TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Tarkvarateaduse instituut

Marian Kuklane 164919IABB

AUTO HOOLDUSAJA BRONEERIMISE MOBIILRAKENDUSE ARENDAMINE

Bakalaureuse töö

Juhendaja: Jekaterina Tšukrejeva
Magistrikraad

Tallinn 2019

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Marian Kuklane

19.05.2019

Annotatsioon

Antud lõputöö eesmärgiks on luua auto hooldusaja broneerimise mobiilrakendus iOS platvormile. Eestis saab auto hooldusaega broneerida ainult läbi veebikeskkonna, mis eeldab igakordset auto- ja kasutajaandmete sisestamist. Kasutaja teekond broneeringu kinnitamiseni on ajamahukam ning ajakonflikti esinemisel eeldab vahetut suhtlust esindusega. Mobiilrakendusega soovitakse kasutajatele teha kiirem, mugavam ja igal ajahetkel kättesaadavam teenus. Antud mobiilrakendus on mõeldud margiesindustele ja nende klientidele. Töö tulemusena valmib mobiilrakenduse prototüüp, mis koosneb kasutajaliidesest, andmebaasist ja serverrakendusest.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 42 leheküljel, 6 peatükki, 12 joonist, 1 tabelit.

Abstract

Mobile application developing for automotive service reservations

The goal of this thesis is to create a mobile automotive service reservation application for iOS platform. In Estonia car service can be booked only via web service that requires car and user specific data each time. Steps to reservation confirmation are more time-consuming and in case of booking conflicts direct communication between client and service provider is needed. With the mobile application, users want to make the service faster, more convenient and more accessible at any given time. As a result of the work, a mobile application prototype consists of a user interface, a database and a server application.

The thesis is in Estonian language and contains 42 pages of text, 6 chapters, 12 figures and 1 table.

Lühendite ja mõistete sõnastik

PHP	*Hypertext Preprocessor* - Skriptimiskeel serveripoolsete rakenduste loomiseks
IDE	Integrated development environment – Integreeritud programmeerimiskeskond
URL	Uniform Resource Locator – Universaalne ressursilokaator
Javascript	Kõrgtasemeline, dünaamiline ja nõrgalt tüpiseeritud skriptimiskeel, mis on standardiseeritud ECMAScript-i poolt
MySQL	Relatsioonile andmebaasi haldamise süsteem, millesse saab päringud esitada standardiseeritult
iOS	Apple'i arendatav mobiilsete seadmete operatsioonisüsteem
UI	User Interface - Kasutaja liides
Https	Hypertext Transfer Protocol Secure – Turvaline hüperteksti edastusprotokoll
IAAS	Infrastructure as service - Infrastruktuur kui teenus
PAAS	Platform as service - Platvorm kui teenus
FQDN	Fully Qualified Domain Name – Täielikult kvalifitseeritud domeeninimi
APT	Advanced Packaging Tool – Täiustatud pakendi tööriist

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Eestis tegutsevatel auto margiesindustel puudub hooldusaja broneerimise mobiilrakendus. Valdaval enamusel margiesindustel on küll hooldusaja broneerimiseks kasutusel veebikeskkond, kuid antud teenus eeldab igakordset auto- ja kasutajaandmete sisestamist. Kasutaja teekond broneeringu kinnitamiseni on ajamahukam ning ajakonflikti esinemisel eeldab vahetut suhtlust esindusega. Alternatiivvõimalus auto hooldusaja broneerimiseks on esindusse või töökotta helistamine. Suured autotootjad nagu Audi, BMW, Tesla, Maserati, Volkswagen, Subaru ja Toyota omavad lisaks veebikeskkonnale ka auto mobiilrakendust, kuid rakenduse eesmärk on auto hetkeseisu monitoorimine. Mobiilrakenduste tähtsus autotööstuses on tõusuteel, seega sai loodud vastav auto hooldusaja broneerimise mobiilrakendus.

Antud töö eesmärgiks on luua toimiv auto hooldusaja broneerimise mobiilrakendus, et kasutaja jaoks oleks hooldusaja broneerimine igal ajahetkel mugav ja lihtne. Kasutajad saavad luua konto, mille tulemusena salvestatakse kliendi andmed andmebaasi. Selline funktsionaalsus tagab, et klient ei peaks iga kord auto hooldusaja broneerimisel nii enda kui ka auto andmeid uuesti lisama. Juhul kui kasutaja on unustanud salasõna, saab turvaküsimusele vastates parooli uuesti seadistada. Lisaks on kasutajal võimalus muuta enda andmeid peale registreerimist. Samuti on kasutajal võimalus vaadata nii tellitud tööde arhiivi kui ka tulevasi töid. Mobiilrakendusega peab kasutaja saama valida tehtava töö liigi: kere poleerimine, rehvide vahetus, korraline hooldus, salongi puhastus, käsipesu ja õlivahetus. Lisaks peab saama valida sobivat töökoda ning täita kontaktandmeid nii auto kui kliendi kohta. Päringu tegemisel peab süsteem verifitseerima sisestatud andmeid ning vajadusel kontrollima andmebaasi sisestatud väärtusi. Mobiilrakendus on mõeldud konkreetse esinduse või auto töökoja jaoks. Hetkel on loodud mobiilrakendus prototüübiks ja ühiseks platvormiks, mille pealt saab arendada konkreetsele kliendile omanäolise auto hooldusaja broneerimise mobiilrakenduse. Lisaks on töö eesmärgiks luua kasutajaliides, mis peaks olema intuitiivne ja lihtsasti kasutatav, et tagada kiire ja mugav broneerimise protsess.

Töö teises peatükis analüüsib autor olemasolevaid lahendusi nii Eestis kui ka välisriikides. Kolmandas peatükis kirjeldatakse valminud mobiilrakenduse arenduskäiku. Tuuakse välja mobiilrakenduse loomiseks kasutatud tööriistu ja räägitakse lahti serveri paigaldamisest ja liidestamisest mobiilrakendusega. Neljandas peatükis kirjeldab autor piltide ja kirjelduste näol reaalset toimuvat mobiilrakendust läbi kasutajaliidese kliendi vaatepunktist. Viiendas peatükis analüüsib töö autor valminud mobiilrakendust.

2 Olemasolevate lahenduste analüüs

Käesolev peatükk kirjeldab olemasolevaid lahendusi nii mobiilrakenduste kui ka veebikeskkonna näol. Töö autor ei leidnud sarnaseid kasutusel olevaid auto hooldusaja broneerimise mobiilrakenduse lahendusi Eestist, küll aga väljaspool Eestit.

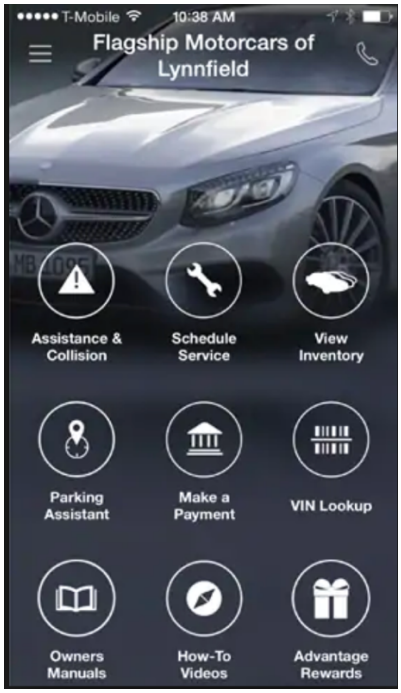
2.1 Mobiilrakendused

Mõnes teises riigis pakub Audi, BMW, Tesla, Maserati, Mercedes, Volkswagen, Subaru ja Toyota autohooldus mobiilrakenduse lahendust. Hõlmates endast teenuserakendust, et jääda seotuks autode omanikega pärast müüki. Rakendused on sünkroniseeritud konkreetsete margiesinduste(kas siis Maserati, Audi, Tesla, WolksVagen, Mercedes, Subaru ja Toyota) sõiduki numbri või esinduse konto e-posti aadressi ja parooli kaudu. Mobiilrakendusel on juurdepääs kasutaja varasematele teenustele ja informatsioonile. Igal ajahetkel on kasutajal võimalus mobiilrakenduse kaudu vaadata auto staatust. Kui sõidukis on vaja midagi hooldada, teavitab rakendus sellest kasutajat ja annab võimaluse broneerida esindusse aeg. Näiteks Maserati (Maserati Car Service) mobiilrakendusega broneerides saab teeninduskeskus kliendile saata rakenduse kaudu arve, mida on võimalus mobiilrakendusega koheselt tasuda. Mobiilrakenduse kaudu on võimalik vaadata ka eelnevate hoolduste ja arvete ajalugu.

Ühiseks jooneks saab tuua, et mõlemal puhul on tarvilik kasutajaks registreerimine. Samuti on võimalus näha tellitud tööde ajalugu. Suur erinevus seisneb selles, et auto hoolduse mobiilrakenduses on diagnostika liides, mis annab kasutajale teada auto hetke seisu ning seejärel saab kasutaja vajamineva hoolduse broneerida. Lõputöö jaoks loodud mobiilrakenduses peab kasutaja olema teadlik auto poolt kuvatavatest hooldus- ja veateadetest, otsene liides auto diagnostikamooduliga puudub. Lisaks loodud mobiilrakendusele on salasõna ununemise korral võimalus kiiresti parool taastada ilma mailiklienti kasutamata. Suureks erinevuseks saab tuua, et lõputöö raames loodud mobiilrakendus on suunitletud ainult auto hooldusaja broneerimisele, võimaldades broneeringu kiire protsessi läbiviimist kasutaja poolt.

Analüüsides väljaspool Eestit kasutusel olevaid mobiilrakenduste disaine võttis autor kasutusele Maserati ja Mercedese rakenduse disainielemente. Eelkõige arvestas autor

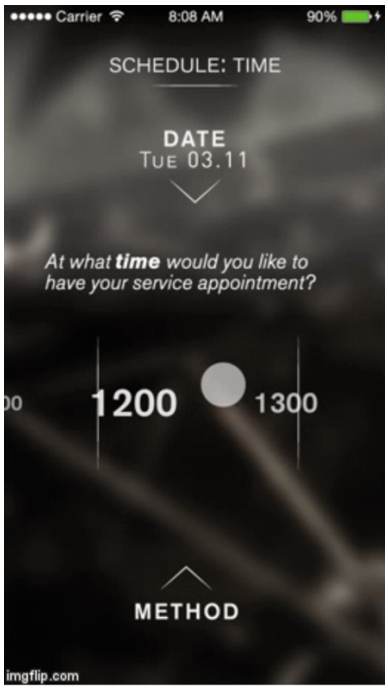
elementide valikul Apple kasutajaliideste juhtnööridega (Human Interface Guidline). Näiteks, käesolevas mobiilrakenduses nuppude väljanägemistel arvestati printsiipi, et ekraani allosas kuvatav täislaiuse nupp näeb parem välja, kui on ümarate nurkadega ja nupu joondus ei tohiks olla servast serva. Lisaks võttis autor kasutusele Mercedese mobiilrakendusest (Joonis 1) loetava ja selge ikoonide väljanägemise ja paigutamise. Ikoonide kasutuselevõtuga lähtuti, et ikoon peab olema äratuntav, tagataust võimalikult lihtne, mitte kirju ja ikoonide nurgad ümarad. Selliselt on ikoonid paremini eristatavad ning hoomatavad. [1]



Joonis 1 Mercedese mobiilrakendus

Teiseks suureks disainielemendiks võttis autor kasutusele ülelehe väljavaate nagu on kasutusel Maserati mobiilrakenduses (Joonis 2). Kerimisvaade jälgib sõrmede liikumist ja kohandub vastavalt päritolule [2]. Kerimisvaade võimaldab erinevaid funktsionaalsusi kokku panna. Vertikaalne kerimine on muutunud nutitelefonide tavaliseks interaktsioonitehnikaks. Läbi on viidud erinevaid kasutaja teste, kus uuritakse, kuidas kerimisfunktsiooni lisamine mõjutab kasutajate kavatsusi rakendust kasutada. Uuringud on näidanud, et kerimismeetodi lisamine mõjub positiivselt. [3] Võttes arvesse eelpool toodud argumenti otsustas autor mobiilrakenduses kasutada kerimisfunktsionaalsust. Kerimisfunktsionaalsus võimaldas märkimisväärselt kokku hoida mobiilrakenduse

lehtede arvu. Märkimist väärib ka asjaolu, et antud lähenemine tagab kasutaja seisukohast kiiremina broneerimise protsessi teostuse.



Joonis 2 Maserati mobiilrakendus

2.2 Veebirakendused

Järgenvalt on võrreldud kolme enim tuntud Eestis tegutseva margiesinduse veebikeskkondi: BMW Inchape Motors, Silberauto ja Audi.

Klientidel on võimalus broneerida auto hooldusaega enda autotootja esinduse veebikeskkonnast. Broneerimise tegevus erinevatel autotootjatel koosneb üldjuhul kolmest etapist:

1. Auto- ja isikuandmete täitmine
2. Esinduse valimine linnade piires
3. Soovitud tööliigi valimine

Silberauto klientidega võetakse ühendust esindusest, et kokku leppida hooldusaeg. BMW Inchape Motors'i ja Audi kliendid saavad valida veebikeskkonnas sobiva hooldusaja. Põhjus, miks võetakse klientidega ühendust seisneb hooldustööde varieeruvates pikkustes. Kolmel esindusel on määratud üldised tööliigid: hooldustööd, remonttööd, rehvivahetus, kere- ja värvitööd, varuosade tellimine, autopesu,

kampaania/tagasikutsumine, infopäring, vahatamine ja autopesu. Kõiki hooldustööde aegu ei saa subjektiivselt hinnata ning see on ka üks põhjus, miks ei ole hetkel veel auto hooldusaja broneerimise mobiilrakendust kasutusele võetud. Käesoleva töö autor valis mobiilrakenduse tegemisel sellised tööliigid, mis on ajaliselt ettehinnatavad. Nagu näiteks salongi pesu, rehvide vahetus, kere poleerimine, õlivahetus, käsipesu ja korraline hooldus.

Veebikeskkonnas hooldusaja broneerimiseks peavad kliendid sisestama nii isiku kui ka auto kohta käivad andmed. Kõigi kolme esinduse puhul väljad, mida klient peab enda kohta täitma on: nimi, telefoni number ja e-maili aadress. Auto kohta käivad andmed varieeruvad erinevate esinduste vahel, kuid üldiselt on nõutud auto registreerimistunnus, mark ja mudel. VIN-koodi, auto margi ja odomeetri näidu sisestamine on reeglina valikuline. Mobiilrakenduses otsustas autor valida nõutavateks väljadeks ees-ja perekonnanime, kontaktnumbri, e-maili, auto registreerimisnumbri, auto väljalaskeaasta. Lisaväljana on tarvilik sisestada salasõna muutmiseks küsimus koos vastusega.

Iga kord kui klient tahab auto hooldusaega broneerida veebikeskkonnas peab ta sisestama nii enda kui ka auto kohta käivad andmed uuesti. Veebikeskkonnas olev lahendus tundub töö autorile aeganõudev tegevus. Sellest lähtudes otsustas töö autor mobiilrakenduses teha sisse logimise funktsionaalsuse. Andmeid peab kasutaja sisestama ainult üks kord registreerimisel. Järgnevatel kordadel saadakse kliendi kohta käivad andmed juba andmebaasist, mis registreerimisel sisestati. Selle tulemusena on auto hooldusaja broneerimise protsess kiirem ja mugavam kliendi jaoks

2.3 Järeldused

Taustauuring teostati, et saada parem ülevaade turu olukorrast. Uurida, kas selline toode on juba olemas ning olemasolu korral selgitada välja, kuidas teostada konkreetse idee jaoks parim lahendus.

Selgus, et Eestis ei ole kasutusel mobiilrakendust, mis võimaldaks broneerida auto hooldusaega, kuid väljaspool Eestit pakuvad Audi, BMW, Tesla, Maserati, Mercedes, Volkswagen, Subaru ja Toyota autohooldus mobiilrakendust.

Olemasolevate veebi- ja mobiilirakenduste analüüsi tulemusel sai autor ideid ja mõtteid, mida projektis rakendati. Autor pidas oluliseks hoida mobiilrakendust kasutaja jaoks võimalikult lihtsa ja mugavana. Võrreldes veebirakendustega, kus kliendid pidid broneerimiseks iga kord sisestama mitmeid välju nii enda kui auto kohta, otsustas autor lihtsama variandi kasuks. Eeskuju võeti ka mitmelt välisriigis kasutusel olevalt mobiilrakenduselt. Näitena võib tuua võimalikult lihtsad ja arusaadavad ikoonid, samuti vertikaalne kerimisfunktsionaalsus.

3 Mobiilrakenduse arendus

Käesolev peatükk kirjeldab valminud mobiilrakenduse arenduskäiku. Tuuakse välja mobiilrakenduse loomiseks kasutatuid tööriistu ja räägitakse lahti serveri paigaldamisest ja liidestamisest mobiilrakendusega.

3.1 Arenduskäik

Käesoleva töö autor alustas arendust sobiva platvormi otsimisega. Mitmete asjaolude põhjal otsustas autor valida iOS platvormi. Seejärel valis autor arenduseks välja sobiva raamistiku, milleks on React Native. Koodi kirjutamiseks ja editeerimiseks kasutati integreeritud programmeerimiskeskkonda (IDE) Visual Studio Code.

Mobiilrakenduse arendamist alustas autor Node.js allalaadimisega, millele järgnes Xcode installeerimine App Store'ist. Järgnevalt oli tarvilik installeerida React Native käsurea liides, mille abil loodi React Native'i projekt. Projekti olemasoluga tekitas autor eraldi kaustad vaadete ja manuste jaoks, et mobiilrakenduse erinevad osad oleksid eraldatud ning kiiresti leitavad. JavaScripti klasside nimed grupeeriti antud mobiilrakenduse vaadete järgi, kus nimi kirjeldab, mida selles klassis tehakse.

Esmalt sai valmis arendatud esialgne mobiilrakenduse kasutajaliides koos lehtede vahelise navigeerimisega. Seejärel paigaldas autor lokaalse andmebaasi enda arvutisse, milleks kasutas avatud lähtekoodiga platvormi veebiserveri lahenduste paketti XAMPP, mis koosneb peamiselt Appache HTTP serverist, MariaDB andmebaasist ja PHP, Perli programmeerimiskeeles kirjutatud skriptidest [4]. Esialgse arenduse ja testimise hõlbustamiseks kasutas autor lokaalset serverit, mis töötles andmebaasi päringuid. Lokaalsesse serveri sai koostatud andmebaasi esialgne mudel, mis koosnes kahest tabelist. Esimene tabel sisaldas isikuandmeid ja teine tabel broneeringu spetsiifilisi andmeid. Arendustegevuse käigus lisandus uusi tabeli elemente vastavalt vajadusele. Esmalt arendas autor isikuandmete salvestamise funktsionaalsuse andmebaasi ning seejärel tabeli broneeringu spetsiifiliste andmete salvestamise andmebaasi. Mõlemate funktsionaalsuste korrektse töötamise järel otsustas autor lokaalse serveri välja vahetada Azure pilveserveri vastu, mis nõudis serveri seadistamist ja konfigureerimist. Lisaks arendas autor välja sisse logimise funktsionaalsuse, uue parooli sätestamise ning andmete

kuvamise ja muutmise võimaluse. Viimasena viimistles autor mobiilrakenduse kasutajaliidest.

Arenduse käigus üritas autor pidada meeles Clean Code printsiipe. Kasutada tähendusrikkaid muutujate nimesid, kasutada selgituseks kommentaare, kasutada koodis treppimist ning pidada meeles, et funktsioon või meetod käsitleks korraga ainult ühte ülesannet. [5]

3.2 Mobiilrakenduse loomiseks kasutatavad tööriistad

Mobiilrakendus sai loodud kasutades IDE-na Visual Studio Code'i, raamistikuna React Native'it ja iOS tarkvara arendustööriista Xcode'i.

3.2.1 React Native

React Native on avatud lähtekoodiga mobiilrakenduste raamistik, mis on loodud Facebooki arendajate poolt. Seda kasutatakse nii Android kui ka iOS mobiilrakenduste arendamises. React Native puhul on tegemist raamistikuga, mis põhineb programmeerimiskeelel Javascript ning kasutab sama disaini, mis React. React'i puhul on tegemist Javascripti teegiga, mis on mõeldud kasutajaliidese loomiseks. Oma olemuselt on antud raamistiku kasutavatel rakendustel sarnane fundamentaalne ülesehitus nagu tavapärastel iOS ja Android mobiilrakendustel. [6]

React Native positiivseteks külgedeks saab välja tuua ühise platvormi. Võimaldades uuesti kasutada sama koodibaasi iOS ja Androidi vahel. Teiseks positiivseks aspektiks on *hot reloading* ehk kuum laadimine, võimaldades arendajal hoida mobiilrakendust töötavana samal ajal kui implementeeritakse uusi versioone ja muudetakse kasutajaliidest. Seetõttu on muutused koheselt nähtavad ilma, et arendaja peaks rakendust uuesti üles ehitama. See omakorda võimaldab säästa aega kompileerimisel ning ei kaota muudatuste tegemisel rakenduse olekut. Suureneb tootlikkus ja väheneb arenduseks kuluv aeg.

React Native raamistikku on kasutatud mitmete tuntud mobiilrakenduste tegemisel. Näitena võib tuua Instagram, Skype, Facebook, Tesla, Bloomberg, Pinterest, WalmartLab.

3.2.2 Visual Studio

Visual Studio Code on Microsofti poolt Windowsi, Linuxi ja MacOSi jaoks välja töötatud lähtekoodi redaktor. Sisaldades debuggimise võimalust, sisseehitatud Git-juhtimist, koodi refaktoorimist ja väljavõtteid.

3.2.3 Xcode

Xcode on integreeritud arenduskeskkond(IDE) MacOS'ile, mis sisaldab Apple'i poolt välja töötatud tarkvaraarendusvahendite komplekti MacOS, iOS, watchOS ja TVOS tarkvara arendamiseks. Pakkudes kõike, mida vajatakse iOS rakenduste loomiseks, testimiseks, kasutuselevõtmiseks ja levitamiseks. [7]

Xcode pakub simulaatorrakendust, mis esitab iPhone'i, iPadi või Apple Watchi kasutajaliidest. Suheldes simulaatoriga, kasutades klaviatuuri ja hiirt, et jäljendada seadme pöörlemist, emuleerida puudutusi ja muid kasutaja toiminguid. [8]

3.3 Mobiilrakenduse liidestamine serveriga

Andmete töötlemiseks ühes keskses infrastruktuuris on projekti kaasatud API server. Mudeleid vastava lahenduse realiseerimiseks on mitmeid. Esimene variant on kogu lahendust ise hallata – kasutada kodus igapäevase töö tegemiseks aeglaseks jäänud lauaarvutit või osta Raspberry PI mikroarvuti, paigaldada sobiv operatsioonisüsteem koos vastavate rakendustarkvaradega. Tegemist võib olla küllaltki ebastabiilse lahendusega, sest riistvara ei pruugi olla eriti töökindel ning kodune internetiteenuse pakkuja võib ühendust ümber seadistades muuta API serveri rakenduse tarbeks kättesaamatuks. Sellise olukorra vältimiseks on võimalik proovida mõnda pilvelahendust, kus füüsiline riistvara on teenusepakkuja hallata. Kliendi kohustuseks jääb operatsiooni ja rakendusserverite töö eest hoolitsemine (IAAS – *infrastructure as service*), või anda ka operatsioonisüsteem teenusepakkuja haldusesse, ostes teenusena sisse rakenduse ja andmebaasi käitamise võimalust (PAAS – *platform as service*).

Mittefunktsionaalse nõudena oli töö autoril soov API serveris kasutada loengutes tuttavaks saanud PHP ja MySQL platvormi. See saigi üheks otsustuskriteeriumiks. Teiseks otsustuskriteeriumiks sai lahenduse majutuse maksumus tudengile ning teatud määral mõjutas ka kasutamise mugavus. Operatiivse halduse huvides valis autor mudeliks

IAAS'i, sest see võimaldab arenduse käigus serveri komponente kõige lihtsamini muuta – serverit saab operatsioonisüsteemist alates seadistada endale sobivaimaks. Hilisemalt, kui rakendus peaks reaalsesse kasutusse minema, tuleks valikut kindlasti uuesti kaaluda, sest teenusepakkujal on oluliselt suurem võimekus teenust kvaliteetsemalt töös hoida.

Tuntumate pilveteenuste pakkujate kodulehtesid läbi vaadates valis autor välja sobivaima IAAS teenusepakkuja. Järgnev valik ei ole ammendav, kuid antud projekti tarbeks enam kui piisav:

Tabel 1 Server

	Virtuaalserveri loomise võimalus	**Hind tudengile**
Microsoft Azure[9]	Jah	0.-
Amazon AWS[10]	Jah	0.-
Google Cloud[11]	Jah	0.-

Pilveteenuse pakkuja valik osutus oodatust lihtsamaks. Selgus, et TalTech üliõpilase konto on Microsofti süsteemidega liidestatud ning Azure portaali saab täiendava vaevata sisse logida ning tasuta ressursse kasutama hakata. Sellest lähtuvalt puudus vajadus teisi teenusepakkujaid lähemalt vaadata ja antud lõputöö raames loodud ja kasutatud API server majutus Microsoft Azure pilves.

API serveri operatsioonisüsteemi valik langes Ubuntu-le. Ühelt poolt on see kõikide teenusepakkujate toetatud operatsioonisüsteemide nimistus olemas ning teisalt leidub internetis palju juhendeid, kuidas PHP + MySQL kombinatsioon serveris käima saada. [9][10][11][12] Andmebaasi haldamiseks leiab kasutust loengutest tuttavat phpMyAdmin tarkvara, mis on internetimaterjalide põhjal Ubuntu-sse lihtsalt paigaldatav. [13]

3.3.1 API serveri paigaldus

API serveri operatsioonisüsteemiks sai Ubuntu 18.04 LTS (*long term support),* mis on majutatud Microsofti Azure pilves. Serveri loomine on kasutaja jaoks tehtud kergeks ja kiireks.

Kasutajale antakse valida loetud parameetrid, nagu virtuaalmasina nimi, andmekeskuse regioon, operatsioonisüsteem ning virtuaalserveri füüsilised parameetrid (protsessori tuumade arv, operatiivmälu kogus ning kõvaketta maht). Hilisemaks serveri haldusliidesele ligipääsuks tuleb seadistada mandaadi (*credentials*) parameetriteks kasutajanimi ning parool ja avada ka tulemüürist serveri teenustele ligipääsuks vajaminevad pordid, nagu 80 (HTTP – *hypertext transport protocol*/hüperteksti edastusprotokoll), 443 (HTTPS – *hypertext transport protocol over SSL*/hüperteksti edastusprotokoll üle turvasoklite kihi) ja 22 (SSH – *secure socket shell*/turvakesta protokoll). Portide 80 ja 443 peal töötavad teenused on kasutuses API serveri ja mobiilirakenduse omavahelises suhtluses, pordil 22 töötava SSH kaudu toimub serveri haldus.

Ülejäänud vajaminevad ressursid, nagu vajaminevad võrguliidesed, tekitab Azure pilv automaatselt. Antud etapis sai jälgida vaid töö staatust, pärast mida sai läbi haldusliidese serverisse logida.

Töökindluse huvides sai seadistatud Azure haldusliidese kaudu serverile veel täielik domeeninimi (FQDN – *Fully Qualified Domain Name*) juhuks kui serveri avalik IP aadress muutub ning kasutame seda mobiilirakenduses serveri poole pöördumisel.

Pärast operatsioonisüsteemi paigaldamist sai serveri haldusliidesega ühendus loodud ning vajaminevad tarkvarapaketid paigaldatud. Kõik vajaminev on alla laetav Ubuntu tarkvarahoidlast (repositooriumist) ja paigalduse kui ka esmase seadistusega saab hakkama Ubuntu paketihaldustööriist APT (*Advanced Packaging Tool*).[14]

Autor käivitas seejärel käsud, mis laadisid repositooriumist tarkvara alla ning paigaldasid need serverisse:

```
sudo apt update
sudo apt install apache2
sudo apt install php libapache2-mod-php php-mysql
sudo apt install phpmyadmin php-mbstring php-gettext
sudo systemctl restart apache2
```

Antud bloki järel oli serverisse paigaldatud veebiserver Apache koos PHP moodulite ja phpMyAdmin haldusliidesega. Seejärel siestatud kombinatsioon „sudo” ütleb operatsioonisüsteemile, et järgnev käsk tuleb käivitada teise kasutaja, ehk antud juhul juurkasutaja õigustes.[15]

Andmebaasimootori paigaldamiseks tuli käivitada käsk:

```
sudo apt install mysql-server
```

ning luua paigaldatud serverisse API tarbeks konto koos kasutajanime ja parooliga:

```
sudo mysql
CREATE USER 'api-user'@'localhost' IDENTIFIED BY 'api
password';
GRANT ALL PRIVILEGES ON *.* TO 'api-user'@'localhost';
FLUSH PRIVILEGES;
```

Loodi „api-user“ nimeline kasutaja koos parooliga. Antud kasutajakontol on õigus MYSQL-is luua vajaminevad andmebaasid, andmetabelid kui ka andmestruktuuri.[16] Andmebaasi haldus hakkab toimuma läbi eelnevalt paigaldatud phpMyAdmin haldusliidese.

Viimase sammuna serveri tehnilisest ettevalmistusest tuli hoolitseda selle eest, et mobiilrakendus API server vaheline andmevahetus oleks krüpteeritud. Selleks paigaldas töö autor API serverisse certbot nimelise tarkvara ja seadistas Apache veebiserveri kasutama Lets Encrypt sertifikaati kasutades järgnevaid käske:

```
Sudo apt install certbot
Sudo apt install python-certbot-apache
Sudo certbot –apache -d
serviceapp.northeurope.cloudapp.azure.com
```

Paigaldusprotsess viis läbi vajamineva seadistuse ning muuhulgas seadistas veebiserveri ka selliselt, et veebiserver hakkas kogu krüpteerimata HTTP liiklust suunama ümber turvalise HTTPS kanalisse. Käesoleva töö puhul puudus selleks küll vajadus, sest mobiilrakendus on eeldatavasti ainus, mis API serveri poole pöördub, kuid teistes kasutusjuhtudes võib sellisest automaatsest ümbersuunamisest kasu olla.[17][18]

Sellega oli API server loodud ja seadistatud. PHP-s kirjutatud programmikood tõsteti serveri vaikimisi loetavasse /var/www/html kausta. Internetiühenduse olemasolul saab mobiilrakendus nüüd pöörduda API serveri poole.

3.4 Funktsionaalsed ja mitte funktsionaalsed nõuded

Peatükis kirjeldatakse auto hooldusaja broneerimise mobiilrakenduse funktsionaalseid ja mittefunktsionaalseid nõudeid.

3.4.1 Funktsionaalsed nõuded

Auto hooldusaja broneerimise mobiilrakendusele on ette seatud järgnevad funktsionaalsed nõuded:

- Mobiilrakenduse kasutamiseks peab olema sisse logitud
- Korraga saab ainult üks kasutaja olla sisse logitud
- Klient saab broneerida aja ainult ühe tööliigi jaoks
- Salasõna ununemise korral, saab klient parooli taastada
- Kliendil on võimalus enda kohta andmeid muuta
- Kasutajaks registreerimise võimalus
- Klient sisestab registreerimisel nii enda kui auto kohta käivad andmed
- Klient saab valida esinduse ja toimumise aja
- Mobiilrakendus peab suhtlema andmebaasiga, et teostada päringuid
- Soovi korral saab klient kirjutada kommentaari
- Eduka broneerimise korral teavitatakse klienti
- Serveripoolne programm peab töö broneerimisel valideerima kasutajanime olemasolu

3.4.2 Mittefunktsionaalsed nõuded

Auto hooldusaja broneerimise kohta käivad mitte funktsionaalsed nõuded:

- Auto hoolduse tööliigi leidmine lehelt ühe klikiga
- Mobiilrakendus põhineb iOS platvormil
- API serveri operatsioonisüsteemiks on Ubuntu 18.04 LTS (*long term support),* asub Microsofti Azure pilves
- Kliendi registreerimisel peab serveripoolne programm valideerima kasutajanime sobivust.
- Süsteemi vastuse aeg peab jääma 1 sekundi piiridesse
- Süsteem peab töötama tõrgeteta 95% ajast

4 Valminud auto hooldusaja broneerimise mobiilrakendus

Mobiilrakenduse eesmärgiks on luua võimalus auto hooldusaja broneerimiseks. Mobiilrakendus koosneb seitsmest mobiilivaatest: pealehe vaade, kasutaja sisse logimise vaade, kasutajaks registreerimise vaade, uue parooli seadistamise vaade, tööliigi valimise vaade, aja ja esinduse valimise vaade ning broneeringu kinnitamise vaade. Projekti kood on üleval GitHub repositooriumis https://github.com/MarianKuklane/Lopu.

Antud peatükk kirjeldab piltide ja kirjelduste näol reaalset toimuvat mobiilrakendust läbi kasutajaliidese kliendi vaatepunktist. Ekraanitõmmised on tehtud reaalselt toimuvast mobiilrakendusest.

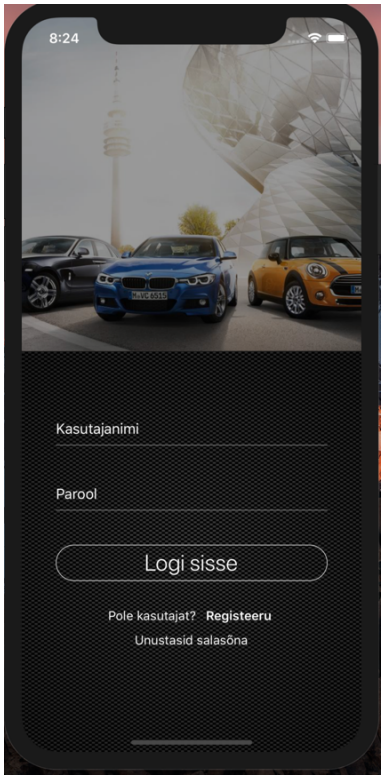
4.1 Sisse logimine ja registreerumine kasutajaks

Mobiilrakenduse avamisel kuvatakse kasutajale järgnev vaade (Joonis 3). Vajutades nuppu „Broneeri hooldusaeg siit“ suunab mobiilrakendus pealehelt järgmisele lehele.

Joonis 3 Pealehe vaade

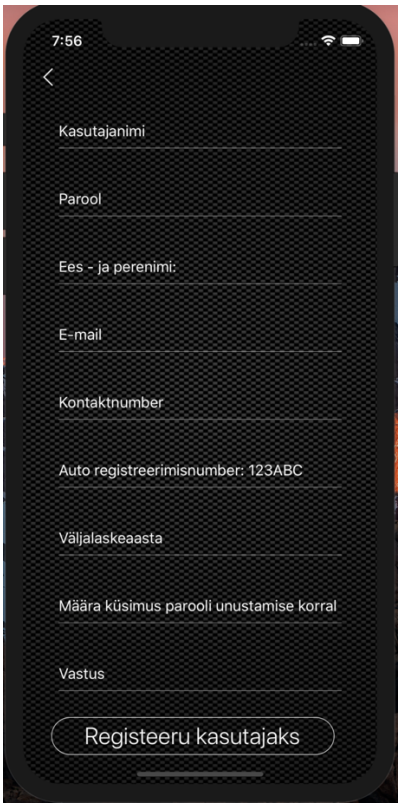
Antud mobiilrakenduse kasutamiseks on tingimus, et igal kliendil, kes soovib auto hooldusaega broneerida, peab olema kasutajakonto. Klient saab mobiilrakenduse

kasutamiseks logida sisse (Joonis 4). Õige kasutajanime ja parooli korral saab klient siseneda süsteemi.



Joonis 4 Sisse logimise vaade

Esimest korda mobiilrakenduse kasutamiseks on vaja luua kasutajakonto. Juhul kui kliendil puudub kasutajakonto olemasolu on võimalus registreerumiseks (Joonis 5). Salasõna ununemise korral saab klient taastada parooli (Joonis 6). Järgneb nii isiku- kui ka autoandmete täitmise vaade. Antud vaates peab klient täitma ankeedi, mis sisaldab kasutajanime, parooli, ees- ja perenime, e-mail aadressi, kontaktnumbrit, auto registreerimisnumbrit ja väljalaskeaastat. Lisaks kui kasutaja unustab salasõna, siis selle tarbeks on vaja märkida registreerimisvaates küsimus ja salasõna.

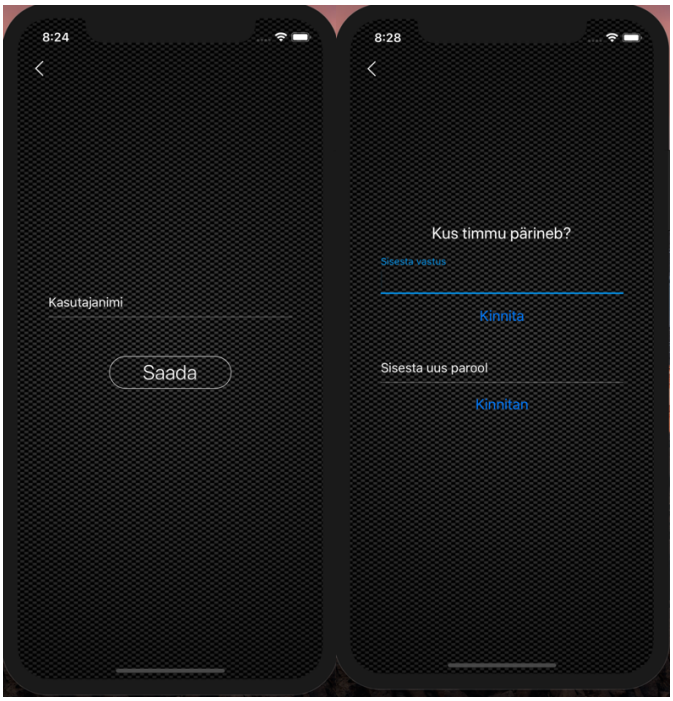


Joonis 5 Kasutajaks registreerimise vaade

Antud vaates enne päringu edastamist teostatakse kontroll väljade üle. Kontrollitakse nii kasutajanime, parooli, ees-ja perenime korrektsust, e-maili vastavust, kontakt- ning registreerimisnumbrit, salaküsimust ja salaparooli korrektsust. Kasutajanimi peab olema miinimum 6 tähemärki. Parool peab sisaldama nii numbreid kui ka tähemärkisid. E-maili sisestamise puhul kontrollitakse, et tähemärkidele järgneks „@“, seejärel oleks vähemalt 4 tähemärki, punkt ning veel vähemalt 2 tähemärki. Kontaktnumber peab sisaldama ainult numbreid. Auto registreerimisnumber peab sisaldama 3 numbrit ja 3 tähemärki. Väljalaskeaasta peab olema minevikus. Kui kõikide väljade kontroll läbitakse edukalt, siis luuakse päring serveri suunas. Päring sisaldab kõiki eelpool sisestatud andmeid. Serveri poolel olev programm dekodeerib JSON’i päringu ning loob uue rea SQL tabelitesse.

4.2 Uue parooli seadistamine

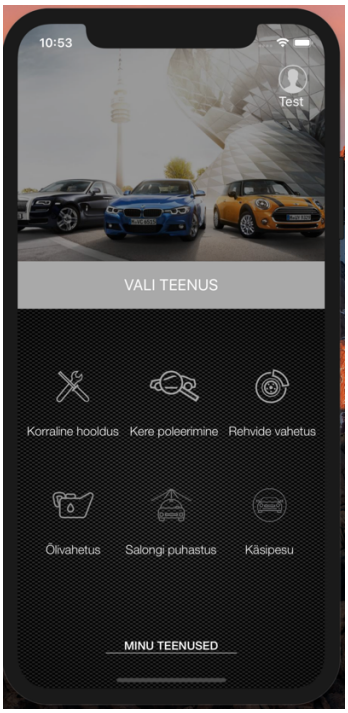
Juhul kui kasutaja on unustanud oma parooli, on võimalus uuesti parooli lähtestada (Joonis 6). Selleks on vaja sisestada turvakontrolliks kasutajanimi. Õige kasutajanime olemasolul peab kasutaja läbima kontrollküsimuse, mille määras registreerimiselehel. Õige vastuse olemasolul on võimalus uus parool määrata. Parooli sobivuse kontrolli läbimisel teavitab süsteem edukast salasõna muutmisest.



Joonis 6 Uue parooli lähtestamise vaade

4.3 Sobiva tööliigi valimine

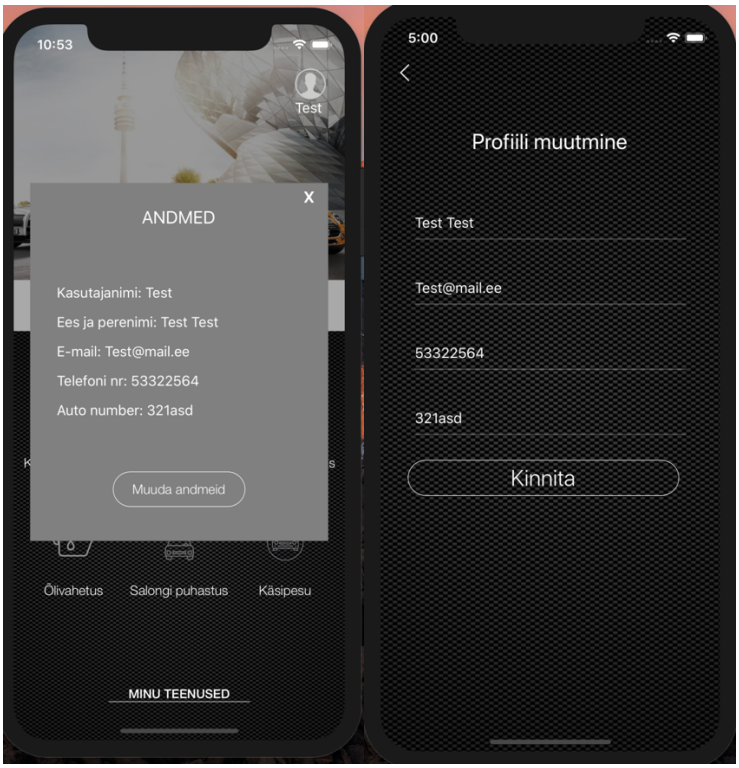
Klient saab valida sobivat tööliiki: õlivahetus, kere poleerimine, rehvitöö, käsipesu, salongipuhastus, korraline hoolduse ja keretöö (Joonis 7). Sõltuvalt tööliigist suunab mobiilrakendus kliendi valitud tööliigi lehele (Joonis 10).



Joonis 7 Teenuste valimise vaade

4.4 Kasutaja andmete kuvamine ja muutmine

Vaate üleval paremas servas kuvatakse kasutajanimi, millega on rakendusse sisse logitud (Joonis 7). Mobiilrakenduse kasutaja saab kasutajanime peale vajutades näha enda kohta käivaid andmeid, mis leidub süsteemis kasutaja kohta. Vajaduse korral saab kasutaja olemasolevaid andmeid muuta (Joonis 8).

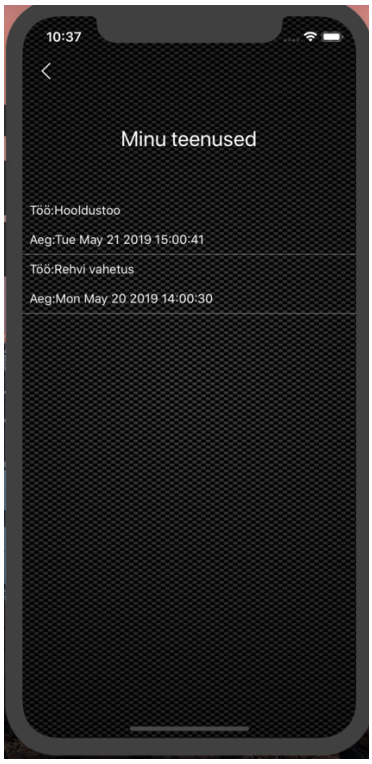


Joonis 8 Kasutaja andmete kuvamise ja muutmise vaade

Kasutajaprofiili muutmisel kuvatakse andmebaasist välja loetud väärtused antud kasutaja kohta. Kasutajal on võimalus kuvada andmeid enda kohta kolmes vaates: valitava tööliigi-, esinduse ja kuupäeva valimise- ning broneeringu lõpetamise vaadetes. Andmeid saab muuta kahes vaates: tööliigi- ning esinduse ja kuupäeva valimise vaadetes.

4.5 Kasutaja teenuste ajaloo kuvamine

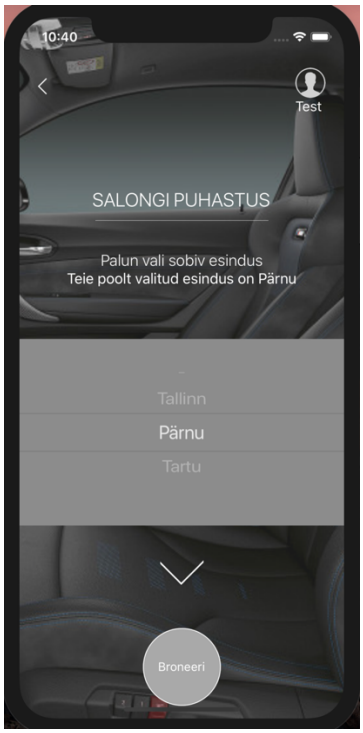
Mobiilrakenduse kasutajal on võimalus vaadata broneeritud tööde ajalugu (Joonis 9). Konkreetses vaates kuvatakse kasutajale tellitud tööliik ning toimumise aeg. Antud vaates kuvatakse kõik eelseisvad ja juba toimunud hooldustööd.



Joonis 9 Teenuste ajaloo vaade

4.6 Sobiva esinduse ja aja valimise vaade

Järgenvalt kui klient on valinud sobiva tööliigi, kuvatakse kliendile ette esinduse ja aja valimise vaade (Joonis 10). Konkreetses vaates saab klient valida teenuse osutamise koha, kuupäeva ja kellaaja. Soovi korral on kliendil võimalus lisada kommentaar.



Joonis 10 Esinduse koha valimine

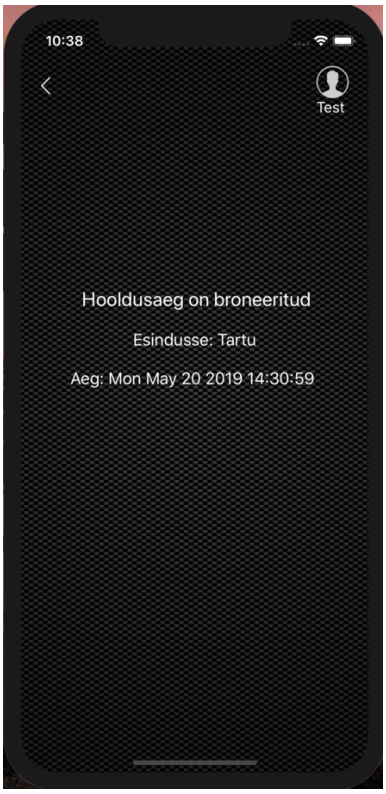
Klient saab valiku teha kolme auto esinduse vahel: Tallinna Pärnu või Tartu. Valitud esindus kuvatakse kliendile. Järgnevalt saab klient valida enda jaoks sobiva toimumise kuupäeva ja kellaaja (Joonis 11).

Joonis 11 Sobiva kuupäeva ja aja valimise vaade

Kui valitud aeg on juba broneeritud, kuvab süsteem teate, et valitud aega ei saa broneerida. Broneeringuid saab teha tööpäevadele 08:00-16:00 vahele. Broneeringuid minevikku, nädalavahetustele ning töövälistele aegadele pole võimalik teha. Valituks sai ajavahemik 8:00 – 16:00 vahel, kuna auto esindused on nimetatud kellaaegadel avatud. Ajavahemikku on võimalik laiendada, kuid antud valik sõltub üheselt margiesinduste soovidest ja vajadustest. Kui klient valib kuupäeva, kontrollib süsteem andmebaasist, kas selleks kellaajaks ei ole juba broneeringut teostatud. Lisaks peab süsteem ka arvestama kliendi poolt valitud esindust ja tööks kuluvat hinnangulist aega. Klient saab soovi korral lisada kommentaari. Kui kontrolli käigus selgub, et valitud esindus on selleks konkreetseks kuupäevaks ja kellaajaks vaba, siis suunatakse klient järgmisele lehele. Lehe navigeerimisega kantakse valitud andmed järgmisele lehele vaike olekus kaasa.

4.7 Lõppvaade

Andmete eduka sisestamise järel edastatakse kasutaja poolt sisestatud andmed viimasele vaatele (Joonis 12). Kuvatakse broneeringu kinnitus, valitud esindus ning kuupäev ja kellaaeg. Lisaks on võimalik näha üleval paremas servas sisse logitud kasutajanime.



Joonis 12 Broneeringu lõpu vaade

5 Auto hooldusaja broneerimise mobiilrakenduse analüüs

Lõputöö käigus valminud mobiilrakendus on kõigest prototüüp, mille pealt saab soovi korral edasi arendada esinduste või töökodade jaoks kasutatavat mobiilrakendust. Antud mobiilrakendus on platvorm, millele saab juurde tekitada lisafunktsionaalsust vastavalt kliendinõuetele.

Kõige suuremaks puudujäägiks autori arvates on tagasisideta aja ja kuupäeva valimise vaade (*date and time picker selector*). Töös kasutati React Native arendajate poolt loodud komponenti *datePickeriOS*'i. Konkreetse komponendi kasutamisega polnud võimalik seda täielikult kohandada autori soovide kohaselt. Komponendil olid antud ette konkreetsed parameetrid, mida sai muuta. Näiteks: keel, minuti intervall, ajavöönd, kuupäeva valija režiim, piirang võimalike kuupäeva ja kellaaja väärtuse vahemikule ning komponendi värv. Käesolevas töös tahtis autor, et ajad, mis on juba broneeritud, kuvatakse komponendis heledamana, et oleks kasutaja jaoks eristatavad. Probleemi uurides selgus, et kasutades React Native enda komponenti pole võimalik sellist muudatust läbi viia. Selleks oleks pidanud kirjutama uue komponenti, mis oleks autori vajadustele konfigureeritav. Uue komponendi kirjutamine oleks kujunenud küllaltki suureks tööks ning vastavalt projekti prioriteetidele see arendustöösse ei läinud.

Teiseks nõrkuseks võib välja tuua turvaprobleemid. Kuigi iga päring, mis andmebaasi poole saadeti, sisaldas andmete korrektsuse kontrolli, jäi siiski variant, et teatavat sisu omavad päringud võivad andmebaasi või selle osasid kahjustada.

Puudusena võib veel välja tuua auto numbrimärgi sisestamise piirangud. Eritellimusel valmistatud registreerimisnumber ei vasta üldjuhul mustrile 123ABC. Puuduse kõrvaldamiseks tuleks teha selgeks, kas ja missugused piirangud registreerimisnumbrile seatud on.

Valminud lõputöö raames oli autor keskendunud mobiilrakenduse ülesehitamisele ning andmebaasi vahelisele suhtlusele. Serveripoolse rakenduse ülesanne oli teostavad vaid primitiivne kontroll päringu sisule, teostada päring ning anda tagasiside mobiilrakendusele. Andmebaasi hallatus polnud töö prioriteediks ning teostatud sai vaid vajalik miinimum rakenduse toimimiseks.

5.1 Mobiilrakenduse võimalikud edasiarendused

Kuna lõputöö käigus valminud rakendus on kõigest prototüüp siis on autoril ideid, milliseid funktsionaalsusi võiks juurde arendada.

Töö autoril oli soov arendada ka SMS teavituse funktsionaalsus. Selle jaoks oleks pidanud autor tekitama uue serveri, mis saadab vastava broneeringu info SMS teavitusena kasutajale välja. Server oleks ühenduses olnud üle USB liidese SIM kaarti omava seadega. Võimalik, et oleks olnud ka lihtsam lahendus probleemi lahendamiseks, kuid selle detailsema analüüsini hetkel ei jõutud.

Töö teostamise ajal ei olnud andmebaasi hallatus töö prioriteediks, siis edasise arenduse käigus on idee luua uus rakenduse, mis on mõeldud nii auto esindustele ja töökodadele andmete paremaks haldamiseks. Rakendus, millel oleks kliendile parem kasutajaliides andmete haldamiseks.

Broneeritud hooldusaega peaks saama sünkroniseerida kliendi personaalse kalendriga. Broneeringu õnnestumisel loodaks kalendrisse sündmus, mis sisaldaks antud broneeringu toimumise aega ja seotud ettevõtte kontakti.

Lisafunktsionaalsusena võiks olla võimalus esindusega ühendust võtta telefoninumbrit käsitsi valimata.

5.2 Testimine

Lõputöö funktsionaalsust ja kasutajamugavust lasi autor testida kasutajatel. Peamised eesmärgid olid rakenduse ülesehituse arusaadavuse ja kasutajasõbralikkuse väljaselgitamine ning märkamatuks jäänud vigade leidmine. Arendusprotsessi juures võivad vead jääda autorile märkamatuks, seetõttu on vajalik rakenduse testimine teiste kasutajate poolt.

Auto hooldusaja ja broneerimise mobiilrakenduse testimine viidi läbi viie inimese vahel kasutades koridor meetodit. Autor võttis kasutusele koridor meetodi, kuna see on kiire ja odav meetod, kus testimisse võetakse suvaliselt ettejuhtuvad inimesed. Testijatele anti suuliselt ette üldised testsammud, näiteks logi sisse, registreeru kasutajaks, broneeri salongipuhastuse aeg, muuda kasutajaandmeid, vaata kasutaja andmeid, sea uus parool.

Detailset testsammud puudusid, et näha kuidas testija antud testülesande täidab. Testitavateks olid igapäevaselt nutitelefoni kasutavad inimesed. Valdav enamus testijatest olid auto omanikud või auto regulaarsed kasutajad. Testimise tulemused on kajastatud Lisa 1.

Testitavad tõid põhiliselt välja, et mobiilrakendus on kasutaja jaoks enamasti kergesti hallatav ja orienteeruv. Viiest testitavast kahel esines raskusi esinduskoha ja kellaaja valimise vaatel, kuna ei tuldud selle peale, et leht on keritav (scroll) ehk antud mobiilvaade on üleleheline. Kuid samas meeldis teistele üleleheline disain. Lisaks tõi üks testija välja, et erinevate hooldustööde lehtede tagataustad võiksid piltide asemel olla ühevärvilised taustad, et vältida teksti segunemist pildiga. Samas ülejäänud testijatele meeldis, et tagataustad kujutavad endast pilte. Lisaks toodi välja, et salasõna vahetamine on paremini lahendatud, kui siiani kokku puututud teenustel. Puudub sidumine kasutaja meili kontoga ning ei esine pikka ootamisaega salasõna kinnitamisel. Samas on antud lahendamisel ka märkimisväärsed puudused, kui registreerimisel sisestatud küsimuse vastus on samuti ununenud.

Mainiti, et võiks olla ka broneerimise tühistamise funktsionaalsus, kuid hetkel polnud antud funktsionaalsus prioriteetne.

Küsimusele, kas võtaksite kasutusele mobiilrakenduse vastasid kõik testijad positiivselt. Selgitusena toodi välja, et broneerimise protsess võrreldes veebikeskkonnaga on kasutaja jaoks hallatavam, kiirem ja lihtsam.

5.3 Esinduste huvi mobiilrakenduse vastu

Töö autor kontakteerus e-maili teel Eestis tegutsevate margiesindustega, et saada tagasisidet mobiilrakenduse vajalikkusest. Autor valis ja võttis ühendust kuue tuntuma margiesindusega: Inchcape Motors OÜ (BMW), Audi Tallinn, SilberAuto (Mercedes), Škoda, Mariine auto(Subaru ja Peugeot) ja Viking Motors (Opel, Kia, Peugeot). Kindlasti saaks võimalusel auto margiesindusi juurde lisada. Tagasiside laekus kahest margiesindusest-Inchape Motors OÜ ja Audi Tallinn.

Nii Inchape Motors OÜ kui ka Audi Tallinn andsid rakendusele positiivset tagasisidet. Mõlemad margiesindused olid huvitatud auto hooldusaja broneerimise mobiilrakendusest ja mainisid valmisolekut kohesest kasutuselevõtust. Kuigi Audi Tallinna jaoks peaks

lisaks hoolduse aja broneerimisele rakendus võimaldama ka kõiki ettevõtte poolt pakutavaid tooteid ja teenuseid.

Tagasisidest saab lähtuda, et antud mobiilrakendusel oleks Eestis potentsiaalset turgu.

6 Kokkuvõte

Eestis tegutsevatel auto margiesindustel puudub hooldusaja broneerimise mobiilrakendus. Antud töö eesmärgiks oli ehitada auto hooldusaja broneerimise mobiilrakendus, mis lihtsustaks ja kiirendaks klientidel hooldusaja broneerimise protsessi esindustesse. Antud eesmärgi saavutamiseks analüüsis töö autor nii olemasolevaid veebilahendusi kui ka väljaspool Eestis kasutatavaid auto mobiilrakendusi.

Valmis mobiilrakendus iOS platvormile, mis on küll hetkel prototüüp, kuid mida saab edasi arendada koostöös margiesindusega. Valminud mobiilrakendus võimaldab kliendil kasutajaks registreeruda ning valida sobiva tööliigi, toimumis koha, kuupäeva ja kellaaja. Antud projekt on leidnud positiivset tagasisidet nii testitavate kui ka margiesinduste seas.

Kasutatud kirjandus

[1]„Human Interface Guidlines“ [Võrgumaterjal]. Available: https://developer.apple.com/design/human-interface-guidelines/ios/visual-design/adaptivity-and-layout/ (07.04.2019)

[2] „UIScrollView” [Võrgumaterjal]. Available: https://developer.apple.com/documentation/uikit/uiscrollview (07.04.2019)

[3] „ Power of the Swipe: Why Mobile Websites Should Add Horizontal Swiping to Tapping, Clicking, and Scrolling Interaction Techniques” [Võrgumaterjal]. Available: https://www.tandfonline.com/doi/full/10.1080/10447318.2016.1147902?scroll=top&needAccess= true (07.04.2019)

[4] „Installing, configuring, and developing with XAMPP,“ [Võrgumaterjal]. Available: http://dalibor.dvorski.net/downloads/docs/installingconfiguringdevelopingwithxampp.pdf. (03.03.2019)

[5] R. C.Martin, Clean Code: A Handbook of Agile Software.

[6] V. Novick, „React Native - Building Mobile Apps with JavaScript,“ [Võrgumaterjal]. Available: https://books.google.ee/books/about/React_Native_Building_Mobile_Apps_with_J.html?id=YJZGDwAAQBAJ&printsec=frontcover&source=kp_read_button&redir_esc=y#v=onepa[1]ge&q&f=false (01.03.2019)

[7] M. Knott, Beginning Xcode: Swift Edition.

[8] „Getting Started in Simulator,“ [Võrgumaterjal]. Available: https://developer.apple.com/library/archive/documentation/IDEs/Conceptual/iOS_Simulator_Guide/GettingStartedwithiOSSimulator/GettingStartedwithiOSSimulator.html. (10.03.2019)

[9] [Võrgumaterjal]. Available: https://azuremarketplace.microsoft.com/en-us/marketplace/apps/Canonical.UbuntuServer. (01.04.2019)

[10] [Võrgumaterjal]. Available: https://aws.amazon.com/marketplace/ (05.04.2019)

[11] „Quickstart Using a Linux VM,“ [Võrgumaterjal]. Available: https://cloud.google.com/compute/docs/quickstart-linux. (05.04.2019)

[12] „How To Install Linux, Apache, MySQL, PHP (LAMP) stack on Ubuntu 18.04,“ [Võrgumaterjal]. Available: https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-ubuntu-18-04. (30.03.2019)

[13] „How To Install Linux, Apache, MySQL, PHP (LAMP) stack on Ubuntu 18.04,“ [Võrgumaterjal]. Available: https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-ubuntu-18-04. (05.04.2019)

[14] „Apt,“ [Võrgumaterjal]. Available: https://help.ubuntu.com/lts/serverguide/apt.html.en. (06.04.2019)

[15] „Ubuntu manuals,“ [Võrgumaterjal]. Available: http://manpages.ubuntu.com/manpages/bionic/man8/sudo.8.html. (06.04.2019)

[16] „How To Create a New User and Grant Permissions in MySQL,,“ [Võrgumaterjal]. Available: https://www.digitalocean.com/community/tutorials/how-to-create-a-new-user-and-grant-permissions-in-mysql. (06.04.2019)

[17] „How to secure Apache with Let’s encrypt on Ubuntu 18.04,,“ [Võrgumaterjal]. Available: https://www.digitalocean.com/community/tutorials/how-to-secure-apache-with-let-s-encrypt-on-ubuntu-18-04 (06.04.2019)

[18] „Let’s Encrypt,,“ [Võrgumaterjal]. Available: https://letsencrypt.org/how-it-works/ (06.04.2019)

Lisa 1 – Testi raport

Registreeru kasutajaks

Testitav 1: Leidis esimese vajutusega üles. Meeldis teavitused valede sisse logimiste puhul

Testitav 2: Leidis esimese vajutusega üles.

Testitav 3: Leidis esimese vajutusega üles. Mainis, et registreerimisel võiks kohe sisse logida.

Testitav 4: Leidis esimese vajutusega üles

Testitav 5: Leidis esimese vajutusega üles

Logi sisse

Testitav 1: Leidis koheselt ja esimese vajutusega üles

Testitav 2: Leidis koheselt ja esimese vajutusega üles. Mainis, et võiks ka olla logi välja funktsionaalsus

Testitav 3: Leidis koheselt ja esimese vajutusega üles

Testitav 4: Leidis koheselt ja esimese vajutusega üles

Testitav 5: Leidis koheselt ja esimese vajutusega üles

Unustasid salasõna

Testitav 1: Leidis esimese vajutusega üles. Hea kerge, meeldib, et ei ole gmaili peal

Testitav 2: Leidis esimese vajutusega üles

Testitav 3: Leidis esimese vajutusega üles

Testitav 4: Leidis esimese vajutusega üles

Testitav 5: Leidis esimese vajutusega üles

Broneeri salongipuhastusaeg

Testitav 1: Leht oli kergesti hallatav, väga meeldis alla kerimine. Võiks olla ühevärviline tagataust

Testitav 2: Leht ei ole kergesti hallatav. Ei saanud koheselt alla liikumisest aru

Testitav 3: Leht ei ole kergesti hallatav. Ei saanud koheselt alla liikumisest aru

Testitav 4: Leht oli kergesti hallatav, väga meeldis alla kerimine

Testitav 5: Leht oli kergesti hallatav, väga meeldis alla kerimine

Vaata kasutaja andmeid

Testitav 1: Leidis kiirelt sobiva koha ülesse, meeldis, et näeb enda kohta andmeid

Testitav 2: Leidis kiirelt sobiva koha ülesse, kuid ei saanud aru, et peale võib klõpsata. Mainis, et võiks olla mingi nooleke, mis näitab, et on nupp

Testitav 3: Leidis kiirelt sobiva koha ülesse, meeldis, et näeb enda kohta andmeid

Testitav 4: Leidis kiirelt sobiva koha ülesse, meeldis, et näeb enda kohta andmeid

Testitav 5: Leidis kiirelt sobiva koha ülesse, meeldis, et näeb enda kohta andmeid

Muuda kasutajaandmeid

Testitav 1: Märkas koheselt nuppu, meeldis, arusaadav, lihtne

Testitav 2: Märkas, meeldis

Testitav 3: Märkas, meeldis

Testitav 4: Märkas, meeldis

Testitav 5: Märkas, meeldis

Vaata minu teenuseid:

Testitav 1: Leidis koheselt minu teenuse nupu üles

Testitav 2: Võttis paar sekundit aega kui avastas minu teenuse nupu

Testitav 3: Ei avastanud kohe minu teenused nuppu. Minu teenused asendada teise nimega

Testitav 4: Leidis koheselt minu teenuse nupu

Testitav 5: Leidis koheselt minu teenuse nupu

Kas kasutaksite sellist mobiilrakendust?

Testitav 1: Jah

Testitav 2: Jah

Testitav 3: Jah

Testitav 4: Jah

Testitav 5: Jah