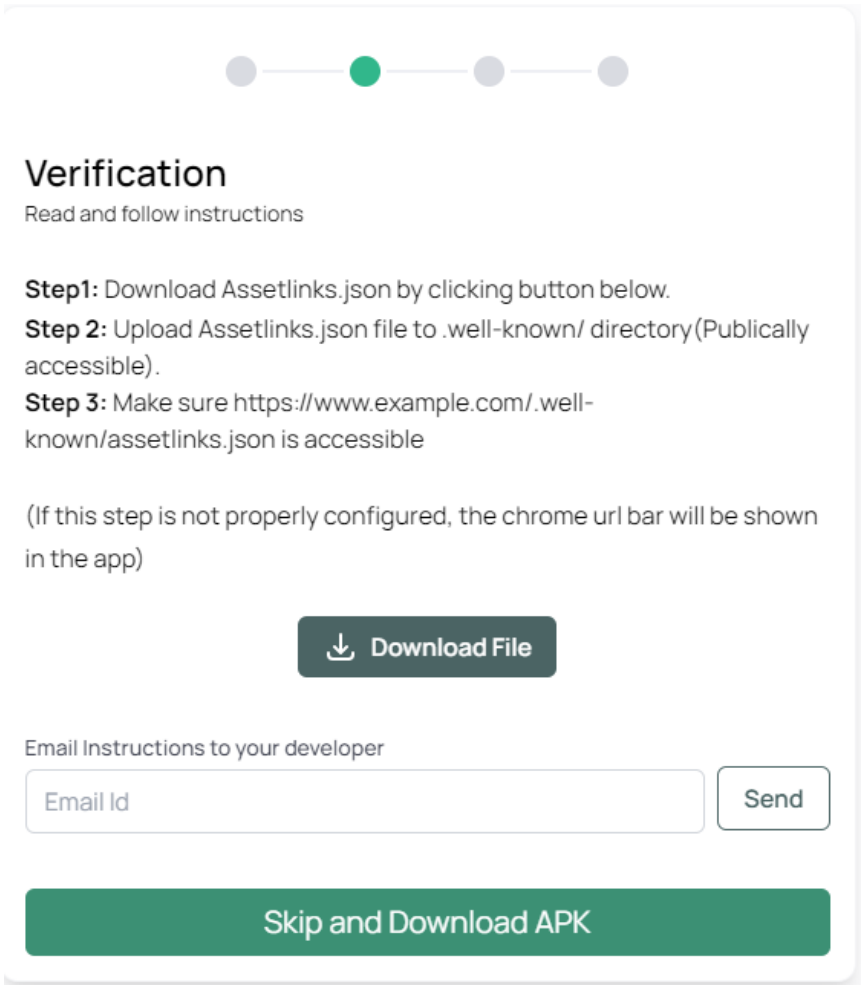
Kuvio 2. Sovelluksen nimen ja kuvakkeen määritys

Kun nimi, kuvake ja sähköposti on asetettu, eteenpäin pääsee Painamalla ”Proceed”-painiketta, jonka jälkeen sovelluksen generointi alkaa. Tämän vaiheen ei pitäisi kestää kauaa. Omassa muunnoksessani aikaa meni noin kaksi minuuttia. (Ks. Kuvio 3.)



Kuvio 3. Sovelluksen generointi

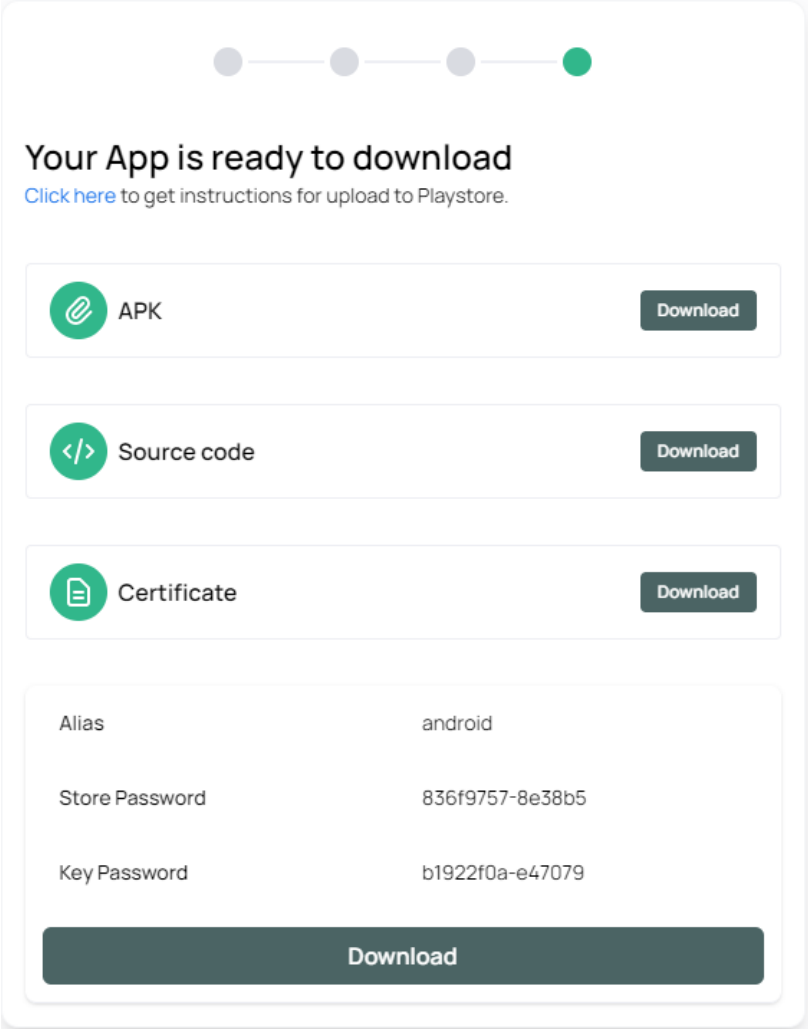
Kun sovellus on generoitu, PWA2APK-työkalu ohjeistaa lataamaan Assetlinks.json-tiedoston. Tämä tiedosto sisältää määritykset, joilla määritetään verkkoselaimen osoitepalkki pois näkyvistä sovelluksesta. Kyseinen tiedosto tulee asettaa julkiseksi oikeaan hakemistoon PWA-sovelluksen verkkosivuille ohjeiden mukaisesti (ks. Kuvio 4). Tämän vaiheen voi tehdä myös myöhemmin ja samat ohjeet saa lähetettyä sähköpostiin antamalla sähköpostinosoitteen siihen tarkoitettuun kenttään.



Kuvio 4. Assetlinks.json-tiedoston asettaminen oikeaan polkuun

Viimeisessä vaiheessa sovelluksen muuntaminen on valmis. Työkalu tarjoaa ladattavaksi neljä erilaista tiedostoa sovellukseen liittyen. Sovelluksesta on ladattavissa pakattu lähdekoodi, APK-, KEYSTORE- ja PROPERTIES-tiedostot (ks. Kuvio 5). Näkyville tulee myös Keystore- ja Key-salasanat ja ne voi ottaa halutessaan talteen. Ne löytyvät kuitenkin myös ladattavissa olevasta PROPERTIES-tiedostosta. Huomioitavaa, että tästä puuttuu AAB-tiedosto, joka on nykyään vaadittu tiedostomuoto Google Play-kauppaan.

Kuitenkin pakattu lähdekoodi sisältää kaikki tiedot, josta APK-tiedosto on luotu. Lähdekoodin voi avata Android Studiolla, joka mahdollistaa sovellustiedostojen muokkaamisen. Android Studiolla sovellusta voi kokeilla etukäteen ja katsoa toimiiko se oikein halutulla tavalla. Tästä tiedostosta saa tehtyä myös Android App Bundlen eli puuttuvan AAB-tiedoston.

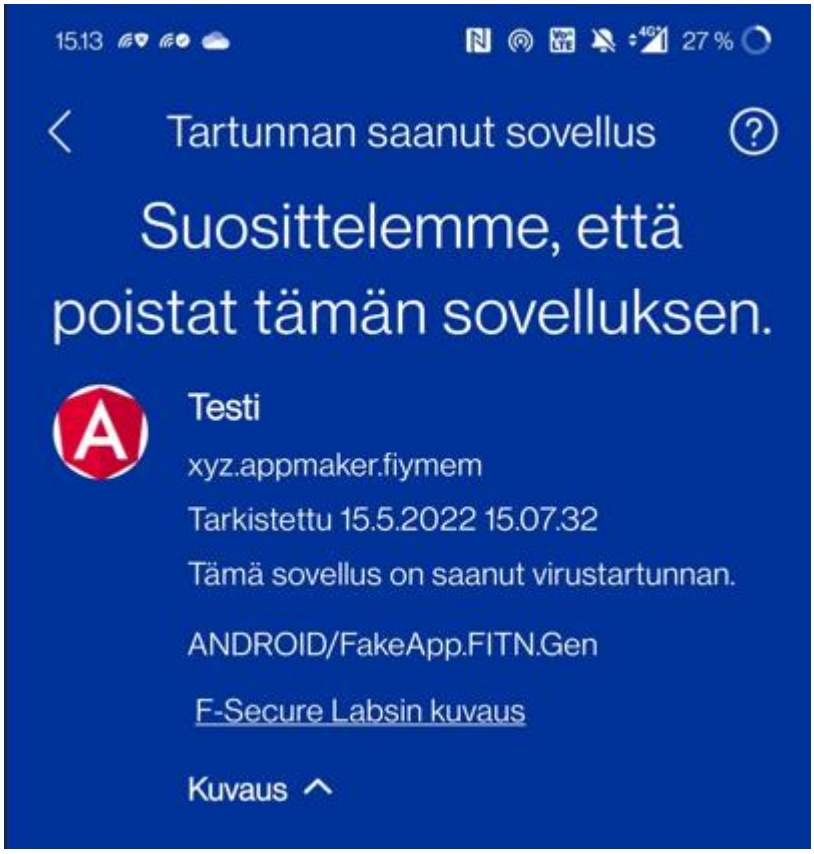


Kuvio 5. Tiedostot valmiina lataamiseen

PWA2APK tarjoaa helpon ja nopean tavan muuntaa PWA-sovellus APK-paketiksi. Tässä on kuitenkin hyvä huomioida, että sovelluksen kääntäminen tapahtuu täysin pilvipalvelun avulla. Koska muuntaminen tapahtuu pilvipalvelun avulla, sovelluksen tiedot menevät palveluntarjoajalle. Tämä kannattaa huomioida siinä vaiheessa, jos ei halua antaa sovellustaan kolmansille osapuolille. Mielenkiintoista tässä on se, että tähän työkaluun syötetään pelkästään PWA-sovelluksen URL-osoite. Eli kuka tahansa, jolla on PWA-sovelluksen URL-osoite voi käyttää tätä työkalua muuntaakseen

PWA-sovelluksen APK-paketiksi. Kuitenkaan se ei ole järkevää, koska verkkoselaimen osoiterivin piilottaminen vaatii kuitenkin pääsyn palvelimelle, jossa PWA-sovellus on.

Lopuksi vielä tallensin APK-tiedoston puhelimeeni ja sovellus näytti toimivan hyvin. Erittäin tärkeä huomioida, että kun sovellus oli asennettu puhelimeen, F-secure Freedome-sovelluksen App Security-tarkistus kertoi, että sovelluksessa on tartunta (ks. Kuvio 6). Kuitenkaan Google Play-kaupan tarkistus kävi sovelluksen läpi, eikä löytänyt siitä tartuntaa. Tämä tiedon huomioiden, vaikka sovellus muunsi PWA-sovelluksen onnistuneesti APK-paketiksi, työkalun käyttöä tulee harkita vakavasti.



Kuvio 6. Mahdollinen tartunta

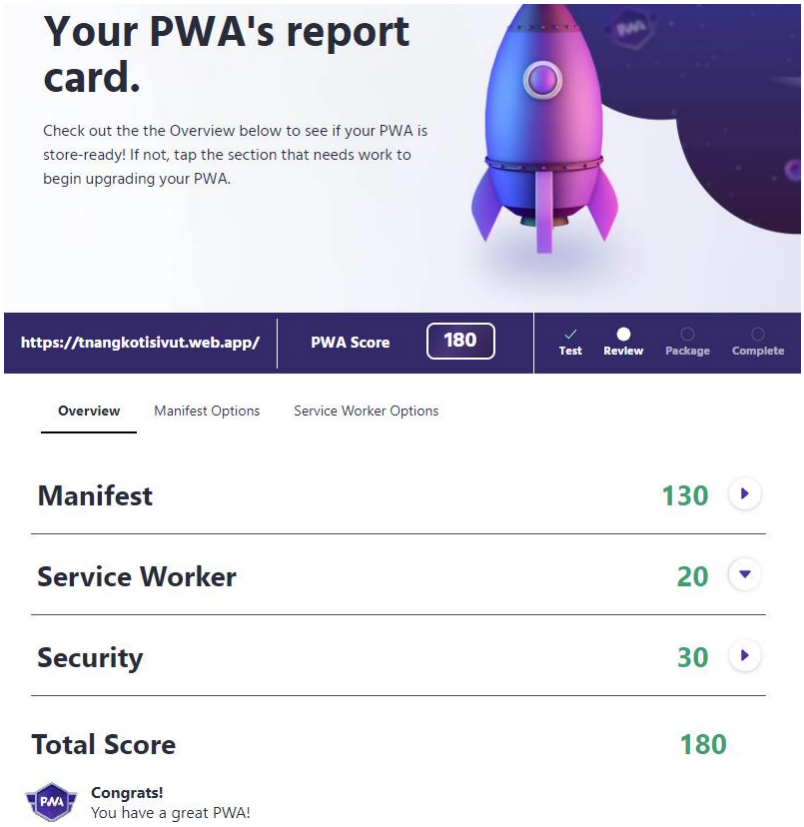
5.2 Muuntaminen PWABuilder-työkalulla

Microsoftin PWABuilder on työkalu, jolla voi muuntaa PWA-sovelluksen APK-paketiksi. Työkalun toimintaperiaate on sama kuin PWA2APK-työkalulla, eli se toimii pilvipalveluna verkossa. Työkalu näyttäisi lupaavan myös mahdollisesti muuntaa sovelluksesta Microsoft- ja iOS App-kauppaan.

PWA-sovelluksen muuntaminen APK-paketiksi alkaa niin, että PWA-sovelluksen URL-osoite syötetään sivustolla sille tarkoitettuun kenttään ja painamalla ”Start”-näppäintä. (Ks. Kuvio 7).

Kuvio 7. PWA-sovelluksen URL-osoite syötetty PWABuilderiin

Todella nopean analysoinnin jälkeen, PWABuilder antaa analysoinnin tuloksista raportin. Raportista näkee, onko PWA-sovellus sovelluskauppaan sopiva. PWABuilder antaa pisteitä kolmesta eri ominaisuudesta: ”Manifest”, ”Service Worker” ja ”Security”. Jos joku näistä kohdista tarvitsee parannusta, pystyy puutteellisia tietoja tarkastelemaan ja lisäämään niitä itse manuaalisesti ja suorittamaan analyysin uudestaan. Jos kaikki on hyvin, PWABuilder kertoo, että sinulla on loistava PWA-sovellus. (Ks. Kuvio 8).



Kuvio 8. Analyysin tulokset. Sovellus ei kaipaa parannusta

Tässä vaiheessa kannattaa vielä huomioida, että PWABuilder tarjoaa loistavat ominaisuudet muokata sovellusta suoraan työkalusta. Manifest Options-välilehdeltä pystyy hyödyntämään PWABuilderin sisäänrakennettua Web Manifest-editoria. Nämä asetukset vaikuttavat siihen, miten sovellus näkyy käyttäjälle milläkin laitteella. Siellä pystyy määrittämään ja esikatselemaan seuraavia asetuksia:

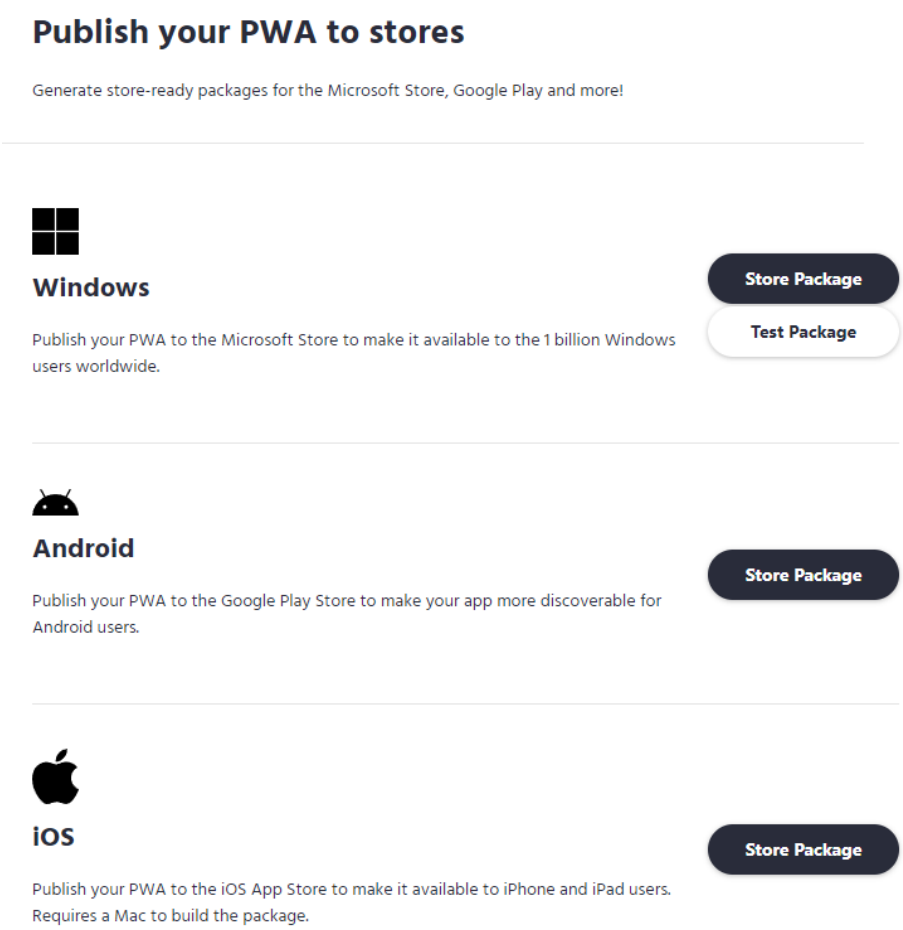
- Sovelluksen nimen
- Lyhyt nimen
- Kuvauksen
- Näkymä-asetuksen (display)
- Sovelluksen teeman ja taustan värit

Lisäksi sieltä pystyy määrittämään reititysasetuksia, eli esimerkiksi miltä PWA-sovelluksen sivulta sovellus aukeaa, sovelluksen kieli- ja orientointiasetukset sekä mahdollisuuden asettaa erikokoiset ikonit. Orientointi asetukset tarkoittavat sitä, esimerkiksi käytetäänkö sovellusta laitteen ollessa pysty (portrait)- vai vaaka-asennossa (landscape). Huomioitavaa on se, että sieltä pystyy määrittämään myös verkkoselaimen hakupalkin piiloon.

Service Worker Options-välilehdeltä pystyy asettamaan sovellukselle esirakennettuja Service Worker-paketteja, jos sellaiset puuttuu tai jos haluaa asentaa jonkun lisäksi. Kaikki niistä ei ole vaadittuja. Haluaman paketin voi ladata painamalla sen vieressä olevaa ”Add to Base Package”-painiketta, jonka jälkeen sivusto antaa ladata Base Packagen laitteelle, joka sisältää tämän Service Workerin ja Web Manifestin. Mukana tulee myös ohjeet, kuinka tiedostot voi liittää sovellukseen.

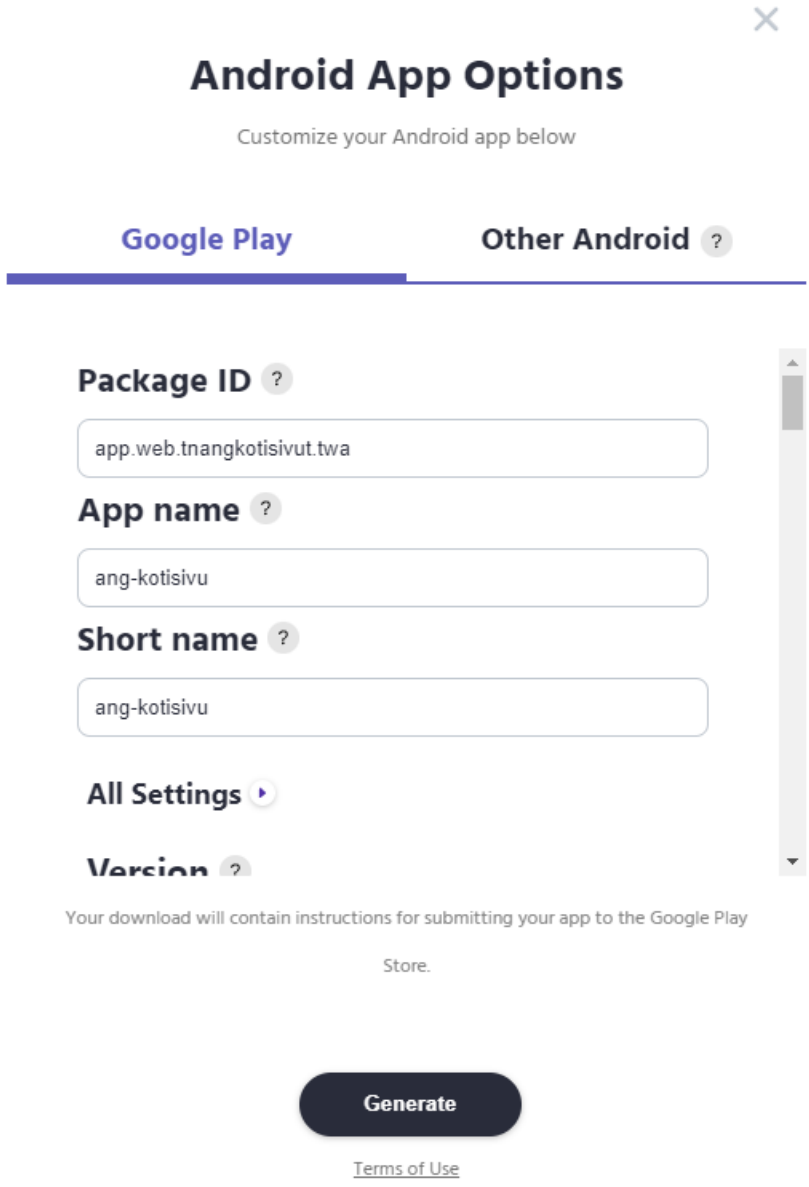
Kun kaikki mahdolliset puuttuvat tiedot on lisätty ja asetukset asetettu, voi seuraavaan vaiheeseen siirtyä Overview-välilehdeltä painamalla ”Next”-näppäintä.

Seuraavassa vaiheessa PWABuilder ilmoittaa, että PWA-sovellus on valmiina sovelluskauppaan. Tässä vaiheesta on mahdollisuus ladata valmiit paketit Microsoft-, Google Play-, ja iOS App -kauppaan. (Ks. Kuvio 9).



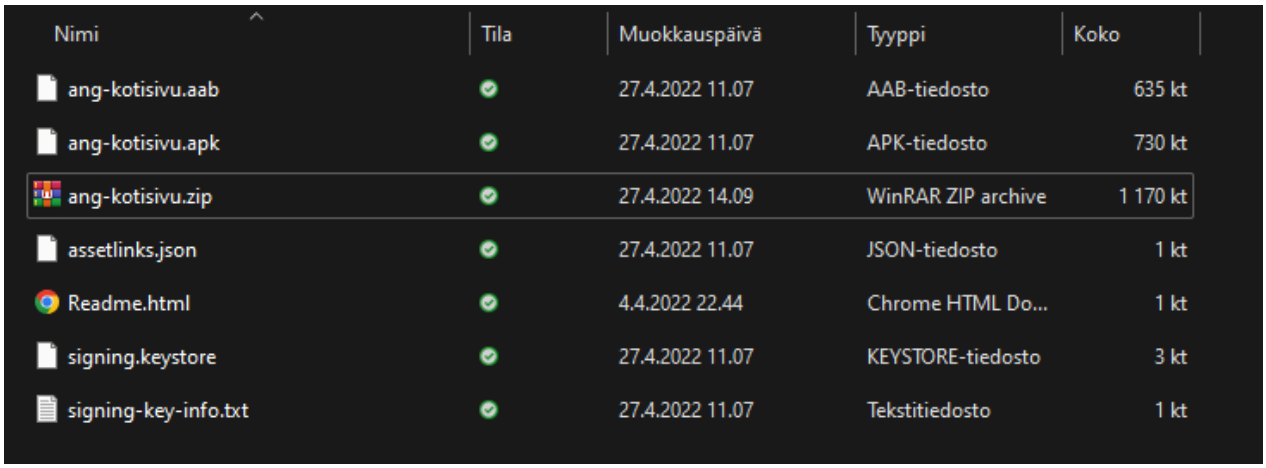
Kuvio 9. Microsoft-, Google Play- ja Ios App-kauppojen paketit

Painamalla halutun sovelluspaketin kohdalta painiketta ’Store Package’, aukeaa näkymä, jossa pystyy vielä muokkaamaan sovelluksen asetuksia. Siellä on muun muassa voi määrittää Google Play maksamiseen liittyviä asetuksia, rajoittaa sovelluksen käyttöä pelkästään ChromeOS-laitteelle ja halutessaan ladata paketin mukana lähdekoodin. Se kannattaa ehdottomasti ladata, jos haluaa tehdä sovellukseen itse myöhemmin muutoksia. Suurin osa on Web Manifest asetuksia, eli asetuksia, jotka määrittelevät kuinka sovellus tulee näkymään käyttäjälle (ks. Kuvio 10.) Kun halutut asetukset on määritetty, painamalla ”Generate”-painiketta, PWABuilder generoi paketin. Hetken kuluttua aukeaa ikkuna, josta sen voi ladata laitteelle.



Kuvio 10. Viimeiset asetukset ja "Generate"-painike

Ladattu tiedosto on pakattu ZIP-tiedosto, jonka purkamalla näkyviin tulee kaikki tarvittavat tiedostot, jotka tarvitaan sovelluskauppaan julkaisemiseksi. Jos edellisessä vaiheessa otti asetuksista lähdekoodin mukaan, se sisältyy samaan ZIP-tiedostoon. Lähdekoodi on tarpeellinen, jos haluaa muokata APK-pakettia Android Studiolla. Huomioitavaa, että sieltä löytyy myös AAB-tiedosto, joka on nykyään vaadittu tiedostomuoto Google Play-kauppaan (ks. Kuvio 11.) Muuntaminen on nyt valmis.



Kuvio 11. Ladattu ja purettu ZIP-tiedosto, joka sisältää APK ja AAB-tiedostot

Lopuksi vielä tallensin APK-tiedoston puhelimeeni ja testasin APK-paketin toimivuutta. APK-paketti asentui onnistuneesti ja sovellus toimi moitteitta.

5.3 Muuntaminen Bubblewrap-työkalulla

Muuntaessa PWA-sovellusta APK-paketiksi Bubblewrap-työkalulla, ohjeena on käytetty Bubblewrapin omaa dokumentaatiota. Bubblewrap on Google-tiimin kehittämä työkalu, jonka käyttö tapahtuu komentorivillä paikallisesti. Tässä muunnostyössä ohjelmana käytetään Visual Studio Coden terminaalia, joka käyttää Windowsin PowerShellia. Työkalun käytön vaatimuksena on Node.js 12 tai suurempi.

Muuntaminen alkaa asentamalla Bubblewrap CLI komennolla: ”npm i -g @bubblewrap/cli”. (Ks. Kuvio 12.)

```
+ @bubblewrap/cli@1.16.0
added 324 packages from 385 contributors in 93.341s
PS C:\Users\tompp\Desktop\Koulu\oppari\ang-kotisivu>
```

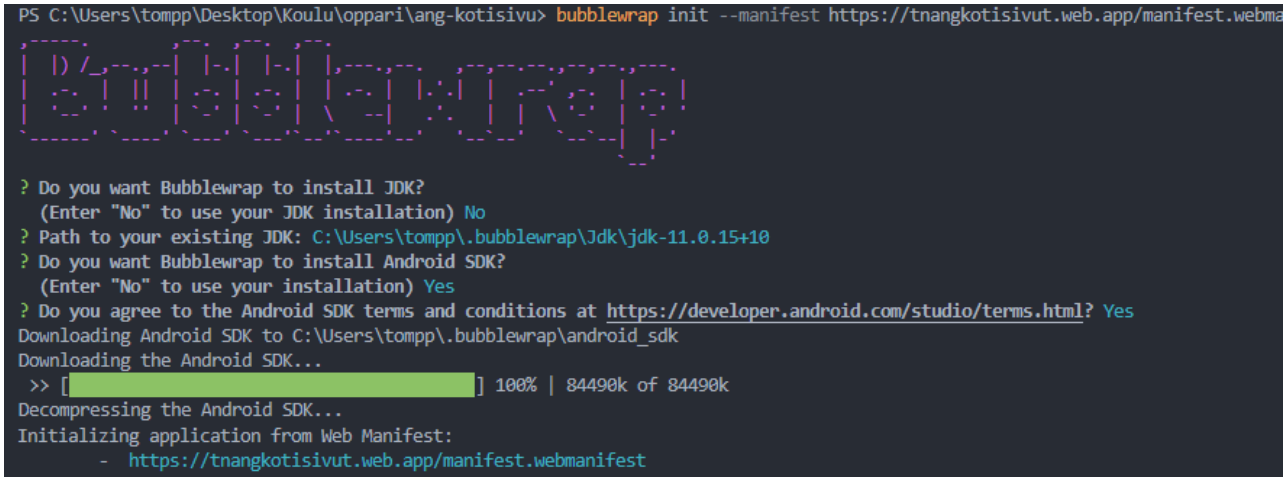
Kuvio 12. Bubblewrap CLI asennettu

Kun Bubblewrap on asennettu, aloitetaan Android-projektin alustaminen. Alustamisessa luodaan Android-projekti olemassa olevasta Web Manifestista syöttämällä komento niin, että komennossa oleva URL-osoite täytyy korvata olemassa olevan PWA-sovelluksen URL-osoitteeseen, joka osoittaa olemassa olevaan MANIFEST-tiedostoon. Korvaan siihen siis PWA-sovellukseni URL-osoitteen.

Tässä tapauksessa komento on: ”bubblewrap init --manifest https://tnangkotisivut.web.app/manifest.webmanifest”. Alkuperäinen muokattava komento: ’bubblewrap init --manifest https://my-twa.com/manifest.json’.

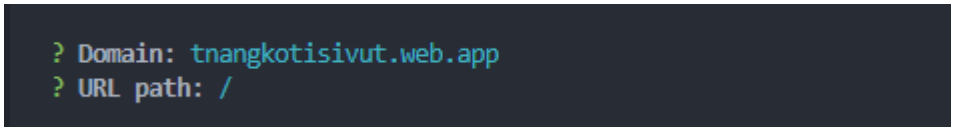
Bubblewrap vaatii toimiakseen Java Development Kitia (JDK) ja Android SDK:ta. Tässä vaiheessa Bubblewrap pyytää asentamaan JDK:n, mutta jostain syystä, Bubblewrap ei onnistu asentamaan sitä suoraan antaen virheilmoituksen: ”cli ERROR ERR_HTTP2_GOAWAY_SESSION”.

Ongelmaan ratkaisu on, että Java Development Kit ja Android SDK on ladattavissa tarvittaessa manuaalisesti niiden omilta verkkosivuiltaan ja sen jälkeen määrittää Bubblewrappiin. Tärkeä huomioida, että vaadittu versio on JDK 11 ja sen täytyy olla pakattu ZIP-tiedosto, joka täytyy purkaa omaan kansioonsa. Kun JDK 11 on purettu omaan kansioon, voi polun määrittää Bubblewrappiin. Seuraavaksi Bubblewrap kysyy Android SDK asentamisesta. Sen pystyi asentamaan suoraan komentoriviltä. (ks. Kuvio 13.)



Kuvio 13. JDK 11 polku määritetty ja Android SDK asennettu

Kun JDK11 ja Android SDK ovat asennettu, seuraavaksi täytyy määrittää asetukset Domain ja URL-polku sovellukselle. Domain on se osoite, mistä PWA-sovellusta isännöidään. URL-polku on tässä tapauksessa se polku, josta sovellus halutaan käynnistyvän, kun sovellus avataan laitteesta. Tässä tapauksessa Domain kohtaan tulee PWA-sovellukseni domain ja URL-poluksi on määritetty: ”/”. Tämä tarkoittaa sitä, että sovellus tulee käynnistymään kyseisen verkkosivuston juuresta. (Ks. Kuvio 14.)



Kuvio 14. Domain ja URL-polku määritetty

Seuraavaksi Bubblewrap pyytää täyttämään seuraavia tietoja, jotka vaikuttavat siihen, miltä sovellus näyttää, kun se on asennettuna laitteeseen (ks. kuvio 15).

- Application name (sovelluksen nimi.)
- Short name (lyhyt nimi, joka näytetään laitteen sovellusvalikossa.)
- Application ID (paketin nimi, joka on yksillöllinen tunniste sovellukselle Android-laitteessa tai Play-kaupassa. Nimessä pitää olla vähintään kaksi segmenttiä eroteltuna pisteillä. Jokainen segmentti täytyy alkaa kirjaimella ja kaikkien merkkien täytyy olla aakkosnumeerisia tai alaviivoja.)

- Display mode (kuinka sovellus näytetään laitteen näytöllä, kun sovellus käynnistetään. Esimerkiksi standalone -toiminto on oletus asetus ja sitä käytetään useimmissa sovelluksissa.)
- Status bar color (tilapalkin väri, jota käytetään, kun sovellus on etualalla.)

```
? Application name: ang-kotisivu
? Short name: kotisivu
? Application ID: app.web.tnangkotisivut.twa
? Starting version code for the new app version: 1
? Display mode: standalone
? Orientation: default
? Status bar color: (#1976D2)
```

Kuvio 15. Tietoja, jotka vaikuttavat siihen miltä sovellus näyttää laitteella

Seuraavaksi määritetään kuva käynnistysikonille. Kuva näkyy myös Splash Screenissa, eli siinä vaiheessa, kun verkkosisältöä ladataan laitteelle. Tällä vältetään näyttämästä tyhjää ruutua käyttäjälle. Seuraavat tiedot tulee täyttää. (Ks. Kuvio 16.)

- Splash screen color (Splash-näytön väri. Asettaa taustavärin Splash-näytölle)
- Icon URL (URL-osoite kuvalle, joka on vähintään 512x512 pikseliä. Tällä luodaan käynnistyskuvake ja kuva Splash-näyttöön.)
- Maskable Icon URL (vaihtoehtoinen)

```
? Splash screen color: #FAFAFA
? Icon URL: https://tnangkotisivut.web.app/assets/icons/icon-512x512.png
? Maskable icon URL: https://tnangkotisivut.web.app/assets/icons/icon-512x512.png
```

Kuvio 16. Sovelluksen käynnistyskuvakkeen määrittäminen

Seuraavana olevat määritykset ovat kaikki valinnaisia ominaisuuksia. Siinä voidaan määrittää seuraavat tiedot:

- Include app shortcuts
- Monochrome icon URL

Koska nämä määritykset ovat valinnaisia ja ne eivät vaikuta PWA-sovelluksen muuntamista APK-paketiksi, niin kyseiset määritteet jätetään pois, eikä niitä tulla kuvaamaan tässä opinnäytetyössä.

Seuraavaksi Bubblewrap kysyy, sisällytetäänkö tuki Google Play Billing-palvelulle sekä pyydetäänkö lupaa maantieteelliseen sijaintiin. Tässä opinnäytetyössä sovellukseen näitä määritteitä ei oteta käyttöön.

Alustuksen viimeisessä vaiheessa määritetään kirjautumisavaimen tietoja. Jos Keystore on jo olemassa, voi seuraavassa vaiheessa syöttää Keystoren polun, jolloin niitä tietoja käytetään. Tässä opinnäytetyössä muuntamiseen käytettävässä PWA-sovelluksessa ei näitä tietoja vielä ole, joten Bubblewrap tulee kysymään halukkuudesta luoda uuden Keystoren. Syötetään oletuspolku, mikä työkalussa näkyy ja luodaan uusi Keystore. Kun avaimen nimeä kysytään (Key name), on syytä huomioida, että annettu nimi tulee olemaan myös seuraavassa vaiheessa luodun avaimen nimi. Lisäksi Keystoren poluksi kannattaa määrittää erillinen tyhjä kansio, sillä kun sovellus lopuksi muunnetaan, lähdekoodi ja muut tiedostot tulevat tähän kansioon.

Avaimen luontivaiheessa, Bubblewrap tulee kysymään seuraavat tiedot, joista salasanat on hyvä ottaa muistiin (ks. Kuvio 17.)

- Haluatko luoda uuden Keystoren?
- Etu- ja sukunimi
- Organisaatioyksikkö
- Yritys
- Maa
- Salasana Keystorelle
- Salasana avaimelle

```
Signing key creation

An existing key store could not be found at "D:\Koulu\Oppari\ang-kotisivu\android.keystore"
.

? Do you want to create one now? Yes
? First and Last names (eg: John Doe): Tomi Naukkarinen
? Organizational Unit (eg: Engineering Dept): opiskelija
? Organization (eg: Company Name): Koulu
? Country (2 letter code): FI
? Password for the Key Store: ********
? Password for the Key: ********
keytool Signing Key created successfully
```

Kuvio 17. Uuden Keystoren luonti

Kun Android-projekti on luotu onnistuneesti, seuraavaksi rakennetaan APK-paketti, joka voidaan julkaista Google Play kaupassa. Syötetään terminaaliin komento: ”bubblewrap build”. Seuraavaksi Bubblewrap kysyy salasanoja Keystorelle ja avaimelle, jotka luotiin edellisessä vaiheessa.

Kun salasanat on syötetty, Bubblewrap yrittää käynnistää analyysin PWA-sovelluksen laatukriteereistä. Tässä opinnäytetyössä tuo vaihe epäonnistuu, eikä sitä saa käytyä loppuun. Bubblewrap käyttää Googlen Lighthouse työkalua PWA-sovelluksen laatukriteerien tarkistamiseen. Tämän voi halutessaan hoitaa manuaalisesti selaimen kehittäjätyökalujen avulla käyttämällä Lighthousea. Vaikka kyseinen analyysin käynnistäminen epäonnistuu, sillä ei ole väliä, koska Bubblewrap luo tarvittavat tiedostot. Nämä tiedostot löytyvät siitä sijainnista, minne Keystore on luotu. Samaan sijaintiin muodostuu lähdekoodi, johon pystyy tarvittaessa tekemään muutoksia Android Studiolla. Huomioitavaa, että Bubblewrap tekee myös suoraan tarvittavan AAB-tiedoston, joka on vaadittu tiedostomuoto Google Play-kauppaan julkaisemiseksi. Muuntaminen on valmis.

5.4 Muuntaminen PwaToTwa-työkalulla

GitHubista löytyy myös Dominik Chrásteckýn (RikudouSage) kehittämä työkalu, jolla PWA-sovelluksen muuntaminen APK-paketiksi on mahdollista. Tutkittuani työkalun dokumentaatiota, itse työkalun käyttäminen pitäisi olla helppoa. Kuitenkin työkalun esivaatimukset ovat hankalat.

Tässä tutkimuksessa kummallakaan tavoista ei onnistuttu muuntamaan PWA-sovellusta APK-paketiksi. Työkalun dokumentaatio ja käyttöönotto vaatii jo syvempää ymmärrystä siihen vaadituista sovelluksista ja kirjastoista, että työkalua ei onnistuttu kokeilemaan.

Työkalua voi käyttää rakentamalla sen suoraan lähteestä tai sen voi rakentaa staattisesti. Suoraan lähteestä rakentaminen edellyttää maksullisen QT Frameworkin käyttöä ja libgit2-kirjastoa, jotka ovat dokumentaation mukaan helppo asentaa Linux-pohjaisille käyttöjärjestelmille. Staattisesti rakentaminen vaatii, että Docker on asennettuna.

Kuitenkin tällainen työkalu on olemassa ja se kannattaa ottaa huomioon. Se toimii mahdollisesti helpommin Linux-pohjaisella käyttöjärjestelmällä paremmin.

6 Tulokset

Tässä luvussa käsitellään tuloksia jokaisen muuntamismenetelmän osalta ja poimitaan niistä tärkeimmät ominaisuudet, hyödyt ja haitat. Esitetään myös vertailua niiden välillä taulukkomuodossa. Nämä tiedot pitäisi vastata tutkimuskysymyksiin.

6.1 PWA2APK-työkalun tulokset

PWA2APK-työkalulla muuntamiseen ei edellytä mitään esivaatimuksia. Työkalua pystyy käyttämään kuka tahansa, jolla on jonkin PWA-sovelluksen URL-osoite. Työkalu onnistui muuntamaan PWA-sovelluksen APK-paketiksi vaivatta ja helposti, eikä siihen tarvitse valtavaa teknistä osaamista. APK-paketti testattiin ja se toimii laitteessa.

Muuntaminen tapahtuu pilvipalvelussa, joka tarkoittaa sitä, että sovelluksen tiedot menevät kolmannelle osapuolelle. Tämä kannattaa huomioida tietoturvan näkökulmasta.

Koska muuntaminen oli todella yksinkertainen, eikä siinä ollut juuri mitään asetuksia, aikaa kului muuntamiseen arviolta noin 15 minuuttia. PWA-sovelluksen koko voi vaikuttaa generointivaiheessa aikaan.

Ongelmia ei tullut vastaan, mutta erittäin tärkeä huomioida, että kun sovellus oli asennettu laitteelle, F-Secure Freedome-sovelluksen App Security-tarkistus kertoi, että sovelluksessa on mahdollinen tartunta. Tämän vuoksi työkalun käyttöä ei voi suositella ja sen käyttöä tulee harkita.

Lopputuloksena sai ladattua seuraavat tiedostot: lähdekoodi (.ZIP), APK-, KEYSTORE- ja PROPERTIES-tiedostot. Android App Bundle eli AAB-tiedosto tästä paketista kuitenkin puuttui. Lähdekoodi kuitenkin mahdollistaa puuttuvan AAB-tiedoston luomisen Android Studiolla. Sillä voi myös tarvittaessa muokata sovellusta. APK-tiedoston koko 1391 kilotavua.

6.2 PWABuilder-työkalun tulokset

PWABuilder on helppo työkalu PWA-sovelluksen muuntamiseen APK-paketiksi. Työkalu ei edellytä esivaatimuksia, vaan riittää, että muunnettava PWA-sovellus on verkossa julkisesti saatavilla. Sovelluksen muuntaminen onnistui ja APK-paketin toimivuus varmistettiin asentamalla se omaan laitteeseen.

PWABuilder-työkalulla muuntaminen tapahtuu pilvipalvelussa, joka tarjoaa erinomaisen sisäänrakennetun Web Manifest-editorin ja valmiiksi rakennettuja Service Workereita. Niiden avulla sovellukseen saa lisättyä puuttuvia toimintoja ohjeiden kera. Pilvipalvelu kuitenkin tarkoittaa sitä, että sovelluksen tiedot menevät kolmannelle osapuolelle. Tämä kannattaa huomioida tietoturvan näkökulmasta.

Muuntaminen on nopea, jos PWA-sovellus on jo valmiiksi määritetty hyvin. Omasta PWA-sovelluksesta PWABuilderin analyysi antoi hyvät pisteet, joten siihen ei tarvinnut lisätä mitään. Aikaa muuntamiseen meni noin 30 minuuttia. Kuitenkin mikäli analyysin tulokset olisivat huonot, niin PWA-sovelluksen parantaminen valmiiden palvelujen avulla on kuitenkin aikaa vievää, joten muuntamisen nopeus on tapauskohtainen. Ongelmia muuntamisessa ei tullut vastaan.

Lopputuloksena työkalu antoi suoraan myös Android App Bundlen eli AAB-tiedoston, joten sitä ei tarvitse lähdekoodista tehdä itse. Lähdekoodi oli mahdollista ladata mukana, mutta se ei ollut oletuksena mukana, vaan se täytyi erikseen osata ottaa mukaan latausasetuksista. Mukana oli myös

kaikki muut tarvittavat tiedostot muun muassa KEYSTORE-tiedosto. APK-tiedoston koko on 730 kilotavua. PWA-Builder menetelmässä on myös mahdollisuus muuntaa PWA-sovellus Microsoft- ja iOS App-kauppoihin.

6.3 Bubblewrap-työkalun tulokset

Bubblewrap-työkalu on hyvä, mutta hieman haastava työkalu PWA-sovelluksen muuntamisessa APK-paketiksi. Bubblewrap on komentorivikäyttöliittymä, joten työkalun käyttö edellyttää tietoa, miten komentorivillä työskennellään. Komentorivikäyttöliittymää on kuitenkin kohtuullisen helppo käyttää, koska se kertoo ja selittää käyttäjälle, mitä tietoja työkaluun täytyy määrittää ja mitä ne tarkoittavat. Työkalun käyttö edellyttää myös asennettua Node.js 12 versiota tai suurempaa, Java Development Kit:ia ja Android SDK:ta. Kaksi viimeisintä ovat kuitenkin ladattavissa suoraan työkalusta.

Bubblewrap onnistui muuntamaan PWA-sovelluksen APK-tiedostoksi. Sama APK-tiedosto todettiin toimivaksi asentamalla se tutkimuksen tekijän omaan laitteeseen.

PWA-sovelluksen muuntaminen tapahtuu paikallisesti omalla laitteella komentoriviltä käsin. Tällä työkalulla muuntamisessa sovelluksen tiedot eivät mene kolmansille osapuolille.

Työkalun käyttö vei aikaa, koska kaikki tiedot piti syöttää työkaluun manuaalisesti. Työkalun käytössä vastaan tuli ongelmia ja niiden ratkaiseminen vei aikaa. Työkalulla muuntamiseen aikaa kului noin 60 minuuttia, mutta muuntaminen voi viedä tapauskohtaisesti aikaa myös enemmän tai vähemmän.

Ongelmia Bubblewrap-työkalulla tuli vastaan. Työkalu ei onnistunut lataamaan Java Development Kit:ia suoraan työkalusta, vaan se piti käydä lataamassa erikseen Oraclen omilta verkkosivuilta ja sitten määrittämällä sen polku työkaluun. Työkalu yritti myös tehdä PWA-sovelluksen laatuanalyysin, joka epäonnistui.

Lopputuloksena työkalu antoi kansion, jossa on kaikki tarvittavat tiedostot, jotta sovelluksen voisi julkaista Google Play-kaupassa. Kansio sisältää APK-, AAB-tiedostot ja työkalussa luodun

KEYSTORE-tiedoston. Lisäksi työkalu antoi suoraan myös sen sovelluksen lähdekoodin, jota pystyy muokkaamaan tarvittaessa Android Studiolla. APK-tiedoston koko on 403 kilotavua.

6.4 PwaToTwa-työkalun tulokset

PwaToTwa-työkalulla PWA-sovelluksen muuntaminen APK-paketiksi ei onnistunut. Työkalun käytön esivaatimukset olivat sen verran hankalat, ettei muunnostyötä onnistuttu tekemään ollenkaan. Työkalun käytöstä ei ole tästä syystä kerättyjä havaintoja, joten tuloksia ei voi kirjata.

Työkalu toimii todennäköisesti parhaiten Linux-pohjaisilla käyttöjärjestelmillä, joten siitä voi olla hyötyä niille, jotka muuttavat PWA-sovellusta APK-paketiksi Linux-pohjaisella käyttöjärjestelmällä.

6.5 Tulosten taulukointi

Yhteenvetona muuntamismenetelmät ovat erikseen laitettu taulukkoon (ks. Taulukko 1), josta eri menetelmien ominaisuuksia pystyy vertaamaan keskenään. Taulukkoon määritetyt vertailuarvot on määritetty sillä perustein, että niillä pyritään kuvaamaan muuntamismenetelmien ominaisuuksia ja tietoja, jotta siitä olisi hyötyä sellaisille ihmisille, jotka haluavat tietoa kyseisistä muuntamismenetelmistä. Ihminen, joka on muuntamassa PWA-sovellusta APK-paketiksi, voi tarkastella taulukossa olevia muuntamismenetelmiä keskenään ja etsimään niistä eroavuuksia peilaten niitä omiin tarpeisiin ja taitoihin. Taulukkoon on kerätty ne muuntamismenetelmät, jotka onnistuneesti muunsivat PWA-sovelluksen APK-paketiksi. Vertailuarvoiksi valikoituivat:

Vertailuarvo ”Toimiiko APK-paketti laitteessa?”, kuvastaa muuntamismenetelmällä muunnetun PWA-sovelluksen lopullista APK-paketin toimivuutta. Jokaisen APK-paketin toimivuutta on kokeiltu fyysisellä laitteella

Vertailuarvo ”Tapahtuuko muuntaminen pilvessä?” kertoo sen, että tapahtuuko muuntamismenetelmällä muuntaminen pilvipalvelussa vai paikallisesti tietokoneelta.

Vertailuarvo ”muuntamisen vaikeus”, on määritetty tutkimuksen tekijän omien havaintojen perusteella, johon vaikuttaa työkalun käytön esivaatimukset, muuntamisen monimutkaisuus ja muuntamiseen kulunut aika. Vertailuarvolle on asetettu seuraava asteikko kuvastamaan sen muuntamisen vaikeutta: ”Helppo”, ”Keskitaso” ja ”Vaikea”.

Vertailuarvo ”Muuntamisen nopeus”, on määritetty kuvaamaan muuntamiseen käytetyn ajan huomioiden muuntamismenetelmän esivaatimukset ja siinä vastaan tulleet ongelmat.

Vertailuarvo ”APK-tiedoston koko”, kertoo muuntamismenetelmän lopputuloksena saadun APK-tiedoston koon, joka on muunnettu PWA-sovelluksesta.

Taulukko 1. Muuntamisessa onnistuneiden muuntamismenetelmien vertailu

Muuntamismenetelmä	PWA2APK	PWABuilder	Bubblewrap
Toimiko APK-paketti?	Kyllä	Kyllä	Kyllä
Tapahtuuko muuntaminen pilvipalveluna?	Kyllä	Kyllä	Ei
Muuntamisen vaikeus	Helppo	Helppo	Keskitaso
Muuntamisen nopeus	~ 15 min	~ 30 min	~ 60 min
APK-tiedoston koko	1391 KT	730 KT	403 KT

7 Johtopäätökset

Tässä luvussa esitetään tulosvaiheesta havaitut johtopäätökset ja peilataan niitä tutkimuskysymyksiin.

Mitä erilaisia vaihtoehtoja on olemassa, joilla voi muuntaa PWA-sovelluksen APK-paketiksi?

PWA2APK-, PWABuilder- ja Bubblewrap-työkalut ovat kaikki menetelmiä, joilla PWA-sovelluksen muuntaminen APK-paketiksi onnistui. Kaikkien menetelmien muuntamat APK-paketit asentuivat onnistuneesti laitteelle ja ne toimivat.

Mitkä ovat menetelmien hyödyt, haitat ja eroavaisuudet?

Tutkimustuloksissa pystyy nostamaan tiettyjä asioita, jotka voivat luokitella suoraan hyödyiksi tai haitoiksi. Esimerkiksi vertailuarvo ”Toimiiko paketti laitteessa” kertoo selkeästi sen, että mikäli muuntaminen onnistui, niin silloin se varmasti on hyödyllinen. Kuitenkin esimerkiksi vertailuarvo ’’Tapahtuuko muuntaminen pilvessä?” onkin jo monitulkintainen. Toinen voi nähdä, että muuntamisen tapahtuminen pilvessä on haittana tietoturvan vuoksi, kun taas toisaalta se voidaan nähdä hyvänä asiana, koska muunnokseen ei vaadita paljoa teknistä osaamista. Hyödyt ja haitat ovat osittain siksi muunnostyön tekijän omia tulkintoja.

Selkeitä hyödyksi havaittuja ominaisuuksia ovat APK-paketin toimivuus laitteessa, muuntamisen helppous ja APK-tiedoston koon pienuus.

Selkeitä haittoja vastakohtaisesti se, että muunnettu APK-paketti ei toimi laitteessa ja silla on suuri koko. Tiedoston kokoon vaikuttaa alkuperäisen muunnettavan PWA-sovelluksen koko, mutta lopullisen muunnetun APK-tiedoston koko vaihtelee melkein 1 megatavulla eri muuntamismenetelmien välillä.

Vertailusta ilmeni myös mielenkiintoisia huomioita. Jokaisen työkalun APK-tiedosto saatiin asennettua onnistuneesti laitteelle ja ne todettiin toimiviksi. Pilvipalveluna toimiva menetelmä tuntui olevan helpompi käyttää kuin paikallisesti toimiva. Myöskin pilvipalveluna toimiva menetelmä oli nopeampi kuin paikallisesti toimiva.

Pilvipalveluina toimivilla muuntamismenetelmillä on kuitenkin isoja eroja ominaisuuksissa. PWABuilderin sisäänrakennetut Web Manifest-editori ja valmiiksi rakennetut Service Workerit tekevät työkalusta ominaisuuksiltaan rikkaamman. Työkalun PWA-sovelluksen analysoiminen ja nämä sisäänrakennetut ominaisuudet erottavat PWABuilder-työkalun edukseen PWA2APK-työkalusta.

Huomioon otettava myös Bubblewrap-työkalu, jonka toteutus tapahtuu paikallisesti. Paikallisesti muuntaessa sovelluksen tietoja ei luovuteta kolmansille osapuolille, siksi tämä vaihtoehto on hyvä

ottaa huomioon tietoturvaa miettiessä. Sen muuntama APK-paketin on koko kaikkein pienin muihin verrattuna. Henkilö, joka ei halua luovuttaa tietojaan kolmansille osapuolille voisi olla Bubblewrap-työkalun käyttäjä.

8 Pohdinta

Tässä luvussa pohditaan tutkimuksen onnistumista ja tutkimuksen luotettavuutta.

8.1 Tutkimuksen onnistuminen

Tutkimuksen tarkoituksena on selvittää erilaisia menetelmiä, joilla voi muuntaa PWA-sovelluksen APK-paketiksi sekä selvittää niiden hyötyjä, haittoja ja eroavuuksia. Tutkimuksessa löydettiin neljä erilaista menetelmää internetistä ja kyseisiä menetelmiä kokeiltiin käytännössä ja ne todettiin toimiviksi yhtä lukuun ottamatta. Näillä perusteilla voin todeta, että muuntamismenetelmien selvittäminen onnistui. Muuntamismenetelmien etsiminen tuntui alusta asti helpolta ja sitä se myös oli. Muuntamismenetelmiä löytyi internetistä aika helposti.

Vaikka muuntamismenetelmät löytyivätkin helposti, niin menetelmiä oli myös tarkoitus kokeilla käytännössä. Niistä oli tarkoitus kerätä havainnot ylös ja selvittää niiden hyödyt, haitat ja eroavaisuudet. Ajatuksena PWA-sovelluksen muuntaminen APK-paketiksi kuulostaa todella monimutkaiselta prosessilta, mutta se olikin yksinkertaista ainakin kahdella menetelmällä. Varsinkin pilvipalveluihin perustuvat PWA2APK- ja PWABuilder-työkalut yllättivät niiden nopeudessa ja työkalun käytettävyyden helppoudessa.

Bubblewrap-työkalun dokumentaatio tarkastellessa, en ollut ollenkaan varma saanko työkalua toimimaan. Työkalun dokumentaatio oli kuitenkin niin hyvin tehty, että siitä löytyi ongelmatilanteisiin ratkaisut ja ohjeet, miten niissä tulee toimia. Itse työkalussa olevat ohjeet ja määritteiden kuvaukset auttavat ymmärtämään työkalussa olevia vaiheita. Niiden avulla muuntaminen onnistui hyvin, alkuoletuksista huolimatta.

Koska lähes kaikki muuntamismenetelmät saatiin onnistuneesti muuntamaan PWA-sovellus APK-paketiksi, saatiin jokaisesta muuntamismenetelmästä kerättyä hyvin havaintoja. Tuloksista ja ha-

vainnoista sai tehtyä taulukoinnin, josta pystyy vertailemaan hyvin eri menetelmien eroavaisuuksia. Tulosten ja taulukon perusteella onnistui tekemään myös merkittäviä johtopäätöksiä menetelmistä ja selvitettyä menetelmien hyötyjä ja haittoja.

Onnistuneista muunnoksista huolimatta, PwaToTwa-muuntamismenetelmä kuitenkin oli sellainen, jonka käyttäminen epäonnistui. Sen käyttövaatimukset olivat sen verran hankalat, ettei koko työkalua päässyt käyttämään. Se kuitenkin voi olla Linux-pohjaisella käyttöjärjestelmällä helpompi suorittaa. Tässä kohtaa kuitenkin herää kysymys, kannattaako sellaista työkalua käyttää, jos olemassa on nyt tutkitusti kolme erilaista menetelmää, jolla muunnoksen saa tehtyä joko pilvipalvelun avulla tai paikallisesti?

Olen vahvasti sitä mieltä, että vaikka yhden menetelmän käyttäminen epäonnistui, tutkimus onnistui myös hyötyjen, haittojen ja eroavaisuuksien selvittämisessä. Tutkimus vastaa siis kattavasti tutkimuksen tutkimuskysymyksiin.

Jatkotutkimusehdotus voisi liittyä tuon epäonnistuneen muuntamismenetelmän tutkimiseen ja selvittämiseen, onko sillä mahdollista muuntaa PWA-sovellus APK-paketiksi. Myöskin oman muuntamismenetelmän kehittäminen voisi olla jatkotutkimusehdotus.

8.2 Luotettavuus

Tutkimuksen luotettavuuden takaamiseksi aineistossa on pyritty käyttämään sellaisia lähteitä, jotka ovat otettu tiedon alkuperäisestä lähteestä. Muuntamiset on tehty hyödyntäen ainoastaan kunkin menetelmän omia dokumentaatioita.

Tuloksiin on kerätty menetelmien eri ominaisuuksia, hyötyjä ja haittoja. Näitä tietoja on verrattu myös puolueettomasti keskenään, jotta niiden käyttämisestä saataisiin mahdollisimman todellinen kuvaus.

Lähteet

About Android App Bundles. 2022. Verkko-artikkeli ja opas. Google Developers. Viitattu 15.5.2022 https://developer.android.com/guide/app-bundle.

Chrástecký, D. 2020. PwaToTwa GitHub--projekti ja tutoriaali. GitHub Inc. Viitattu 24.03.2022. https://github.com/RikudouSage/PwaToTwa.

Converting your web app to a Progressive Web App with PWABuilder. 2021. Blogiteksti. PWABuilder by Microsoft. Viitattu 23.03.2022 https://blog.pwabuilder.com/docs/converting-your-web-app-to-a-progressive-web-app-with-pwabuilder/.

Digital Asset Links Overview. 2022. Googlen opas. Google Developers. Viitattu 17.4.2022. https://developers.google.com/digital-asset-links/v1/getting-started.

Enge, E. 2021. Mobile vs. Desktop Usage in 2020. Verkko-artikkeli. Perficient Inc. Viitattu 25.03.2022. https://www.perficient.com/insights/research-hub/mobile-vs-desktop-usage.

Feature: Trusted Web Activities. 2021. Ominaisuuden kuvaus. Chrome Platform Status. Viitattu 11.4.2022. https://chromestatus.com/feature/4857483210260480.

Gillis, Alexander S. N.d. Native app. Verkko-artikkeli natiivisovelluksesta. TechTarget. Viitattu 11.4.2022. https://www.techtarget.com/searchsoftwarequality/definition/native-application-native-app.

GoogleChromeLabs. 2022. Bubblewrap GitHub-projekti ja tutoriaali. GitHub Inc. Viitattu 24.03.2022. https://github.com/GoogleChromeLabs/bubblewrap.

Helping developers build and publish PWAs. N.d. PWA Builder. Työkalu ja sen verkkosivut. PWA builder by Microsoft. Viitattu 24.03.2022. https://www.pwabuilder.com/.

Introduction to Service Worker. 2019. Verkko-artikkeli Service Workerista. Google Developers. Viitattu 6.5.2022. https://developers.google.com/web/ilt/pwa/introduction-to-service-worker.

Juhila, K. N.d. Laadullisen tutkimuksen ominaispiirteet. Tampere: Yhteiskuntatieteellinen tietoarkisto. Viitattu 11.5.2022. https://www.fsd.tuni.fi/fi/palvelut/menetelmaopetus/kvali/mita-on-laadullinen-tutkimus/laadullisen-tutkimuksen-ominaispiirteet/.

Juviler, J. 2021. What Is GitHub? (And What Is It Used For?). Blogi-teksti. HubSpot Inc. Viitattu 17.4 2022. https://blog.hubspot.com/website/what-is-github-used-for.

Kananen, J. 2017. Laadullinen tutkimus pro graduna ja opinnäytetyönä. Jyväskylän ammattikorkeakoulu. Viitattu 11.5.2022. https://janetfinna.fi, Booky.

LePage, P & Bandarra, A. 2020. Trusted Web Activity Overview. Verkko-artikkeli. Chrome Developers. Viitattu 11.4.2022. https://developer.chrome.com/docs/android/trusted-web-activity/.

LePage, P., Beaufort, F. & Steiner, T. 2021. Add a web app manifest. Dokumentaatio Web manifestista. Viitattu 16.5.2022. https://web.dev/add-manifest/.

Lighthouse. 2021. Verkko-artikkeli ja opas työkalusta. Google Developers. Viitattu 22.4.2022. https://developers.google.com/web/tools/lighthouse.

Richard, S. & LePage, P. 2020. What are Progressive Web Apps? Verkko-artikkeli. Web.dev by Google Developers. Viitattu 23.02.2022. https://web.dev/what-are-pwas/.

Service Worker Overview. 2021. Dokumentaatio Service Workerista. Google Developers. Viitattu 6.5.2022. https://developer.chrome.com/docs/workbox/service-worker-overview/

Sign your app. 2022. verkko-opas sovelluksen rekisteröintiin. Viitattu 15.5.2022. https://developer.android.com/studio/publish/app-signing.

Stegner, B. 2021. What Is an APK File and What Does It Do? Explained. Verkko-artikkeli. MUO. Viitattu 24.03.2022. https://www.makeuseof.com/tag/what-is-apk-file/.

Upload PWA to Playstore. N.d. Tutoriaali ja työkalu PWA-sovelluksen muuntamiseen APK-paketiksi. Appmaker.xyz. Viitattu 24.03.2022. https://appmaker.xyz/pwa-to-apk.

What is GitHub? 2016. YouTube-video Githubista. YouTube by Google LLC. Viitattu 17.4.2022. https://www.youtube.com/watch?v=w3jLJU7DT5E.