TAMPEREEN
AMMATTIKORKEAKOULU

MUXUNI

Mobiilisovellus

Santeri Kivikoskela

Mikki Levón

Opinnäytetyö
Helmikuu 2018
Tieto- ja viestintätekniikka
Ohjelmistotekniikka

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tieto- ja viestintätekniikka
Ohjelmistotekniikka

SANTERI KIVIKOSKELA & MIKKI LEVÓN:
Muxuni
Mobiilisovellus

Opinnäytetyö 46 sivua, joista liitteitä 4 sivua
Helmikuu 2018

Tässä opinnäytetyössä tarkastellaan kehitteillä olevaa Muxuni-mobiilisovellusta, jota on kehitetty Ionic- ja AngularJS-ohjelmistokehyksiä käyttäen. Lisäksi sovellus käyttää Apache Cordovaan päästäkseen kiinni mobiililaitteiden sisäisiin API:hin.

Muxuni on sovellus, joka tarjoaa nopean ja kätevän käyttöliittymän, jolla voi tilata lastenhoitajan ja sopia työn yksityiskohdista lastenhoitajan kanssa. Sovelluksessa on myös käyttöliittymä lastenhoitajille, josta he voivat hallita töitään ja profiiliaan.

Sovellus on alfa-testausvaiheessa ja sen tärkeimmät ominaisuudet ovat tasolla, jolla sovellusta voidaan testata kontrolloidussa ympäristössä. Beta-testaukseen päästään, kunhan tärkeimmät ominaisuudet ovat kokonaan valmiita. Sovellus julkaistaan Play Store ja App Store -kauppapaikoissa vuoden 2018 aikana.

Opinnäytetyössä käsitellään sovelluksessa käytettyjä tekniikoita ja teknologioita, sen arkkitehtuuria, sovelluksen käyttöä ja käyttöliittymää, sekä suunnittelu- ja kehitysprosessia.

Asiasanat: mobiilisovellus, mobiilikehitys, hybridisovellus, ionic

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Information and Communication Technology
Software techniques

SANTERI KIVIKOSKELA & MIKKI LEVÓN:
Muxuni
Mobile application

Bachelor's thesis 46 pages, appendices 4 pages
February 2018

This thesis examines Muxuni mobile application which is being developed with Ionic and AngularJS frameworks. In addition, the application uses Apache Cordova to make use of the device APIs of the mobile device.

Muxuni is an application that offers a fast and easy way to order a nanny and agree on the details of the task. The application also has a user interface for nannies from which they can manage their profile and work schedule.

The mobile application is in alpha testing phase and its most important features are on a level that they can be tested in a controlled environment. Beta testing phase can start when those features are done. The application will be released within the year 2018 in Play Store and App Store.

This thesis addresses the techniques and technologies used in the application, its architecture, the usage of the software and its user interface. The designing and development process is also discussed.

Key words: mobile application, mobile development, hybrid application, ionic

SISÄLLYS

LYHENTEET JA TERMIT

Ohjelmistokehys	(Engl. framework) Pohjakoodi ja komponentit, joiden muodostaman rungon päälle voi rakentaa uuden ohjelmiston
Natiivisovellus	Sovellus, joka on toteutettu suoraan mobiililaitteen käyttöjärjestelmän ohjelmointikielellä ja rajapinnoilla.
Hybridisovellus	Mobiilisovellus, joka on toteutettu HTML5:n, CSS:n ja JavaScriptin avulla.
SPA	(Engl. Single Page Application) Yhden sivun sovellus.
VCS	(Engl. Version Control System) Versionhallintajärjestelmä.
Front-end	Ohjelmiston esityksellinen kerros
Modaali	Näytön ylle tuleva ikkuna, jonka takaa vielä näkyy alkuperäinen, mutta estää alkuperäisen käytön ennen ikkunan sulkemista
API	(Engl. Application Programming Interface) Ohjelmointirajapinta, jonka mukaan ohjelmat tai ohjelman osat voivat vaihtaa tietoja keskenään tavalla, joka on tarkasti määrätty rajapinnan luojan toimesta.

1 JOHDANTO

Muxuni-applikaatio sai alkunsa, kun huomattiin, että markkinoilta ei löytynyt palvelua, josta saisi nopeasti lastenhoitajan kotiin pienellä varoitusajalla. Useilla toimijoilla kesti yli 24 tuntia tilauksen vastaanottamisesta lastenhoitajan saapumiseen. Nämä siis eivät toimisi, mikäli lastenhoitajan tarve olisi akuutti.

Projektin aikana toteutettava mobiilisovellus pyrkii olemaan mahdollisimman nopea ja luotettava tapa yhdistää vanhemmat ja lastenhoitajat, sekä tekemään sen mahdollisimman yksinkertaisesti. Molemmat osapuolet voivat käyttää samaa sovellusta ja ilmoitus uudesta tilauksesta tulee heti lastenhoitajalle sovelluksen kautta. Sovellusta ei ole vielä julkaistu, koska sen kehitys on vielä kesken.

Luvussa kaksi käydään läpi projektissa käytetyt teknologiat ja työkalut, sekä kuvataan niiden ominaisuuksia. Kolmannessa luvussa käydään läpi mobiilisovelluksen valmiina olevat ominaisuudet ja toiminnot. Tässä luvussa kerrotaan myös Muxunin kohderyhmästä ja kilpailijoista. Luvussa neljä käsitellään applikaation loogista arkkitehtuuria ja rakennetta ja lisäksi kerrotaan liitännöistä muihin API:hin. Käyttöliittymän toteutuksesta kerrotaan luvussa viisi, jossa käydään läpi sovelluksessa navigointia ja käyttökokemusta. Luvussa kuusi käydään läpi projektin suunnittelua, toteuttamista ja näissä käytettyjä toimintamalleja. Luku seitsemän tiivistää applikaation tulevaisuudennäkymät ja tarkastellaan jo ajateltuja toiminnallisuuksia. Työn viimeinen luku sisältää pohdinnan projektin nykytilanteesta ja toimintatavoista.

2 KÄYTETYT TEKNOLOGIAT JA TYÖKALUT

2.1 Ionic ohjelmistokehys

Ionic on ohjelmistokehys hybridisovelluksille, jolla on käytössä seuraavat webteknologiat: HTML, CSS ja Javascript. Ionic keskittyy pitkälti sovelluksen ulkoasuun ja käyttäjäkokemukseen ja yrittää saada siitä mahdollisimman natiivin tuntuisen. Ionickia voi käyttää monilla eri Javascript-kirjastoilla, mutta monet tärkeät ominaisuudet vaativat AngularJS:n, jolloin sitä suositellaan käyttämään Ionic-sovelluksia kehittäessä. Ionicillä itsellään ei ole pääsyä puhelimen rajapintoihin, joten on suositeltavaa käyttää Cordovaa Ionicin kanssa (Drifty, 2016).

Ionic sai virallisen julkaisunsa vuonna 2015 yrityksessä Drifty kolmen kehittäjän toimesta. Tätä ennen Ionic oli alfavaiheessa 2013 ja betassa 2014. Nykyään Ionickia kehittää laaja kansainvälinen yhteisö. Vuonna 2015 Ionicilla rakennettiin yli 1,3 miljoonaa sovellusta (Drifty, 2016).

2.1.1 Webteknologiat

Webteknologioilla viitataan moniin kieliin ja multimediapaketteihin, joiden yhdistäminen muodostaa websivuja. Web-teknologiat ovat toiminnaltaan yksinään suhteellisen rajallisia, jolloin ne vaativat toisiaan muodostaakseen jotain järkevää (Collins, 2018). Tässä kappaleessa kerrotaan klassisimmista webteknologioista: HTML, CSS ja Javascript.

HTML tulee sanoista Hypertext Markup Language. Se ei siis ole ohjelmointikieli vaan merkintäkieli (engl. markup language) eli sillä laitetaan tekstin sekaan metadataa, joka ohjaa tekstin rakennetta ja ulkoasua. HTML muodostuu elementeistä, joita ilmaistaan sisältöä ympäröivillä tageilla. Elementeillä voidaan luoda esimerkiksi kursivoitua tekstiä, otsikoita, linkkejä tai jopa taulukoita (Mozilla, 2018).

CSS on tyylisivukieli, jonka nimi muodostuu sanoista Cascading Style Sheets. Siinä, missä HTML määrittelee sivun rakenteen ja että millaisiin loogisiin kokonaisuuksiin, CSS määrittelee miltä kaikki näyttää. CSS:llä määritellään erilaisia tyylejä eri palasille

sivulla ominaisuuksilla, jotka voivat määritellä esimerkiksi tekstin fonttikoon, alueen taustavärin tai marginaalin. Selaimet asettavat näiden ominaisuuksien asettamat säännöt esitettävään dokumenttiin muokatakseen sen ulkoasua (Mozilla, 2017).

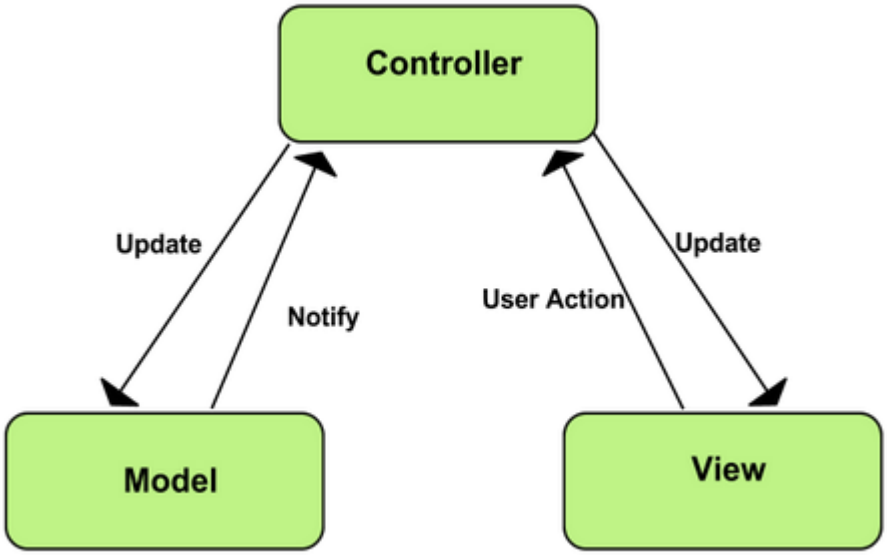
JavaScript on prototyyppipohjainen olio-ohjelmointikieli eli siinä luodaan olioita määrittelemättä niille luokkia ensin. Se on kevyt kieli, jota useimmiten käytetään HTML-sivujen yhteydessä tekemässä websovelluksista dynaamisia. HTML:n kanssa käytettynä JavaScriptillä voi luoda tapahtumia napin painalluksille ja sivujen navigoinnille (Mozilla, 2018).

2.1.2 AngularJS

AngularJS on JavaScript-ohjelmistokehys, jota Google ylläpitää (Viskari, 2015). Sen lähdekoodi on MIT-lisenssillä saatavilla ilmaiseksi GitHubista tai AngularJS:n omilta sivuilta (AngularJS Github).

AngularJS toteuttaa SPA-arkkitehtuuria, joka tarkoittaa, että sovelluksen käyttöliittymä ja taustajärjestelmä erotetaan vahvasti toisistaan, jolloin selaimen sivunlatauksia ei tapahdu eri toimintojen välillä. Tämä nopeuttaa sovelluksen toimintaa, koska koko sivua ei tarvitse ladata uudestaan ja näin käytettävyys paranee. (Viskari, 2015.). Tästä seuraa kuitenkin se, että sovelluksen prosessointi tapahtuu suurimmaksi osaksi käyttäjän laitteella, mikä voi aiheuttaa ongelmia varsinkin mobiililaitteilla (Lourenço, 2014).

AngularJS:n käyttämä MVC-malli (engl. model-view-controller model) auttaa ohjelmistokehittäjiä tuottamaan modulaarisempia sovelluksia, mikä helpottaa ohjelmakoodin uudelleenkäytettävyyttä, usean kehittäjän samanaikaista työskentelyä ja sovelluksen näkymälogiikan ja liiketoimintalogiikkan erottamista toisistaan. MVC malli koostuu kolmesta komponentista, jotka näkyvät kuviossa 1. (Chrome).



KUVIO 1. MVC malli (Chrome)

Malli (engl. model) tallettaa sovelluksen datan ja kun tämä data muuttuu, ilmoittaa siitä tämän kuuntelijoille. Näkymä (engl. view) on mitä näytetään käyttäjälle ja minkä kautta applikaatiota käytetään. Käsittelijä (engl. controller) on logiikka, joka päivittää näkymää, kun malli muuttuu ja päivittää mallia, kun käyttäjä käyttää näkymää. (Chrome).

2.1.3 Apache Cordova

Apache Cordova, entiseltä nimeltään Phonegap, on avoimeen lähdekoodiin perustuva ohjelmistokehys. Sen avulla pystytään tekemään hybridisovelluksia eli käyttämään HTML5:tä, CSS:ää ja JavaScriptiä mobiiliapplikaation tekemiseen. Apache Cordova tukee lähes kaikkia suurimpia mobiilialustoja (ml. Android, iOS, Windows Phone). (Virinchy, 2014).

Hybridiapplikaatiossa käyttöliittymä on koko näytön täyttävä natiivi WebView, jonka sisällä web-applikaatio ajetaan. Tällöin vain WebView-elementti muuttuu mobiilialustan mukaan ja pystytään uudelleenkäyttämään koko webapplikaatio eri mobiilialustoille. (Virinchy, 2014).

Normaalit websivustot eivät pääse kiinni mobiililaitteen laite-API:hin, kuten kameraan tai GPS anturiin, kiinni, mutta Apache Cordovalla tehdyt applikaatiot voivat käyttää sen

tuomia kirjastoja, jotka ottavat yhteyttä laite-API:hin. Tällöin pystytään käyttämään esimerkiksi kameraa Apache Cordovalla tehdyllä applikaatiolla. Lisäksi mikäli valmista kirjastoa ei ole saatavilla, pystytään niitä luomaan itse. Tällöin pitää tehdä toteutus kaikille halutuille alustoille niiden natiivilla ohjelmointikielellä. (Virinchy, 2014).

2.2 Git-versionhallinta

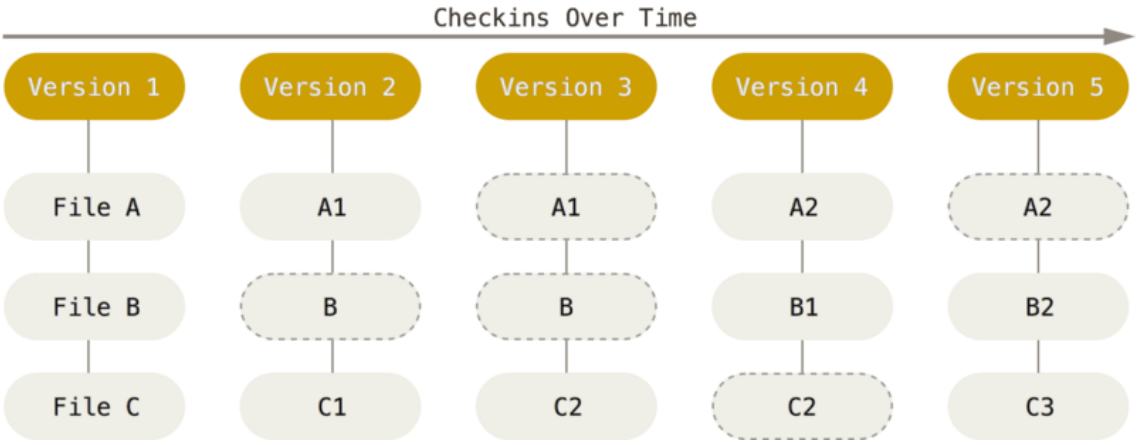
Versiohallintajärjestelmä (myöh. VCS) pitää kirjaa tiedostoon tai tiedostoihin tehdyistä muutoksista. Tätä tarvitaan siihen, että voidaan pitää muistissa vanhat versiot tiedostoista ja palata niihin tarvittaessa. VCS:n toinen tehtävä on varmuuskopioida tiedostoja. VCS:t voidaan jakaa kolmeen eri luokkaan: lokaaliin, keskitettyyn ja hajautettuun VCS:ään. (Chacon;ym., 2014).

Lokaalissa VCS:ässä on yksinkertainen tietokanta, jossa pidetään kaikki muutokset tiedostoihin. Tämä tietokanta sijaitsee omalla koneella ja siihen pääsee käsiksi vain tältä koneelta. (Chacon;ym., 2014).

Keskitetty VCS oli ratkaisu, kun haluttiin usean kehittäjän tiimin pystyvän tekemään projekteissa yhteistyötä. Tässä järjestelmässä yhdellä palvelimella on tietokanta, jossa pidettään muutokset tiedostoihin. Tästä tietokannasta käyttäjä pystyy hakemaan haluamansa version tiedostot, sekä lähettämään uuden version tietokantaan. Tällöin myös helpottuu VCS:in valvonta, kun pystytään katsomaan, kuka on tehnyt minkäkin muutoksen. Lisäksi pystytään antamaan eri oikeuksia eri käyttäjille. Ongelmana järjestelmässä on, että jos palvelin hajoaa tai tietokannan data korruptoituu ja ei olla pidetty hyviä varmuuskopioita, koko projektin historia katoaa, paitsi yksittäisten käyttäjien tilannevedokset. (Chacon;ym., 2014).

Hajautettu VCS tuo tähän ongelmaan ratkaisun. Jokainen käyttäjä hakee itselleen palvelimelta koko historian omalle koneelleen. Mikäli palvelin hajoaa, pystytään koko projektin historia tuomaan takaisin käyttäjän omasta tietokannasta. (Chacon;ym., 2014)

Git on hajautettu VCS, joka luotiin Linuxin kernel-projektiin vuonna 2005. Sen toiminta perustuu siihen, että joka Gittiin tallentamisen (commit) yhteydessä, Git ottaa periaatteessa tilannevedoksen muutetuista tiedostoista ja lisää versionhallintaan viitteen tähän tilannevedoksen. Kuviossa 2 on havainnollistettu Gitin tapa tallentaa. (Chacon;ym., 2014).



KUVIO 2. Gitin tapa tallentaa tiedostojen tilannevedoksia. (Chacon;ym., 2014)

Kuten kuvassa 2 nähdään, jos tiedostoa ei ole muutettu, tulee uuteen versioon vain viite siihen tilannevedokseen, jossa tiedostoa on muutettu. Esimerkiksi kuvan versiossa 2 on muutettu tiedostoa A ja C, mutta tiedosto B on muuttumaton, jolloin tietokantaan ei tallenneta tätä tiedostoa uudestaan vaan viite ensimmäisen version tiedostoon B. (Chacon;ym., 2014).

Lisäksi lähes jokainen Git-operaatio on lokaali, jolloin voidaan tarkastella vanhoja versioita tiedostoista ilman internetyhteyttäkin. Myös muutoksia pystytään tekemään lokaalisti tallentamalla ne lokaaliin tietokantaan ja kun on mahdollista taas yhdistää internettiin, voidaan muutokset puskea palvelimelle. (Chacon;ym., 2014).

2.3 Google Maps

Google Maps on Googlen omistama karttapalvelu, jossa on useita toiminnallisuuksia kartan näyttämisen lisäksi, kuten reittien haku ja näyttäminen kartalla. Se toimii usealla alustalla, kuten mobiilissa sekä websivustoilla. (Google Maps). Google Maps julkistettiin 8.2.2005 Googlen blogissa. Tuotepäällikkö Bret Taylor kertoo postauksessaan, että sen

ominaisuuksiin kuuluu reittien haku, esimerkiksi hotellien haku alueelta ja kartan siirtäminen dynaamisesti ilman odotusaikaa tai uuden kuvan latausta.

2.4 Teknologioiden valinta

Muxuni on sovellus, jonka tavoite olisi tavoittaa useat suomalaiset ja toimia monia vuosia julkaisunsa jälkeen, joten oli erittäin tärkeää valita teknologiat, joilla sovelluksen vaatimukset toteutuvat huomioiden myös tiimin taidot ja tilanteen. Sovelluskehitys on tapahtunut ilman yritystä tai minkäänlaista rahoitusta, joten sovelluskehityksen pitää olla ilmaista. Kehityksen aloituksen aikaan myös tiimin sovelluskehittäjillä ei ollut kunnollista kokemusta sovelluskehityksestä ja vaativammista ohjelmointikielistä, joten Javascript-pohjainen ohjelmistokehys sopi hyvin tilanteeseen.

Muxuni julkaistaan kuluttajamarkkinoille, jossa käyttäjillä voi olla hyvin monenlaisia mobiililaitteita ja sovelluksen toivotaan saavuttavan mahdollisimman monet asiakkaat. Ionic mahdollistaa saman koodin kääntämisen Androidille ja iOS:lle ja sen vaatimukset näiltä käyttöjärjestelmiltä ovat matalat. Näkyvyyden mahdollistamista mahdollisimman monelle laitteelle auttaa myös, ettei sovellus ole kovin raskas, jolloin voidaan valita einatiivi tapa kehittää sovellusta ilman, että sovelluksen muistin kulutus nousisi kovin korkealle.

3 MUXUNI-SOVELLUS

3.1 Sovelluksen tarve

Huhtikuussa 2016 jo ennen Muxunin kehittämisen aloitusta suoritettiin Tampereen lähikunnissa nettikysely, johon osallistui 480 vanhempaa. Vastanneista 95% tavoitettiin eri lastenhoidon palveluiden kautta (MLL, Huusholli jne.). 65% osallistuneista oli Tampereelta, 15% Nokialta, 6% Ylöjärveltä, 6% Lempäälästä, 5% Kangasalalta ja 3% Pirkkalasta.

Tutkimukseen 64% olivat käyttäneet viimeisen yhdeksän kuukauden aikana kerran tai kaksi kuussa lastenhoitajan palveluita, 32% vielä useammin ja 4% eivät olleet käyttäneet ollenkaan. Osallistujista 55% käyttivät lastenhoitajapalveluita useimmin arki-iltoina, mutta loput tilasivat lastenhoitajan viikonlopuksi vanhempien omaa aikaa varten (liite 1).

Tutkimuksen mukaan tyypillisesti 38%:lle lastenhoitajista maksetaan 70€ tai enemmän yhdestä illasta. Huomattava trendi on myös, että nuorille lastenhoitajille maksetaan huomattavasti vähemmän kuin vanhemmille lastenhoitajalle. Keskimääräinen lasten hoitajan palkka osuu haarukkaan 8-16€/h (liite 1).

Näistä tuloksista huomaa, että on paljon vanhempia, jotka tarvitsevat huomattavan määrän lastenhoitajan palveluita ja käyttävät niitä myös pakollisen tarpeen lisäksi oman ajan saamiseksi. Tutkimuksen ulkopuolisista keskusteluista on myös tullut ilmi, että lapsiperheissä on usein yllättäviä tilanteita, joihin on lyhyen varoitusajan takia erittäin vaikeaa palkata lastenhoitaja. Erityisesti näitä kiireisiä ja paljon lastenhoitajan palveluita käyttäville vanhemmille lähdettiin suunnittelemaan Muxunia, joka helpottaisi heidän arkeansa.

Työmarkkinoilla myös Muxuni on hyvä, sillä uuden aktiivimallin mukaan työttömän tulee tehdä 18 tuntia palkkatöitä, tienata yritystoiminnalla 241€ tai olla viisi päivää TE-toimiston työllistymistä edistävässä palvelussa kolmen kuukauden aikana säilyttääkseen työttömyystukensa (Työ- ja elinkeinoministeriö, 2018). Henkilöt, jotka haluavat tehdä lastenhoitajan töitä, saavat Muxunin kautta näkyvyyttä itselleen ja kykenevät keräämään itselleen vähintäänkin vaaditun määrän säilyttääkseen työttömyystukensa. Muxuni on

myös sopiva lastenhoitajan ammattia opiskelevalle, joka voi saada lisätuloja toimimalla lastenhoitajana Muxunin kautta.

3.2 Tarkoitus

Muxuni on sovellus, joka tarjoaa nopean ja kätevän käyttöliittymän, jolla voi tilata lastenhoitajan ja sopia työn yksityiskohdista tarkemmin lastenhoitajan kanssa. Sovelluksessa on myös käyttöliittymä lastenhoitajille, josta he voivat hallita töitään ja lastenhoitajaprofiiliaan.

Muxunin tarkoitus on helpottaa vanhempien mahdollisuuksia järjestää toisilleen aikaa tai saada lastenhoitaja nopeasti yllättäviin tilanteisiin. Sovellus on myös mahdollisuus työllistää itsensä yksityisen lastenhoitajan tehtäviin tai hyvä markkinointialusta isommillekin lastenhoitajan palveluja tarjoavalle yritykselle.

Toiminta aloitetaan ensin vain Tampereen alueella tarkoituksena hankkia käyttäjiä vähitellen lisää ja sitten laajentaa toimintaa muihinkin Suomen kuntiin.

3.3 Kohderyhmä

Muxunilla on kaksi kohderyhmää: vanhemmat ja lastenhoitajat. Muxunia käyttävillä vanhemmilla on pieniä lapsia, jotka tarvitsevat hoitoa vanhempien ollessa poissa. Vuonna 2017 ensimmäisen lapsensa synnyttänyt äiti on ollut keskimäärin 29-vuotias (Tilastokeskus, 2017), mutta haarukkaa on kaksikymppisestä nelikymppiseen.

Lastenhoitajat voivat esimerkiksi olla nuoria sosiaali- ja terveysalan opiskelijoita tai lastenhoidon kokeneita eläkeläisiä, jotka kaipaavat lisätienestiä. Myös lastenhoitofirmat ja yksityiset ammattilaiset voivat hyödyntää sovellusta saadakseen itselleen näkyvyyttä.

3.4 Kilpailijat

Muxuni-sovellusta vastaavia ei ole Suomessa kuin Sitly, joka on Alankomaisen, kuuteen maahan levinneen firman, 2Care4Kids:n tekemä mobiilisovellus, jota voi myös käyttää vanhempana tai lastenhoitajana. Sitlyn rinnalla toimii Suomessa babysitter.fi-sivusto, jolle on vastaava sivusto jokaisessa kuudessa maassa, missä 2Care4Kids:kin toimii. Kirjautuminen palveluun tehdään sähköposti-salasana-parilla tai Facebook-kirjautumisella. Palvelu vaatii Premium-jäsenyyden, jotta siinä voi sopia lastenhoitajan/vanhemman kanssa töistä (2Care4Kids Group, 2018).

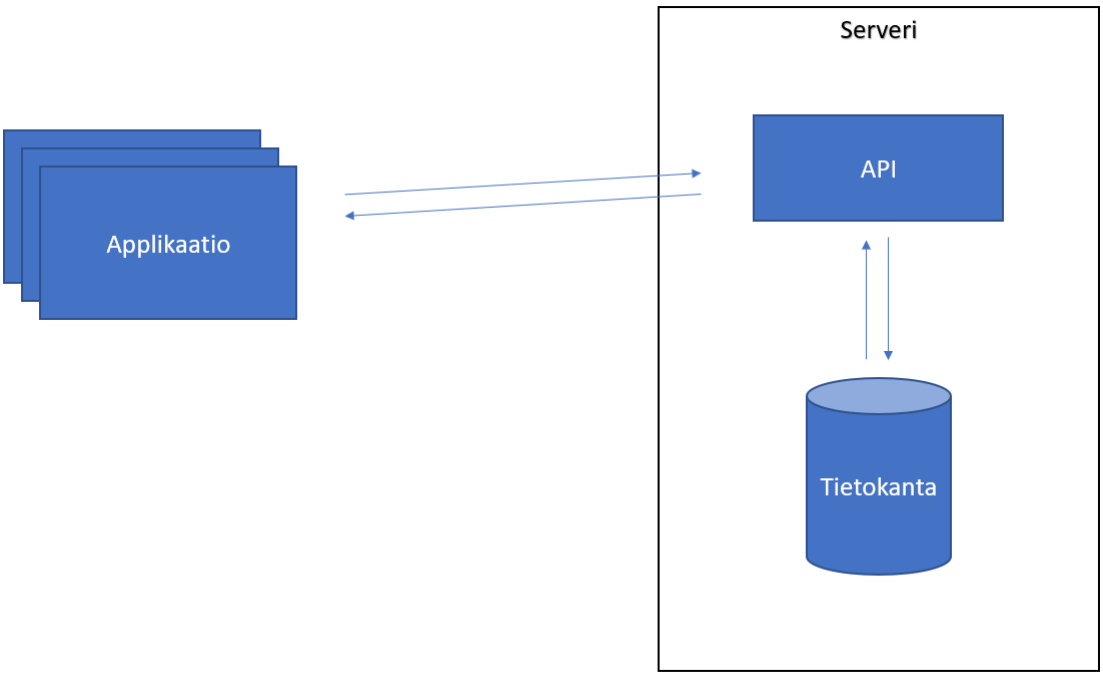
Muxunin kilpailijoina toimivat myös muunlaiset websivustot, jotka välittävät lastenhoitajien palveluita. Osa näistä palveluista välittää yksityisiä lastenhoitoa tarjoavia henkilöitä asiakkaille ja osa näistä on suurempien firmojen, kuten MLL, välityspalveluita. Isommat firmat voivat vaatia suhteellisen suuriakin hintoja lastenhoidosta eikä kaikki raha mene itse lastenhoitajalle (Hoivanet, 2018).

Muita mobiili- ja/tai websovelluksia, joiden kanssa Muxuni kilpailee, ovat keikkatyönvälityssovellukset. Niissä henkilö voi kirjautua töitä etsiväksi tai työnantajaksi ja hakea työntekijöitä tehtäviin, joita kohtaan työntekijää ilmoittaa kiinnostuksensa. Yksi esimerkki tällaisista sovelluksia on Treamer. Treamerista löytyy profiilit työntekijöistä ja työnantajista. Työnantajat hakevat työntekijöitä tehtäviin ja työntekijät osoittavat kiinnostusta töihin. Työnantajat maksavat Treamerin kautta ja pystyvät arvostelemaan työntekijöitä (Treamer Oy, 2017).

Muxunin etu kilpailijoihinsa nähden on, että se tarjoaa turvallisen ja nopean käyttöliittymän eksklusiivisesti lastenhoidon tilaamiseen. Sovellus vaatii jokaiselta käyttäjältään pankkitunnistautumisen, jotta voidaan varmistaa jokaisen rekisteröityneen henkilöllisyys. Turvallisuuteen liittyen kenenkään tarkkaa sijaintia ei näytetä ennen kuin tilaus on vahvistettu ja maksettu. Sovelluksessa on kattavat profiilit kaikista käyttäjistään ja arvostelut lastenhoitajista. Lastenhoitajat määrittelevät itse oman hintansa ja kaikkiin maksuihin lisätään pieni prosentti, joka otetaan Muxunin jatkokehitykseen. Tämä ei kuitenkaan nosta lastenhoitajan hintaa paljon. Lastenhoitajan kännykkään tulee ilmoitus töistä ja hän pystyy hyvinkin vähäisellä varoitusajalla hyväksymään töitä.

3.5 Toiminta

Järjestelmään kuuluu itse applikaatio, järjestelmän tietokanta ja API, joka antaa ja muokkaa tietokannan tietoja. Näiden väliset suhteet ovat myös kuvattuna kuviossa 3. Tämä opinnäytetyö käsittelee sovelluksen front-end-kehitystä, joten API:a ja tietokantaa ei kuvata työssä.



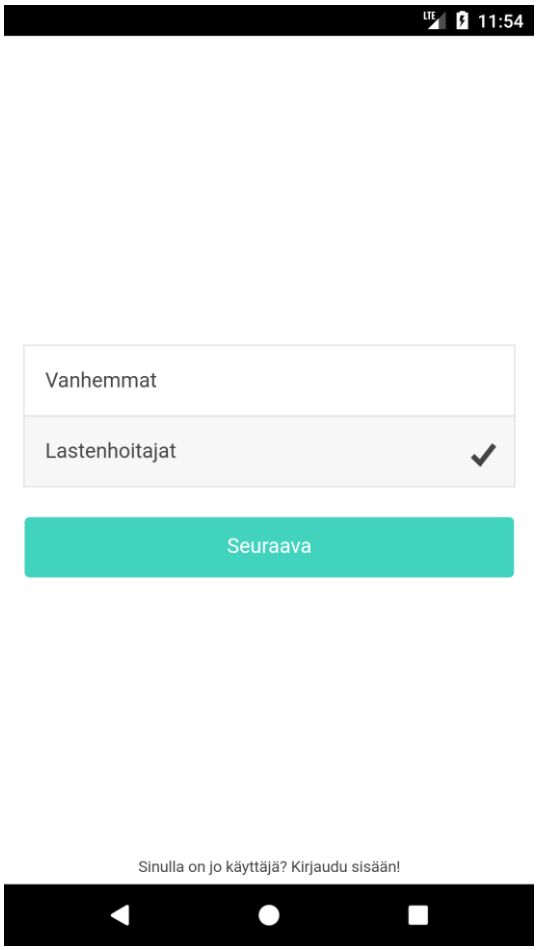
KUVIO 3. Järjestelmän karkea rakenne

Muxuni-sovelluksen kaikki toiminnot tukevat lastenhoitajien tilaamista, mikä on sovelluksen tärkein tarkoitus. Sovellukseen voi kirjautua vanhempana tai lastenhoitajana ja molemmille on täysin omat käyttöliittymät samassa sovelluksessa. Molemmilla rooleilla on omanlaiset profiilinsa. Vanhemmat voivat tilata lastenhoitajia ja lastenhoitajat voivat hallita töitään.

3.5.1 Kirjautuminen ja rekisteröinti

Vakikäyttäjän näkökulmasta sovelluksen käyttö alkaa vain kirjautumisesta sisään, jolloin käyttäjä ohjataan oman roolinsa käyttöliittymän puolelle. Kuitenkin ennen kuin voi kirjautua sovellukseen sisään, pitää rekisteröityä Muxuni-järjestelmään. Rekisteröityessä

valitaan käyttäjän rooli kuvan 1 ikkunassa, syötetään sähköpostiosoite, joka toimii käyttäjän kirjautumistunnuksena, ja salasana käyttäjätilille. Tämän jälkeen käyttäjälle lähetetään sähköposti, jonka kautta käyttäjä voi aktivoida tilinsä. Tiliä ei voi käyttää ennen sen aktivointia.



KUVA 1. Roolin valinta Muxuniin rekisteröityessä.

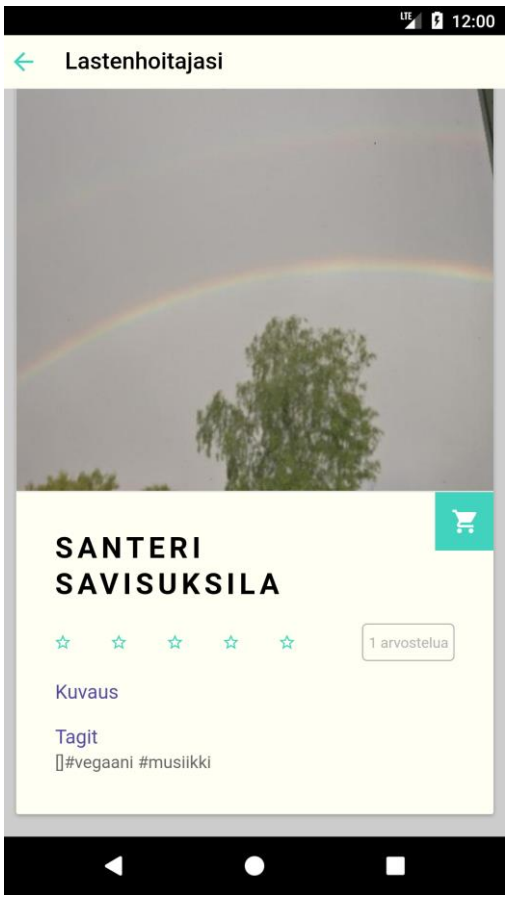
Tärkeä osa rekisteröitymisprosessia on ensimmäinen kirjautuminen, jossa käyttäjää vaaditaan asettamaan profiilinsa tiedot ensimmäistä kertaa. Perustietoja, jotka molemmilta rooleilta löytyy, on nimi, kontaktitiedot, osoitetiedot ja profiilikuva. Tämän lisäksi vanhemmat lisäävät järjestelmään lapsensa tiedot, joihin kuuluu nimi, ikä, sukupuoli, allergiat/erityisruokavaliot ja huomioitavat sairaudet tai vammat. Myös sekä yleisesti vanhemman profiilissa ja jokaisen lapsen tiedoissa on ”muuta”-kohta, johon voi laittaa muuta huomioitavaa. Lastenhoitajat täyttävät itsestään tilitiedot, palkkansa, työaikansa, työalueensa, kuvauksen itsestään ja tageja, jotka toimivat kuvauksen tukena.

3.5.2 Vanhempien käyttöliittymä

Kirjauduttuaan sovellukseen vanhemmat näkevät ensimmäisenä listauksen tulevista tilauksistaan. Tilausta koskettamalla aukeaa yhteenveto, jossa on hoitava lastenhoitaja, työn sijainti, aika ja linkki lastenhoitajan profiiliin. Tilauksesta löytyy myös nappi chattiin lastenhoitajan kanssa. Tulevien tilausten yläpuolella on välilehtivalinta tulevien ja menneiden tilauksien välillä. Menneistä tilauksista saa samalla tavalla yhteenvedon kuin tulevistakin, mutta chat-mahdollisuuksia ei ole. Jokaisessa menneessä tilauksessa on nappi lastenhoitajan arvostelemiselle, kunnes tilaus on kerran arvosteltu.

Sovelluksen otsakkeesta löytyy nappi, jota koskettamalla avautuu pieni valikko ruudun yläkulmaan. Valikosta pääsee profiilin muokkaukseen, asetuksiin ja kirjautumaan ulos. Profiilin muokkauksessa voi säätää kaikkea aiemmin mainittuja vanhemman profiilitietoja paitsi käyttäjän sähköpostia.

Tulevien tilausten alareunassa on ’+’-nappi uuden tilauksen tekemiselle. Nappia painettua avautuu ruutu, jossa valitaan tilauksen ajankohta, kestääkö tilaus yön yli ja tapahtuuko se vanhemman vai lastenhoitajan luona. Tämän jälkeen lastenhoitajan luokse tehdyt tilaukset ohjataan karttanäkymään, jossa näkyy lastenhoitajien lukumäärän sisältäviä merkkejä alueilla, joissa on lastenhoitajia vapaana tilauksen määrittelemänä aikana. Merkkiä kosketettuaan avautuu lista alueella olevista vapaista lastenhoitajista. Näkymään päädyttäisiin suoraan, jos on määritelty tilauksen tapahtuvan vanhemman luona, jolloin näytettäisiin vanhemman kodin alueella työskentelevät vapaat lastenhoitajat. Listassa näkyy lastenhoitajien kuvat, arvostelut, tagit ja palkka. Listaprofiilista pääsee kuvassa 2 näkyvään hoitajan täyteen profiiliin, jossa näkyy kaikki listaprofiilissakin olevat tiedot, isompi kuva ja täysi kuvaus lastenhoitajasta, ja tilauksen vahvistukseen. Tilauksen vahvistuksessa maksetaan lastenhoitajan palkka ja siirrytään takaisin vanhemman aloitusruutuun.



KUVA 2. Alfa-vaiheen keskeneräinen lastenhoitajaprofiili

3.5.3 Lastenhoitajien käyttöliittymä

Lastenhoitajien näkymästä löytyy samanlainen valikko ja välilehtivalinta kuin vanhempien käyttöliittymästäkin. Samoin myös kaikista töistä pystyy aukaisemaan yhteenvedon ja tulevien töiden tiimoilta on chat-mahdollisuus vanhempien kanssa.

Vanhempien puolelta tuttujen ominaisuuksien lisäksi lastenhoitajat hallitsevat töitään. Uuden työn saatuaan lastenhoitajan tulee hyväksyä työ tai kieltäytyä siitä työn alla olevista napeista. Myös hyväksytyn työn voi peruuttaa myöhemmin. Jokaisesta työstä pääsee vanhemman profiiliin, josta näkyy heidän tietonsa ja lastensa tiedot.

4 LOOGINEN ARKKITEHTUURI

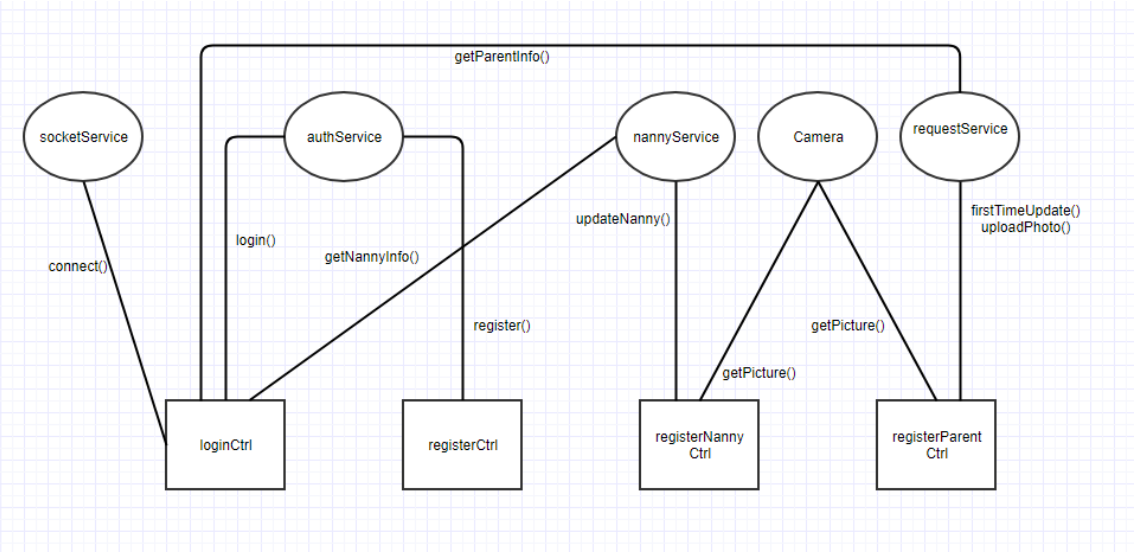
4.1 Yleistä arkkitehtuurista

AngularJS-applikaation loogiset osat ovat käsittelijät (engl. controller) ja palvelut (engl. service). Nämä yhdessä ovat MVC mallin käsittelijä. Käsittelijät ovat suorassa yhteydessä näkymäkomponentteihin. Yksi käsittelijä pystyy hoitamaan useaa näkymäkomponenttia, mutta yhdellä näkymäkomponentilla voi olla vain yksi käsittelijä. Palvelut taas ovat uudelleenkäytettäviä liiketoimintalogiikkakomponentteja, jotka eivät ole suorassa yhteydessä näkymään, vaan ovat käsittelijöiden käytössä. Niitä käytetään datan säilyttämiseen eri näkymien välillä, ulkoisista API:sta tai käyttäjältä saadun datan muokkaamiseen käsittelijän käyttöön ja monimutkaisten toiminnallisuuksien häivyttämiseen yksinkertaisen API:n taakse.

Seuraavissa kappaleissa on esitelty applikaation käsittelijät ja palvelut, sekä niiden väliset kutsut. Ne ovat ryhmitelty neljään eri osaan lukemisen helpottamiseksi. Alla olevissa kaavioissa ovaalit ovat palveluja ja nelikulmiot käsittelijöitä. Funktiokutsut ovat myös näytetty ilman parametrejä lukemisen helpottamiseksi.

4.2 Rekisteröityminen ja sisäänkirjautuminen

Rekisteröintiä ja sisäänkirjautumista hoitavat käsittelijät, palvelut ja niiden väliset kutsut näkyvät kuviossa 4. Sisäänkirjautumisnäkymää hoitaa loginCtrl-käsittelijä. Tämä käyttää authService-palvelua käyttäjän tunnistuksessa. Se myös asettaa oletustunnisteet seuraavia API-kutsuja varten, jotta backend tietää, että käyttäjä on kirjautunut sisään. Kirjautuessa sisään nannyService ja requestService hakevat käyttäjän tiedot ja sekä tulevat että menneet tilaukset. NannyService hakee nämä lastenhoitajille ja requestService vanhemmille. Sisäänkirjautuessa käyttäjä lisätään myös chat palveluun. Tästä huolehtii socketService. Lisäksi loginCtrl hoitaa automaattisen sisäänkirjautumisen, mikäli käyttäjä on sen sallinut.

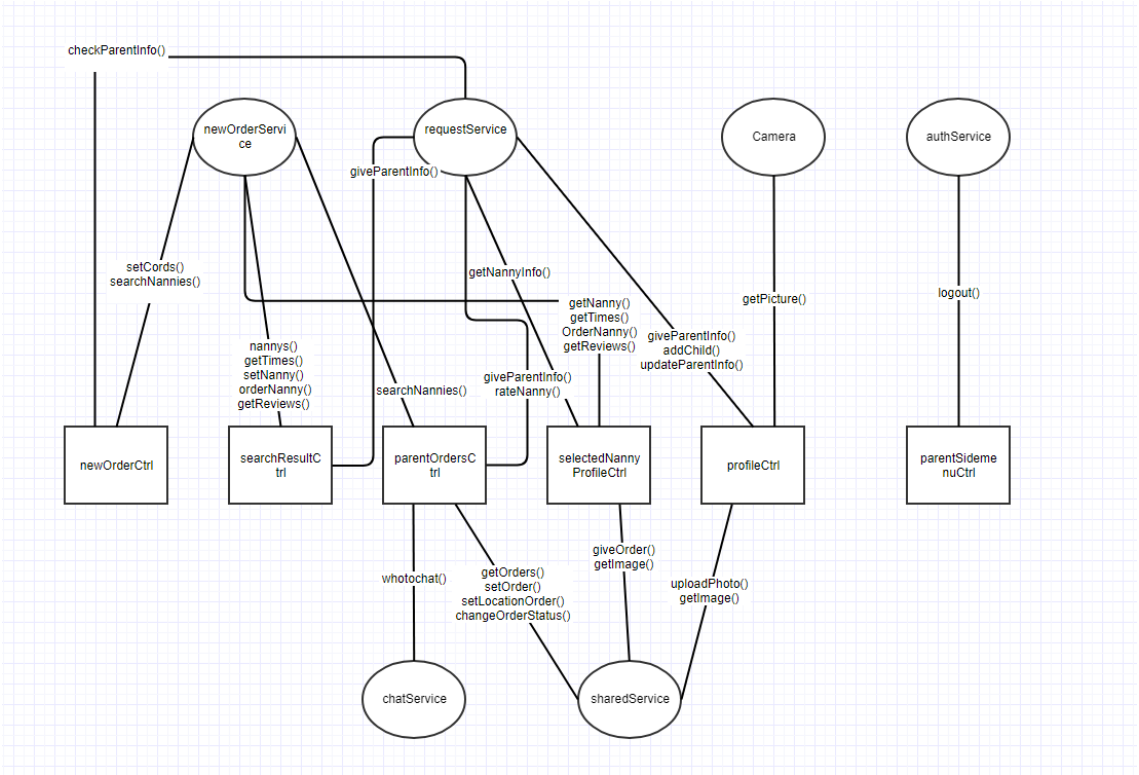


KUVIO 4. Rekisteröitymistä ja sisäänkirjautumista hoitavat käsittelijät ja palvelut

AuthService sisältää myös uuden käyttäjän luontiin tarvittavia API-kutsuja, joita registerCtrl-käsittelijä käyttää. Tästä siirrytään käyttäjän valinnan mukaan lastenhoitajan tai vanhemman tietojen täyttämiseen. Nämä käyttävät kuvion 4 osoittamalla tavalla palveluita päivittääkseen käyttäjän tiedot palvelimelle. Camera-palvelu hoitaa kameran operoinnin tai kuvatiedostojen haun laitteen muistista.

4.3 Vanhempien puoli

Vanhemmilla on kaksi omaa kuviossa 5 havainnollistettua palvelua nimeltään requestService, joka hoitaa vanhemman omien tietojen säilytyksen ja päivityksen, ja newOrderService, joka hoitaa lastenhoitajien haun ja tilauksen. Lisäksi vanhemmilla ja lastenhoitajilla on yksi yhteinen palvelu, sharedService, jonka tehtävänä on säilyttää käyttäjän tilaukset ja niiden muokkaamisen. Myös kuvien tallentaminen palvelimelle kuuluu sharedServicelle. ChatService-palvelu hoitaa chat-toiminnot käyttäjien välillä.



KUVIO 5. Vanhempiin vaikuttavat käsittelijät ja niiden käyttämät palvelut

Tulevien ja menneiden tilauksien näkymää hoitaa parentOrdersCtrl. Tämä käsittelijä saa käyttäjän tiedot requestServiceltä sekä tilauksien tiedot sharedServiceltä ja pystyy myös perumaan tilauksensa sen kautta. Tästä näkymästä myös aloitetaan lastenhoitajien haku, johon newOrderService erikoistuu. Lisäksi pystytään aloittamaan chat-keskusteluja tilattujen lastenhoitajien kanssa. Tällöin annetaan chatServicelle tieto, kenen kanssa halutaan keskustella.

SearchResultCtrl hoitaa hakukriteerit täyttävien lastenhoitajien näytön. Tämä hakee lastenhoitajien tiedot newOrderServiceltä sekä omat tietonsa requestServiceltä. Tästä näkymästä pystytään tilaamaan lastenhoitaja, sekä siirtymään näkemään lastenhoitajan koko profiilin.

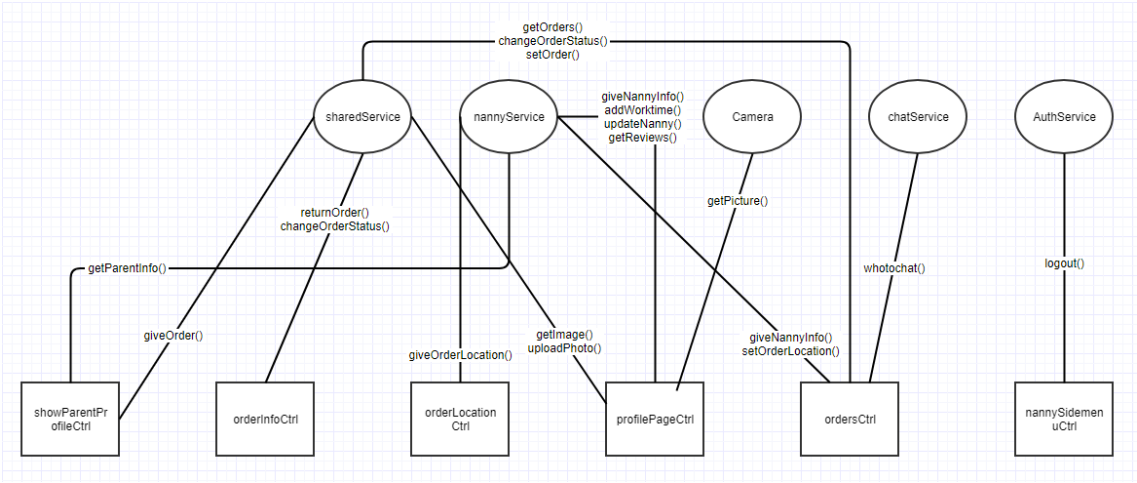
Halutessa katsella tarkemmin lastenhoitajan profiilia, mennään näkymään, jota hallinnoi selectedNannyProfileCtrl. Tähän näkymään pääsee uudessa tilauksessa tai vanhan tilauksen kautta. Käsittelijä hakee lastenhoitajan tiedot joko newOrderServicen kautta, mikäli on tultu lastenhoitajien haun kautta, tai requestServiceltä jos on tultu tilauksen kautta. Lisäksi käsittelijä muuttaa näkymää hieman, riippuen siitä kumpaa reittiä näkymään on tultu, esimerkiksi haun kautta tultaessa on näkymässä nappi, jolla voidaan tilata lastenhoitaja.

Käyttäjän profiilitietojen päivityksen näkymää hallinnoi profileCtrl-käsittelijä. Tämä hoituu requestServicen kautta, pois lukien kuvien lähettäminen palvelimelle, jonka hoitaa sharedService.

Viimeisenä vanhemman käsittelijänä on parentSidemenuCtrl, jonka tehtävä on hoitaa useassa näkymässä olevaa valikon toimintaa. Näistä suurin osa on navigointiin liittyviä tehtäviä, mutta käyttäjä pystyy sieltä kirjautumaan ulos järjestelmästä. Tällöin käsittelijä käyttää authServicen logout-funktiota tuhotakseen kirjautumistiedot.

4.4 Lastenhoitajien puoli

Kuviossa 6 näytetään lastenhoitajien puolen arkkitehtuuri. Osa toiminnallisuuksista on samankaltaisia vanhempien puolen kanssa. Lastenhoitajilla on yksi oma palvelu nimeltä nannyService, joka hoitaa lastenhoitajan profiilitietojen tallentamisen ja päivittämisen. Lisäksi pystyy hakemaan vanhemman tiedot tilauksen perusteella. SharedServicen toiminta-alue on sama kuin vanhemman puolella.



KUVIO 6. Lastenhoitajiin vaikuttavat käsittelijät ja niiden käyttämät palvelut

Tulevien ja menneiden tilauksien näkymän käsittelijänä toimii ordersCtrl. Tämän tehtäviin kuuluu tilauksien listaaminen, hyväksyminen ja peruminen. Tilaukset käsittelijä saa sharedServiceltä samalla tavalla kuin vanhempien parentOrdersCtrl. Lisäksi ordersCtrl

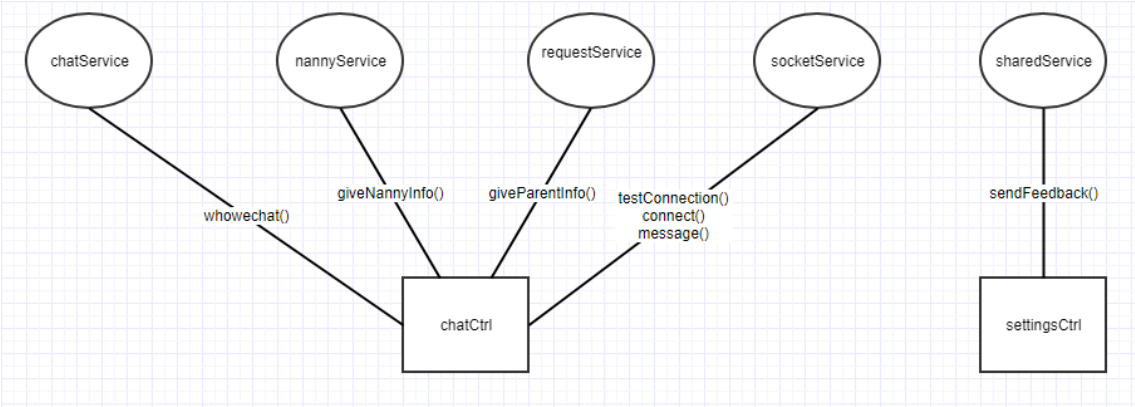
avaa modaalin, jota hallinnoi orderInfoCtrl. Tämän näkymän tehtävänä on näyttää tarkemmat tiedot tilauksesta sekä ohjata käyttäjä vanhemman profiiliin, jota hallinnoi showParentProfileCtrl. OrderInfoCtrlin hallinnoimasta näkymästä pääsee myös karttanäkymään, jota hallinnoi orderLocationCtrl, jossa näytetään tilauksen suorittamispaikka.

Käyttäjän profiilitietojen päivityksen näkymää hallinnoi profilePageCtrl-käsittelijä. Tämä hoituu nannyServicen kautta, pois lukien kuvien lähettäminen palvelimelle, jonka hoitaa sharedService.

NannySidemenuCtrlin tehtävä on sama kuin vanhemmalla, mutta navigaatiolinkit ohjaavat lapsenvahdin näkymiin. Myös uloskirjautuminen toimii samalla tavalla kuin vanhemmalla.

4.5 Yhteiset käsittelijät

Applikaatiossa on myös kaksi yhteistä käsittelijää, joilla on eri tehtävät. Nämä ovat kuvattu kuviossa 7



KUVIO 7. Yhteiset käsittelijät ja niiden käyttämät palvelut

Chat-näkymään mentäessä, chatCtrl hakee tiedot käyttäjästä joko nannyServiceltä, mikäli käyttäjä on lastenhoitaja, tai requestServiceltä, jos käyttäjä on vanhempi. Lisäksi se hakee tiedon kenen kanssa chatataan chatServiceltä, johon se on tallennettu juuri ennen chat näkymään mentäessä. Chat itsessään toimii socketein socketServicen kautta. Kun käyttäjä saapuu näkymään, testataan, onko hän yhteydessä chat-palveluun, johon liitytään jo sisäänkirjautuessa. Mikäli yhteyttä ei ole, yhdistetään chat-palveluun. Lähettäessä viestin

emitoidaan se socketin avulla palvelimelle, josta se lähetetään vastaanottajalle. Vastaanottajan socketService saa tapahtumasta ilmoituksen, ja ilmoittaa tällöin chatCtrille viestin sisällön, joka lisätään chat-historiaan ja näytetään käyttäjälle.

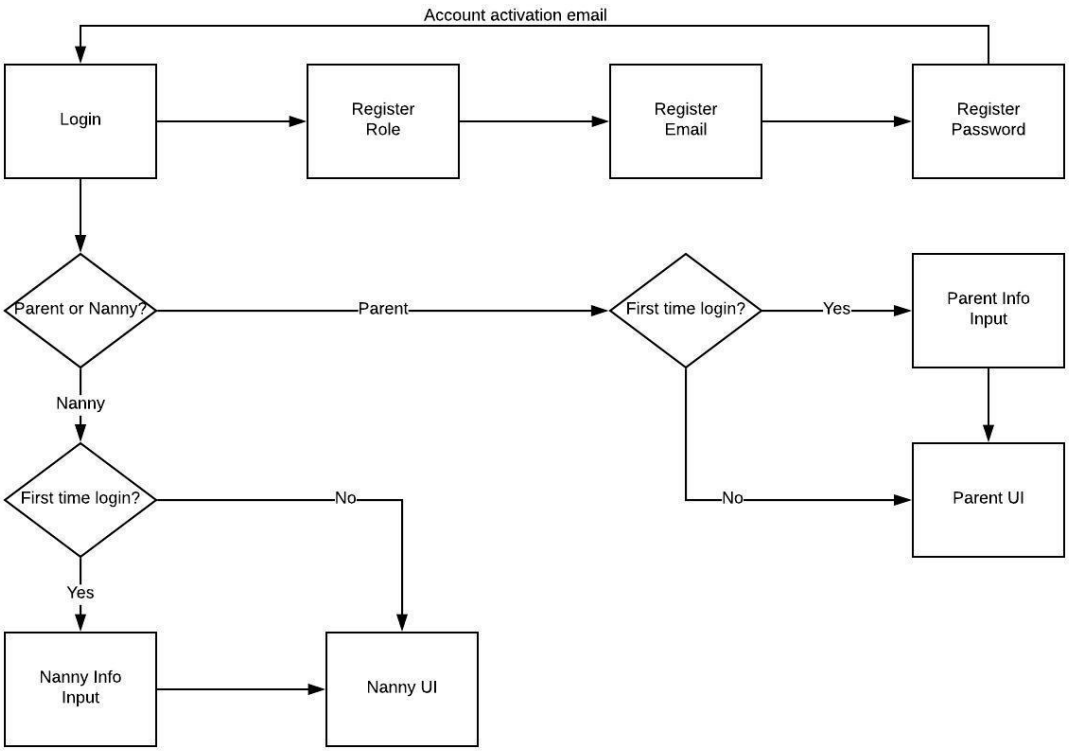
Applikaation asetuksissa pystytään muokkaamaan asetuksia, kuten haluaako käyttäjä kirjautua sisään automaattisesti. Lisäksi täältä pystyy ilmoittamaan ongelmista applikaation käytössä. Tämä käyttää sharedServicen sendFeedback-funktiota lähettääkseen palvelimelle viestin ongelmasta.

5 KÄYTTÖLIITTYMÄ

5.1 Sovelluksessa navigointi

Muxuni on laaja sovellus, jossa on kahden eri käyttäjäroolin omat käyttöliittymät ja molemmissa on monia näyttöjä. Sovelluksen navigointi on jaettu neljään eri osaan: kirjautuminen ja rekisteröinti, lastenhoitajan käyttöliittymä, vanhemman käyttöliittymä ja tilauksen tekeminen.

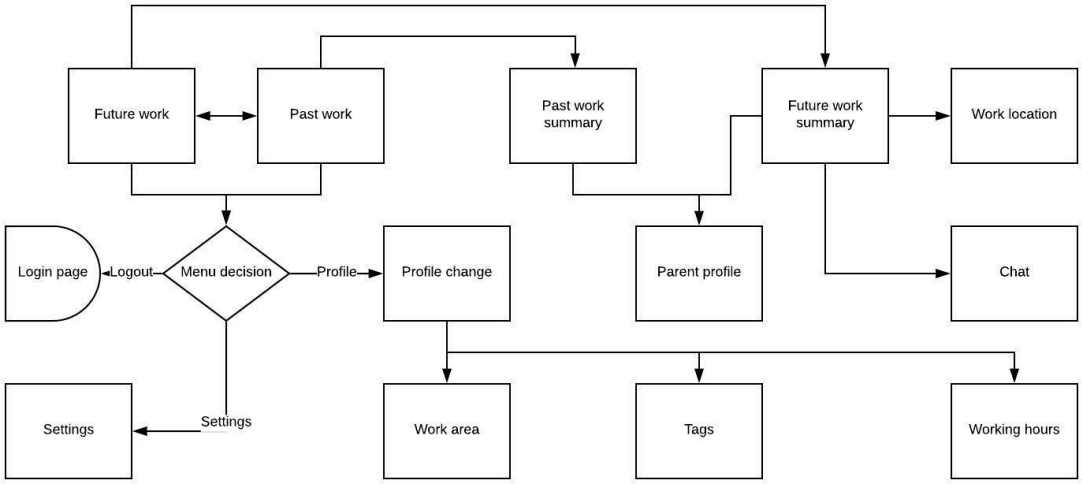
Ensimmäisenä sovelluksen käyttäjä näkee kirjautumissivun, josta kuvion 8 mukaisesti oikeilla tunnuksilla normaalisti siirryttäisiin roolin mukaisen käyttöliittymän puolelle tai voidaan siirtyä rekisteröintiin. Ensimmäisellä kirjautumisella tai jos käyttäjä ei ole aiemmin täyttänyt kaikkia tietojaan väliin tulee sivu, jossa käyttäjä täyttää tietonsa ensimmäistä kertaa profiiliinsa. Nämä tallennettuaan käyttäjä siirretään omaan käyttöliittymäänsä.



KUVIO 8. Kirjautumisen ja rekisteröinnin navigaatiokartta

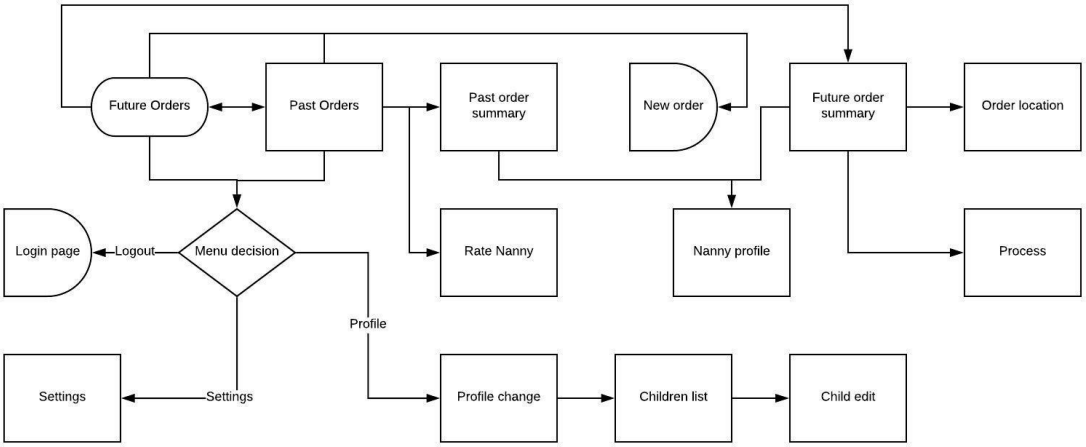
Kirjautumissivun alareunassa on teksti ”Eikö sinulla ole käyttäjää? Rekisteröidy täältä”, jota painamalla siirrytään kolmivaiheiseen rekisteröintiin, jossa on vastaava nappi teksti takaisin kirjautumissivulle siirtymiseen. Rekisteröinnissä on kolme sivua, joiden välillä pääsee kulkemaan, kun edellisen sivun tiedot on täytetty. Lopuksi käyttäjä palautetaan kirjautumissivulle, josta hän pääsee kirjautumaan, kun hän ensin aktivoi tilinsä saadustaan aktivointisähköpostin linkistä.

Lastenhoitajan ja vanhemman käyttöliittymät ovat pitkälti samanlaiset, kuten kuvioissa 9 ja 10 esitetään. Molemmissa ensimmäisenä nähdään työt, joista voi välilehdillä valita, katsooko tulevia vai menneitä töitä. Töistä saa esiin modaalin, jossa näkyy työn aika, paikka ja toisen osapuolen tiedot. Työn yhteenvedosta löytyy linkki toisen osapuolen profiiliin. Tulevista töistä myös voi navigoida chattiin toisen osapuolen kanssa ja katsomaan työn sijaintia.



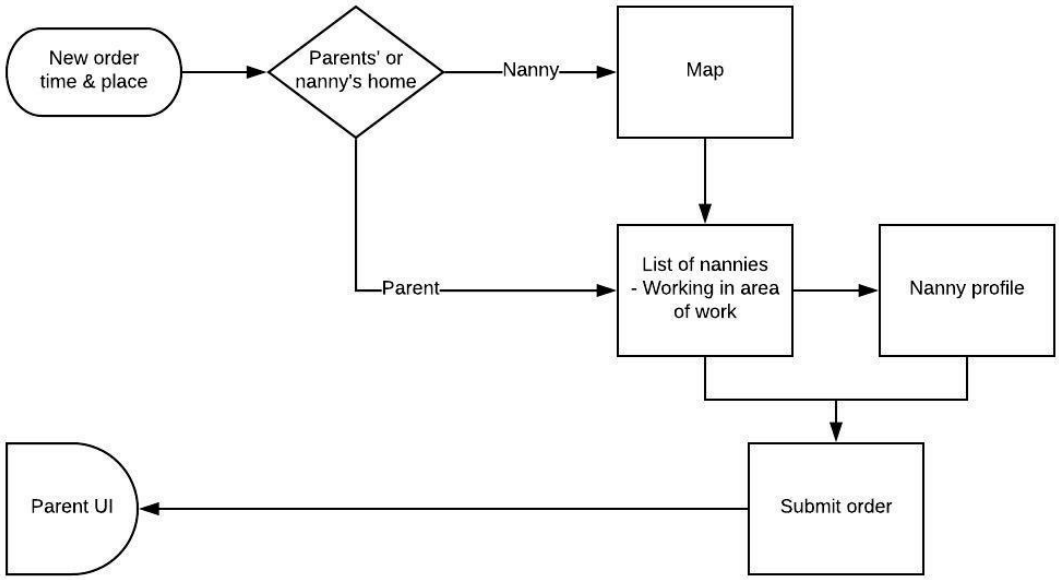
KUVIO 9. Lastenhoitajan puoli sovelluksesta

Molemmilla olevien työkorttiin ja -modaaliin liittyvien ominaisuuksien lisäksi molemmilla rooleilla on omia ominaisuuksia. Lastenhoitajalle on tärkeää pystyä hallitsemaan töitään, joten jokaisessa työssä on napit juuri siihen. Saadessaan uuden työn lastenhoitaja voi hyväksyä työn tai kieltäytyä siitä ja hyväksymisen jälkeen työ voidaan vielä perua. Vanhemmilla on myös oikeus perua työ ja sen lisäksi he voivat arvioida kerran jokaisen menneen työn.



KUVA 10. Vanhemman puoli sovelluksesta.

Molemmissa käyttöliittymissä työsivulla on yläkulmassa nappi, josta yläkulmaan aukeaa valikko. Valikosta palaamaan kirjautumissivulle kirjautumalla ulos, mennä muokkaamaan profiiliaan ja muuttamaan asetuksia, joita ei ole vielä tarkkaan määritelty. Profiilissaan molemmat roolit voivat vaihtaa kaikkia perustietojaan, paitsi nimeään tai sähköpostiaan. Lastenhoitajat voivat tämän lisäksi navigoida sivuille, joissa he muuttavat työaikojaan ja -alueitaan ja tagejaan. Vanhemmat pääsevät navigoimaan lastenhallintasivulle, josta he voivat muokata tai lisätä lapsen modaalissa.



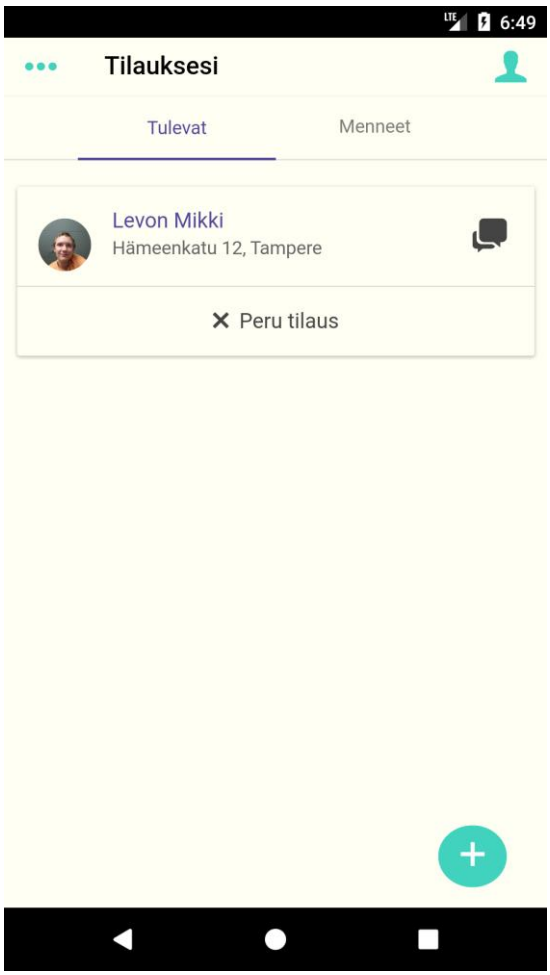
KUVA 4. Tilauksen tekeminen

Uusi tilaus tuo ensin ruudulle lomakkeen, jossa valitaan kestääkö työ yön yli, sen aloitus ja tarvittaessa lopetus päivämäärä ja aloitus- ja lopetusajat. Lopuksi vielä valitaan, halutaanko lastenhoitaja vanhempien kotiin vai muualle. Valittaessa muualle lomakkeen hyväksyminen tuo ruudulle kartan, missä näkyy merkeillä yhteen lastenhoitajien määrä eri alueille, lastenhoitajista ei tässä kohtaa näytetä tarkkaa sijaintia. Merkkiä koskettamalla ruutuun tulee lista alueella olevista lastenhoitajista. Kun halutaan lastenhoitaja vanhemman kotiin, ruutuun tulee suoraan lista lastenhoitajista, jotka työskentelevät vanhemman kodin alueella. Listoissa näkyy vain tilauksen hetkellä vapaana olevat lastenhoitajat. Listalta näkee nopeasti lastenhoitajan nimen, kuvan, arvostelut, palkan ja tagit. Tästä (kuten myös tilauksen koosteesta) pääsee selaamaan lastenhoitajan julkista profiilia. Profiilissa näkyy lisäksi vielä tarkempi kuvaus lastenhoitajasta. Sopivan lastenhoitajan löydettyään käyttäjä tekee tilauksen, mikä tuo esiin maksutapahtuman ja palauttaa lopulta käyttäjän tilausnäkymään.

5.2 Käyttäjäkokemus

Muxuni on teknisten taitojen suhteen hyvin eritasoisten käyttäjien käytössä eikä se saa pelottaa ketään pois sovelluksen ääreltä. Siksi sovellus on suunniteltava helpoksi ja intuitiiviseksi käyttää. Tässä osiossa sovelluksen toimintaa peilataan Linda Clairen kirjoittamassa blogissa oleviin kymmeneen interaktiosuunnittelun periaatetta vasten (Claire, 2017).

Sovelluksen käyttöliittymän suunnittelussa tulisi huomioida, että suurin osa käyttäjistä käyttää sovelluksia oman vaiston varassa ilman minkäänlaista ohjetta. Tällöin kaikkien interaktiota tarjoavien elementtien pitäisi toimia tavalla, jolla käyttäjä olettaa sen toimivan. Muxunissa käytetään nappeja, jotka joko suoraan kertovat tekstinä, mitä niistä tapahtuu, tai ovat ymmärrettäviä ikoneita. Esimerkiksi kuvassa 3 näkyy, että vanhemman tilaussivulla on neljä ikonipainiketta, joista on helppo päätellä niiden tarkoitus. Ylävasemman kolmesta pisteestä avautuu ”lisää vaihtoehtoja” eli valikko, yläoikean kulman hahmosta avautuu profiilin muokkaus, tilauksen puhekuplista pääsee chattiin ja alaoikeassa oleva huomiota kiinnittävä ’+’-painike aloittaa uuden tilauksen (Claire, 2017).

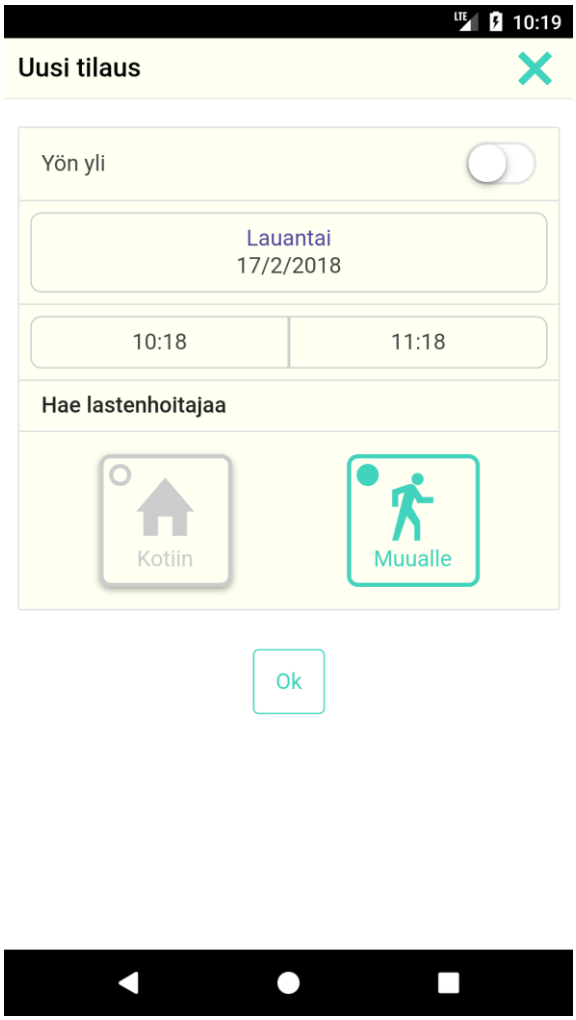


KUVA 3. Vanhemman käyttöliittymän tilaussivu

Toinen interaktiosuunnittelun periaate on kohdata käyttäjän tarpeet. Sovelluksen pitäisi täysin pohjautua käyttäjäkuntansa tarpeisiin ja mahdollisimman tehokkaasti tyydyttää ne. Muxunin koko määrittelyprosessi on lähtenyt yhden tiimiläisen omasta tarpeesta ja vielä sen jälkeen tehdystä kyselystä lastenhoidon tarpeesta. Myös sovelluksen suunnittelussa on ollut tärkeässä osassa käyttäjän tarpeet. Esimerkiksi päätettiin, että sovellus tarvitsee luotettavan tunnistautumisen, koska ihmiset eivät halua antaa lapsiaan kenenkään käsiin varmistamatta, kuka heitä hoitaa (Claire, 2017).

Sovelluksien käyttökokemusta parantaa konsistenssi eli pysyvyys kaikilla osa-alueilla, kuten performanssissa, ulkoasussa ja interaktioiden tuntumassa. Muxunissa käyttöliittymä on kaikkialla suhteellisen samantyyppinen. Esimerkiksi kaikki modaalien ja korttien alareunoissa olevat napit ovat tekstillisiä, mutta muutoin sovelluksessa käytetään pelkkiä ikoneita nappeina. Otsake näkyy aina kirjautuneena ja näyttää samalta kaikkialla (Claire, 2017).

Kolmantena periaatteena on esitelty ajatus ”vähemmän on enemmän”, jolla tarkoitetaan, että hyvässä interaktiosuunnittelussa koitetaan vähentää käyttäjän kognitiivista ja operatiivista kuormaa. Muxunissa on aina pyritty siihen, että yhdellä sivulla on mahdollisimman vähän vastuita ja tällöin kaikki informaatio pidetään minimissään. Esimerkiksi kuvan 4 esittelemällä sivulla on hyvin vähän dataa selkeästi aseteltuna käyttäjän käytettäväksi (Claire, 2017).

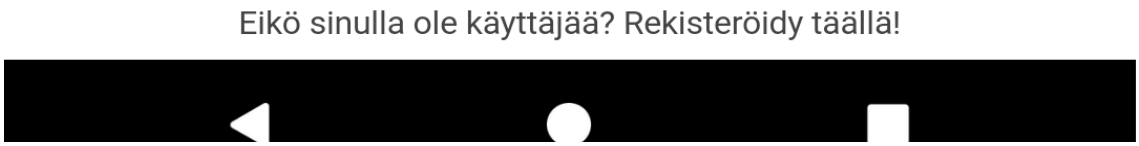


KUVA 4. Uusi tilaus -sivu Muxunissa

Käyttäjäystävällisesti suunnitellun kuluttajasovelluksen tulisi välttää teknistä jargonia ja yksinkertaistaa sovelluksessa käytettyä kieltä. Alfa-vaiheen sovelluksessa tässä aiheessa olisi vielä parantamisen varaa. Muxunin profiilien muokkauksessa lukee tallennuksessa vielä ”Lähetä” viitaten, että uudet muokatut tiedot lähetetään palvelimelle, kun käyttäjälle luonnollisempaa olisi lukea sana ”Tallenna” ja ajatella, että muutokset tallennetaan johonkin. On sovelluksessa kuitenkin jo monessa paikassa huomioitu ymmärrettävä kieli, kuten kuvan 3 ”Peru tilaus”-napissa (Claire, 2017).

On houkuttelevaa tehdä sovelluksesta graafisesti näyttävä uhraten joitakin käyttäjäystävällisyyden osa-alueita, mutta Linda Claire blogissa kuudes periaate ohjeistaa juuri tällaista ajatusmallia vastaan. Periaatteen kaltainen ajatustapa on ollut Muxunin toteutuksessa helppoa, sillä kehittäjätiimin käyttäjäkokemuksesta vastaava on nimenomaan kiinnostunut käyttöliittymäsuunnittelusta eikä grafiikasta. Sovelluksessa on käytetty laajalti muissa sovelluksissa käytettyjä sommittelumalleja ja mustan ja valkoisen lisäksi sovelluksen väriteemassa on kaksi muuta väriä käytössä (Claire, 2017).

Sovelluksen ei kuuluisi vaatia käyttäjältään juurikaan ajattelua. Jos on tilanne, jossa intuitiivinen ikoni ei pysty kertomaan kaikkea, asiat, kuten virheilmoitukset, napit ja syötteet, kuuluu kirjoittaa selkeällä kielellä. Kuvassa 5 näkyy Muxuni-sovelluksen rekisteröitymisnappi, jossa selkeästi ilmaistaan, mistä voi luoda uuden käyttäjän (Claire, 2017).



KUVA 5. Sovelluksen rekisteröitymislinkki kirjautumissivun alareunassa.

Sovelluksissa on huomioitava, että toiminnan kannalta tärkeät toiminnot on jotenkin nostettu esiin muusta käyttöliittymästä. Tämä on Muxunissa huomioitu esimerkiksi kuvan AAA ’+’-napin värivalinnassa, kun muut napit sivulla ovat harmaita ikoneita ja lisäysnapilla on pastellinvihreä pyöreä tausta. Sovelluksessa myös hyödynnetään oman väriteemansa lisäksi punaista ja vihreää hyväksy/peruuta-tilanteissa, joissa peruutus esitetään punaisena ja hyväksyminen vihreänä (Claire, 2017).

Ohjelmistokehityksessä on huomioitava mahdolliset virhetilanteet ja olla rajoittamatta käyttäjän tekemistä virhetilanteiden pelossa. Virhetilanteissa pitää osoittaa selkeästi, mitä on tapahtunut ja miksi, ja osoittaa käyttäjä taas oikealle tielle. Samaan teemaan myös kuuluu varmistusdialogin esittäminen kriittisten toimintojen kohdalla. Muxunissa ei hirveästi virhetilanteita tapahdu, mutta mahdollisista yhteysongelmista on tehty ilmoitukset ja väärien kirjautumistietojen syöttämisestä annetaan selkeä virhe kirjautumiskenttien

yläpuolelle. Sen sijaan Muxunissa on monta kriittistä tilannetta, kuten tilaamisen hyväksyminen ja työn peruminen, joissa annetaan varmistusdialogi, jossa voidaan vielä perua toiminto (Claire, 2017).

Viimeisenä blogin periaatteena on palautteen anto käyttäjälle tilanteista. Tähän kuuluu yllämainittujen virheilmoitusten lisäksi esimerkiksi valitut vaihtoehdot ja toiminnan toteutumisen varmistus. Muxunissa on esimerkiksi kuvassa 5 näkyvä ”Kotiin/Muualle”-valinta, jossa valittu vaihtoehto on korostettu vihreällä värillä ja sovellus myös tilauksen tehtyä ilmoittaa käyttäjälle onnistumisesta ja siirtää hänet aloitussivulle, jossa tilaus on lisätty listaan (Claire, 2017).

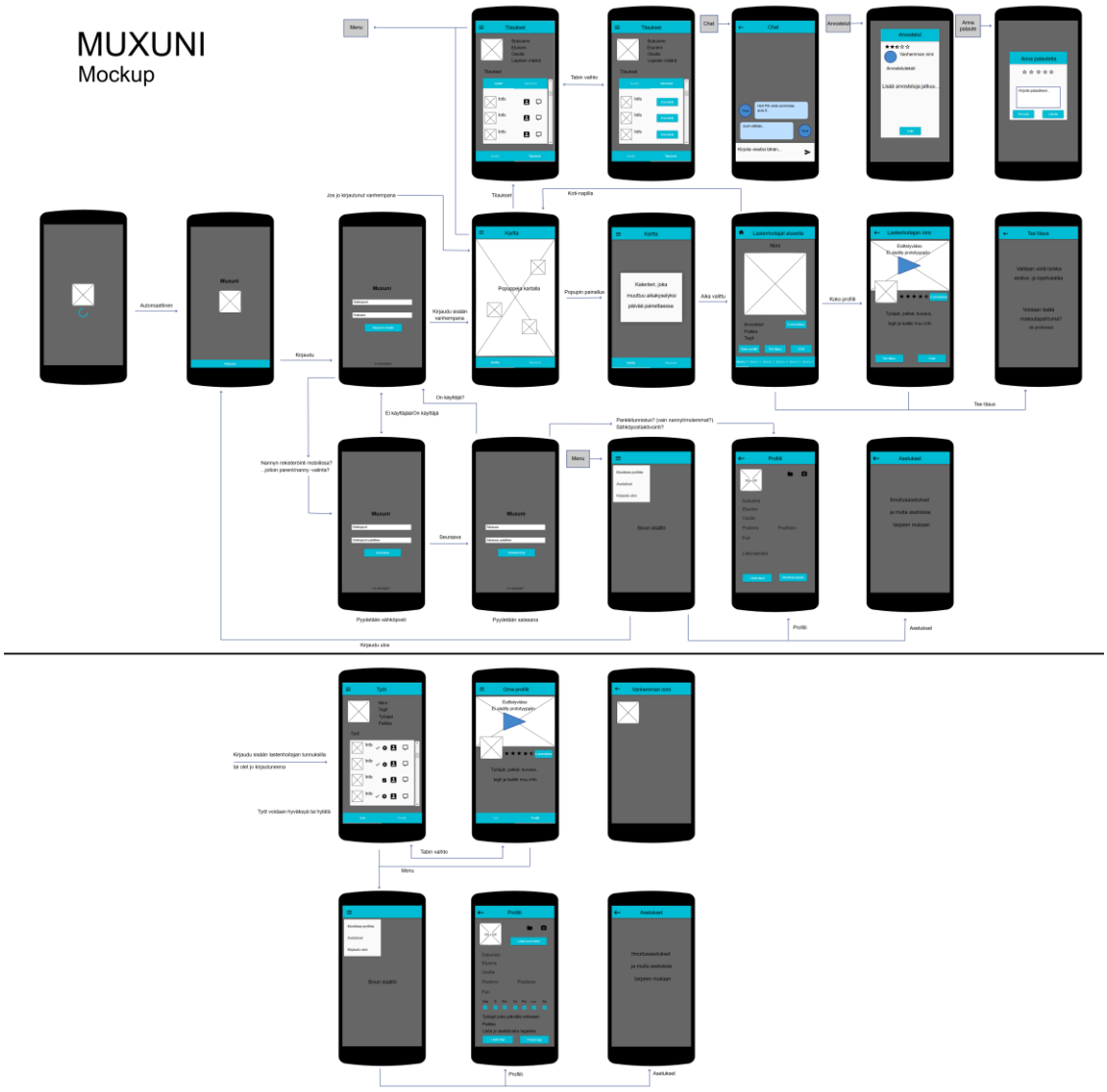
6 TOTEUTUSPROSESSI

6.1 Suunnittelu

Projektin alkaessa sovelluksen suunnittelu jäi hyvin vähäiseksi, koska kehittäjillä ei ollut kokemusta sovellusarkkitehtuurin suunnittelusta eivätkä he tunteneet käytettäviä teknologioita. Suunnittelu rajautui teknologioiden valintaan ja siitä siirryttiin suoraan teknologioihin tutustumiseen ja kokeilun kautta toteutukseen.

Pienen teknologioihin tutustumisen jälkeen sovelluksen arkkitehtuuri alkoi muovautumaan automaattisesti teknologian asettamien mahdollisuuksien ja rajoitusten mukaiseksi ja nämä mallit, kuten AngularJS:n käyttämä MVC-malli, otettiin käyttöön kaikkialla sovelluksessa. Arkkitehtuuri on pysynyt vakiona, mutta siihen on lisätty erilaisia osia tarpeen mukaan ja mukautettu sopivaksi sovelluksen arkkitehtuurimallille.

Sovelluksella ei ollut alkuun myöskään tarkkaa määrittelyä navigaation suhteen, vaan määrittely muodostui varhaisessa vaiheessa toteutuksen yhteydessä. Ensimmäinen, kuvassa 11 näkyvä, esityksellinen suunnitelma tehtiin muutaman kuukauden aloituksen jälkeen ja se sisälsi sovelluksen navigaatiologiikan ja käyttöliittymäsuunnittelua. Ensimmäisen suunnitelman jälkeen on tullut tarpeita lisätä sivuja sovellukseen ja käyttöliittymän asettelua on muutettu, mutta tämäkin suunnitelma on ollut hyvä pohja Muxunille.



KUVIO 11. Muxunin ensimmäinen esityksellinen suunnitelma

Nykytilanteessa sovelluksen vaatimukset ovat hyvin selvillä ja tiedetään mitä mihinkin vielä tarvitaan sovelluksen julkaistavaan versioon. Kehittäjätiimin toiminta on jo muovautunut vaatimusten selkiytymisen myötä määrätietoisemmaksi ja selkeämmäksi.

6.2 Kehittäminen

Projektin alussa myös sovelluksen kehittäminen usean henkilön voimin tuotti vaikeuksia versionhallintataitojen puuttuessa. Git on todella tehokas työkalu, mutta vain jos sitä osaa käyttää oikein. Jokaisella kehittäjällä oli oma nimikoitu Git-haara sovelluksesta ja näitä yhdistettiin (engl. merge), kun tarvittiin toisen kehittäjän tuottamia toiminnallisuuksia

omaan haaraan. Tämä tuotti ongelmia konfliktien suhteen ja päähaaralle (master) ei tullut ollenkaan uusia toiminnallisuuksia.

Myöhemmin projektissa otettiin käyttöön parempi työnkulku. Tässä jokaiselle uudelle toiminnallisuudelle tai korjaukselle tehdään uusi haara, ja valmistuessaan yhdistetään päähaaraan. Myös jokaisen haaran alku piste on päähaara. Tällöin usean kehittäjän työskentely keskenään helpottuu ja yhdistämiskonfliktien määrä pienenee. Lisäksi päähaaralla pyritään pitämään toimiva ohjelmakoodi ja se pitää sisällään vain valmiita toiminnallisuuksia.

7 JATKOKEHITYS

7.1 Testauksen vaiheet

Muxuni on tällä hetkellä alfa-vaiheessa. Tärkeimmät asiat, kuten rekisteröinti molempiin rooleihin, pääasialliset profiilit, tietojen täyttö, tilaaminen ja töiden käsittely toimivat tasolla, jolla sovellusta voidaan testata kontrolloiduissa ympäristössä. Testaajat ovat sovelluksen kehittäjät itse.

Beta-vaiheeseen vaaditaan ainakin maksujärjestelmän valitsemisen ja käyttöönoton, kuittien näkemisen molemmille osapuolille, varsinkin lastenhoitajille verotuksellisista syistä, sekä chatin uudelleen suunnittelun ja ohjelmoinnin. Lisäksi beta-vaiheessa täytyy tarkastaa sovelluksen ja koko järjestelmän haavoittuvuus, sillä järjestelmä sisältää arkaluontoisia tietoja.

Mahdollisia beta-testaajia on ilmoittautunut useita molemmista ryhmistä. Tämän takia sovellus yritetään saada mahdollisimman nopeasti tähän pisteeseen, jotta päästään testaamaan oikeilla käyttäjillä, sekä saamaan heiltä palautetta sovelluksesta.

7.2 Julkaisu

Julkaisu on tavoitteena toteuttaa vuonna 2018, kunhan beta-vaiheen toiminnallisuudet ovat saatu valmiiksi, sovelluksen graafisesta ilmeestä tehty mahdollisimman helppo ja miellyttävä, sekä paikattu sovelluksen mahdolliset haavoittuvuudet. Sovellus tullaan julkaisemaan Google Playssa sekä App Storessa.

7.3 Ylläpito ja tulevaisuus

Sovellus tullaan ensin ottamaan käyttöön Tampereen alueella ja tarkoitus on hankkia hyvä käyttäjäkunta, jotta sovelluksesta tulee käyttökelpoinen. Tämän jälkeen on tarkoitus laajentaa muihin Suomen kuntiin. Ylläpidollisesti tämä tarkoittaa helppoa aikaa, jollei

vastoinkäymisiä satu. Tällöin voidaan jatkaa sovelluksen kehittämistä ja uusien toimintojen suunnittelua. Suurin vastuu tässä vaiheessa on markkinoinnilla, jotta saadaan sovellukselle näkyvyyttä ja useammat voivat alkaa käyttämään sovellusta.

8 POHDINTA

Projektin tavoitteena on kehittää mobiilisovellus, joka auttaa vanhempia löytämään lastenhoitajan nopeasti ja vaivattomasti. Molemmat osapuolet voivat käyttää samaa sovellusta toistensa kanssa asioimiseen.

Sovellus tehdään käyttäen Ionic-ohjelmistokehystä, jolla voidaan tehdä webohjelmoinnin tekniikoin mobiilialustoille sovelluksia yhdellä ohjelmistokoodilla. Tämä ohjelmistokehys valittiin, jotta saadaan nopeammin kehitettyä usealle alustalle ja täten saada sovellus laajemmille markkinoille.

Sovellus on alfatestausvaiheessa, jossa testataan sovelluksen tärkeimmät toiminnot, kuten lastenhoitajien etsiminen ja tilaamine. Lisäksi uusia toimintoja kehitetään testauksen lomassa. Sovellus julkaistaan Androidille sekä iOS laitteille vuoden 2018 aikana.

Aikaisen vaiheen tarkka vaatimus- ja toiminnallinen määrittely olisivat nopeuttaneet sovelluksen kehittämistä. Tämä voidaan selittää kehittäjien kokemattomuudella suurien järjestelmien tekemisestä. Kuitenkin kehittäjien taidot ovat kaikin puolin kehittyneet ja kehitysnopeus on parantunut.

LÄHTEET

2Care4Kids Group. 2018. Lisää Babysitter.fi-palvelusta. 2018. Viitattu: 30.1.2018. https://www.babysitter.fi/tietoa-babysitter

AngularJS Github. AngularJS Github. Viitattu: 20.1.2018. https://github.com/angular/angular

Chacon, Scott ja Straub, Ben. 2014. *Pro Git.* Toinen painos. Apress, 2014. sivut. 5 - 11. ISBN 1484200772.

Chrome. MVC Architecture. Viitattu: 23.1.2018. https://developer.chrome.com/apps/app_frameworks

Claire, Linda. 2017. 10 Basic Interaction Design Principles to Boost the UX Design. 2017. Viitattu: 16.2.2018. https://blog.prototypr.io/10-basic-interaction-design-principles-to-boost-the-ux-design-c05afbaf2068

Collins, John. 2018. The Basics of Web Technologies. 2018. Viitattu: 8.2.2018. http://www.alphadevx.com/a/7-The-Basics-of-Web-Technologies

Drifty. 2016. Core Concepts. 2016. Viitattu: 5.2.2018. https://ionicframework.com/docs/intro/concepts/

Drifty. 2016. Ionic Documentation Overview. 2016. Viitattu: 5.2.2018. https://ionicframework.com/docs/v1/overview/

Google Maps. About. *Google Maps.* Viitattu: 14.2.2018. https://www.google.com/maps/about/

Hoivanet. 2018. Hinnasto. 2018. Viitattu: 30. 1.2018. https://www.hoivanet.fi/hinnasto/

Lourenço, Alexandre Eleutério Santos. 2014. AngularJS. *Technology Explained.* 22. 10 2014. Viitattu: 23.1.2018. https://alexandreesl.com/tag/angularjs/

Mozilla. 2018. Getting started with HTML. 2018. Viitattu: 9.2.2018. https://developer.mozilla.org/en-US/docs/Learn/HTML/Introduction_to_HTML/Getting_started

Mozilla. 2017. How CSS works. 2017. Viitattu: 9.2.2018. https://developer.mozilla.org/en-US/docs/Learn/CSS/Introduction_to_CSS/How_CSS_works

Mozilla. 2018. Introduction. 2018. Viitattu: 9.2.2018 https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Introduction

Taylor, Bret. 2005. Mapping your way. *Google Official Blog.* 8. 2 2005. Viitattu: 14.2.2018 https://googleblog.blogspot.fi/2005/02/mapping-your-way.html

Tilastokeskus. 2017. Äidit tilastoissa 2017. 2017. Viitattu: 24.1.2018. https://www.stat.fi/tup/tilastokirjasto/aidit-tilastoissa-2017.html

Treamer Oy. 2017. Näin löydät apulaisia Treamerin avulla. 2017. Viitattu: 30.1.2018. https://www.treamer.com/ohjeet

Työ- ja elinkeinoministeriö. 2018. Aktiivimalli - usein kysyttyjä kysymyksiä. 2018. Viitattu: 15.2.2018. http://toimistot.te-palvelut.fi/uusimaa/aktiivimalli

Virinchy, P. 2014. What is Cordova and how does it work ? *SAP*. 27. 7 2014. Viitattu: 23.1.2018. https://blogs.sap.com/2014/07/27/what-is-cordova-and-how-does-it-work/

Viskari, Mikko. 2015. Tilaisitko yhden sivun web-sovelluksen (SPA) vai arkkitehtuuriltaan valmiiksi vanhentuneen järjestelmän? *Eatech.* 18. 11 2015. Viitattu: 14.1.2018. https://www.eatech.fi/spa-sovellukset/

LIITTEET

Liite 1. Muxuni – Research

1 (4)

MUXUNI

Agenda

- **About MUXUNI**
- **Background and Methodology**
- **Summary of results**
- **Detailed results**
- **Council**
- **Denial (disclaimer)**

(jatkuu)

Methodology

- I have built up a team in our university (TAMK) and together we have started to develop a mobile application for nannies and families. The name of the app is called: MUXUNI.
- In April 2016 MUXUNI conducted a survey for a demand of babysitter.
- **480 parents** from different province in Tampere participated in the survey.
- **95% of participants** were members of different child cares, for example MLL (Mannerheimin Lastensuojeluliijon) and Huusholli. The other 5% were part of my other online survey.
- **65% are from Tampere**, 15% are from Nokia, 6% from Ylöjärvi and 5% from Kangasala, 3% from Pirkkala and the rest of 3 % are from Lempäälä

Definition

Definition:

Babysitter **(Lapsenvahti),** is a service to take care of a child (or more children) during the day offered by a person which most often is someone outside the child's immediate family other than the child's legal guardians.

Summary: Babysitters

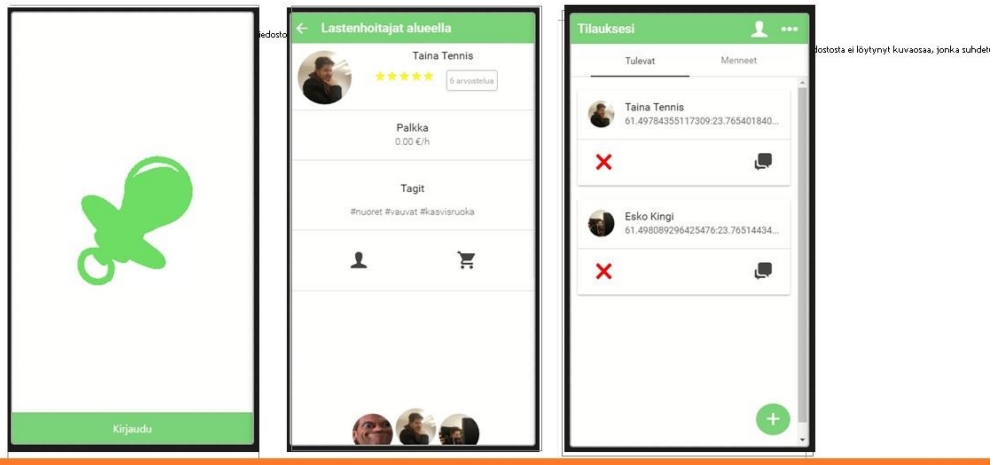
- **64% (more than the half)** of the parents which have been interviewed, they have used once – twice a month over the last 9 months , a babysitter. 32% of them have used a babysitter more often (twice a week or even more), whereas 4% have order a babysitter not at all.
- **More than 55% of the parents use a babysitter on weeknights** and the rest use a sitter on Saturdays, for **"parents – date –night".**
- **Our research showed that four in ten (38%) of the babysitters get payed around 70€ or even** more on a typical night.

Summary: Pay

- **64% (more than the half)** of the parents which have been interviewed, they have used once – twice a month over the last 9 months , a babysitter. 32% of them have used a babysitter more often (twice a week or even more), whereas 4% have order a babysitter not at all.
- **Babysitters average cost:**
  - Babysitter from 16 years to 20 years 8,20 € - 16,40 €
  - Babysitter from 21 years to 25 years 16,40 € - 24,00 €
  - Babysitter over 26 years 24,00 € - more
- Parents pay for babysitters under 20 years less per hour than those over 21 years.
- The average salaries in Finland for babysitters are around 8 € to 16 €.

Product

LOG IN & SEARCH BY CITY ⟶ CHECK PROFILE ! ⟶ REVIEW LISTINGS



Summary: Pay