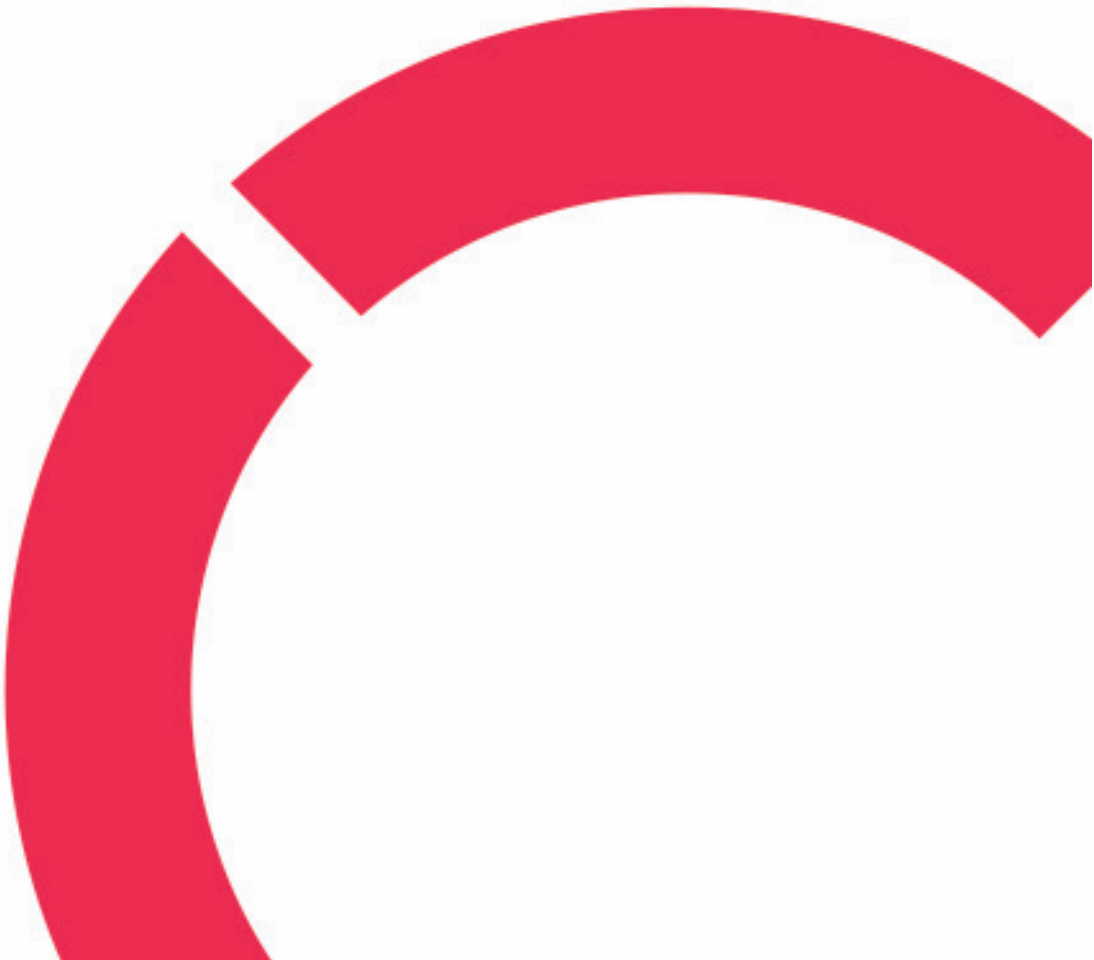
Tina Sonntag

DIGITALISAATION HYÖDYNTÄMINEN KANTA-ASIAKASOHJELMASSA

Leimakorttisovelluksen toteutus kauneusalan yritykselle

Opinnäytetyö
CENTRIA-AMMATTIKORKEAKOULU
Tieto- ja viestintätekniikan koulutus
Maaliskuu 2024



TIIVISTELMÄ OPINNÄYTETYÖSTÄ

Centria-ammattikorkeakoulu	**Aika** Maaliskuu 2024	**Tekijä/tekijät** Tina Sonntag
Koulutus Tieto- ja viestintätekniikka		☒ AMK ☐ YAMK
Työn nimi DIGITALISAATION HYÖDYNTÄMINEN KANTA-ASIAKASOHJELMASSA. Leimakorttisovelluksen toteutus kauneusalan yritykselle.		
Työn ohjaaja Anne Keskitalo		**Sivumäärä** 33 + 0
Työelämäohjaaja -		

Tämän opinnäytetyön tavoitteena oli tutustua mobiilisovellusten kehittämiseen sekä toteuttaa osioita kanta-asiakassovelluksesta kauneusalan yritykselle. Tarkoituksena oli kehittää konsepti digitaalisesta leimakorttisovelluksesta, joka on nykyaikaisempi ratkaisu fyysiseen paperiversioon verrattuna. Kanta-asiakasohjelmat ovat tärkeitä asiakasuskollisuudelle, ja digitaalisella leimakorttisovelluksella voidaan kasvattaa myyntiä ja asiakastyytyväisyyttä. Mobiilisovelluksen tuli olla yhteensopiva sekä Android- että iOS-käyttöjärjestelmien kanssa.

Opinnäytetyössä luotiin konsepti digitaalisesta kanta-asiakasohjelmasta ja kehitettiin osioita kanta-asiakassovellukseen Flutter-ohjelmalla Dart-ohjelmointikieltä käyttäen. Valitsin toteutukseen Flutterin, koska se tarjoaa mahdollisuuden luoda hybridisovelluksen, joka toimii sekä Androidilla että iOS:lla samaa koodipohjaa käyttäen.

Opinnäytetyöni tutkimuskysymyksinä toimivat “Miksi markkinoille tarvitaan digitaalista kanta-asiakassovellusta?” ja “Millainen kanta-asiakassovellus palvelisi parhaiten sen käyttäjäryhmää?”.

Opinnäytetyö on tyypiltään toiminnallinen, joten sen pääpaino oli kanta-asiakassovelluksen kehitysprosessissa ja mobiilisovelluksen osioiden luomisessa. Sovelluksen osioiden kehittämisessä otettiin huomioon toimeksiantajan tarpeet ja vaatimukset. Tulevaisuuden suunnitelmana on jatkaa sovelluksen kehittämistä tuomalla mukaan uusia toiminnallisuuksia ja hienosäätämällä käyttöliittymän ulkoasua entistäkin paremmaksi.

Asiasanat
Android, Dart, digitalisaatio, Flutter, iOS, kanta-asiakassovellus, sovelluskehitys

ABSTRACT

Centria University of Applied Sciences

Centria University of Applied Sciences	Date March 2024	Author Tina Sonntag
Degree programme Information technology		
Name of thesis UTILIZING DIGITALIZATION IN A LOYALTY PROGRAM. Developing a stamp card app for a beauty industry company.		
Centria supervisor Anne Keskitalo		**Pages** 33 + 0
Instructor representing commissioning institution or company -		

The aim of this thesis was to familiarize myself with mobile application development and to implement sections of a loyalty program application for a beauty industry company. The purpose was to develop a concept for a digital stamp card application, which is a more modern solution compared to the physical paper version. Loyalty programs are important for customer retention, and the use of a digital stamp card can increase sales and customer satisfaction. The mobile application was intended to be compatible with both the Android and iOS operating systems.

In the thesis a concept for a digital loyalty program was created and sections of the loyalty program application were developed using the Flutter framework with the Dart programming language. Flutter was chosen for implementation because it offers the possibility to create a hybrid application that works on both Android and iOS using the same codebase.

The research questions of the thesis were “Why is there a need for a digital loyalty program on the market?” and “What kind of a loyalty program application would best serve its user group?”.

The thesis was of a practical nature, so its main focus was on the development process of the loyalty program application and creating sections of the mobile application. As part of the development of the application sections, the needs and requirements of the client were taken into consideration. In the future, the application will be developed further by introducing new functionalities and improving its user interface.

Key words
Android, customer loyalty application, Dart, digitalization, Flutter, iOS, software development

KÄSITTEIDEN MÄÄRITTELY

ANDROID
Googlen kehittämä mobiilikäyttöjärjestelmä.

ANDROID STUDIO
Androidin virallinen kehitysympäristö.

AOT
(Ahead-of-time) on tehokas käännöstekniikka, joka muuntaa Dart-koodin halutulle laitteelle.

API
(Application Programming Interface) on ohjelmointirajapinta, joka mahdollistaa sovellusten kommunikoinnin keskenään.

BAAS
(Backend-as-a-Service) on pilvipalvelumalli, joka tarjoaa valmiita taustapalveluita ja infrastruktuurin osia sovellusten backend-toiminnallisuuksien helpottamiseksi ja nopeuttamiseksi.

AVOIN LÄHDEKOODI
(Open source englanniksi) on lähdekoodi, joka on julkisesti saatavilla sekä vapaasti muokattavissa, tarkasteltavissa ja uudelleenjaettavana.

DART
Googlen kehittämä ohjelmointikieli, jota käytetään Flutterissa.

EMULAATTORI
Ohjelmisto- tai laitteistopohjainen järjestelmä, joka matkii toista laitteistoa tai ohjelmistoympäristöä.

FLUTTER
Googlen kehittämä avoimen lähdekoodin ohjelmistokehityspaketti (SDK).

IOS

Applen kehittämä ja ylläpitämä käyttöjärjestelmä mobiililaitteille ja tableteille.

JIT

(Just-in-time) on käännöstekniikka, jossa ohjelman lähdekoodi käännetään konekieleksi ajon aikana.

NOSQL

(Not only SQL) on käsite, joka kuvaa ei-relaationaalisia tietokantoja.

OHJELMISTOKEHYS

(Framework) on ohjelmistotuote, joka toimii runkona sen päälle rakennettavalle tietokoneohjelmalle.

SDK

(Software Development Kit) on kokoelma ohjelmistonkehityksen työkaluja yhdessä paketissa, jolla voi kehittää sovelluksia.

SOVELLUS

Ohjelma, ohjelmien kokonaisuus tai peli, joka suorittaa tiettyjä toimintoja. Esimerkiksi Instagram ja WhatsApp.

SYNTAKSI

Lauseoppi, jota ohjelmointikieli noudattaa. Se määrittää ohjelmointikielen rakenteen ja miten ohjelmointikieltä tulee kirjoittaa.

VS CODE

(Visual Studio Code) on avoimen lähdekoodin tekstieditori, jota käytetään ohjelmoinnissa.

WIDGET

Widget, eli pienoisohjelma, joka näyttää laitteessa oleellista tietoa käynnistämättä niitä erikseen. Nämä pienoisohjelmat ovat esimerkiksi kalenteri, kello ja sääennuste.

KUVIOT

KUVAT

1 JOHDANTO

Mobiilisovellusten latausmäärä on kasvanut nopeasti maailmanlaajuisesti vuodesta 2016 lähtien ja ylitti 200 miljardia latausta vuonna 2019. Kuluttajat latasivat laitteisiinsa 255 miljardia mobiilisovellusta vuoden 2022 aikana, mikä merkitsi yli 80 prosentin kasvua vuoteen 2016 verrattuna. (Ceci 2023.) Nykypäivän liiketoimintamaailmassa kanta-asiakasohjelmat ovat olennainen osa asiakasuskollisuuden ylläpitämistä ja asiakassuhteiden kehittämistä. Näiden ohjelmien avulla yritykset voivat palkita ja sitouttaa asiakkaitaan, mikä puolestaan vaikuttaa myönteisesti niiden tulokseen ja kilpailukykyyn. Asiakkaiden tarpeiden ymmärtäminen on avainasemassa. (Black 2017.)

Kanta-asiakkuuteen liittyy yleensä olennaisesti leimakortti, joka oli tämän opinnäytetyön kehitettävä asia. Leimakorttisovelluksella voidaan lisätä yrityksen myyntiä sekä kasvattaa asiakastyytyväisyyttä (Volarević 2022). Työn tavoitteena on suunnitella konsepti kanta-asiakasohjelmalle sekä kehittää osioita kanta-asiakassovelluksen käyttöliittymään, josta myöhemmän jatkokehityksen jälkeen syntyisi käyttäjäystävällinen sovellus. Sovelluksen kehittämisessä otetaan huomioon sekä toimeksiantajan näkemykset ja toiveet että oma osaaminen ja aiempaan tutkittuun tietoon sovelluskehityksestä. Hyödynnän prosessissa puolistrukturoitua haastattelumenetelmää, jonka avulla saan syvällistä ymmärrystä toimeksiantajan tarpeista ja odotuksista.

Opinnäytetyössäni kehitän kanta-asiakasohjelmaa Flutter-nimisellä ohjelmalla, joka on Googlen kehittämä avoimen lähdekoodin ohjelmistokehityspaketti. Opinnäytetyöni rakenne koostuu kolmesta osasta: työn lähtökohdista, sovelluskehityksestä ja yhteenvedosta. Ensimmäisessä osassa tutkin digitaalisen kanta-asiakasohjelman sovelluksen tarvetta ja kysyntää: miksi yritysten tulisi panostaa tällaiseen sovellukseen, ja millaisia etuja se voi tarjota niin yrityksille kuin asiakkaillekin. Työn toisessa osassa keskityn itse sovelluskehitykseen ja tutustuin Flutterin käyttöön. Tarkoituksena on avata prosessia digitaalisen kanta-asiakasohjelman sovelluksen kehittämisestä. Tavoitteena on kuvata, miten sovelluksen osioita voidaan suunnitella ja rakentaa tehokkaasti, jotta se vastaisi kanta-asiakasohjelman tarpeita. Viimeisessä osassa on yhteenveto tuloksista ja kerron opinnäytetyön etenemisestä.

Työn tutkimuskysymyksinä toimivat “Miksi markkinoille tarvitaan digitaalista kanta-asiakassovellusta?” ja “Millainen kanta-asiakassovellus palvelisi parhaiten sen käyttäjäryhmää?”.

2 TOIMEKSIANTO

Toimeksiannon lähtökohtien selvittämistä varten käytin puolistrukturoitua haastattelua. Haastattelun tarkoituksena oli selvittää haastateltavan ajatuksia, käsityksiä, kokemuksia ja toiveita toimeksiannon suhteen. Lisäksi selvitin sovelluksen käyttötarpeiden määrityksen ja arkkitehtuurin suunnittelun.

Toimeksiantajani toimii Iina Tikkanen, It's Beauty.fi:n omistaja. Olen ollut Iinan asiakas jo vuoden verran. Eräänä kertana rupesimme puhumaan kanta-asiakaskorteista, ja niiden puutteesta. Iinaa harmitti, ettei Suomen markkinoilla ollut olemassa sähköistä kanta-asiakaskorttia. Tästä saimmekin idean lähteä kehittämään hänen yrityksensä ja asiakkaansa tarpeisiin sopivaa kanta-asiakasohjelman konseptia sekä luomaan digitaalisen leimakorttisovelluksen osioita.

Minulla oli jo ennen opinnäytetyön aloittamista jonkinlainen ajatus sovelluksesta, mutta halusin vielä syventää tietämystäni haastattelemalla Iinaa. Halusin pitää haastattelun rentona ja avoimena, mutta luonnostelin haastattelua varten kuitenkin runkokysymykset, jotka johdattelisivat keskustelun suuntaa. Iina on kauneusalan yrittäjä, kynsiteknikko ja tuleva kosmetologi. Yrityksen päätoimiala on kauneudenhoito. It's Beauty.fi:n tärkeimmät asiakkaat, eli yrityksen palveluja käyttävä kohderyhmä on pääasiassa kaikenikäiset naiset, ei kuitenkaan nuoriso.

Iinan toive sovelluksen suhteen oli toimiva ja käyttäjäystävällinen leimakorttisovellus, joka myötäilee yrityksen brändiä, kuten värit, fontit ja logo. Myös sosiaalisen median kuvakkeet ja linkit yrityksen sivustoille mainittiin. Toiveena oli myös sovellus, joka on soveltuva sekä Android- että iOS-käyttöjärjestelmille. Tulevan leimakortin tietoturva ja helppokäyttöisyys ovat myös tärkeitä. Iinan ajatuksena on palkita asiakkaat ilmaisella käynnillä joka kymmenes kerta, mikä oli asiakkaiden kovasti odotettu etuus. Tämä palkitseminen myös sitouttaa asiakkaat yritykseen. Asiakkaiden ensisijainen toive ja tarve kanta-asiakkuudesta ja leimakorttisovelluksesta oli saada käyttöönsä digitaalinen leimakortti, joka on helppokäyttöinen ja kulkee lähestulkoon aina mukana. Digitaalinen leimakortti on niin sanottu ekoteko sekä asiakkaiden että toimeksiantajan kannalta. (Tikkanen 2023.)

3 DIGITAALISEN KANTA-ASIAKASOHJELMAN MARKKINATUTKIMUS

Toimeksiantajan haastattelun pohjalta tein analyysia markkinatutkimusta varten. Analyysissä otettiin myös huomioon haastattelemani toimeksiantajan keskustelut hänen asiakkaidensa kanssa heidän tarpeistaan. Tämän pohjalta tunnistettiin tarve kehittää kanta-asiakasohjelmalle konsepti ja luoda digitaalisen kanta-asiakassovelluksen osioita. Toimeksiantajan sekä asiakkaiden toiveena oli nykyaikaistaa kauneusalan yrityksen kanta-asiakaskorttia digitaaliseen muotoon.

Maailmassa tunnetuin kanta-asiakasohjelman malli on fyysinen leimakortti, jossa kanta-asiakas kerää leimoja ostoksistaan. Täydestä leimakortista kanta-asiakas voi lunastaa palkinnon, joka yleensä on ilmainen tuote tai palvelu. Digitaaliset leimakortit ovat yleistyneet viime vuosikymmenellä, koska ne tarjoavat asiakkaille kätevyyttä ja asiakastietojen seurantamahdollisuuden toimijoille. Mobiililleimakortteja on kolmea eri tyyppiä: perus, kompleksinen sekä moninainen. Perustyypissä on yksi toiminnallisuus, kuten leimakortti. Kompleksisessa tyypissä on usein kaksi toiminnallisuutta, kuten leimakortin lisäksi ostohistoria. Moninaisessa tyypissä on edellisten esimerkkien lisäksi useita eri toimintoja, tuotteita sekä näiden lisäksi jopa yhteistyökumppaneiden etuja tai tarjouksia. (Pay 2022.)

Mobiili kanta-asiakaskortti kulkee lähestulkoon aina mukana ja sen lisäksi se sitouttaa asiakkaat palveluun entistä paremmin. Asiakkaat ja yritykset hyötyvät digitaalisista leimakorttisovelluksista. Asiakkaan kannalta suurin hyöty on, että fyysistä korttia ei tarvitse kantaa mukana, koska korttisovellus on asiakkaan puhelimessa. Yritykset voivat hyödyntää sovellusta monella tapaa, kuten lisäämällä yhteystiedot ja aukioloajat, sekä tarjoamalla alennuksia. Lisäksi fyysisten korttien tulostamisesta ei tarvitse huolehtia, kun tiedot löytyvät digitaalisena. Digitaalisen sovelluksen käyttö kanta-asiakasohjelmassa vahvistaa asiakasuskollisuutta. (Ziemianek 2020.)

4 TEKNOLOGIAT & TYÖKALUT

Flutter on moderneista mobiilisovelluskehityskehyksistä yksi monipuolisimmista ja suositummista vaihtoehdoista, sillä se tarjoaa mahdollisuuden kehittää mobiilisovelluksia iOS- ja Android-alustoille yhdellä ja samalla koodipohjalla (Neshkoska 2023). Seuraavissa alaluvuissa paneudutaan tarkemmin siihen, mikä Flutter on, sen taustoihin ja siihen, miksi se on noussut suosioon mobiilikehityksessä.

Dart-ohjelmointikieli on olennainen osa Flutter ohjelmistokehitystä. Se on Googlen omistama ja ylläpitämä kieli. Dart on luotettava ohjelmointikieli, jonka oppiminen on vaivatonta riippumatta siitä, onko dynaaminen vai staattinen kieli entuudestaan tuttu. (Windmill 2020a, 4.) Dart-kieli on avainasemassa Flutter-sovellusten tehokkaassa kehityksessä.

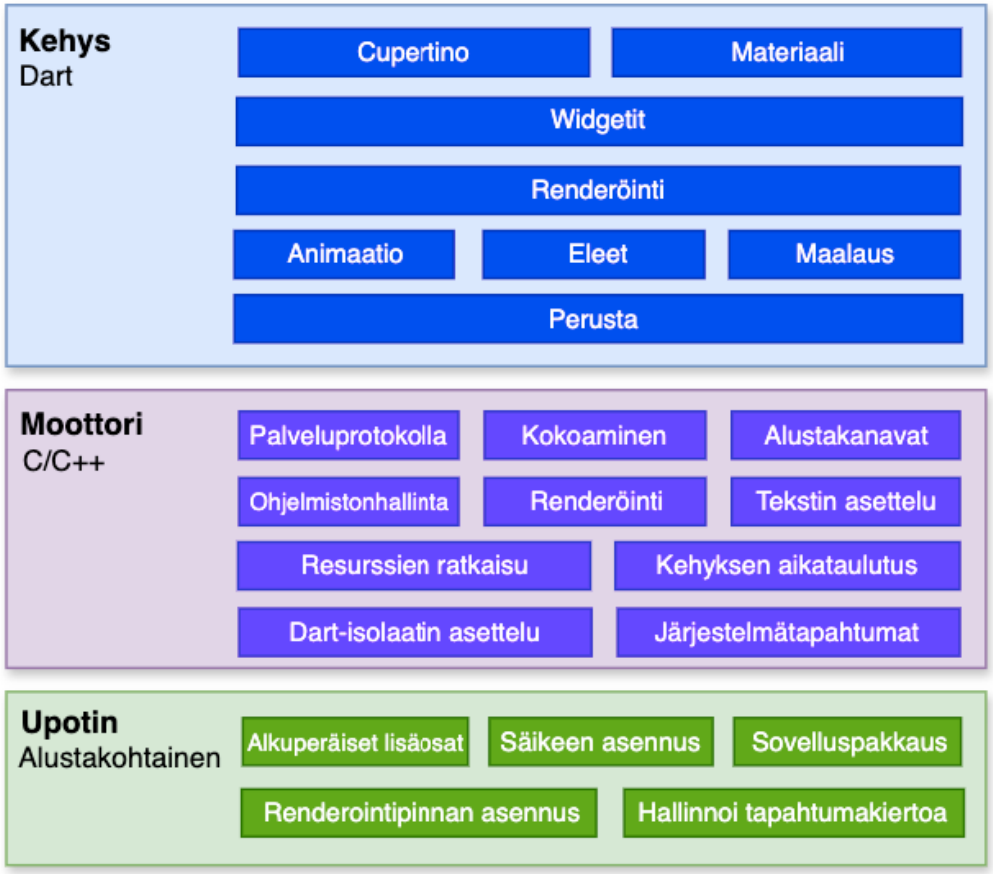
Käsittelen myös erilaisia widgettejä, jotka ovat Flutterin rakennuspalikoita sovellusten käyttöliittymän luomisessa. Widgetit ovat keskeinen osa Flutteria, ja ne tarjoavat monipuolisia komponentteja sovellusten graafiseen suunnitteluun sekä käyttökokemuksen luomiseen (Windmill 2020b, 57). Lopuksi käyn läpi mikä Firebase on ja mihin sitä tarvitaan Flutter-kehityksessä.

4.1 Flutter

Flutter on nouseva teknologia, jolla voi luoda sekä iOS- että Android-käyttöjärjestelmille tarkoitettuja mobiilisovelluksia yhtä koodipohjaa käyttäen. Kehitysympäristön ohjelmointikielenä on Dart, joka on tehokas ja nopea jokaisella käyttöalustalla. (Flutter.) Flutter on avoimen lähdekoodin ohjelmistokehityspaketti (SDK), jonka Google on kehittänyt. Sen avulla on mahdollista rakentaa mobiili-, web- ja työpöytäsovelluksia yhdestä koodipohjasta. Flutter julkaistiin virallisesti avoimen lähdekoodin projektina toukokuussa 2017, ja se saavutti ensimmäisen vakaan version joulukuussa 2018. Flutter oli alun perin suunniteltu kohdistumaan iOS- ja Android-laitteille vaihtoehtona olemassa oleville monialustakehityskehyksille. (Bartosińska & Dembny 2023.)

Flutter on vuonna 2022 tehdyn kehittäjäkyselyn mukaan maailmanlaajuisesti suosituin monialustainen mobiilikehitystyökalu. Peräti 46 prosenttia ohjelmistokehittäjistä käyttää Flutteria ja kolmannes mobiilikehittäjistä käyttää monialustaisia teknologioita tai kehyksiä. (Vailshery 2023.)

Flutter on laajennettava ja kerrostettu järjestelmä, joka muodostuu itsenäisistä kirjastoista (KUVIO 1). Flutter-sovellukset pakataan samalla tavalla kuin muutkin natiivisovellukset. Ne hyödyntävät alustakohtaista liitäntäkerrosta, joka toimii yhteistyössä käyttöjärjestelmän kanssa tarjoten pääsyn palveluihin, kuten renderöintipintoihin, saavutettavuuteen ja syötteisiin. Flutter-moottori on pääosin toteutettu C++ -kielellä ja se tarjoaa matalan tason toteutuksen Flutterin ydinpohjaisesta ohjelmointirajapinnasta. Tämä sisältää muun muassa grafiikan, tekstin asettelun ja saavutettavuustuen sekä Dart-käännöstyökalut ja ajonaikaympäristön. Sovelluskehittäjät käyttävät Flutter-kehystä kommunikoidakseen Flutterin kanssa. Tämä ohjelmistokehys tarjoaa moderneja ja reaktiivisia työkaluja, jotka on kirjoitettu Dart-ohjelmointikielellä. (Flutter docs a.)



KUVIO 1. Arkkitehtuurin kerrokset Flutterissa (mukaillen Flutter docs a)

Flutter käyttää Hot-reload-toimintoa, jolla voi rakentaa käyttöliittymiä, lisätä ominaisuuksia sekä korjata ohjelmointivirheitä. Tämä toiminto mahdollistaa muutosten tarkastelun säilyttäen sovelluksen tilan. (Flutter docs b.)

4.2 Dart-ohjelmointikieli

Dart on Googlen kehittämä oliopohjainen avoimen lähdekoodin ohjelmointikieli. Se on asiakaspohjainen ohjelmointikieli, joka tarjoaa useita työkaluja sovellusten kehittämiseen. Sitä käytetään erityisesti nopean suorituskykyisten sovellusten tekemiseen eri laitteille, ja se tarjoaa selkeän syntaksin sekä laajan sisäänrakennetun kirjaston. Dart-kielen koodia voi kääntää alkuperäiseksi koodiksi, jolloin se toimii suoraan tietokone- ja mobiilialustoilla ilman erillistä siltaa, toisin kuin React Native, joka on monialustaisen sovelluskehityksen viitekehys. Tämä tekee siitä tehokkaan työkalun monialustaiseen sovelluskehitykseen. (Sahu 2022.) Dart-kielellä on kyky kääntää koodi nopeasti, sillä se tukee sekä JIT (just-in-time) että AOT (ahead-of-time) –käännöstekniikoita (Dart).

Dart-ohjelmointikielen virallinen ensimmäinen versio julkaistiin vuonna 2013, jolloin vain Googlen oma henkilöstö oli innokas käyttämään sitä. Viisi vuotta myöhemmin Google uudisti kielen versiolla Dart 2.0, joka osoitti sen sitoutumistaan kyseiseen kieleen. Nykyään Dart on yhä suositumpi ohjelmointikieli kehittäjien keskuudessa etenkin niillä käyttäjillä, jotka ovat tottuneet käyttämään Javaa ja C# -kieliä. Dart-ohjelmointikielellä on monia vahvuuksia, koska siinä yhdistyvät kahden maailman parhaat puolet: se on tyyppiturvallinen ja käännettävä ohjelmointikieli, kuten Java ja C# sekä myös komentokieli, kuten Python ja JavaScript. Lisäksi se toimii monessa eri ympäristössä ja koodi voidaan muuntaa natiiviksi mobiilisovellukseksi, mikä tekee siitä monikäyttöisen. Myös Dart-kielen oppiminen on nopeaa, sillä sen syntaksi on samantyyppinen kuin Javassa ja C-ohjelmointikielissä. (Ford 2019.)

4.3 Widgets

Widgetit ja niiden asettelu muodostavat Flutterin ytimen. Flutter-sovellus rakentuu erilaisista widgeteistä jotka voivat olla näkyviä, kuten kuvakkeita ja tekstiä, mutta myös näkymättömiä elementtejä, kuten ruudukoita ja sarakkeita. (Flutter docs c.) Flutter käyttää widgettejä käyttöliittymän luomiseen, jotka kuvaavat miltä näkymä näyttää niiden nykyisen tilan ja kokoonpanon perusteella. Kehys käyttää modernia rakennetta, joka on saanut vaikutteita React-kirjastosta. Kun widgetin tila muuttuu, se rakentaa uudelleen kuvauksensa. Tämän jälkeen kehys vertailee sitä edelliseen kuvauskertaan määrittääkseen tarvittavat muutokset renderipuussa siirtyäkseen yhdestä tilasta seuraavaan. (Flutter docs d.)

Widgettejä on olemassa kahta eri tyyppiä: stateful ja stateless. Stateful, eli tilallinen widget, voi muuttua sovelluksen käytön aikana. Nämä widgetit voivat muuttaa ulkonäköään dynaamisesti, silloin

kun ne vastaanottavat dataa tai niitä käynnistetään käyttäjän toimesta. Stateless widgetissä mikään ei voi muuttua, jolloin sitä kutsutaan tilattomaksi widgetiksi. (Flutter docs e.) Nämä widgetit eivät tallenna dataa, eli ne ovat luonteeltaan staattisia (Tillu 2019).

4.4 Tilanhallinta

Flutter koostuu widgeteistä, jotka luodaan sovelluksen tilan perusteella. Flutterin state, eli tila, tarkoittaa tietoa, jota käytetään käyttöliittymän luomisessa. (Tillu 2022.) Flutterin tila voidaan jakaa kahteen kategoriaan, jotka ovat hetkellinen tila ja sovelluksen tila. Hetkellinen tila tunnetaan myös nimellä käyttöliittymän tila tai paikallinen tila, jolla voi järjestää tiettyjä widgettejä. Tämä tilanhallintatekniikka ei ole paikallisesti suosittu. Sovelluksen tila on kompleksisempi. Se tunnetaan myös jaettuna tilana. Tämä tila säilyy, koska se jaetaan useiden sovellusten osien kesken ja sen tarkoitus on säilyttää tiedot käyttäjien istuntojen välillä. (Flutter docs f.)

KUVIO 2. Flutterin tilanhallintakaava (mukaillen Flutter docs f)

Tilanhallinta tarkoittaa käyttöliittymän datan muutosta. Muutokset voivat johtua käyttäjän syötteestä tai taustaprosessista, jotka käsittelevät taustapalvelun tietoja. Tilan tarkoituksena on varmistaa, että sovelluksen tiedot ja käyttöliittymä ovat aina yhdenmukaiset ja estävät epäjohdonmukaisuudet, kun data muuttuu. (Kodeco 2023.) Flutter käyttää deklaratiivista ohjelmointitapaa, joka tarjoaa monia etuja (KUVIO 2). Käyttäjä voi valita deklaratiivisen viitekehyksen, jolloin tilanhallinta on kehittäjän vastuulla. Jotta sovellus vastaa ajantasaista tilaa, täytyy kehittäjän manuaalisesti käynnistää käyttöliittymän päivitys. (Flutter docs g.)

4.5 Firebase

Firebase on Googlen kehittämä alusta, joka tarjoaa monipuolisia pilvipalveluja. Se toimii Backend-as-a-Service (BaaS) -ratkaisuna ja sisältää isännöityjä palveluita kuten reaaliaikaisen tietokannan, käyttäjän tunnistautumisen, isännöinnin staattisille tiedostoille ja pilvitallennustilaa. (Flutter docs h.) Firebase Realtime Database on osa Googlen Firebase-tuoteperhettä, joka on pilvessä sijaitseva NoSQL-tietokanta. Tämä tietokanta tarjoaa kehittäjille tehokkaan ja joustavan ratkaisun reaaliaikaisten sovellusten luomiseen sekä ratkaisun reaaliaikaisen datan hallintaan pilvessä, joka on olennainen osa mobiilisovelluskehitystä. (Khan 2023.)

Firebase Realtime Database –tietokanta tallentaa datan JSON-muodossa ja synkronoi sen reaaliajassa jokaisen siihen liitetyn asiakaslaitteen kanssa. Päivitetty data on myös saatavilla sovelluksen offline-tilassa. Reaaliaikainen tietokanta jaetaan kaikkien asiakaslaitteiden kesken, kun luodaan monialustaisia sovelluksia käyttäen SDK:ta Apple-alustoille, Androidille ja JavaScriptille. (Firebase.)

Firebase tarjoaa reaktiivisen NoSQL-tietokannan nimeltä Firestore, mikä muodostaa erinomaisen yhteensopivuuden Flutter-sovelluksille. Firestore toimii tietokantapalveluna, joka mahdollistaa tehokkaan viestinnän sovelluksen ja Firestore-tietojen välillä. (Windmill 2020c, 281.)

5 SOVELLUKSEN KEHITYSPROSESSI

Lähtökohtien selvittämisen ja markkina-analyysin laatimisen jälkeen suunnittelin sovelluksen toteutustapaa. Halusin valita sovelluksen ohjelmistokehitystyökalukseni mahdollisimman helppokäyttöisen, helposti omaksuttavan sekä yhdellä koodipohjalla toimivan työkalun. Edellisessä luvussa käsitelty Flutter sopi näihin kriteereihin mainiosti. Olen myös opiskelujeni kautta saanut vahvistusta Flutterille, sillä opiskelupiireissä niin opettajat kuin opiskelukollegat ovat kehuneet Flutteria markkinoiden parhaaksi työkaluksi.

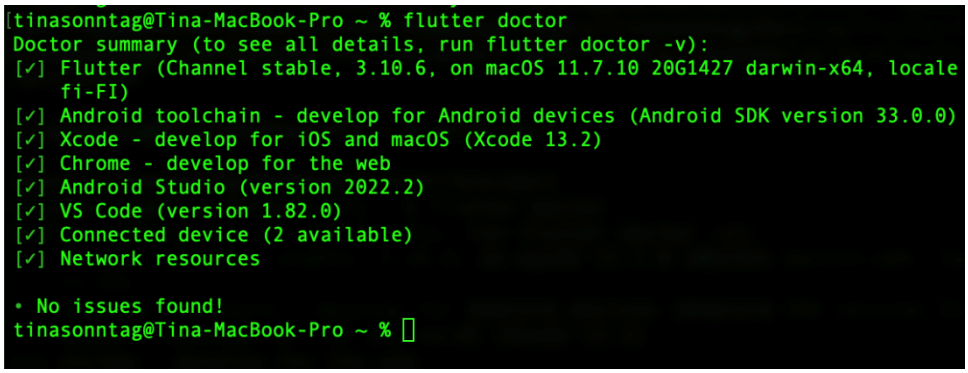
Tarkoituksenani oli kehittää mobiilisovellukseen osioita kuten leimakortti, asiakasprofiili sekä mahdollisesti sisällyttää siihen myös yrityksen yhteystiedot. Sovelluksessa on myös asiakaskohtainen QR-koodi, jonka avulla toimeksiantajani, eli tässä tapauksessa kauneusalan yrittäjä, voi skannata asiakkaan tiedot sekä lisätä leimoja leimakorttiin.

5.1 Kehitysympäristön rakentaminen

Sovelluskehityksen ensimmäinen vaihe oli kehitysympäristön lataaminen omalle tietokoneelle sekä siihen tutustuminen, sillä ohjelma oli minulle entuudestaan tuntematon. Kehitysympäristön rakentamiseen käytin Flutterin dokumentaatiosivuston englanninkielistä ohjeistusta. Flutterin asentaminen omalle tietokoneelle vaati useita vaiheita ja ensimmäiseksi tuli varmistaa tarvittavat järjestelmävaatimukset.

Kehitysympäristön rakentaminen alkoi Flutter-ohjelmistokehityspaketin lataamisella ja asentamisella. Tämän jälkeen lisäsin PATH-ympäristömuuttujan pääte-emulaattorilla, joka määrittää Flutterin hakemistopolun. Flutterin kehittäjätiimi suosittelee käyttämään kehitysympäristöä tai tekstieditoria, joka on varustettu Flutterin laajennuksella tai liitännäisellä, kuten Visual Studio Code ja Android Studio. Näitä lisäosia voidaan käyttää hyödyksi tarjoamalla automaattista koodin täydennystä, korostamalla koodin syntaksia, tukemalla widgetien muokkausta ja tarjoamalla toiminnallisuutta sovellusten suorittamiseen sekä vianetsintään (Flutter docs i.) Android Studio sekä Visual Studio Code löytyivät jo omalta tietokoneeltani ja hyödynsin molempia kehitystyössäni. Lisäksi alkuvaiheeseen kuului koodaamisen testausta, jotta sain kokemusta Flutterilla ohjelmoinnista sekä miten Dart syntaksi toimii.

Pääte-emulaattorilla voidaan tarkistaa flutter doctor –komennolla, että Flutter-kehys on asennettu ja konfiguroitu oikein. Samalla se tunnistaa mahdolliset ongelmat tai puuttuvat työkalut ja riippuvuudet. Flutter-ohjelmistokehys on asennettu onnistuneesti omalle tietokoneelleni (KUVA 1).

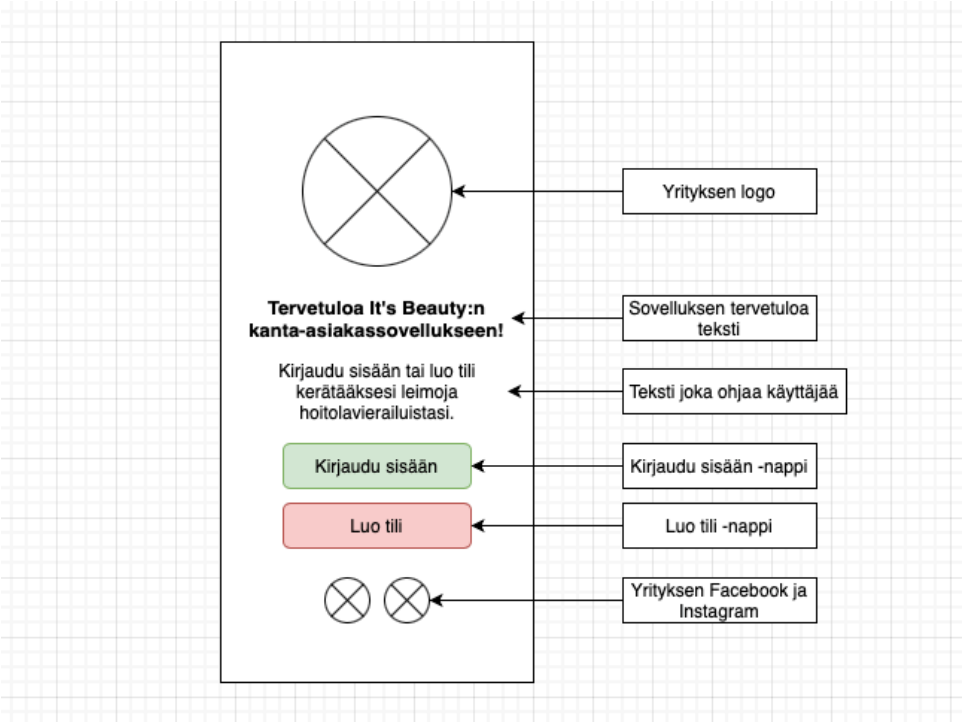


KUVA 1. Flutter ohjelmistokehyksen asennus terminaalissa

Tämän jälkeen suunnittelin, minkälainen leimakorttisovelluksesta tulee. Testaamisen ohella ohjelmoin samalla sovelluksen. Ensimmäisen version valmistuttua esitin sovelluksen prototyypin toimeksi -antajalle, jonka jälkeen toimeksiantajalla oli mahdollisuus esittää mahdollisia korjausehdotuksia ja muutoksia sovellukseen. Toimeksiantajalla ei ollut tässä vaiheessa korjausehdotuksia ensimmäiseen versioon, joten jatkoin sovelluksen kehittämistä eteenpäin. Testasin myös sovellusta asiakkaan näkökulmasta jokaisessa kehitysvaiheessa, sillä olen toimeksiantajan kanta-asiakas.

5.2 Suunnittelu

Sovelluksen käyttötarkoituksena on kerätä leimoja hoitolavierailuista. Palveluntuottaja antaa asiakkaalle leiman jokaisesta käynnistä skannaamalla asiakkaan QR-koodin sovelluksesta. Jotta leimat kohdistuisivat oikealle henkilölle, on jokaisen käyttäjän luotava oma tili saadakseen henkilökohtaisen QR-koodin. Sovelluksen etusivulla on mahdollista navigoida kirjautumissivulle kirjaudu sisään -napista ja rekisteröitymissivulle luo tili -napista (KUVA 2). Etusivulta löytyy myös yrityksen Facebook- ja Instagram-linkit.



KUVA 2. Rautalankamalli sovelluksen etusivusta

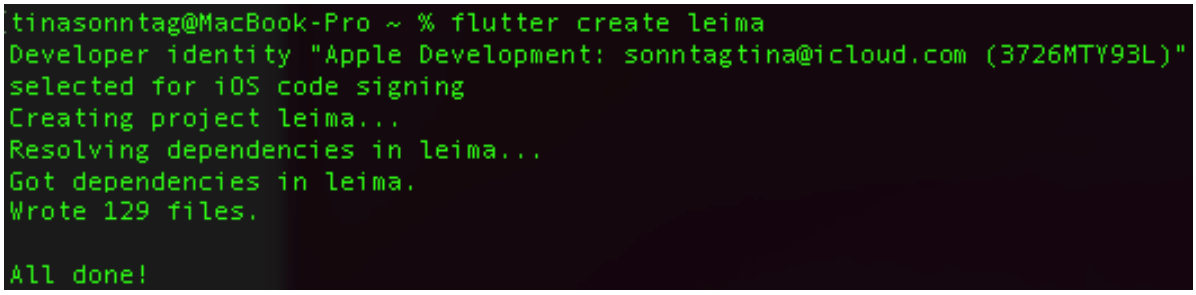
Projektin suunnittelussa ja toteutuksessa käytin yrityksen brändin mukaista typografiaa ja värimaailmaa. Brändin mukaisia fontteja en löytänyt Googlesta, joten käytin typografiaa, joka muistuttaa eniten alkuperäistä fonttia. Ensisijaisena typografiana toimii Googlen fontti Montserrat sekä vaihtoehtoisena typografiana Inter (KUVA 3).

KUVA 3. Sovelluksessa käytettävät typografiat Montserrat ja Inter

Suunnittelutyökaluna käytin FlutterFlow -ohjelmaa, jotta pystyin paremmin havainnollistamaan sovelluksen ulkoasua. FlutterFlow tarjoaa kätevän ratkaisun sovelluksen visuaalisen käyttöliittymän suunnitteluun (FlutterFlow docs). Ohjelman avulla sain paremman käsityksen siitä, miten käytän yrityksen brändin mukaisia värejä sekä hahmotin, miten widgetit asettuvat sovellukseen. FlutterFlow:n avulla pääsin tarkemmin tutustumaan widgettien koodaamiseen ja sen oikeanlaiseen rakenteeseen, joka oli minulle uusi asia.

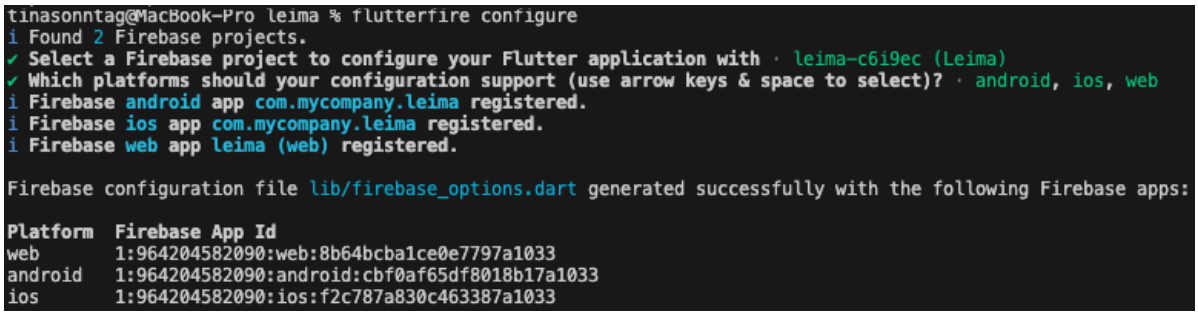
6 KEHITYSTYÖ

Sovelluksen kehittäminen käynnistyi uuden projektin luomisella. Flutter -projektin aloittaminen tapahtui komennolla “flutter create [sovelluksen nimi]”. Nimesin oman projektini nimellä “leima” (KUVA 4). Flutter luo automaattisesti valmiin demoprojektin, jossa on yksinkertainen laskuri napin painallusten laskemiseksi. Tämän demoprojektin käynnistäminen antaa välittömän varmuuden siitä, että käytettävät työkalut toimivat oikein. Demoprojektin välitön käynnistäminen mahdollistaa myös Flutterin Hot Reload -toiminnon hyödyntämisen, joka päivittää muutokset koodiin välittömästi emulaattorin näytölle, kun kehittäjä tallentaa muutokset.



KUVA 4. Oman projektin luominen terminaalissa

Firebase-tietokantapalvelu on myös konfiguroitu projektiin web-, android- ja iOS-alustoille (KUVA 5). Kirjautuminen ja rekisteröityminen sovellukseen tapahtuu käyttäen sähköpostiosoitetta ja salasanaa. Tämä oli mielestäni helpoin ja turvallisin kirjautumistapa, sillä Firebase varmistaa, että annetut tiedot vastaavat tietokannassa olevan käyttäjän tietoja. Muita kirjautumisvaihtoehtoja Firebase-palveluun olisi ollut esimerkiksi kirjautuminen sosiaalisen median kautta, mikä ei ollut minusta niin yksinkertainen ja käytännöllinen vaihtoehto.



KUVA 5. Firebase-tietokantapalvelun lisäys

6.1 Etusivun luominen

Aloitin etusivun luomisen lisäämällä siihen kaksi painiketta, tekstiruudut ja kuvakkeen. Etusivun kuvana toimii yrityksen logo (KUVA 6). Ensimmäisessä painikkeessa on teksti “Kirjaudu sisään”, joka vie kirjautumissivulle. Toisessa painikkeessa on teksti “Luo tili”, joka taas ohjaa rekisteröitymissivustolle. Lisäsin sivuun myös toimeksiantajan toivomat Facebook- ja Instagram-linkit.

```
Align(
  alignment: const AlignmentDirectional(0, 0),
  child: ClipRRect(
    borderRadius: BorderRadius.circular(8),
    child: Image.asset(
      'assets/images/its.png',
      width: 290,
      height: 200,
      fit: BoxFit.cover,
      alignment: const Alignment(0, 0),
    ), // Image.asset
  ), // ClipRRect
), // Align
```

KUVA 6. Etusivun logon koodi

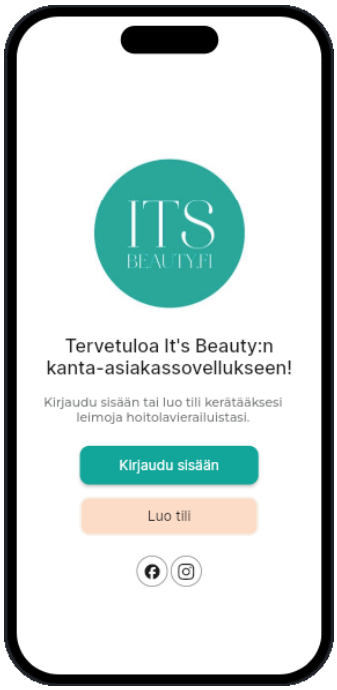
Tervetuloviesti näkyy ensimmäisenä heti logon alapuolella, ja siinä toivotetaan It’s Beautyn asiakkaat tervetulleiksi kanta-asiakassovellukseen. Seuraavana näkyy toinen tekstielementti tervetuloviestin alla, ja siinä pyydetään käyttäjää kirjautumaan sisään tai luomaan tili valitsemalla alla olevista painikkeista. Valitsin tämän asettelun, koska se vaikutti minusta selkeältä ja se sopii hyvin yrityksen brändiin.

Seuraava koodi kuvaa etusivun käyttöliittymän ulkoasua, jossa on kaksi tekstielementtiä, tervetuloviesti ja kirjautumistieto, järjestettyinä sarakkeeseen tiettyjen tyylimäärittelyjen ja tilan asetusten kanssa (KUVA 7). “Padding” lisää tyhjää tilaa sen lapsielementin ympärille ja “Column” järjestää lapsielementit pystysuuntaisesti maksimikokoiseksi.

```
Padding(
  padding: const EdgeInsetsDirectional.fromSTEB(
      16.0, 0.0, 16.0, 0.0), // EdgeInsetsDirectional.fromSTEB
  child: Column(
    mainAxisSize: MainAxisSize.max,
    crossAxisAlignment: CrossAxisAlignment.start,
    children: [
      Align(
        alignment: const AlignmentDirectional(0.0, 0.0),
        child: Text(
          '\nTervetuloa It\'s Beauty:n \nkanta-asiakassovellukseen!',
          textAlign: TextAlign.center,
          style: FlutterFlowTheme.of(context)
              .headlineMedium
              .override(
                fontFamily: 'Inter',
                color:
                    FlutterFlowTheme.of(context).primaryText,
                fontSize: 24.0,
                fontWeight: FontWeight.w500,
                useGoogleFonts:
                    GoogleFonts.asMap().containsKey('Inter'),
              ),
        ), // Text
      ), // Align
      Align(
        alignment: const AlignmentDirectional(0.0, 0.0),
        child: Padding(
          padding: const EdgeInsetsDirectional.fromSTEB(0.0, 0.0, 16.0, 0.0),
          child: Text(
            '\nKirjaudu sisään tai luo tili kerätääksesi leimoja hoitolavierailuistasi.\n',
            textAlign: TextAlign.center,
            style: FlutterFlowTheme.of(context)
                .bodyMedium
                .override(
                  fontFamily: 'Montserrat',
                  color: FlutterFlowTheme.of(context)
                      .secondaryText,
                  fontSize: 16.0,
                  fontWeight: FontWeight.w500,
                  useGoogleFonts: GoogleFonts.asMap()
                      .containsKey('Montserrat'),
                ),
          ), // Text
        ), // Padding
      ), // Align
```

KUVA 7. Etusivun tekstielementtien koodi

Kanta-asiakassovelluksen etusivun ensivaikutelma on ratkaiseva, sillä se heijastaa yrityksen brändiä. Yrityksen logo, värit ja typografia sekä sivun helppokäyttöisyys muodostavat yhtenäisen ja houkuttelevan kokonaisuuden, joka vahvistaa brändikokemusta. Huolellisesti suunnitellut visuaaliset elementit eivät ainoastaan houkuttele käyttäjiä, vaan myös välittävät yrityksen persoonallisuutta. Helppokäyttöinen sivusto parantaa käyttäjäkokemusta, mikä puolestaan lisää käyttäjien sitoutumista yritykseen ja sen tarjoamiin palveluihin. Kanta-asiakassovelluksen etusivun lopullinen ulkoasu on tärkeä osa tätä kokonaisuutta (KUVA 8).

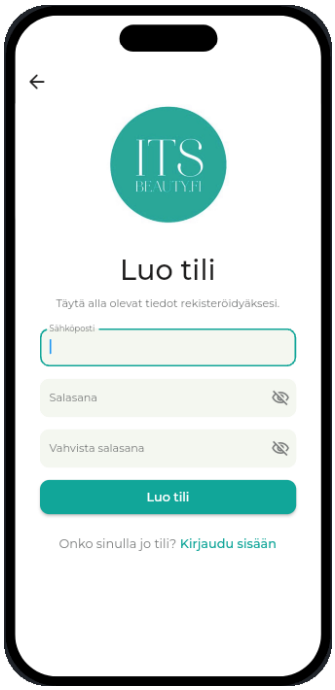


KUVA 8. Kanta-asiakassovelluksen etusivu toiminnallisuuksilla

6.2 Kirjautumis- ja rekisteröitymissivun luominen

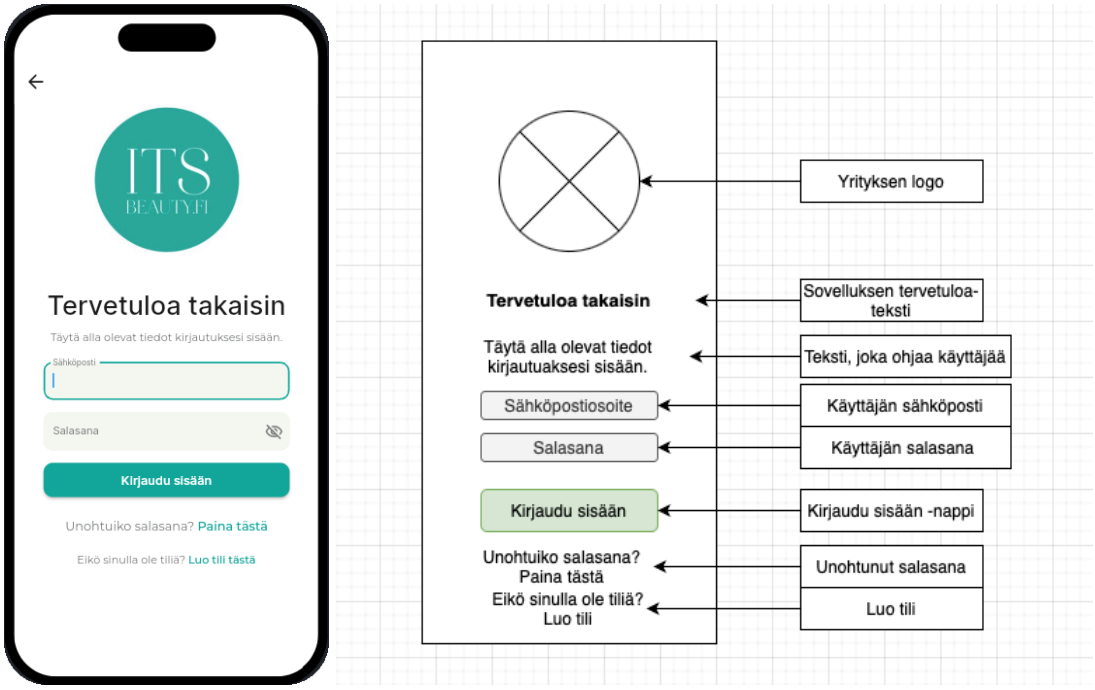
Etusivulta on mahdollista siirtyä kirjautumissivulle tai rekisteröitymissivulle. Näiden sivujen välillä voi myös navigoida, jos käyttäjä on valinnut väärän sivun. Käyttäjän rekisteröitymiseen tarvitaan toimiva sähköpostiosoite ja salasana, joka on vahvistettava syöttämällä salasana kaksi kertaa. Kirjautumis- ja rekisteröitymissivuilla on myös nuoli, joka vie takaisin etusivulle (KUVA 9).

Kirjautumissivulta löytyvät yrityksen logo, tekstikentät, “Kirjaudu sisään” -painike sekä linkit sivuille “Unohtunut salasana” ja “Luo tili”. Jos käyttäjä unohtaa salasanansa, hän voi tilata uuden painamalla “Unohtunut salasana” -kohtaa. Testasin tätä itse ja sain sähköpostiini salasanan palautusviestin. Kirjautumisen jälkeen käyttäjä ohjautuu kotisivulle.



KUVA 9. Rekisteröitymissivun ulkoasu

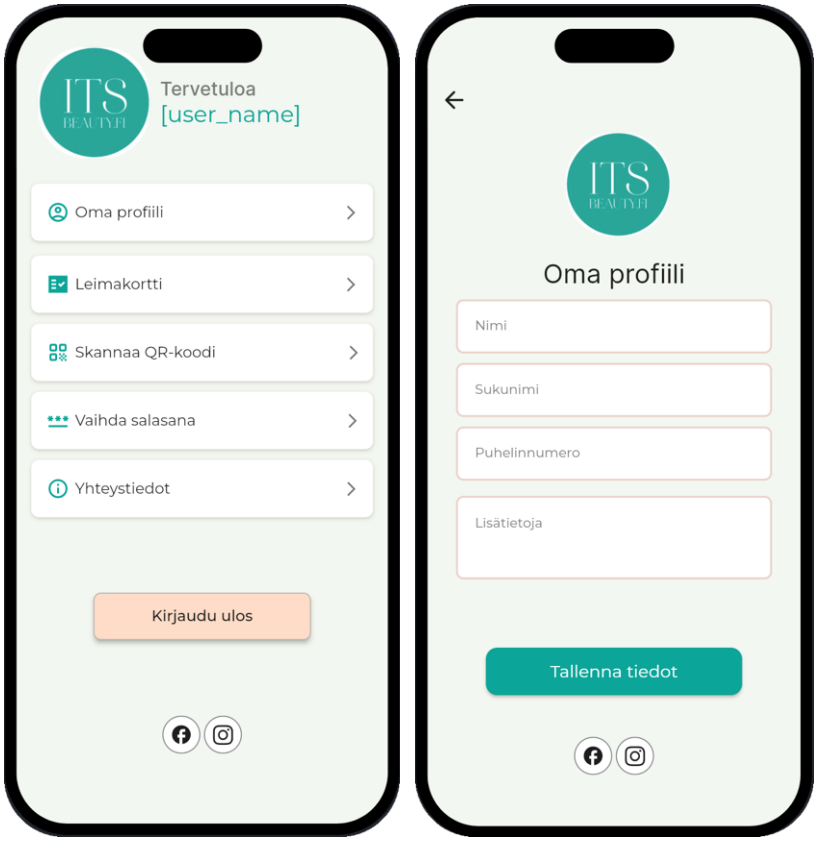
Tässä on kirjautumissivun lopullinen ulkoasu sekä sen rautalankamalli (KUVA 10). Alussa tekemäni rautalankamalli osoittautui lopulta erittäin hyödylliseksi työvaiheeksi, sillä sen avulla pystyin hahmottamaan ja rakentamaan helpommin kanta-asiakassovelluksen käyttöliittymää.



KUVA 10. Kirjautumissivun ulkoasu ja rautalankamalli

6.3 Kotisivu ja oma profiili

Sisäänkirjautumisen jälkeen käyttäjä siirtyy sovelluksen kotisivulle, josta löytyy valikko eri osioihin: oma profiili, leimakortti, QR-koodi, salasanan vaihto ja yhteystiedot (KUVA 11). Kotisivu toivottaa käyttäjän tervetulleeksi henkilökohtaisella tervehdyksellä “Tervetuloa [user_name]”. Käyttäjän syöttäessä nimensä omaan profiiliinsa, tervehdyksen teksti “[user_name]” vaihtuu omaan nimeen. Omaan profiiliin käyttäjä voi syöttää etu- ja sukunimensä, puhelinnumeronsa sekä mahdollisia lisätietoja. Nämä tiedot tallennetaan profiiliin painamalla “Tallenna tiedot” -painiketta. Kotisivulta käyttäjä voi myös halutessaan kirjautua ulos sovelluksesta. Molemmille sivuille on lisätty linkit toimeksiantajan Facebook- ja Instagram-profiileihin.



KUVA 11. Kotisivu ja oma profiili

Tämä ohjelmakoodi liittyy käyttäjätietojen päivittämiseen ja navigointiin sovelluksessa. Kun käyttäjä painaa “Tallenna tiedot” -painiketta, hänen tietonsa päivitetään ja tallennetaan, minkä jälkeen hän siirtyy takaisin kotisivulle (KUVA 12).

```
Align(
  alignment: const AlignmentDirectional(0.0, 0.05),
  child: Padding(
    padding: const EdgeInsetsDirectional.fromSTEB(
        0.0, 6.0, 0.0, 0.0), // EdgeInsetsDirectional.fromSTEB
    child: FFButtonWidget(
      onPressed: () async {
        await currentUserReference!
            .update(createUsersRecordData(
          userName: _model.nameController.text,
          phoneNumber: _model.phoneNumberController.text,
          bio: _model.bioController.text,
          surname: valueOrDefault(
              currentUserDocument?.surname, ''),
          email: '',
        ));

        context.pushNamed('HomePage');
      },
      text: 'Tallenna tiedot',
```

KUVA 12. Tallenna tiedot -painikkeen toiminnallisuus

Seuraava ohjelmakoodi luo tekstinäkymän, joka näyttää käyttäjän nimen etusivulla kohdassa “[user_name]”, mikäli käyttäjä on syöttänyt etunimensä omassa profiilissa (KUVA 13).

```
), // Text
if (widget.firstName != '')
  AuthUserStreamWidget(
    builder: (context) => Text(
      valueOrDefault(
          currentUserDocument?.userName, ''),
      style: FlutterFlowTheme.of(context)
          .headlineSmall
          .override(
            fontFamily: 'Montserrat',
            color: FlutterFlowTheme.of(context)
                .primary,
            fontWeight: FontWeight.normal,
            useGoogleFonts: GoogleFonts.asMap()
                .containsKey('Montserrat'),
          ),
    ), // Text
  ), // AuthUserStreamWidget
```

KUVA 13. If-ehtolause, joka näyttää käyttäjän nimen etusivulla

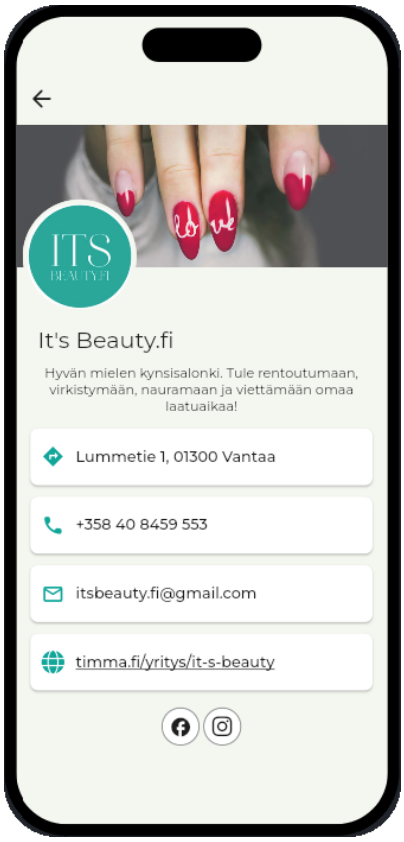
6.4 Yrityksen yhteystietosivu

It’s Beauty.fi:n yhteystietosivulta löytyy yrityksen osoite, puhelinnumero, sähköposti ja linkki varaussivustoille. Lisäksi sivulle on lisätty yrityksen logo sekä Facebook- ja Instagram-painikkeet, jotka vievät suoraan It’s Beauty:n omille sosiaalisen median sivuille (KUVA 14).

```
Align(
  alignment: const AlignmentDirectional(0.0, 1.0),
  child: Padding(
    padding: const EdgeInsetsDirectional.fromSTEB(
        0.0, 0.0, 2.0, 0.0), // EdgeInsetsDirectional.fromSTEB
    child: FlutterFlowIconButton(
      borderColor:
          FlutterFlowTheme.of(context).secondaryText,
      borderRadius: 20.0,
      borderWidth: 1.0,
      buttonSize: 40.0,
      fillColor: FlutterFlowTheme.of(context)
          .secondaryBackground,
      icon: Icon(
        Icons.facebook_sharp,
        color: FlutterFlowTheme.of(context).primaryText,
        size: 24.0,
      ), // Icon
      onPressed: () async {
        await launchURL(
            'https://www.facebook.com/itsbeauty.fi/');
      },
    ), // FlutterFlowIconButton
  ), // Padding
), // Align
```

KUVA 14. Yrityksen Facebook-painike ikonilla

Sivulta löytyy myös tervetuloviesti asiakkaille sekä kuva huolitelluista ja herkullisen värisistä kynsistä, mikä kuvastaa yrityksen tarjoamaa palvelua. Kuva on ladattu Unsplash-kuvapankista, joka tarjoaa vapaasti käytettäviä kuvia. Tässä on yrityksen yhteystietosivun lopullinen ulkoasu, joka sisältää kaikki tarvittavat toiminnot ja toiminnallisuudet (KUVA 15).



KUVA 15. Yrityksen yhteystietosivu

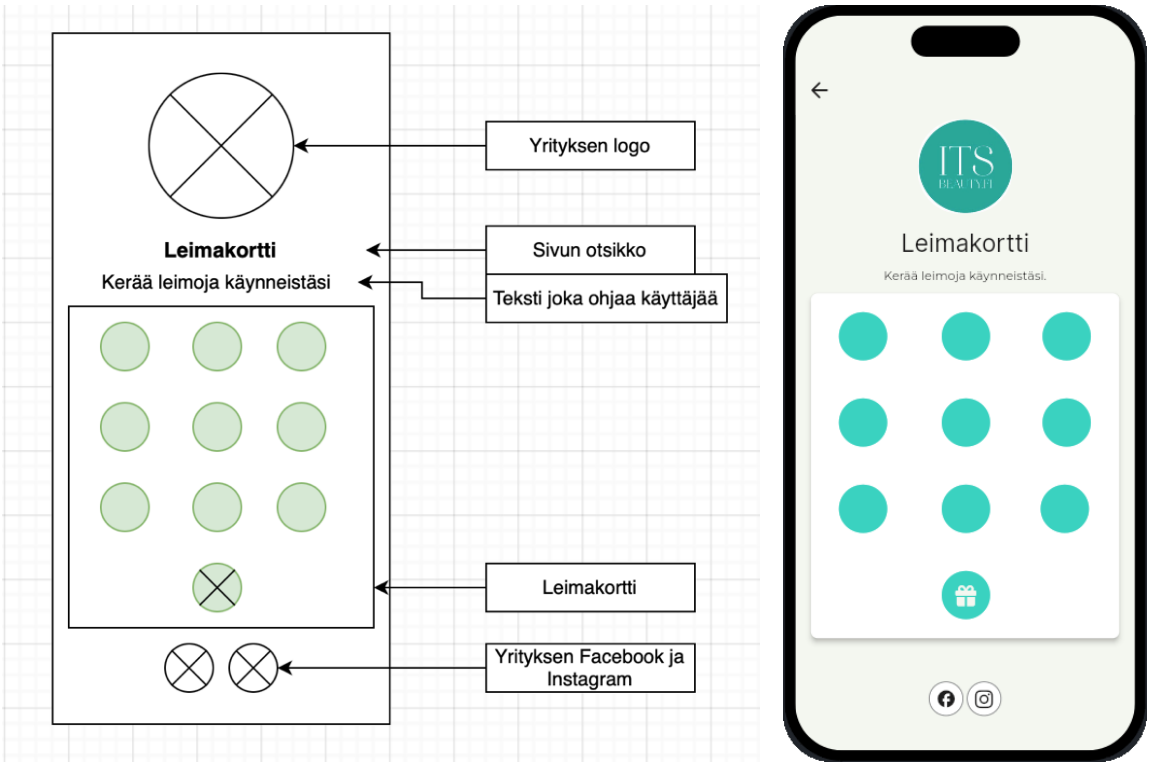
6.5 Leimakortti ja QR-sivu

Leimakorttisivussa on 10 leimaruutua, joista viimeisessä on lahjapaketin kuvake kuvaamaan alennettua käyntikertaa kauneushoitolassa. QR-sivulla asiakkailla on henkilökohtainen QR-koodi ja toimeksiantajalla on QR-skanneri, jolla hän saa kuvattua asiakkaan oman QR-koodin (KUVA 16).
Kun leimakortti on täynnä, sovellus palauttaa sen alkuperäiseen tilaan, jotta uusi leimojen keräyskierros voi alkaa.

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    body: Center(
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          QrImageView(
            data: "$userId ${DateTime.now()}",
            version: QrVersions.auto,
            size: 320.0,
            gapless: false,
          ), // QrImageView
          const SizedBox(
            height: 140,
          ), // SizedBox
          MaterialButton(
            onPressed: () {
              setState(() {});
            },
          ) // MaterialButton
        ],
      ), // Column
    ), // Center
  ); // Scaffold
}
```

KUVA 16. Asiakkaan QR-koodin koodi

Laadin erilaisia luonnoksia leimakorteista ja valitsin niistä kaksi parasta vaihtoehtoa. Näistä kahdesta versiosta päätin toteuttaa sen, joka mielestäni parhaiten edusti yrityksen brändiä. Tämä valittu versio oli myös loogisempi ja visuaalisesti houkuttelevampi. Tässä on alla rautalankamalli, joka oli lähtökohtana, sekä lopullinen versio leimakortista ilman toiminnallisuuksia (KUVA 17).



KUVA 17. Leimakortin rautalankamalli ja lopullinen ulkoasu

Leimakortin turkoosien värisiin ympyröiden ruutuihin lisätään merkintä jokaisen hoitolakäynnin jälkeen. Kortissa on yhteensä 10 ruutua, joista jokaiseen ruutuun merkintään leima aina hoitolakäynnin yhteydessä. Kun kaikki ruudut on leimattu QR-skannerilla, viimeisen ruudun lahjapaketin symboli ilmestyy, osoittaen että asiakas on saavuttanut täyden kortin. Tämä oikeuttaa asiakkaan alennettuun käyntikertaan tai muihin etuihin, kuten ilmaiseen palveluun tai tuotteeseen. Lisäksi digitaalinen leimakortti toimii kätevänä ja ympäristöystävällisenä vaihtoehtona perinteiselle paperiselle kortille. Asiakkaat voivat seurata käyntiensä määrää ja etujaan suoraan älypuhelimen sovelluksen avulla, mikä tekee kanta-asiakasohjelman hyödyntämisestä entistäkin vaivattomampaa.

7 POHDINTA

Toimeksiantajan ja asiakkaiden toiveiden ja tarpeiden myötä oli syntynyt tarve kehittää digitaalinen leimakorttisovellus, jonka suunnittelin ja kehitin Flutter-alustalla. Kanta-asiakassovellukset ovat nykyaikaisia, ympäristöystävällisempiä sekä monipuolisia. Niiden avulla asiakkaat saadaan sitoutumaan yritykseen hyödyntämällä tarjouksia ja palkintoja kanta-asiakkuudesta. Sekä toimeksiantaja että asiakkaat toivovat helppokäyttöistä ja toimivaa sovellusta, joka on yrityksen brändin mukainen.

Toimeksiantajan haastattelun myötä sain hyvän kokonaisnäkemyksen siitä, millaista kanta-asiakassovellusta haetaan sekä asiakkaan että yrittäjän kannalta. Toimimme toimeksiantajan kanssa tiiviissä yhteistyössä, jotta lopputulos olisi mahdollisimman hyvä. Tämä oli ensimmäinen kehittämäni sovellus, joten olin hieman epävarma, pystynkö luomaan sellaisen lopputuloksen, joka miellyttää sekä asiakkaita että toimeksiantajaa. Alussa etsin mahdollisimman paljon tietoa alan sivustoilta sekä verkostoiduin ohjelmointiosaajien kanssa, jotta pystyin tarvittaessa kääntymään heidän puoleensa ongelmatilanteissa.

Olin erittäin innoissani ja motivoitunut opinnäytetyöprojektistani, sillä tulevaisuuden haaveeni liittyvät mobiilisovelluskehitykseen. Koulun myötä olen saanut hyvät perustaidot aiheeseen liittyen, ja tämän projektin tarkoituksena oli syventää ohjelmoinnin osaamistani. Oman näkemykseni mukaan Flutter tarjoaa parhaat edellytykset monialustaiseen mobiilikehitykseen nykyaikaisessa ympäristössä.

Alussa kohtasin haasteita Flutterin asentamisessa ja laajennusten käyttöönotossa, mutta ajan myötä mobiilisovelluksen kehittäminen kyseisen, minulle uuden työkalun avulla sujui hyvin. Kanta-asiakassovelluksen testaaminen vei odotettua enemmän aikaa, sillä Flutter-alustaan tuli päivityksiä suhteellisen säännöllisesti. Siihen liittyvät laajennukset eivät pysyneet päivitysten tahdissa, mikä vaikeutti uusien ominaisuuksien käyttöönottoa.

Opinnäytetyön tietoperustan haasteena oli englanninkielisten termien suomentaminen oikeaan muotoon. Myös suomenkielisten lähteiden löytäminen oli haastavaa, joten olen hyödyntänyt työssäni paljon englanninkielisiä lähteitä. Koodin virheiden paikantamisessa ja korjaamisessa oli haasteita, sillä minulle ei ollut vielä opiskelijana ehtinyt kertyä tarpeeksi kokemusta sovelluskehityksestä, ja siksi

ongelmanratkaisuun meni odotettua enemmän aikaa. Ohjelmakieli on suhteellisen uusi, joten ohjelmointivirheisiin ei välttämättä löytynyt ratkaisuja tai tahoa, jolta olisi saanut neuvoja. Lisäksi sovelluksen toiminnallisuuksien löytäminen ja koodaaminen oli hankalaa.

Toiminnallisen työn osuus valmistui suunnitelmien mukaisesti. Onnistuin luomaan konseptin kanta-asiakasohjelmalle sekä kehittämään osioita kanta-asiakassovelluksesta. Mobiilisovelluksen jatkokehitykselle on tarve, jotta saadaan viimeistellyksi sovelluksen tärkeät ominaisuudet, kuten käyttäjän oikeaan profiiliin sisäänkirjautuminen, käyntihistorian tallentaminen ja leimakortin keskeisin toiminnallisuus leimojen lisäämistä varten. Sain toteutettua ison osan kanta-asiakassovelluksesta, mutta puuttuvat toiminnot vaikuttivat sovelluksen kokonaisvaltaiseen käyttökokemukseen ja käytännöllisyyteen. Tästä syystä aion jatkaa mobiilisovelluksen kehittämistä, jotta voin tuoda markkinoille toimivan digitaalisen leimakortin, joka täyttää käyttäjien tarpeet ja odotukset mahdollisimman hyvin.

Opinnäytetyön aikana sain laajan näkemyksen mobiilikehittämisestä, mobiilisovellusten toiminnasta ja uuden ohjelmointikielen käytöstä. Kaiken kaikkiaan tämä opinnäytetyö tarjosi erinomaisen oppimiskokemuksen ja oli riittävän haastava kehittääkseen taitojani edelleen.

LÄHTEET

Bartosińska, I., & Dembny, M. 2023. *What is Flutter?*. Saatavissa: https://www.thedroidsonroids.com/blog/what-is-flutter-app-development#Summary_%E2%80%93_why_is_everyone_talking_about_Flutter. Viitattu 29.9.2023.

Black, K. 2017. *Why customer loyalty programs are so important*. Forbes 13.9.2017. Saatavissa: https://www.forbes.com/sites/kpmg/2017/09/13/why-customer-loyalty-programs-are-so-important/?sh=207999eb2bd4. Viitattu 14.9.2023.

Ceci, L. 2023. *Number of mobile app downloads worldwide from 2016–2022. Annual number of global mobile app downloads 2016–2022.* Saatavissa: https://www.statista.com/statistics/271644/worldwide-free-and-paid-mobile-app-store-downloads/. Viitattu 20.10.2023.

Dart. *Dart overview. Dart: The platforms.* Saatavissa: https://dart.dev/overview#platform. Viitattu 12.10.2023.

Firebase. *Firebase Realtime Database.* Saatavissa: https://firebase.google.com/docs/database. Viitattu 10.11.2023.

Flutter. *Beautiful apps for every screen*. Saatavissa: https://flutter.dev/. Viitattu 22.9.2023.

Flutter docs a. *Architectural layers.* Saatavissa: https://docs.flutter.dev/resources/architectural-overview. Viitattu 29.9.2023.

Flutter docs b. *Hot reload.* Saatavissa: https://docs.flutter.dev/tools/hot-reload. Viitattu 18.10.2023.

Flutter docs c. *Layouts in Flutter.* Saatavissa: https://docs.flutter.dev/ui/layout. Viitattu 18.10.2023.

Flutter docs d. *Building user interfaces with Flutter.* Saatavissa: https://docs.flutter.dev/ui. Viitattu 18.10.2023.

Flutter docs e. *Add interactivity to your Flutter app. Stateful and stateless widgets.* Saatavissa: https://docs.flutter.dev/ui/interactivity#stateful-and-stateless-widgets. Viitattu 18.10.2023.

Flutter docs f. *Differentiate between ephemeral state and app state.* Saatavissa: https://docs.flutter.dev/data-and-backend/state-mgmt/ephemeral-vs-app. Viitattu 19.10.2023.

Flutter docs g. *Start thinking declaratively*. Saatavissa: https://docs.flutter.dev/data-and-backend/state-mgmt/declarative. Viitattu 19.10.2023.

Flutter docs h. *Firebase*. Saatavissa: https://docs.flutter.dev/data-and-backend/firebase. Viitattu 10.11.2023.

Flutter docs i. *Set up an editor.* Saatavissa: https://docs.flutter.dev/get-started/editor. Viitattu 24.10.2023.

FlutterFlow docs. *Export FlutterFlow UI code to your Flutter project.* Saatavissa: https://docs.flutterflow.io/flutter/export-flutterflow-ui-code-to-your-flutter-project. Viitattu 17.11.2023.

Ford, S. 2019. *The Dart Language: When Java and C# Aren't Sharp Enough*. Saatavissa: https://www.toptal.com/dart/dartlang-guide-for-csharp-java-devs. Viitattu 11.10.2023.

Khan, S. 2023. *Real-time Data Management in Flutter with Firebase Database.* Saatavissa: https://medium.com/@samra.sajjad0001/real-time-data-management-in-flutter-with-firebase-database-458f81667a6c. Viitattu 10.11.2023.

Kodeco. 2023. *The Top 5 Flutter State Management Solutions: A Deep Dive. What is State Management?* Saatavissa: https://www.kodeco.com/39848254-the-top-5-flutter-state-management-solutions-a-deep-dive. Viitattu 19.10.2023.

Neshkoska, M. 2023. *Why are Flutter developers in high demand.* Saatavissa: https://proxify.io/articles/why-are-flutter-developers-in-high-demand. Viitattu 23.10.2023.

Pay, K. 2022. *Loyalty frameworks, punch & stamp cards.* Saatavissa: https://loyaltyrewardco.com/loyalty-frameworks-punch-stamp-cards/. Viitattu 18.9.2023.

Sahu, A. 2022. *What is Best for App Development: Flutter, Dart Language or Something Else?* Turing 11.8.2022. Saatavissa: https://www.turing.com/blog/flutter-vs-dart-best-platform-for-app-development/. Viitattu: 12.10.2023.

Tikkanen, I. 2023. Toimeksiantajan haastattelu. 4.10.2023. It's Beauty. Vantaa.

Tillu, J. 2019. *4. What is Widget in flutter? Let's clear the basics first. Stateless Widgets.* Saatavissa: https://medium.com/jay-tillu/4-what-is-widget-in-flutter-lets-clear-the-basics-first-82f501c8d0f0. Viitattu 18.10.2023.

Tillu, J. 2022. *What is State in Flutter?. What is State?* Saatavissa: https://jaytillu.medium.com/what-is-state-in-flutter-dbee1522fb59. Viitattu 19.10.2023.

Vailshery, L. 2023. *Cross-platform mobile frameworks used by developers worldwide 2019–2022*. Saatavissa: https://www.statista.com/statistics/869224/worldwide-software-developer-working-hours/. Viitattu 20.10.2023.

Volarević, M. 2022. *5 top punch card apps in 2022: An honest review.* Saatavissa: https://blog.loopyloyalty.com/5-top-punch-card-apps-in-2022-an-honest-review-69a5a1807400. Viitattu 14.9.2023.

Windmill, E. 2020a. *Flutter in action. Why does Flutter use Dart?* Shelter Island: Manning Publications.

Windmill, E. 2020b. *Flutter in action. Again, everything is a widget.* Shelter Island: Manning Publications.

Windmill, E. 2020c. *Flutter in action. Working with Firebase in Flutter.* Shelter Island: Manning Publications.

Ziemianek, M. 2020. *The benefits of a digital stamp cards.* Saatavissa: https://appinstitute.com/digital-stamp-cards/. Viitattu 18.9.2023.