Android-sovelluksen kehittäminen infonäyttöön — Case Mercantus Oy

Matias Kohanevic

2022 Laurea

Laurea-ammattikorkeakoulu

Android-sovelluksen kehittäminen infonäyttöön — Case Mercantus Oy

Matias Kohanevic

Tietojenkäsittely

Opinnäytetyö

Marraskuu, 2022

Laurea-ammattikorkeakoulu **Tiivistelmä**

Tietojenkäsittely

Tradenomi (AMK)

Matias Kohanevic

Android-sovelluksen kehittäminen infonäyttöön

— Case Mercantus Oy

Vuosi 2022 Sivumäärä 25

Tämän toiminnallisen opinnäytetyön tilaajana toimi ohjelmistoyritys Mercantus Oy. Kehittämistyön tavoitteena oli selvittää, miten tulee suunnitella ja kehittää Android-pohjaista sovellusta, jota on tarkoitus käyttää erilaisten varattavien tilojen ovenpielinäytöissä. Sovelluksen tarkoituksena on estää ovenpielinäyttöjen väärinkäyttöä ja hyödyntää laitteen tärkeimmät ominaisuudet.

Selvitystyössä vertailtiin erilaisia mahdollisia toteutustapoja ja niiden soveltuvuutta toimeksiantajan tarpeisiin. Työkaluiksi valikoitui teknologioita, kuten .NET ja JavaScript, joita oli ennestään käytössä toimeksiantajan muissa projekteissa.

Työn tuloksena syntyi käyttökelpoinen sovellus, joka saavutti kaikki sille asetetut tavoitteet. Sovellus on Mercantus Oy:n asiakkaiden jokapäiväisessä käytössä ja on osoittautunut käyttökelpoiseksi ja pitkäaikaiseksi ratkaisuksi.

Asiasanat: android, dotnet, ohjelmointi, kiosk-sovellus

Laurea University of Applied Sciences **Abstract**

Degree Programme in Business Information Technology

Bachelor's Thesis

Matias Kohanevic

Developing an Android Application for an Information Display Device

— A Case Study of Mercantus Oy

Year 2022 Pages 25

This functional bachelor's thesis was commisioned by the software company Mercantus Oy. The purpose of the thesis was to find out how to develop an Android application, and then design and actually develop the application. The application is intended to be used in information displays located on various bookable premises. The purpose of the application is to prevent misuse of information displays and utilize key features of the device.

In the research phase, several technologies were compared to find out the most appropriate tools for the job. While comparing technologies, those that were already in use in Mercantus' other projects were prioritized and eventually selected, such as .NET and JavaScript.

As a result of this thesis, a new functional Android application, which achieved all the goals set for it, was created. The application is in a daily use by Mercantus' costumers and it has proven to be a usable and long-term solution.

Keywords: android, dotnet, programming, kiosk application

Sisällys

1 Johdanto

Tässä opinnäytetyössä käydään läpi kehittämistehtävää, joka käsittelee Android-sovelluksen kehittämistä infonäyttöä varten. Sovelluksen tarkoituksena oli mahdollistaa tilavarauksien tarkastelun ja uusien varausten luomisen infonäytöstä, hyödyntäen kyseisen laitteen tarjoamat erikoisominaisuudet. Kehittämistyöhön kuului käytettävien työkalujen ja teknologioiden selvitystyöt, sovelluksen suunnittelu, ohjelmointi, testaus ja käyttöohjeiden laatiminen. Kehitystyön tilasi ohjelmistoyritys Mercantus Oy, joka erikoistunut varausjärjestelmiin.

Opinnäytetyö alkaa toimeksiantajan ja Timeworks-varausjärjestelmän esittelyllä, jonka jälkeen käydään läpi projektin tavoitteet. Tavoitteiden läpikäynnin jälkeen selvitetään, millä työkaluilla ja teknologioilla pystytään saavuttamaan kaikki sovellukselle asetetut tavoitteet. Valitut työkalut ja teknologiat testataan yksi kerrallaan, jonka jälkeen sovellusta kasataan toimivaksi kokonaisuudeksi.

Sovelluksen ohjelmoinnin jälkeen luodaan liittymä, jonka avulla Timeworks-varausjärjestelmä pystyy keskustelemaan sovelluksemme kanssa. Kehitystyön päätteeksi käydään läpi millä tavoin sovellusta on testattu ja miten valmista sovellusta toimitettiin asiakkaille.

Opinnäytetyön päätteeksi pohditaan, miten kehitystyössä on onnistuttu ja mitä sen aikana on tullut opittua. Sen lisäksi käydään läpi asiakkailta saatua palautetta ja arvioidaan, onko sovellukselle olemassa potentiaalisia jatkokehityskohteita.

2 Toimeksiantajan esittely

Mercantus Oy on vuonna 2005 perustettu suomalainen ohjelmistoyritys, jonka päätuotteena on Timeworks-ohjelmisto. Yritys koostuu noin 10 hengen tiimistä, joka toimii kokonaan Suomesta käsin. Yrityksen käyttämät teknologiat perustuvat moderneihin ja tietoturvallisiin Microsoft-kehitysalustoihin.

Timeworks-ohjelmiston avulla ratkaistaan varausten hallinnan tarpeet kokonaisvaltaisesti. Se on kattava tilavarausohjelmisto, jolla varataan neuvotteluhuoneita ja työpisteitä sekä niihin liittyviä palveluita, kuten tarjoilutilauksia ja tarvittavia laitteita. Timeworks koostuu moduuleista, joista voidaan räätälöidä asiakkaiden käyttötarpeita vastaava kokonaisuus.

Mercantuksen asiakkaina ovat muun muassa kunnat ja kaupungit, valtionhallinto, yritykset ja yhteisöt, yrityspuistot ja coworking-tilat, museot ja tapahtumatalot (Mercantus 2022).

3 Kehitystyön tavoitteet

Mercantuksen asiakkaiden varattavien tilojen ulkopuolelle on asennettuna ovenpielinäyttöjä, joiden tarkoituksena on näyttää tilojen varaajille sen hetkisen varaustilanteen lisäksi tulevat varaukset, kellonaika, huoneen nimi/numero sekä varaajan nimi ja kokouksen aihe. Päivän tulevat kokoukset ovat myös nähtävissä. Nämä kosketusnäytöillä varustetut ovenpielinäytöt mahdollistavat myös uusien varausten tekemisen.

Kuvio 1: Ovenpielinäytössä näkyvä varaustilanne (Mercantus 2022)

Kehitystyön päätavoitteena oli luoda kyseisille ovenpielinäytöille sovellus, jonka tarkoituksena oli tehdä ovenpielinäytöstä laite, joka on rajoitettu ainoastaan tilavarauksien tarkasteluun ja uusien luomiseen. Sovellus tulisi olla Android-pohjainen ja sen täytyi tukea vanhoja Android-versioita.

3.1 Kiosk-sovellus

Kun julkiseen paikkaan asennetaan kosketusnäyttö, jossa pystyy tekemään varauksia tai muuta asiointia, on tärkeää, että käyttäjä ei pääse poistumaan sovelluksesta tai tekemään mitään muuta kuin sitä, mihin kosketusnäyttö on tarkoitettu. Hyviä esimerkkejä ovat kauppakeskuksen digitaaliset opastekyltit, lipunmyyntiautomaatit tai itsepalvelukassat. Laitteen rajoittamista yhteen sovellukseen kutsutaan kiosk-tilaksi (Google Developers 2022).

Kiosk-tilan implementointi tähän projektiin oli välttämätöntä, koska ilman sitä on aina olemassa riski, että laitteeseen kohdistuu väärinkäyttöä. Kioskitilassa voidaan ehkäistä lukuisia

mahdollisia väärinkäyttöjä kuten esimerkiksi näytön tai laitteen sammuttaminen, sovelluksen sulkeminen, laitteen asetusten muuttaminen ja navigointi sopimattomiin sivustoihin.

3.2 Tilanvarausnäkymä

Timeworks-varausjärjestelmässä on jo olemassa web-käyttöliittymä tilanvarauksia varten. Kyseinen web-sivu skaalautuu erinomaisesti myös pienempiin näyttöihin. Jotta ei tarvitsisi rakentaa uusia monimutkaisia rajapintoja ja uutta samanlaista käyttöliittymää sovellusta varten, yksi kehitystyön tavoitteista oli selvittää, miten tässä sovelluksessa voitaisiin hyödyntää tätä olemassa olevaa, hyväksi todettua selainnäkymää.

3.3 Virheenhallinta

Yhtenä kehitystyön keskeisimmistä tavoitteista oli kehittää tapa käsitellä odottamattomia tilanteita, kuten esimerkiksi sähkökatkokset tai internet-yhteyden menettäminen, jotka voivat aiheuttaa laitteen uudelleenkäynnistyksen tai sovelluksen pysähtymisen.

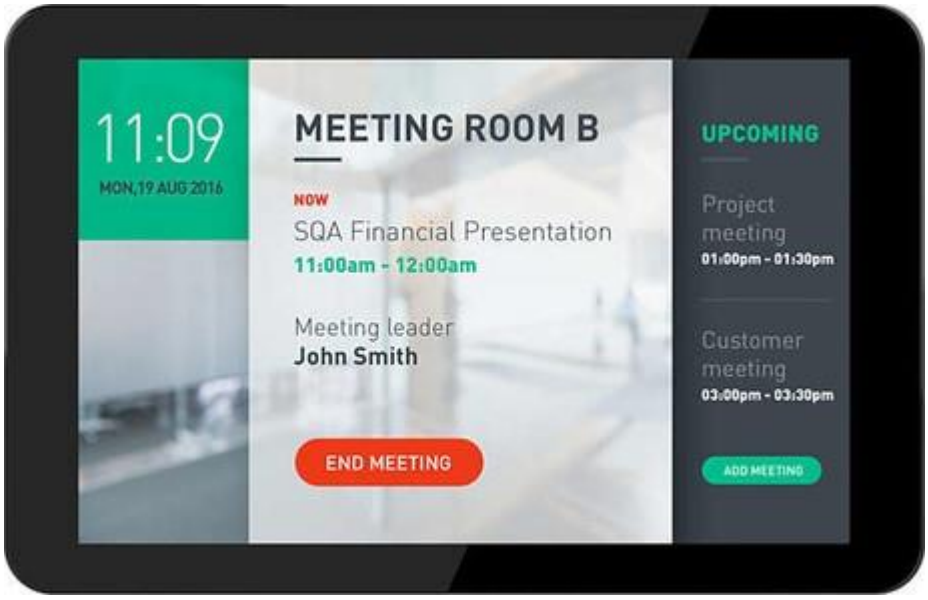
Verkkokatkon sattuessa, käyttäjälle on näytettävä asianmukainen virheviesti samalla kun taustalla yhteyttä yritetään muodostaa uudelleen. Kun internet-yhteys on palautunut, sovelluksen on pystyttävä jatkamaan normaalia ajoa ilman käyttäjän väliintuloa.

Sähkökatkosten varalta toivottiin ominaisuutta, joka avaisi sovelluksen automaattisesti uudelleen ja laittaisi sen takaisin kiosk-tilaan aina kun ovenpielinäyttö käynnistyy uudelleen syystä tai toisesta.

Ideaalitilanteessa sovellusta asennettaisiin ovenpielinäyttöön, jonka jälkeen tehtäisiin alkumääritykset. Tämän vaiheen jälkeen ovenpieli olisi käyttövalmis, ja virhetilanteiden sattuessa ei tarvittaisi ikinä ihmisresursseja tai minkäänlaisia toimenpiteitä ongelmien korjaamiseksi.

3.4 Ovenpielinäyttö

Ovenpielinäytön malli, jossa sovellusta käytetään, on Philips 10BDL3051T, joka on Android 4.4 -pohjainen 10 tuuman tabletti. Kyseinen laite ja sen ohjelmisto ovat tarkoitettuja käytettäväksi nimenomaan erilaisiin varausjärjestelmiin tai ohjauspaneeleihin.



Kuvio 2: Philips 10BDL3051T (Philips 2022)

Ovenpielinäytön tärkeimpiin ominaisuuksiin kuuluu näytön molemmilla sivuilla olevat LED-valot, johon on mahdollista asettaa miljoonia eri värejä. Toimeksiantajan toiveena oli saada nämä LED-valot muuttamaan värejään varattavien tilojen varaustilanteiden mukaan.

LED-valojen ohjaamisen mahdollistamiseksi oli myös selvitettävä, miten saataisiin nykyinen JavaScript-pohjainen Timeworksin tilanvaraussivu kertomaan Android-sovellukselle, että nyt haluttaisiin vaihtaa ovenpielinäytön LED-valot tiettyyn väriin. Tämä tarkoitti sitä, että Timeworks-varausjärjestelmän lähdekoodia olisi hieman päivitettävä.

4 Työkalut ja teknologiat

Android-sovellusten tekemiseen on nykyään tarjolla monia varteenotettavia vaihtoehtoja. Työkalujen ykkösvalintaperusteena oli se, että ne mahdollistaisivat kaikkien asetettujen tavoitteiden saavuttamisen. Valinnassa priorisoitiin sellaisia teknologioita, jotka ovat valmiiksi käytössä toimeksiantajan muissa projekteissa. Tutut työkalut tekevät mahdollisesta jatkokehityksestä ja ylläpidosta huomattavasti helpompaa, sen lisäksi että Mercantuksen koodikanta säilyy yhtenäisempänä. Näiden seikkojen vuoksi kehitystyössä käytetyt teknologiat pohjautuivat vahvasti Microsoftin tarjoamiin työkaluihin.

4.1 .NET

.NET on avoimen lähdekoodin kehittäjäalusta erityyppisten sovellusten rakentamiseen (Microsoft 2022). Tähän alustaan kuuluu erilaisia koodieditoreita, kirjastoja ja ohjelmointikieliä, kuten esimerkiksi C#-ohjelmointikieli, jolla Android-sovellusta kehitettiin.

4.1.1 NuGet-paketit

NuGet on .NET:n ylläpitämä paketinhallintatyökalu, joka tarjoaa mahdollisuuden hankkia ja hallita kolmannen osapuolen ohjelmistokomponentteja, joita voi implementoida omaan koodiin (Microsoft 2022).

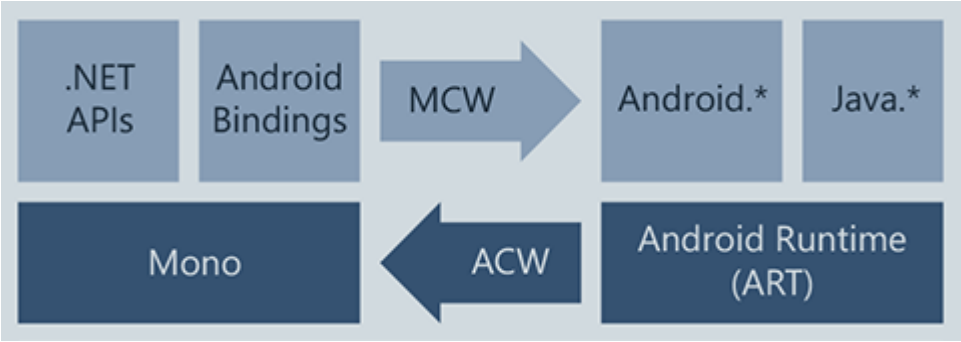
Philips 10BDL3051T tabletin ohjaamiseksi on olemassa PhilipsSignageDisplaySicp -NuGet-paketti, jonka avulla voidaan ohjelmallisesti käskeä tablettia tekemään noin 50 erilaisia toimintoa, kuten esimerkiksi äänenvoimakkuuden säätäminen, näytön sammuttaminen tai kosketusnäytön ottaminen pois käytöstä (aolde 2019). Tässä projektissa tätä pakettia tarvittiin ovenpielinäytön LED-valojen käyttöönottoon ja niiden värien vaihtamiseen varaustilanteiden mukaan.

4.1.2 Visual Studio

Visual Studio on kattavin IDE (ohjelmointiympäristö) .NET kehittäjille. Koodieditorin lisäksi Visual Studio yhdistää versiohallinnan, laajennukset ja monia muita ominaisuuksia yhteen paikkaan. Tehokas IDE tekee ohjelmoinnista sujuvampaa ja vähentää virheiden mahdollisuutta (Microsoft 2022).

4.1.3 Xamarin

Xamarin on .NET:n päälle rakennettu avoimen lähdekoodin alusta nykyaikaisten Windows, iOS ja Android-sovellusten kehittämiseen. Xamarin tarjoaa mahdollisuuden implementoida olemassa olevia Android-kirjastoja omaan projektiin, mikä antaa kehittäjälle pääsyn laajaan valikoimaan kolmannen osapuolen koodia (Microsoft 2022).



Kuvio 3: Xamarin.Androidin arkkitehtuuri (Microsoft 2022)

Xamarin mahdollistaa myös monialustaisten sovellusten kehittämisen, mutta koska tässä tapauksessa sovelluksen kohteena on Android-laite, sovelluksesta tulee natiivi Xamarin.Android -sovellus. Xamarin oli kehitystyölle luonnollinen valinta, koska sen lisäksi, että se pystyy

täyttämään applikaatiolle asetetut tavoitteet, se käyttää C#-ohjelmointikieltä ja Visual Studiota, jotka ovat käytössä muissakin toimeksiantajan projekteissa.

4.2 Webview

Webview on työkalu, jonka avulla voidaan upottaa web-sivuja sovellukseen. Webview:ssa näytetään web-sivun sisältöä ilman verkkoselaimen osoitepalkkia tai navigointinappeja. Se on hyödyllinen, kun tarvitaan enemmän käyttöliittymän hallintaa (Google Developers 2022).

Tässä projektissa Webview mahdollistaa Timeworksin varaussivunäkymän upottamiseen Android-sovellukseen, joten ei ole tarvetta rakentaa uutta Android-pohjaista käyttöliittymää. Se on merkittävä etu, sillä mikäli Mercantus haluaa tulevaisuudessa päivittää varaussivun ulkoasua, tätä sovellusta ei tarvitsee päivittää, koska vain varausjärjestelmän web-sivua tarvitsee muokkausta.

4.3 Vue.js

Vue on moderni JavaScript-kehys, jolla Timeworksin varausjärjestelmän käyttöliittymä on toteutettu. Timeworksin Vue-koodiin jouduttiin tekemään pieniä säätöjä, jotta se osaisi kommunikoida Android-sovelluksellemme kanssa.

5 Kehitystyön kulku

Kehitys alkoi varmistamalla, että valitut teknologiat olivat varmasti sopivia sovelluksen tarkoitukseen. Olennaisinta alussa oli testata, toimiiko LED-valojen ohjaus, WebView-näkymä ja kiosk-ominaisuudet. Näiden ominaisuuksien testaamisen aikana syntyi jonkun verran valmiita sovelluksen osia, joita pystyttiin hyödyntämään projektin myöhemmässä vaiheessa.

5.1 Uusi Xamarin-applikaatio

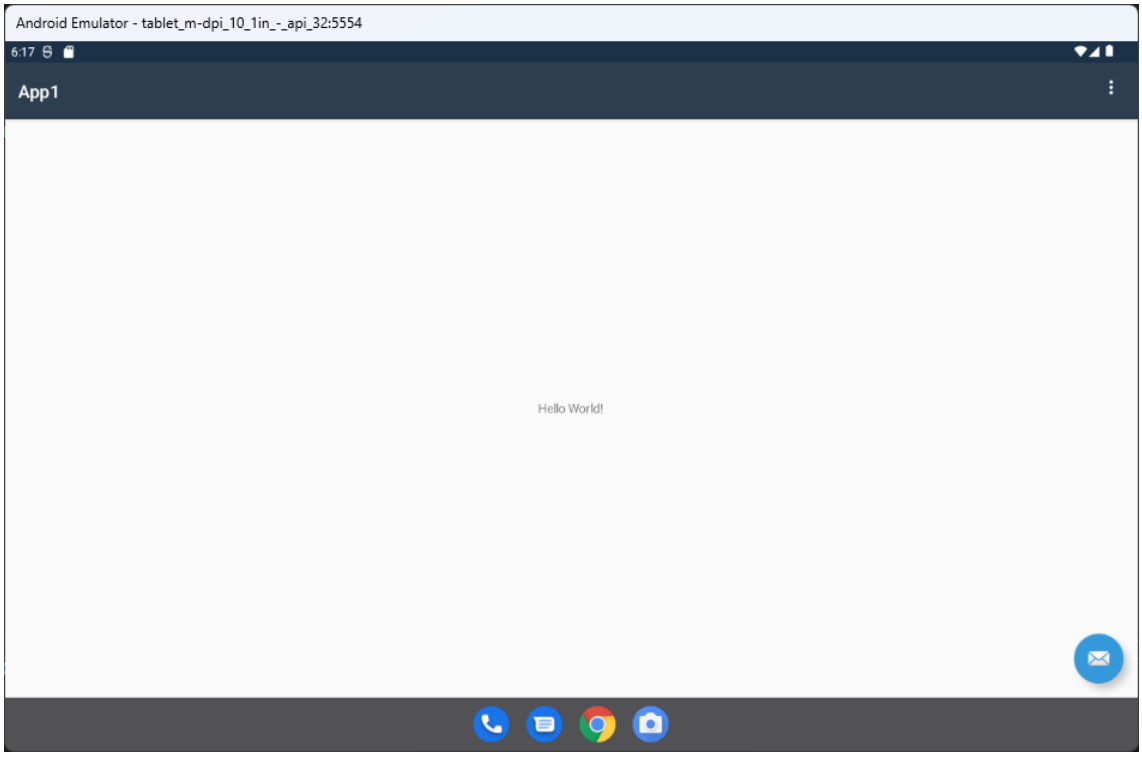
Visual Studion projektipohjat tarjoavat käyttäjille uudelleenkäytettäviä malleja, jotka sisältävät peruskoodin ja -rakenteen. Mallit tarjoavat käyttäjille kätevän lähtökohdan projektien luomiseen, jonka jälkeen niitä voi lähteä muokkailemaan omien tarpeiden mukaan. Käytettävissä on oletuksena asennettuja malleja tai yhteisön luomia, ladattavia malleja (Microsoft 2022).

Uuden sovelluksen luomiseksi, Visual Studiossa luotiin uusi projekti käyttämällä valmista ”Android App (Xamarin)” -mallia, joka luo yksinkertaisen Android-sovelluksen. Projektikansiossa on kaikki tiedostot, jotka tarvitaan sovelluksen käynnistämiseksi. Tässä sovelluksessa oli tässä vaiheessa vain perusaloitusnäkymä, jossa oli muutama komponentti.

5.2 Emulaattorin käyttöönotto

Emulaattori on ohjelmisto, jolla voi jäljittää toisen laitteen toimintaa. Android-emulaattori mahdollistaa sovellusten testaamisen eri laitteilla, ilman aktuaalista fyysistä laitetta. Emulaattorilla pystyy simuloimaan lähes kaikki todellisen Android-laitteen ominaisuuksia, kuten saapuvia puheluita ja tekstiviestejä, sijainninmääritys, näytön kääntämistä ja paljon muuta (Google Developers 2022).

Sovelluksen testaamisen helpottamiseksi, kehitysympäristöön asennettiin Android-emulaattori, joka jäljittää 10 tuumaista tablettia ja joka on mahdollisimman samankaltainen kuin Mercantuksen käyttämä ovenpielinäyttö.



Kuvio 4: Xamarinin sovelluspohja emulaattoriajossa

Vaikka emulaattoria voidaan käyttää useiden ominaisuuksien testaamiseen, parhaimman toimivuuden varmistamiseksi ja tiettyjen asioiden testaamiseen tarvitaan oikea infonäyttö.

5.3 Ovenpieninäytön käynnistysapplikaation määrittäminen

Ovenpieninäytöissä on mahdollisuus määrittää sovellus, joka avautuu automaattisesti silloin kun laite käynnistyy. Tämä asetus oli testattava oikealla laitteella, sillä emulaattorin jäljittämällä laitteella ei ollut vastaavaa ominaisuutta. Ovenpielinäytön laiteasetuksista asetettiin tätä testisovellusta käynnistysapplikaatioksi, tarkoituksena testata, saadaanko sovellus takaisin päälle ilman ylimääräisiä toimenpiteitä, mikäli laite sammuu syystä tai toisesta. Laitetta

käynnistettiin uudelleen, jonka jälkeen testisovellus avautui automaattisesti, juuri niin kuin oli tarkoituskin.

5.4 LED-valojen käyttöönotto

LED-valojen testaamista varten asennettiin aiemmin mainittu PhilipsSignageDisplaySicp -NuGet-paketti. Asennus suoritettiin Visual Studiossa NuGet Package Managerin kautta. Tämän jälkeen sovelluksen lähdekoodia muokattiin niin, että käynnistyessä sovellus käskee NuGet-paketin ohjeiden mukaisesti laitetta sytyttämään LED-valojaan ja vaihtamaan niiden väriä vihreäksi.

LED-valojen toimivuutta ei pystytty testaamaan emulaattorilla, koska sellaista toiminnallisuutta ei yksinkertaisesti pysty emuloimaan. Testaamista varten jouduttiin luomaan sovelluksen asennustiedosto, joka asennettiin USB-tikulla oikeaan laitteeseen. Sovelluksen käynnistyessä LED-valot syttyivät odotetusti ja muuttuivat vihreiksi.

Kuvio 5: Ovipielinäytön LED-valo

5.5 WebView:n käyttöönotto

WebViewn testaamiseksi, mallisovelluksen aloitusnäkymän kaikki esimerkkikomponentit poistettiin ja niiden tilalle luotiin uusi WebView. WebViewn leveydeksi ja korkeudeksi asetettiin kaikki saatavilla oleva tila, ja URL-osoitteeksi annettiin erään Timeworksin demona käytetyn varaussivun osoitetta. Sovellusta käynnistettiin emulaattorilla, jonka jälkeen aloitusnäkymäksi avautui sovellukseen upotettu varaussivu. Seuraavaksi testattiin tabletilla ja kuten odotettua, varaussivu skaalautui tablettiin erinomaisesti.

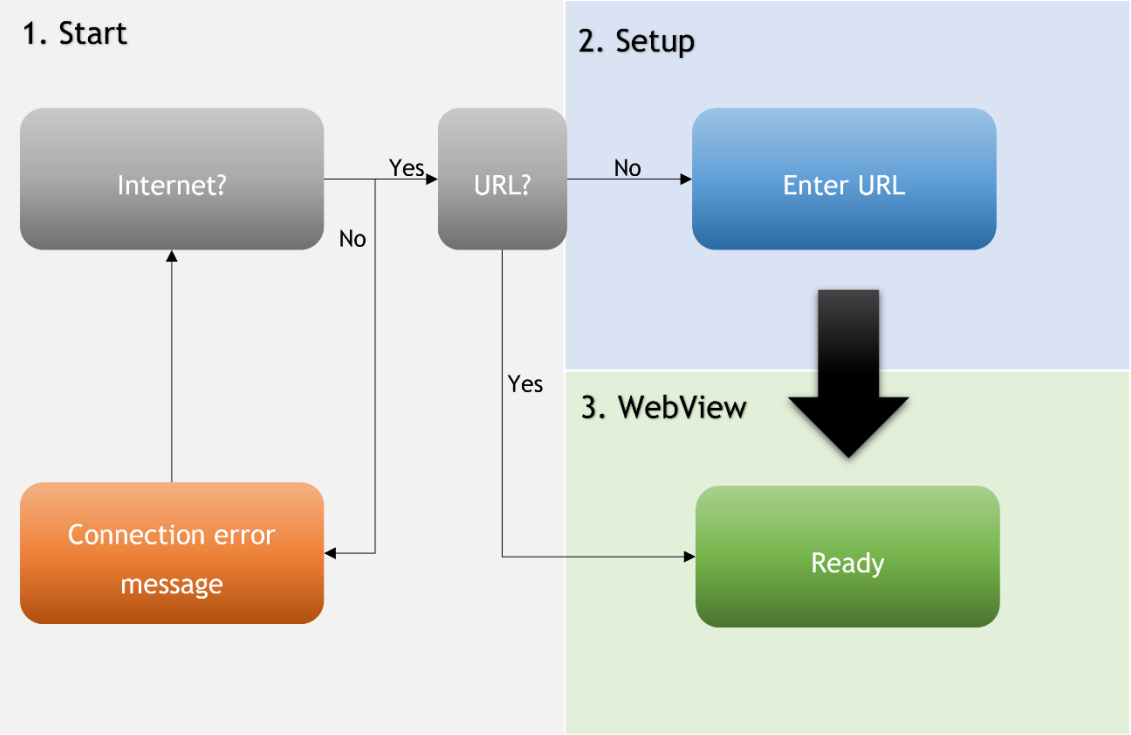
5.6 Kiosk-ominaisuuden käyttöönotto

Kun varaussivua saatiin upotettua WebView-näkymään, oli vielä selvitettävä, miten laitetta pystyy lukitsemaan yhteen sovellukseen, jotta käyttäjä ei voisi poistua siitä. Sopivaksi ratkaisuksi valikoitui Androidin lock task-tila, joka rajoittaa laitteen toimintoja ja lukitsee sen yhteen tai useampaan valittuun sovellukseen (Google Developers 2022).

Lock task-tilan implementoimiseksi, aloitusnäkymän luomisvaiheessa yksinkertaisesti kutsuttiin StartLockTask()-metodia, joka aktivoi lock task-tilaa. Tabletilla testatessa tuli huomattua, että lock task-tila tosiaankin toimii. Se toimi niin hyvin, että keskeneräistä sovellusta ei saanut enää millään suljettua ilman erikoistoimenpiteitä. Tämä oli juuri sitä mitä haluttiin. Kehitystyön ja testauksen jatkuessa oli kuitenkin pyrittävä välttämään laitteen totaalista lukitsemista kiosk-tilaan, koska se aiheutti ylimääräistä säätöä.

6 Sovelluksen toiminta

Onnistuneen selvitystyön jälkeen, sovellus oli valmis ohjelmoitavaksi. Ajatuksena oli jakaa sovellus kolmeen komponenttiin: käynnistys, määritys ja WebView-päänäkymä.



Kuvio 6: Sovelluksen rakenne

6.1 Käynnistyskomponentti

Sovelluksen käynnistyessä ensimmäiseksi tarkistetaan, onko internet-yhteys muodostettu. Internet-yhteys on välttämätön sovelluksen toiminnan kannalta, joten ilman sitä ei voida missään nimessä edetä seuraavaan näkymään. Internet-virheen sattuessa näytetään yksinkertainen virheviesti, jossa kerrotaan käyttäjälle, että internet-yhteyttä ei ole ja sitä yritetään muodostaa. Samaan aikaan taustalla tarkistetaan kolmen sekunnin välein, onko internet-yhteys muodostunut.

Kun internet-yhteys on muodostettu, sovellus tarkistaa onko määrityksiin tallennettu tulevan päänäkymän WebView:n URL-osoite, ja sen perusteella ohjataan seuraavaan näkymään. Jos määrityksistä löytyy tallennettu URL-osoite, ohjataan suoraan WebView-päänäkymään, muuten ohjataan käyttäjää syöttämään WebView:n URL osoitetta.

6.2 URL-osoitteen määritys

Määritysnäkymään ohjataan ainoastaan silloin, kun sovellus ei löydä tallennettua URL-osoitetta, eli käytännössä sovelluksen ensimmäisellä käynnistyskerralla. Näkymän ulkoasuna on vain yksinkertainen tekstikenttä, johon pyydetään käyttäjää syöttämään varaussivun URL-osoitetta. Ulkoasuun ei ole juurikaan panostettu, koska URL-syötön jälkeen tätä näkymää ei näytetä käyttäjälle enää ikinä.

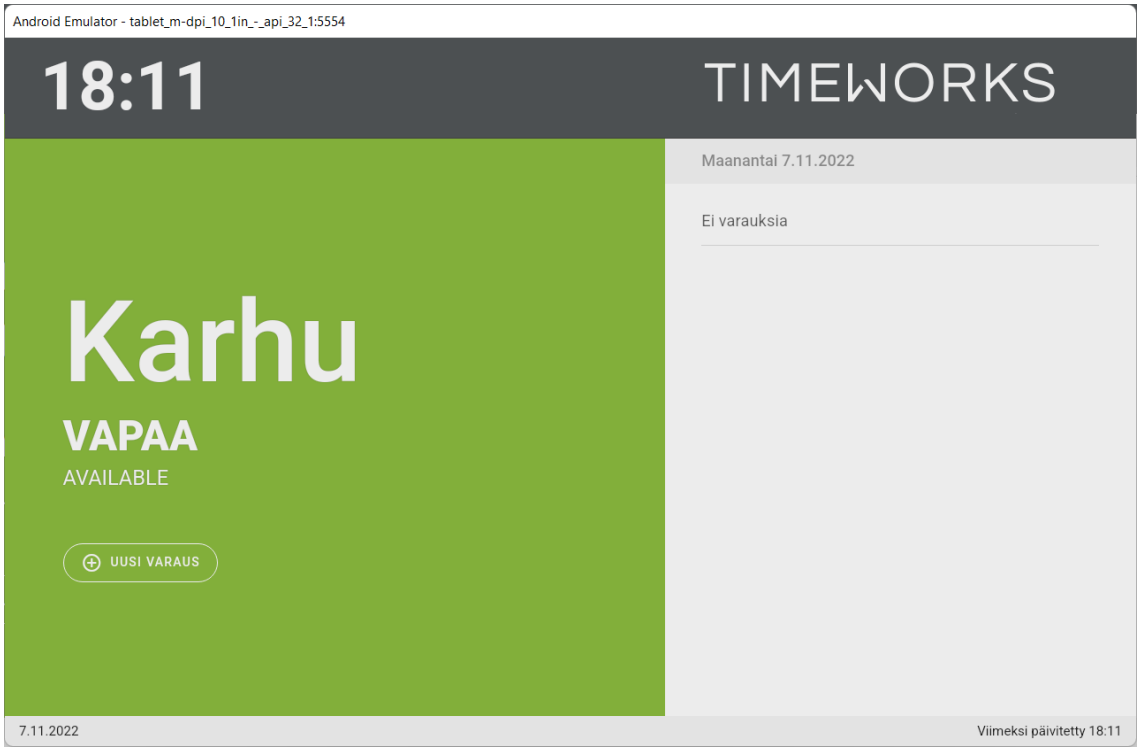
Kun URL-osoite on syötetty onnistuneesti, sovellus tallentaa sen omiin asetuksiinsa, jonka jälkeen URL-osoitetta ei voi enää vaihtaa. Ainoa tapa määrittää URL-osoite uusiksi on asentaa sovellus uudelleen.

6.3 WebView-näkymä

Sovelluksen päänäkymä on varaussivu, johon edellisessä näkymässä syötetty URL-osoite vie. Näkymän ulkoasuna toimii Timeworksin varausnäkymä, joka on upotettu sovellukseen WebView:n avulla.

6.3.1 Lock task-tila

Kun tämä näkymä latautunut, sovellusta asetetaan välittömästi kiosk-tilaan kutsumalla StartLockTask()-metodia. Tämän tilan tehostamiseksi, näkymää asetetaan ohjelmallisesti kokonäyttötilaan, jonka lisäksi piilotetaan tabletin navigointipainikkeet, yläpalkki ja alapalkki.



Kuvio 7: Webview:iin upotettu demovaraussivu kiosk-tilassa, emulaattoriajossa

Tässä vaiheessa tabletti on lukittuna Webview:iin upotettuun varaussivuun. Tabletin näyttöä ei voi enää sammuttaa, sovelluksesta ei voi navigoida pois, eikä laitteen asetuksia pysty tarkastelemaan

6.3.2 Virheenhallinta

Androidin dokumentaation mukaan, OnReceivedError()-metodin avulla on mahdollista siepata yhteysvirheitä. Tämän metodin toiminnallisuuksia korvaamalla, voidaan käskeä metodia tekemään erilaisia toimintoja aina kun se saa virheen siepattua (Google Developers 2022). Virheiden sieppaaminen ja käsitteleminen on tässä vaiheessa kriittistä, koska kaatunut web-sivu ei osaa itsestään uudelleen latautua.

Kuvio 8: Kaatunut WebView

OnReceivedError()-metodin toimintaa korvataan eri toiminnallisuuksilla niin, että kun se saa yhteysvirheen kiinni, se yksinkertaisesti käskee WebView:tä lataamaan sivun uudelleen. Jos sivu ei vieläkään lataudu, se antaa uuden yhteysvirheen, joka siepataan ja taas käsketään lataamaan sivua uudelleen. Näin jatketaan, kunnes virheitä ei enää tule. Jos yhteys saadaan muodostettua, sivu latautuu uudelleen ja sovelluksen ajo jatkuu normaalina.

7 Web-sivun kommunikointi sovelluksen kanssa

WebView:ssä käytetty varaussivu on toteutettu Vue.js:llä, joka on JavaScript-ohjelmistokehys (framework) modernien web-käyttöliittymien rakentamiseen. Koska kyseinen web-sivu käyttää JavaScript-ohjelmointikieltä, se ei osaa suoraan kommunikoida Android-sovelluksen kanssa, joka on tässä tapauksessa toteutettu C#-ohjelmointikielellä.

Tabletin LED-valojen ohjaamiseksi, varaussivun täytyy jotenkin kertoa Android-sovellukselle, mihin väriin halutaan vaihtaa LED-valot. JavaScript on WebView:ssa oletuksena poistettu käytöstä, joten kommunikoinnin mahdollistamiseksi se on otettava käyttöön. JavaScriptin ottaminen käyttöön tapahtuu WebView-asetuksiin liitettyjen WebSettings-asetusten kautta (Google Developers 2022).

7.1 JavaScript interface

JavaScriptin ottaminen käyttöön antaa kehittäjän luoda liittymiä JavaScript-koodin ja Android-koodin välille. Tämän jälkeen varaussivun JavaScript-koodi voi kutsua Android-sovelluksessa olevaa C#-ohjelmointikielellä kirjoitettua metodia, jonka avulla vaihdetaan tabletin LED-valojen väriä.

7.2 Värinvaihtometodi

Android-sovelluksessa kirjoitettu värinvaihtometodi on hyvin yksinkertainen: se ottaa yhden parametrin vastaan, joka on merkkijono, jossa lukee haluttu väri hex-muodossa. Esimerkiksi punainen väri hex-muodossa on #ff0000. Tämän jälkeen metodi käskee laitetta vaihtamaan tabletin LED-valot annettuun väriin.

Varaustilanne	Väri	Hex
Vapaa	Vihreä	#00FF00
Odottaa	Sininen	#0000FF
Varattu	Punainen	#FF0000

Taulukko 1: Varaustilanteet ja niiden värit

7.3 Vue Watcher

Tässä vaiheessa projektia jouduttiin hieman muokkaamaan varaussivun lähdekoodia Timeworksin puolella, jotta Vue.js:lla toteutettu varaussivu osaisi kertoa Android-sovellukselle, milloin vaihtaa tabletin LED-valot ja mihin väriin.

Niin kuin nimikin jo kertoo, Watcher-toiminnolla voidaan tarkkailla tietyn arvon tilanmuutoksia ja tehdä toimenpiteitä näiden muutosten perusteella (Vuejs 2022). Tässä tapauksessa tarkkaillaan varaustilannetta. Jos varaustilanne muuttuu esimerkiksi varatuksi, Vuen tarkkailija huomaa statuksen muuttuneen, jonka jälkeen tätä tilanmuutosta käsitellään niin, että kutsutaan Android-sovelluksen värinvaihtometodia ja annetaan sille parametriksi #FF0000, joka on punaisen värin hex-muoto. Infonäytön LED-valot vaihtuvat välittömästi punaisiksi.

Värin kertominen Android-sovellukselle on ainoa asia, mikä vaaditaan web-käyttöliittymältä. Sovellus hoitaa loput värinvaihtamiseen liittyvät toimenpiteet.

8 Testaus

Sovelluksen testaus on yksi tärkeimmistä vaiheista sovelluksen kehitysprosessissa. Jatkuvalla testaamisella voidaan varmistaa sovelluksen toimivuutta ja käytettävyyttä ennen sen julkistamista (Google Developers 2022). Sovelluskehityksen aikana testattiin jokaista vaihetta perusteellisesti sekä emulaattorilla että infonäytöllä.

Testauksen tavoitteena oli kaikin tavoin etsiä mahdollisia puutteita sovelluksesta, josta voisivat olla loppukäyttäjälle jotain haittaa. Huolellisen testauksen aikana ilmeni muutamia pieniä puutteita, jotka korjattiin heti. Testilaitteeseen asennettiin testauksen aikana useita versiota sovelluksesta.

8.1 Virheenhallinnan testaus

Sovelluksen toimivuuden testaamiseksi aiheutettiin ongelmatilanteita tarkoituksella. Sovelluksen virheenhallintaa testattiin itseaiheutetuilla internet-katkoilla. Tämän testin ideana oli varmistaa, että käyttäjä saa virheviestin nopeasti ja sovellus palaa normaaliin ajoon heti kun mahdollista.

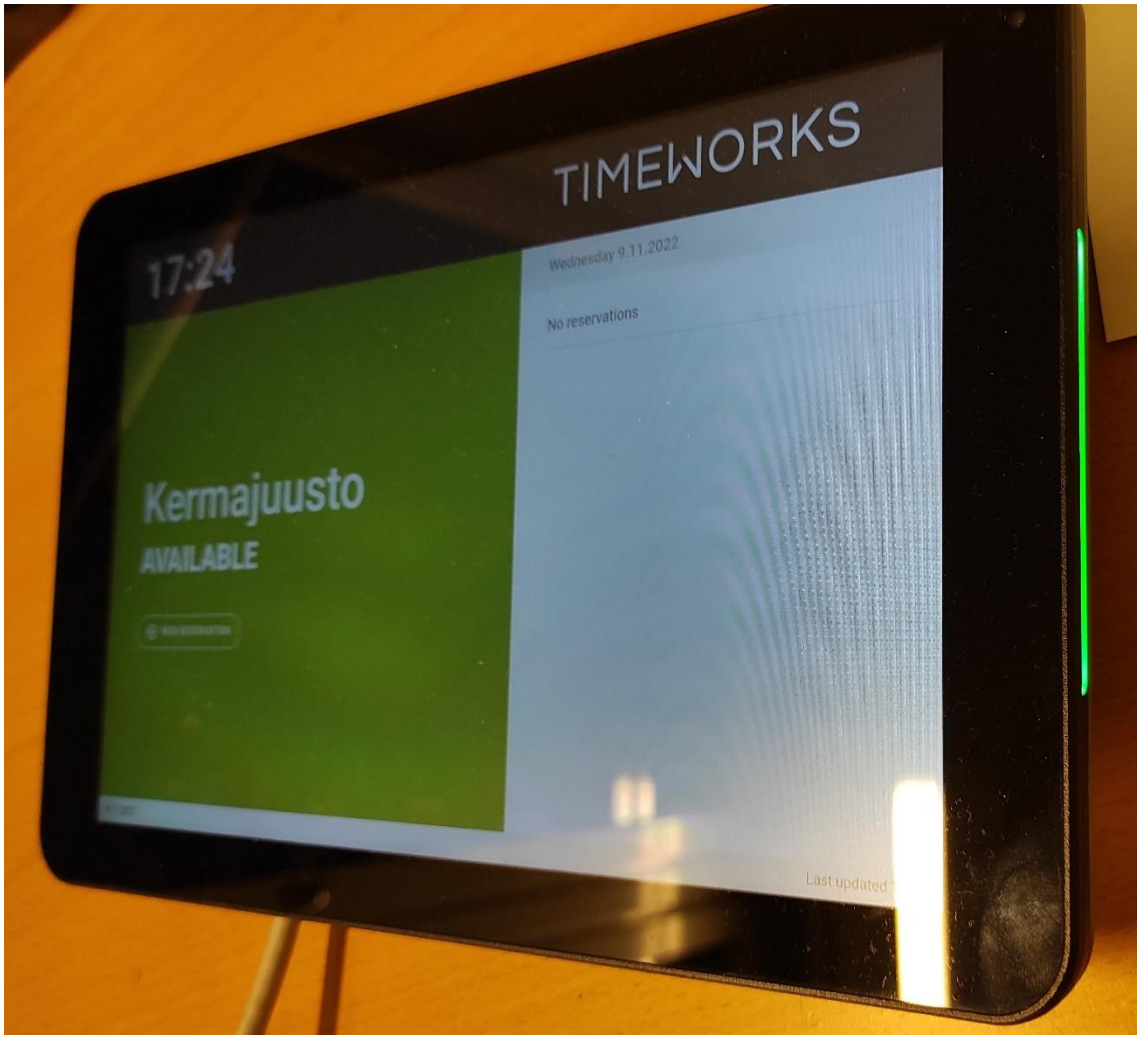
Sähkökatkoja simuloitiin irrottamalla tabletin virtajohtoa. Kyseinen laite käynnistyy automaattisesti, kun virtapiuha on kiinni. Tällä testillä haluttiin varmistaa, että laitteen uudelleenkäynnistyttyä, sovellus avautuu automaattisesti ja menee takaisin kiosk-tilaan.

8.2 Kiosk-tilan testaus

Kiosk-tila on tämän sovelluksen avainominaisuus, joten sitä tuli testattua erittäin perusteellisesti. Kaikkia mahdollisia tabletin näppäimiä tuli painettua, jonka jälkeen yritettiin myös kaikkia mahdollisia näppäinyhdistelmiä. Kosketusnäyttöä koitettiin pyyhkäistä joka suuntaan, mutta kiosk-tila pysyi päällä. Testi osoitti, että sovelluksesta on mahdotonta poistua ja aiheuttaa häiriötilanteita.

8.3 Yleisen toimivuuden testaus

Testitabletti jätettiin päälle useiksi viikoiksi, jonka aikana tehtiin satoja testivarauksia Timeworksin demotilaan. Testivarauksen alkaessa ja loppuessa tarkkailtiin tabletin LED-valoja, jotka vaihtuivat odotetusti varaustilanteen mukaan. Samalla tuli testattua, että sovellus toimii hyvin, vaikka se on ajossa pitkiä aikoja.



Kuvio 9: Vapaana oleva demotila

9 Jakelu ja asennusohjeet

Sovellus ei ole perinteinen julkisessa levityksessä oleva sovellus, jonka pystyy lataamaan Google Play:stä. Tämän sovelluksen tapauksessa jakelu tapahtuu lähettämällä asennustiedostoa tarvitseville asiakkaille.

9.1 Asennustiedosto

Asennustiedoston luominen tapahtuu Android SDK-työkalujen avulla, jotka kokoavat sovelluskoodin ja kaikki muut tarvittavat resurssit yhdeksi APK-tiedostoksi (Google Developers 2022). Tämän sovelluksen asennustiedoston koko on vain 7,25 megatavua, joten sen pystyy helposti toimittamaan esimerkiksi sähköpostin välityksellä. Asentaminen tapahtuu siirtämällä asennustiedostoa USB-tikkuun, joka asetetaan laitteen takana olevaan USB-porttiin.

9.2 Käyttöohjeet

Sovelluksen käyttöönoton helpottamiseksi, asiakkaille toimitettiin erittäin yksityiskohtaiset käyttöohjeet, jotta he pystyisivät asentamaan ja määrittämään sovellusta kokonaan itse. Tarkkojen käyttöohjeiden ansiosta, Mercantuksen ei tarvinnut lähettää ammattilaista asentajaa kiertämään asennuspaikkoja USB-tikun kanssa. Käyttöohjeet sisältävät jokaisesta asennusvaiheesta otettuja näytönkaappauksia, johon on lisätty tarkennuksia kuvanmuokkaustyökaluilla ja kuvateksteillä.

10 Yhteenveto

Opinnäytetyön tuloksena syntyi toimeksiantajan kaikkia vaatimuksia täyttävä uusi Android-sovellus. Mercantuksen tiimi ja sen asiakkaat olivat hyvin tyytyväisiä sovellukseen, sillä infonäyttöjen kiosk-tila ja LED-valojen ohjaus olivat pitkään kaivattuja ominaisuuksia.

Kehitystyön aikana oppi paljon asioita Android-kehityksestä, joista voi potentiaalisesti olla paljon hyötyä kehittäjän uralla. Verkossa on nykyään tarjolla paljon materiaalia Android-kehittämisen oppimiseen ja erittäin kattavat dokumentaatiot. Tiedonhankinnassa käytettiinkin paljon Microsoftin ja Androidin dokumentaatioita. Näitä dokumentaatioita seuraamalla ja yleisellä ohjelmointiosaamisella pääsee helposti alkuun.

Sovelluksen jatkokehitykselle ei ole tarvetta, koska sen käyttötarkoitusta ei tulla muuttumaan tulevaisuudessa. Sen päivittäminen olisi muutenkin työlästä, koska sovellus täytyisi asentaa uudelleen jokaiseen infonäyttöön erikseen, jossa se on käytössä. Sovelluksen käytössä ei ole ilmennyt ongelmia, joten ilman yllättävän bugin löytämistä, sovellusta ei tulla päivittämään.

Lähteet

Sähköiset

aolde 2019. PhilipsSignageDisplaySicp 1.0.1. Viitattu 2.11.2022. https://www.nuget.org/packages/PhilipsSignageDisplaySicp

Google Developers 2022. Application Fundamentals. Viitattu 9.11.2022. https://developer.android.com/guide/components/fundamentals

Google Developers 2022. Dedicated devices overview. Viitattu 28.10.2022. https://developer.android.com/work/dpc/dedicated-devices

Google Developers 2022. Lock task mode. Viitattu 9.11.2022. https://developer.android.com/work/dpc/dedicated-devices/lock-task-mode

Google Developers 2022. Run apps on the Android Emulator. Viitattu 3.11.2022. https://developer.android.com/studio/run/emulator

Google Developers 2022. Test apps on Android. Viitattu 7.11.2022. https://developer.android.com/training/testing

Google Developers 2022. Use JavaScript in WebView. Viitattu 7.11.2022. https://developer.android.com/develop/ui/views/layout/webapps/webview#UsingJavaScript

Google Developers 2022. WebViewClient. Viitattu 10.11.2022. https://developer.android.com/reference/android/webkit/WebViewClient

Google Developers 2022. Web-based content. Viitattu 2.11.2022. https://developer.android.com/develop/ui/views/layout/webapps

Mercantus 2022. Näyttö parantaa tilojen käytettävyyttä. Viitattu 10.11.2022. https://mercantus.fi/tilavararaus-jarjestelma-timeworks-meeting/integraatiot/#hfaq-post-4000

Mercantus 2022. Tilavarausjärjestelmä – tilojen ja työpisteiden hallinta. Viitattu 9.11.2022. https://mercantus.fi/tilavararaus-jarjestelma-timeworks-meeting/

Mercantus 2022. We Make Your Business Run. Viitattu 1.11.2022. https://mercantus.fi/yritys-mercantus-oy/

Microsoft 2022. Code faster work Smarter. Viitattu 2.11.2022. https://visualstudio.microsoft.com/vs/

Microsoft 2022. Project and item templates. Viitattu 9.11.2022. https://learn.microsoft.com/en-us/visualstudio/ide/creating-project-and-item-templates?view=vs-2022

Microsoft 2022. What is .NET? Viitattu 1.11.2022. https://dotnet.microsoft.com/en-us/learn/dotnet/what-is-dotnet

Microsoft 2022. What is NuGet? Viitattu 2.11.2022. https://www.nuget.org/

Microsoft 2022. What is Xamarin? Viitattu 2.11.2022. https://learn.microsoft.com/en-us/xamarin/get-started/what-is-xamarin

Philips 2022. Signage Solutions Multi-Touch-näyttö. Viitattu 2.11.2022. https://www.philips.fi/c-p/10BDL3051T_00/signage-solutions-multi-touch-naeyttoe

Vuejs 2022. Watchers. Viitattu 7.11.2022. https://vuejs.org/guide/essentials/watchers.html

Kuviot

Taulukot