České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra počítačů

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: Šidlovský Marko

Studijní program: Otevřená informatika
Obor: Softwarové systémy

Název tématu: Webová aplikace správy vozového parku pro systém sdíleni automobilu vice uživateli

Pokyny pro vypracování:

Navrhnete a implementujte webovou aplikaci správy vozového parku pro systém sdíleni automobilu vice uživateli.
1) Dohodnete s ostatními studenty pracujícími na tomto systému funkcionalitu systému a zdokumentujte všechny způsoby použiti funkci pomoci UML diagramu.
2) Navrhnete vzhled webové aplikace pro správu vozového parku určeného pro provozovatele systému.
3) Implementujte webovou aplikaci a napojte ji na databázi vytvořenou pro tento systém.
4) Vytvořte návod pro správce, který bude dostupný vhodným a intuitivním způsobem v aplikaci.
5) Navrhnete budoucí rozšířeni webové aplikace.

Seznam odborné literatury:

[1] Fowler, M.: UML Distelled - Third Edition. Addison-Wesley, 2003.
[2] Fain, Y., Moiseev, A.: Angular 2 Development with Type Script. Manning, 2016.
[3] Novotny, T.: Technicka zprava: Technicka vize CarSharingu. CVUT FS, 2015.

Vedoucí: doc. Ing. David Šišlák, Ph.D.

Platnost zadání do konce letního semestru 2017/2018

prof. Dr. Michal Pěchouček, MSc.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 25.1.2017

České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra počítačů

Bakalárska práca

Webová aplikace správy vozového parku pro systém sdílení automobilu více užívateli

Marko Šidlovský

Vedúci práce: doc. Ing. David Šišlák, Ph.D.

Študijný program: Otevřená informatika, bakalársky

Odbor: Softwarové systémy

24. mája 2017

Poďakovanie

Chcem sa poďakovať doc. Ing. Davidovi Šišlákovi, Ph.D. za odbornú pomoc, konzultácie a rady, ktoré mi pomohli pri vypracovaní mojej bakalárskej práce.

Prehlásenie

Prehlasujem, že som predloženú prácu vypracoval samostatne a že som uviedol všetky použité informačné zdroje v súlade s Metodickým pokynom o dodržiavaní etických princípov pri príprave vysokoškolských záverečných prác.

V Prahe dňa 24. 5. 2016 .....

Abstract

ŠIDLOVSKÝ, Marko: Web application for carsharing system. [Bachelor Thesis] - Czech Technical University in Prague. Faculty of Electrical Engineering, Department of Computer Science. Supervisor: doc. Ing. David Šišlák, Ph.D.

The object of the thesis is to design and implement a fleet management web application for the carsharing system used by several users. Introduction describes the problems of user applications and the way of their development, but the main emphasis is put on creating a prototype of a functional web application with connection to the server. Also included is the design of communication between the server and the web application. The result of the bachelor thesis is a web administration application for a carsharing system by multiple users. The proposed application meets basic functionality and is ready for a future expansion.

Keywords: carsharing, web application, application interface, angular, Uniqway.

Abstrakt

ŠIDLOVSKÝ, Marko: Webová aplikace správy vozového parku pro systém sdílení automobilu více užívateli. [Bakalárska práca] - České vysoké učení technické v Praze. Fakulta elektrotechnická, Katedra počítačů. Vedúci: doc. Ing. David Šišlák, Ph.D.

Objektom tejto práce je návrh a implementácia webovej aplikácie správy vozového parku pre systém zdieľania automobilov viacerými užívateľmi. V úvode je popísaná problematiky užívateľských aplikácií a spôsob ich vývoja, no hlavný dôraz sa kladie na vytvorenie prototypu funkčnej webovej aplikácie s pripojením na server. Súčasťou je aj návrh spôsobu komunikácia medzi serverom a webovou aplikáciou. Výsledkom bakalárskej práce je správcovská webová aplikácie pre systém zdieľania automobilov viacerými užívateľmi. Navrhnutá aplikácia spĺňa základnú funkcionalitu a je pripravená pre jej budúce rozšírenie.

Kľúčové slová: zdieľanie áut, webová aplikácia, aplikačné rozhranie, angular, Uniqway.

Obsah

Zoznam obrázkov

Zoznam tabuliek

Kapitola 1

Úvod

Fenoménom posledných rokov je zdieľaná ekonomika. Pojmom zdieľaná ekonomika označujeme ekonomiku, v ktorej si bežní ľudia vymieňajú tovar a služby medzi sebou. Jej rozvoj nastal začiatkom 21. storočia, kedy moderné technológie priniesli možnosti pre obyčajné domácnosti ako jednoducho prenajať byt či vypožičat si automobil. Takáto ekonomika prináša rozšírenie zdieľania na veľkú, anonymnú skupinu ľudí, ktorí sa nielenže nikdy nemuseli stretnúť, oni ani nemusia byť z rovnakého kontinetu a nemusia hovoriť rovnakým jazykom. Zdieľaná ekonomika dnes už zďaleka nie je iba o zdieľaní, ale aj o vymieňaní, spolupráci a o prenájme. Iný názov by mohol byť „párovacia ekonomika“, od slova párovať niečo s niečím, v tomto prípade párovanie dopytu a ponuky cez internetovú platformu. Dnes celosvetovo najznámejšími spoločnosťami, ktoré sú postavené na princípoch tohto druhu ekonomiky sú Airbnb a Uber.

Mobilita a zdieľaná ekonomika hrajú vo veľkých mestách stále väčšiu úlohu. V najväčších českých mestách je dopravná situácia problematická a medzi kľúčové dopravné prostriedky v nich patria autá. Carsharing je zdieľanie osobných automobilov viacerými osobami, ktorým by sa z dôvodu malej frekvencie využívania nevyplatilo vlastniť a prevádzkovať automobil. Môže mať formu oficiálneho či neoficiálneho združovania ľudí, ktorí sú potom spoluvlastníkmi automobilov, ale aj formu podnikateľskú, teda ako služba verejných požičovní automobilov. Systém, v rámci ktorého viacerí vodiči zdieľajú jedno auto sa rozvíja v posledných rokoch vo viacerých svetových metropolách. Zdieľaním áut je možné šetriť životné prostredie, menším počtom áut na ceste znížiť hodnotu emisií a menšou potrebou parkovacích miest šetriť priestor v mestách.

Projekt Uniqway je spoločným študentským projektom pražských univerzít (České vysoké učení technické v Praze, Vysoká škola ekonomická v Praze a Česká zemědělská univerzita v Praze). Partnerom projektu je ŠKODA AUTO a.s.. Hlavnou myšlienkou je poskytnúť jednoduchú, cenovo výhodnú službu mobility pre cieľovú skupinu projektu, ktorou sú študenti hore uvedených univerzít. Služba je určená pre tých z nich, ktorí nevlastnia automobil a potrebujú sa prepraviť do školy, na internát, na nákup či výlet. Veľmi dobre doplňuje ich každodenné používanie hromadnej dopravy.

Mojou úlohou na projekte je vytvoriť ľahko použiteľnú webovú aplikáciu pre správcu (administrátora) carsharingového systému. Správca sa stará o vozový park, užívateľov a rezervácie, preto potrebuje možnosť prístupu k dátam a ich vhodnú vizualizáciu.

Cieľom bakalárskej práce je analyzovať potrebnú funkcionalitu správcu systému, navrhnúť koncept riešenia, užívateľsky prívetivý vzhľad a implementovať webovú aplikáciu. Súčasťou práce je aj návrh a implementácia aplikačného rozhrania servera pre výmenu dát nevyhnutných pri práci s webovou aplikáciou. V práci sa kladie dôraz na vytvorenie prototypu webovej aplikácie s prihliadnutím na jej budúcu rozšíriteľnosť. Teoretický základ a výber technológií obsahuje druhá kapitola. V jej podkapitolách je popísaný koncept problematiky tvorby užívateľských aplikácií, rôzne možnosti pre vývoj aplikácií ako aj odôvodnenie voľby konkrétnej technológie pre túto prácu. Tretia kapitola prináša praktickú implementáciu webovej správcovskej aplikácie. Jej pod-

kapitoly definujúce požiadavky na aplikáciu a správcovské role prinášajú analytický pohľad na riešenú problematiku. V podkapitole o aplikačnom rozhraní sa zameriavame na jeho návrh a v samotnom konci kapitoly je popísaná implementácia navrhnutej aplikácie. Posledná, štvrtá kapitola je záverom práce a prináša zhodnotenie práce ako aj návrh rozšírenia aplikácie v budúcnosti. Prílohu práce tvoria náhľady užívateľského rozhrania aplikácie, diagram komunikácie správcovskej aplikácie so serverom a popis budúcej funkcionality.

Kapitola 2

Užívateľská aplikácia

2.1 Popis užívateľskej aplikácie

V dnešnej dobe používame počítač každý deň. Na počítači máme spustený náš obľúbený webový prehliadač na surfovanie na internete, textový editor na písanie a úpravu textov. Pomocou počítača spolu komunikujeme, či už priamo volaním alebo emailom, počúvame hudbu či pozeráme film. Počítač využívame aktívne v práci pri programovaní, projektovaní, riadení výroby, vyhodnocovaní ekonomických výsledkov a pri rôznych iných činnostiach. Pri každej tejto aktivite využívame rôzne **aplikácie**.

Aplikácia je systém na zber, ukladanie, spracovanie a prezentáciu dát pomocou počítača [2]. Užívateľ zapína aplikáciu s jasným cieľom, vykonať nejakú úlohu. Aplikácie prešli svojim vývojom a ak ste niekedy na ich ovládanie potrebovali skúsenosti programátora so znalosťou programovacích jazykov a príkazov operačného systému, dnes to zvládnu malé deti vďaka jednoduchému užívateľskému rozhraniu.

2.2 Druhy užívateľských aplikácií

Aplikácie na počítači určené pre užívateľa môžeme rozdeliť na:

- **Konzolová aplikácia** - aplikácia bez grafického rozhrania, ktorá beží v príkazovom riadku operačného systému. Zväčša jednoduchá aplikácia na testovanie.
- **Desktopová aplikácia** - aplikácia, ktorá vyžaduje inštaláciu v rámci operačného systému počítača, napríklad textový editor alebo prehrávač filmov.
- **Webová aplikácia** - webová stránka s ktorou môže užívateľ interagovať - odosielať a prijímať dáta. Webová aplikácia je spustná vo webovom prehliadači. Pre jej korektný chod je nutné pripojenie zariadenia k internetu.
  - **Single page aplikácia** - špeciálny typ webovej aplikácie, ktorá pri používaní nevyžaduje opätovné načítanie celého obsahu. Medzi serverom a užívateľom sú prenášané iba potrebné dáta a aplikácia ponúka užívateľské rozhranie pre prácu s nimi vo webovom prehliadači.

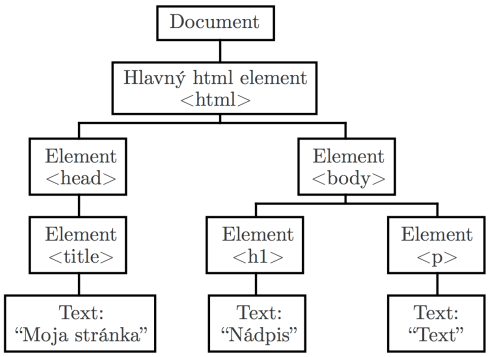
Vďaka rozvoju rýchlosti a pokrytia internetu sa po roku 2000 rozvinul koncept *Softvér ako služba* [4], model kedy zákazník po zaplatení neinštaluje desktopovú aplikáciu ale prostredníctvom webového prehliadača prístupuje ku vzdialenej aplikácií na serveri poskytovateľa. Prinieslo to výhody ako rýchlejšie nasadenie, nižšie počiatočné náklady či kvalitnejšia podpora. Avšak tento model kládol veľkú záťaž na servery poskytovateľa, ktoré spracovávali množstvo požiadaviek od užívateľov. Single page aplikácia rieši práve tento problém a je istým hybridom medzi desktopovou aplikáciou a webovou aplikáciou. Spája ich výhody, ktorými sú **rýchla odozva** dektopovej aplikácie a **multiplatformnosť** webovej aplikácie. To sú hlavné dôvody, prečo som sa v tejto práci rozhodol pre tvorbu single page aplikácie.

2.3 Single page aplikácia

Pojmom *Single page aplikácia* skrátene SPA označujeme webové aplikácie bežiace vo webovom prehliadači, ktoré pri používaní nevyžadujú opakované načítavanie celej aplikácie. Na rozdiel od klasickej webovej stránky, renderovanie SPA prebieha na klientovi a dáta sa prenášajú na pozadí. U SPA sa nejedná iba o prezenčnú vrstvu, ale časť logiky aplikácie prebieha na klientovi, teda o SPA môžeme hovoriť aj ako o **hrubom klientovi** bežiacom v prehliadači [7].

2.3.1 Document Object Model

Document Object Model skrátene DOM je rozhranie pre prácu s HTML alebo XML dokumentom. Definuje logickú štrúktúru dokumentu a spôsoby, akými sa s dokumentom dá pracovať [10]. Webový prehliadač pri načítaní webovej stránky zo serveru zostaví DOM danej HTML stránky, s ktorým následne pracuje. DOM HTML dokumentu má po zostavení stromovú štruktúru. Jednoduchý príklad ilustruje obrázok 2.1.



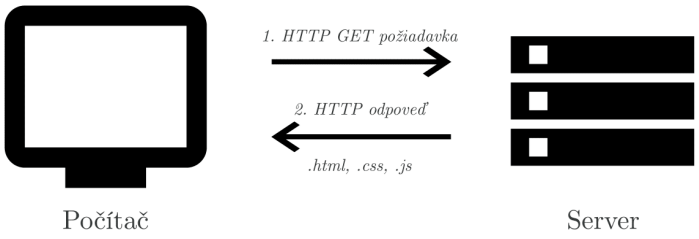
Obr. 2.1: Document object model príklad

2.3.2 Asynchronous JavaScript And XML

Asynchronous JavaScript And XML skrátene AJAX je súborom viacerých technológií. Ide o využitie programovacieho jazyka JavaScript, ktorý na pozadí webového prehliadača dokáže komunikovať pomocou protokolu HTTP so serverom. JavaScript na základe užívateľských vstupov alebo odpovedí zo servera mení DOM, čím modifikuje užívateľské rozhranie stránky [6].

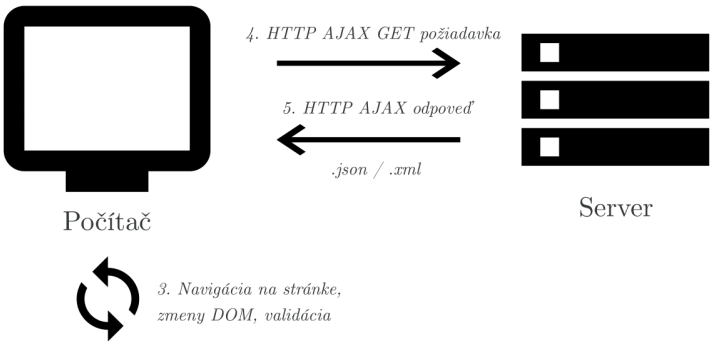
2.3.3 Komunikácia single page aplikácie so serverom

Pri prvom načítaní SPA si webový prehliadač stiahne kód (HTML, CSS, JavaScript) potrebný na chod aplikácie, ako ilustruje obrázok 2.2.



Obr. 2.2: Inicializácia single page aplikácie

Následná interakcia užívateľa s aplikáciou prebieha na strane klienta, DOM SPA je menený pomocou JavaScriptu, vďaka čomu stránka nepotrebuje komunikovať so serverom. Nutná komunikácia so serverom pre načítanie aktuálnych dát z databázy prebieha na pozadí pomocou technológie AJAX(viď obrázok 2.3). Dáta sú prenášané vo formáte JSON alebo XML.



Obr. 2.3: Chod single page aplikácie

Výhody použitia SPA pre užívateľa podľa [7] sú:

- Aplikácia je spustená vo webovom prehliadači, čím je **nezávislá** na **type zariadenia** a **operačného systému** užívateľa.
- Aplikácia je **uložená na serveri**, jej aktualizácia je iba otázkou nahratia novej verzie na server. Užívatelia tak majú novú verziu okamžite k dispozícií.
- Aplikácia ponúka **rýchlu odozvu** ako desktopová aplikácia.

2.4 Tvorba single page aplikácie

Framework je všeobecná aplikácia (časť kódu), ktorá môže byť došpecifikovaná pre vytvorenie vlastnej aplikácie [9]. Väčšina aplikácií má určité časti spoločné, napríklad ošetrenie vstupov od užívateľa, prihlásenie či registrácia užívateľa. Práve tieto časti obsahuje framework, a preto využitie frameworku urýchľuje a uľahčuje vývoj aplikácií.

Frameworky pre tvorbu single page aplikácie sú napríklad Angular, React, Ember alebo VueJS.

Mojou voľbou bol Angular. Je to komplexný framework s dobrou dokumentáciou, veľkou komunitou a rýchlym nasadením, za ktorým stojí internetový gigant Google.

2.4.1 Angular

Angular 2[1] je frontendový TypeScriptový framework pre tvorbu SPA vydaný spoločnosťou Google v roku 2016. Predchodca, AngularJS, bol publikovaný v roku 2010. Angular 2 prešiel podstatnými zmenami, a preto nie je spätne kompatibilný so svojím predchodcom. Angular 2 je kompletný balík, ktorý rieši prezenčnú vrstvu aplikácie, interakciu užívateľa s aplikáciou, dátový model, HTTP volania a iné. Základným stavebným blokom Angular aplikácie je komponenta. Vďaka Angular 2 CLI je vytvorenie a spustenie aplikácie otázkou 4 príkazov v príkazom riadku. Vývojári tohto frameworku taktiež ponúkaju peknú a užívateľsky prívetivú dokumentáciu. Príklady známych aplikácií naprogramovaných pomocou frameworku Angular sú Prezi, Wolfram Alpha, Google Assistent, Tesla či Udemy.

Stavebné bloky Angular aplikácie podľa [5] sú:

- **Modul** - je kontajnerom pre skupinu spoločných komponent, servís, direktív a podobne. Ide o knižnicu komponent a servís, ktoré implementujú určítú funkcionalitu. Malé aplikácie majú zvyčajne iba jeden hlavný modul.
- **Komponenta** - je hlavným stavebný blokom Angular aplikácie. Každá komponenta pozostáva z dvoch častí. Prvou je prezenčná časť, ktorá definuje užívateľské rozhranie a druhou trieda, ktorá implementuje logiku prezenčnej časti. Každá Angular aplikácia sa skladá z minimálne jedného modulu a jednej komponenty.
- **Direktívum** - dovoľuje pridať vlastné, špeciálne chovanie HTML elementu. Napríklad umožnuje pridať samodoplňovaciu funkciu k HTML <input> elementu. Každá komponenta je direktívum spolu s pripojenou prezenčnou časťou, ale na rozdiel od komponenty, direktíva nemá vlastnú prezenčnú časť.

[1] Viac informácií o Angulare nájdete na https://angular.io/

2.4.2 TypeScript

TypeScript je rozšírením jazyka JavaScript, ktorého cieľom je uľahčiť vývoj JavaScriptových aplikácií. Ide o open-source programovací jazyk vyvinutý a udržiavaný spoločnosťou Microsoft. TypeScript je navrhnutý pre vývoj veľkých aplikácií a zdrojový kód napísaný v jazyku TypeScript je kompilovaný do jazyka JavaScript. TypeScript obohacuje JavaScript o modulový systém, triedy a rozhrania. Primárnym cieľom jazyka je poskytnúť systém statického typovania pri vývoji JavaScriptových aplikácií [3].

2.5 Komunikácia single page aplikácie so serverom

2.5.1 Server

Server je hardvérový systém alebo softvérový program, ktorý poskytuje službu klientom [2]. Ide o počítač pripojený do siete, ktorý reaguje na požiadavku klienta a generuje odpoveď - klient/server architektúra. Medzi známe druhy serverov patrí webový server, databázový server, DNS a iné.

Server projektu Uniqway slúži ako:

- **Webový server** - poskytuje správcovskú aplikáciu.
- **Aplikačný server** - sprístupňuje aplikačné rozhranie, ktoré slúži na prenos dát medzi serverom a ostatnými komponentami systému.
- **Databázový server** - slúži na uchovanie perzistentných dát.

2.5.2 Webové aplikačné rozhranie

Webové aplikačné rozhranie slúži pre definíciu komunikácie s aplikáciou, predpisuje druhy požiadaviek, ktoré môže klient odosielať na server. Každá požiadavka má definované svoje parametre a odpoveď ktorú poskytne.

Aplikačné rozhranie serveru Uniqway má tieto parametre:

- pri komunikácií využíva šifrovaný protokol HTTPS,
- pre autentifikáciu užívateľa slúžia Java Web Tokens, skrátene JWT[2],
- dáta sú prenášané vo formáte JSON.

[2]Viac informácií o JWT na https://jwt.io/

Kapitola 3

Implementácia správcovskej aplikácie

3.1 Požiadavky na správcovskú aplikáciu

Po konzultáciách s členmi a vedúcim technického tímu projektu Uniqway boli definované požiadavky na správcovskú aplikáciu. Z dôvodou dodržania rozsahu bakalárskej práce je v nej riešená iba časť všetkých požiadaviek. V prílohe D.1 sú uvedené ostatné požiadavky.

3.1.1 Nefunkčné požiadavky

Nefunkčné požiadavky na softvér sú definované ako požiadavky, ktoré nepopisujú, čo softvér bude robiť, ale ako to bude robiť. Príkladom možu byť požiadavky na návrh a výkonnosť softvéru, bezpečnosť, požiadavky na externé rozhranie softvéru, čas vývoja a iné. Nefunkčné požiadavky sa niekedy ťažko testujú, preto sú zvyčajne hodnotené subjektívne [2].

Nefunkčné požiadavky na správcovskú aplikáciu:

- **Muliplatformný systém** - podmienkou pre systém je jeho funkčnosť na rôznych typoch zariadení (počítač, tablet, telefón), ako aj na rôznych platformách (Android, iOS a iné).
- **Intuitívne rozhranie** - systém má mať prehľadné a intuitívne užívateľské rozhranie, aby práca s ním bola čo najjednoduchšia.
- **Bezpečnosť** - pri návrhu systému je potrebné klásť dôraz na bezpečnosť systému ako celku. Zabezpečenie prístupu do aplikácie iba autentifikovaným užívateľom či použitie šifrovanej komunikácie so serverom má byť samozrejmosťou.

3.1.2 Funkčné požiadavky

Funkčné požiadavky tvoria podmnožinu požiadaviek používateľa, ktoré opisujú, čo softvér robí z hľadiska úloh a služieb [2]. Definujú základné činnosti, ktoré sa musia v softvéri vyskytnúť pre správne prijímanie vstupov, spracovanie vstupov a generovanie správnych výstupov [1].

Funkčné požiadavky na správcovskú aplikáciu:

- **Zobrazenie všetkých áut** - systém má podporovať zobrazenie všetkých áut. Pre prehľadnosť zobrazí v tabuľke iba základné údaje (poradové číslo auta, meno auta, ŠPZ, typ auta, stav nádrže a stav auta), bližšie informácie poskytne po kliknutí na auto.
- **Zobrazenie aktuálnej polohy áut** - systém prehľadne zobrazí polohu áut na Google Maps. Poloha auta sa má obnovovať v reálnom čase.
- **Pridanie užívateľa** - systém umožní pridať informácie o užívateľoch. Systém má o užívateľovi v databáze evidovať priezvisko, meno, titul, štátna príslušnosť, číslo občianskeho preukazu/pasu, platnosť OP/pasu, dátum narodenia, miesto narodenia, rodné číslo, adresa trvalého pobytu, mailová adresa, číslo vodičského preukazu, platnosť VP, číslo RFID karty a stav užívateľa.

- **Editácia užívateľa** - systém umožní správcovi editovať informácie o užívateľoch.
- **Vymazanie užívateľa** - systém umožní vymazať užívateľa, presnejšie zmeniť jeho stav na Inactive (Neaktívny). Užívateľ v stave Inactive nemá oprávnenie sa do systému prihlásiť.
- **Zmena stavu užívateľa** - systém umožní meniť stav užívateľa. Stavy užívateľa sú Created (Vytvorený), Active (Aktívny) a Inactive (Neaktívny).
- **Zobrazenie detailných informácií o užívateľovi** - systém umožní prehľadne zobraziť detailné informácie o užívateľovi.
- **Zobrazenie všetkých užívateľov** - systém v tabuľke zobrazí všetkých užívateľov. Uvedené budú pri nich iba základné informácie (poradové číslo klienta, meno, priezvisko, číslo OP/pasu, číslo VP, stav užívateľa). Detailné informácie sa zobrazia až po kliknutí na konkrétneho užívateľa.
- **Aktivácia karty užívateľa** - systém umožní aktivovať užívateľovu registráciu, presnejšie zmeniť jeho stav na Active (Aktívny).
- **Zobrazenie všetkých rezervácií** - systém v tabuľke zobrazí všetky rezervácie. V tabuľke sa pri jednotlivých rezerváciách zobrazia len základné údaje (poradové číslo rezervácie, užívateľ, dátum a čas vytvorenia rezervácie, stav rezervácie). Detailné informácie o rezervácií sa zobrazia po kliknutí na konkrétnu rezerváciu.
- **Vymazanie rezervácie** - systém umožní vymazať rezerváciu, presnejšie zmeniť jej stav na zrušená. Takáto rezervácia sa už ďalej nezobrazuje v prehľade rezervácií v systéme. Informácia o zrušení rezervácie vrátane dôvodu je zaslaná užívateľovi mailom resp. priamo správou do mobilnej aplikácie.

3.2 Užívateľské role

Koncept riadenia prístupu do aplikácie založený na užívateľských rolách vznikol v 70. rokoch minulého storočia. Prístupové práva do aplikácie sú spojené s užívateľskými rolami a užívatelia s rovnakým prístupom sú priradení k príslušnej užívateľskej role. Užívateľské role sú vytvorené podľa typov pracovných pozícií v organizácií a užívateľom sú priradené na základe ich zodpovednosti a kvalifikácie [8].

Na základe požiadaviek bola pre správcovskú aplikáciu definovaná jedna užívateľská rola, a to administrátor. Užívateľ s rolou administrátor má prístup do všetkých častí aplikácie. V budúcnosti sa počíta s rozšírením rolí o role dispečer, správca užívateľov, správca vozového parku a ekonóm. Príloha práce D.2 obsahuje kompletný diagram správcovských rolí spolu s popisom ich oprávnení.

3.3 Aplikačné rozhranie na serveri

Pre zabezpečenie požadovanej funkcionality definovanej vo funkčných požiadavkách na správcovskú aplikáciu (viď kapitola 3.1.2) bolo navrhnuté webové aplikačné rozhranie na serveri. Aplikačné rozhranie je popísané zoznamom požiadaviek, na ktoré server dokáže odpovedať. Pri každej požiadavke sú definované typ HTTP požiadavky, URI, očakávaná odpoveď. Každá nižšie definovaná požiadavka, okrem požiadavky na prihlásenie, **musí** ešte v hlavičke obsahovať **JWT token** slúžiaci na autentifikáciu. V prípade nevalidného tokenu server vráti odpoveď s kódom 403.

3.3.1 Prihlásenie užívateľa

Táto požiadavka slúži na prihlásenie užívateľa.

Typ HTTP požiadavky: **POST**
URI: /**login**
Parametre v hlavičke požiadavky:

Parameter	**Typ parametru**	**Dátový typ**
email	povinný	reťazec
password	povinný	reťazec

Tabuľka 3.1: Parametre požiadavky na server pre prihlásenie užívateľa

Očakávaná odpoveď zo servera:

```
{
        "message": "User successfully logged in"
}
```

Iné odpovede (HTTP status kód 401):

- nevalidné prihlasovacie údaje.

3.3.2 Zoznam všetkých áut

V odpovedi na túto požiadavku sa vráti zoznam všetkých áut so základnými údajmi o každom vozidle.

Typ HTTP požiadavky: **GET**
URI: /**admin/cars**

Očakávaná odpoveď zo servera:


```
[{
        "carId": 1,
        "name": "Columbus",
        "licenceNumber": "3SY 5562",
        "status": "Available",
        "carModel": {
                "brand": "Skoda",
                "type": "Fabia"
        },
        "position": {
                "latitude": 50.083588,
                "longitude": 14.441225
        },
        "tankPercentage": 92.0
}]
```


3.3.3 Zoznam všetkých užívateľov

V odpovedi na túto požiadavku sa vráti zoznam všetkých užívateľov spolu so základnými údajmi o užívateľoch.

Typ HTTP požiadavky: **GET**
URI: /**users**
Očakávaná odpoveď zo servera:


```
[{
        "userId": 1,
        "firstName": "Peter",
        "surname": "Novak",
        "email": "peter.novak@test.com",
        "status": "new",
        "university": {
                "name": "Ceske vysoke uceni technicke",
                "acronym": "CVUT",
                "city": "Praha"
        }
}]
```


3.3.4 Vytvorenie nového užívateľa

Táto požiadavka slúži na pridanie nového užívateľa.

Typ HTTP požiadavky: **POST**
URI : /**users**

Parametre v tele požiadavky:

Parameter	Typ parametru	Dátový typ
firstName	povinný	reťazec
surname	povinný	reťazec
email	povinný	reťazec
gender	povinný	celé číslo
identificationNumber	povinný	reťazec
university	povinný	celé číslo
nationality	povinný	celé číslo
dateOfBirth	povinný	reťazec
addresses	nepovinný	pole Address
address.type	nepovinný	celé číslo
address.street	nepovinný	reťazec
address.doorNumber	nepovinný	reťazec
address.city	nepovinný	reťazec
address.postCode	nepovinný	reťazec
address.country	nepovinný	celé číslo
groups	nepovinný	pole celých čísel

Tabuľka 3.2: Parametre požiadavky na server pre vytvorenie užívateľa

Parameter *address.type* určuje typ adresy. V aplikácií rozlišujeme 3 druhy adries, a to permanentnú adresu(1), adresu narodenia(2) a kontaktnú adresu(3).

Očakávaná odpoveď zo servera:

```
{
        "message": "User successfully generated"
}
```

Iné odpovede (HTTP status kód 400):

- v požiadavke chýba povinný parameter,
- parameter odoslaný v požiadavke neexistuje v databáze (poradové číslo),
- parameter nie je validný.

3.3.5 Detail užívateľa

Odpoveď na túto požiadavku sa vráti detailné informácie o užívateľovi. Súčasťou URI je identifikačné číslo užívateľa.

Typ HTTP požiadavky: **GET**
URI: **/users/{id}**
Očakávaná odpoveď zo servera:


```
{       "userId": 1,
        "firstName": "Peter",
        "lastName": "Novak",
        "email": "peter.novak@test.com",
        "status": "new",
        "identificationNumber": "9503232929",
        "addedAt": "2015-03-02 03:23:33",
        "nationality": "Slovak",
        "university": {
                "name": "Ceske vysoke uceni technicke",
                "acronym": "CVUT",
                "city": "Praha"
        },
        "groups": [{
                "name": "Ambasador *"
        }],
        "cards": [{
                "type": 1,
                "name": "ISIC",
                "validity": "2015-03-04",
                "canOpen": true
        }],
        "address": [{
                "type": 1,
                "name": "Permanent residence",
                "street": "Svabska",
                "doorNumber": "39",
                "city": "Presov",
                "postcode": "08001",
                "country": "Slovakia"
        }],
        "reservations": [{
                "reservationId": 3,
                "createdAt": "2015-10-26 07:46:36",
                "finishedAt": "2015-10-26 09:46:36",
                "price": 15.03,
                "reservationStatus": "closed"
}]}
```

Iné odpovede (HTTP status kód 400):

- identifikačné číslo odoslané v URI neexistuje v databáze.

3.3.6 Editácia užívateľa

Táto požiadavka slúži na editáciu existujúceho užívateľa. Súčasťou URI je identifikačné číslo užívateľa.

Typ HTTP požiadavky: **PUT**
URI: **/users/{id}**
Parametre v tele požiadavky:

Parameter	Typ parametru	Dátový typ
firstName	povinný	reťazec
surname	povinný	reťazec
email	povinný	reťazec
status	povinný	celé číslo
gender	povinný	celé číslo
identificationNumber	povinný	reťazec
university	povinný	celé číslo
nationality	povinný	celé číslo
dateOfBirth	povinný	reťazec

Tabuľka 3.3: Parametre požiadavky na server pre editáciu užívateľa

Očakávaná odpoveď zo servera:

```
{
        "message": "User successfully updated"
}
```

Iné odpovede (HTTP status kód 400):

- identifikačné číslo odoslané v URI neexistuje v databáze,
- v požiadavke chýba povinný parameter,
- parameter odoslaný v požiadavke neexistuje v databáze (poradové číslo),
- parameter nie je validný.

3.3.7 Aktivácia užívateľa

Táto požiadavka slúži na aktiváciu užívateľa. Súčasťou URI je identifikačné číslo užívateľa. Na serveri prebieha kontrola či užívateľ má pridané všetky povinné atribúty.

Typ HTTP požiadavky: **GET**
URI: /**users**/**{id}**/**activate**

Očakávaná odpoveď zo servera:

```
{
        "message": "User successfully activated"
}
```

Iné odpovede (HTTP status kód 400):

- identifikačné číslo odoslané v URI neexistuje v databáze,
- užívateľovi chýba povinný parameter,
- kontrolovaný parameter nie je validný.

3.3.8 Vymazanie užívateľa

Táto požiadavka slúži na vymazanie užívateľa. Súčasťou URI je identifikačné číslo užívateľa.

Typ HTTP požiadavky: **DELETE**
URI: /**users**/**{id}**

Očakávaná odpoveď zo servera:

```
{
        "message": "User successfully deleted"
}
```

Iné odpovede (HTTP status kód 400):

- identifikačné číslo odoslané v URI neexistuje v databáze.

3.3.9 Zoznam všetkých rezervácií

Odpoveď na túto požiadavku vráti zoznam všetkých rezervácií so základnými informáciami.

Typ HTTP požiadavky: **GET**
URI: /**reservations**

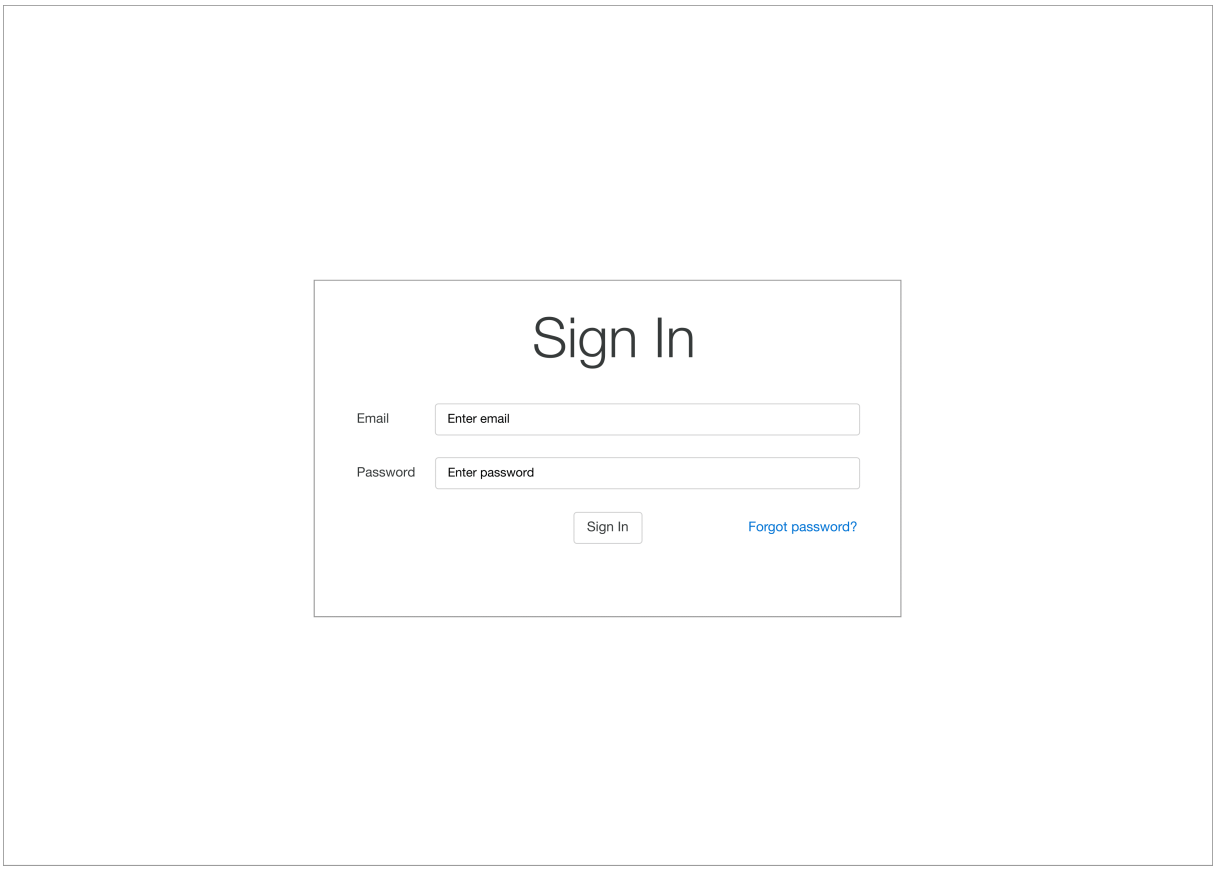
Očakávaná odpoveď zo servera:


```
[{
        "reservationsId": 1,
        "status": "closed",
        "user": {
                "userId": 1,
                "firstName": "Peter",
                "surname": "Novak"
        },
        "car": {
                "carId": 1,
                "name": "Galileo"
        },
        "createdAt": "2015-10-26 07:46:36",
        "finishedAt": "2015-10-26 09:46:36",
        "price": 15.03
}]
```


Parametry *createdAt* a *finishedAt* označujú čas a dátum začiatku a konca rezervácie. Parameter *createdAt* je automaticky zaznamenaný prijatím rezervácie na serveri z klienstskej aplikácie. Rezervácia sa končí začatím jazdy alebo zrušením rezervácie Sever poznamená čas v parametri *finishedAt*. Obe tieto atribúty slúžia k výpočtu ceny za rezerváciu.

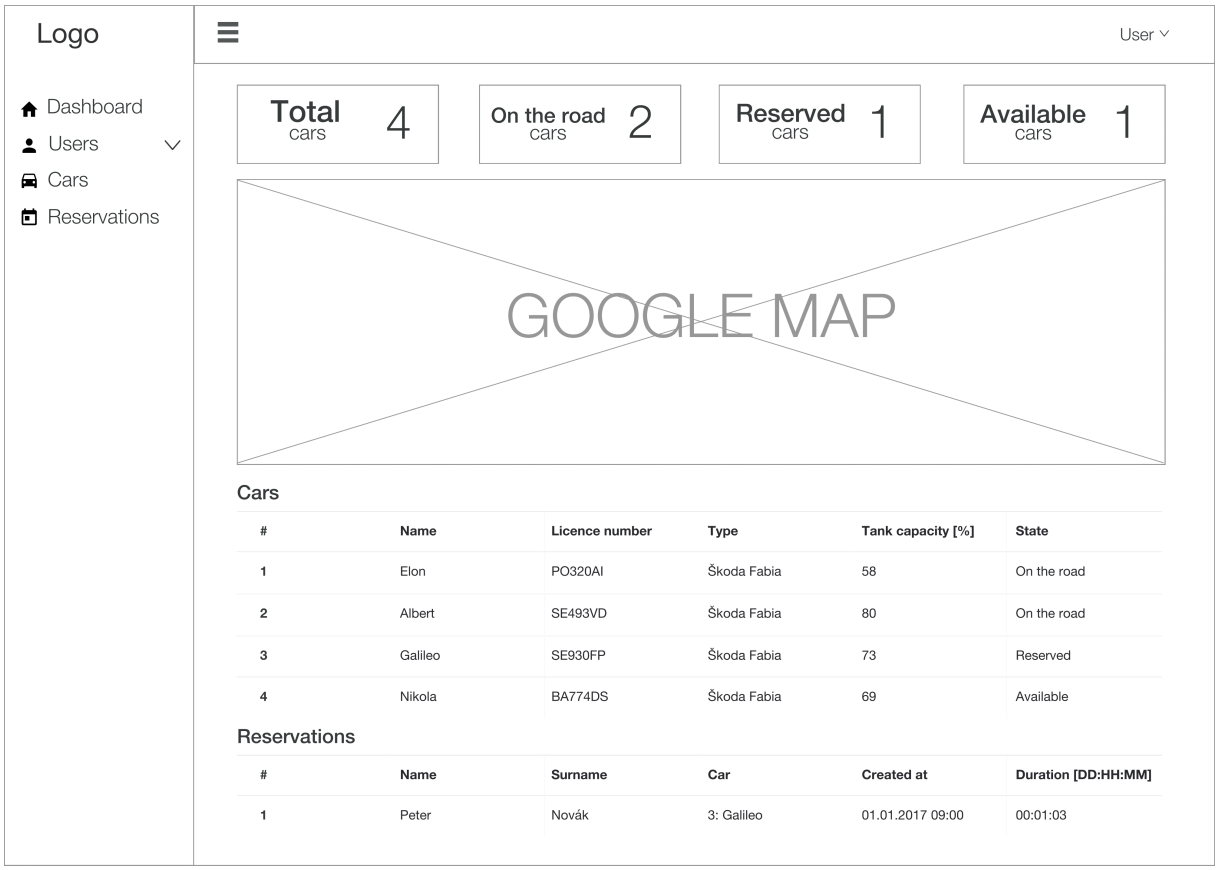
3.4 Návrh vzhľadu aplikácie

Pri návrhu užívateľského rozhrania sa kládol dôraz na intuitívnosť a prehľadnosť, podľa jednej z nefunkčných požiadaviek. Náhľady obrazoviek vznikali za účelom dať administrátorovi možnosť splniť aspoň jednu funkčnú požiadavku, ako aj logicky oddeliť časti správcovskej aplikácie.



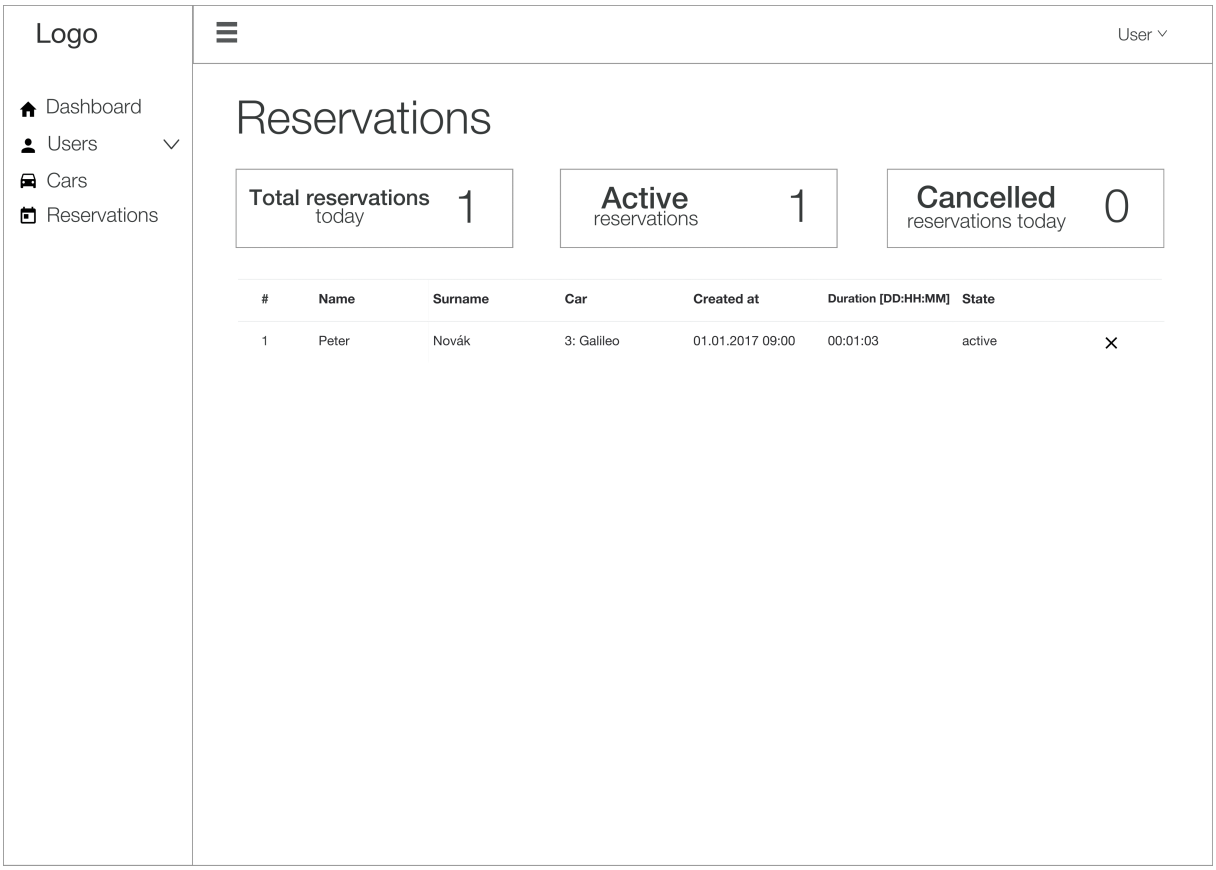
Obr. 3.1: Návrh obrazovky pre prihlásenie

Táto obrazovka vznikla na základe nefunkčnej požiadavky na Bezpečnosť aplikácie. Slúži na autentifikáciu administrátora pracujúceho s aplikáciou. Prostredníctvom užívateľského rozhrania, ktoré je zobrazené na obr. 3.1 sa administrátor prihlasuje do správcovskej aplikácie. Po zadaní prihlasovacích údajov, ktorými sú email a heslo administrátor klikne na prihlasovacie tlačidlo (Sign In). Ak systém vyhodnotí, že zadané údaje sú korektné, presmeruje administrátora na novú obrazovku, ktorej názov je Úvodná obrazovka. V prípade zadania neplatných prihlasovacích údajov ho aplikácia upozorní na túto skutočnosť a neumožní mu pokračovať v ďalšej práci s aplikáciou. Administrátor má možnosť prostredníctvom tohto užívateľského rozhrania resetovať svoje heslo, a to kliknutím na tlačidlo resetovať heslo (Forgot password?).



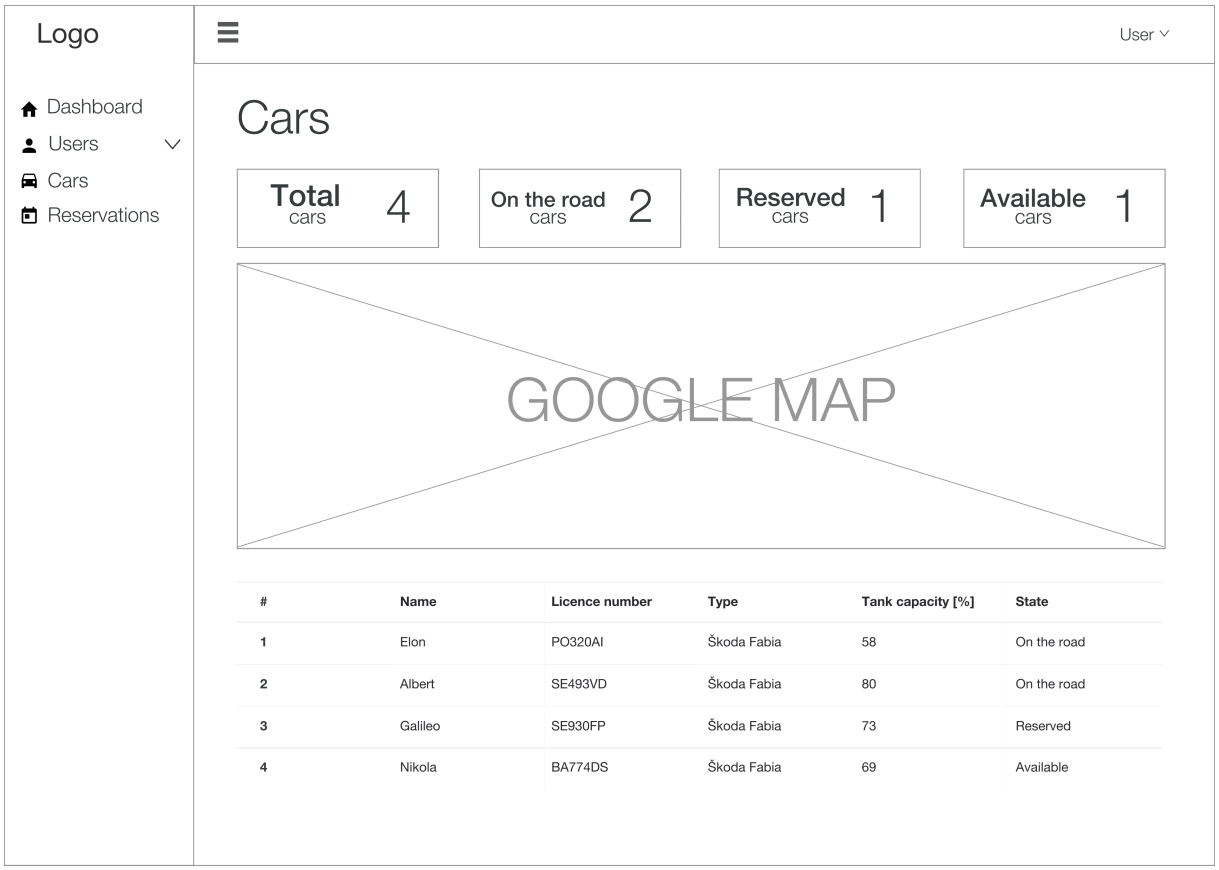
Obr. 3.2: Návrh úvodnej obrazovky

Užívateľské rozhranie, ktorého návrh je na obr. 3.2, slúži ako úvodná obrazovka aplikácie. Po korektnom prihlásení administrátora sa objaví obrazovka podľa návrhu. V pravej časti horného menu obrazovky má administrátor možnosť kliknúť na šípku dole. Po kliknutí sa zobrazí rozbaľovacie menu a v ňom má možnosť vybrať odkaz pomoc (Help) alebo na odkaz odhlásiť (Sign out), čím sa odhlási z aplikácie a je presmerovaný na stránku prihlásenia. Po kliknutí na odkaz pomoc (Help) sa adminstrátorovi zobrazia základné informácie pre prácu s aplikáciou Ľavé menu užívateľského rozhrania slúži na navigáciu na stránke. Horné a ľavé menu aplikácie sú rovnaké na všetkých obrazovkách, okrem obrazovky prihlásenia. Toto užívateľské rozhranie zobrazuje prehľadne vybrané informácie o celom systéme. V hornej časti administrátor vidí celkový počet áut, ktoré sú momentálne k dispozícii, koľko áut je momentálne požičaných, koľko je rezervovaných a koľko dostupných. Nižšie vidí aktuálnu polohu áut na Google Mape. Administrátorovi sú taktiež na obrazovke prehľadne v tabuľke zobrazené informácie o vozovom parku a o rezerváciách.



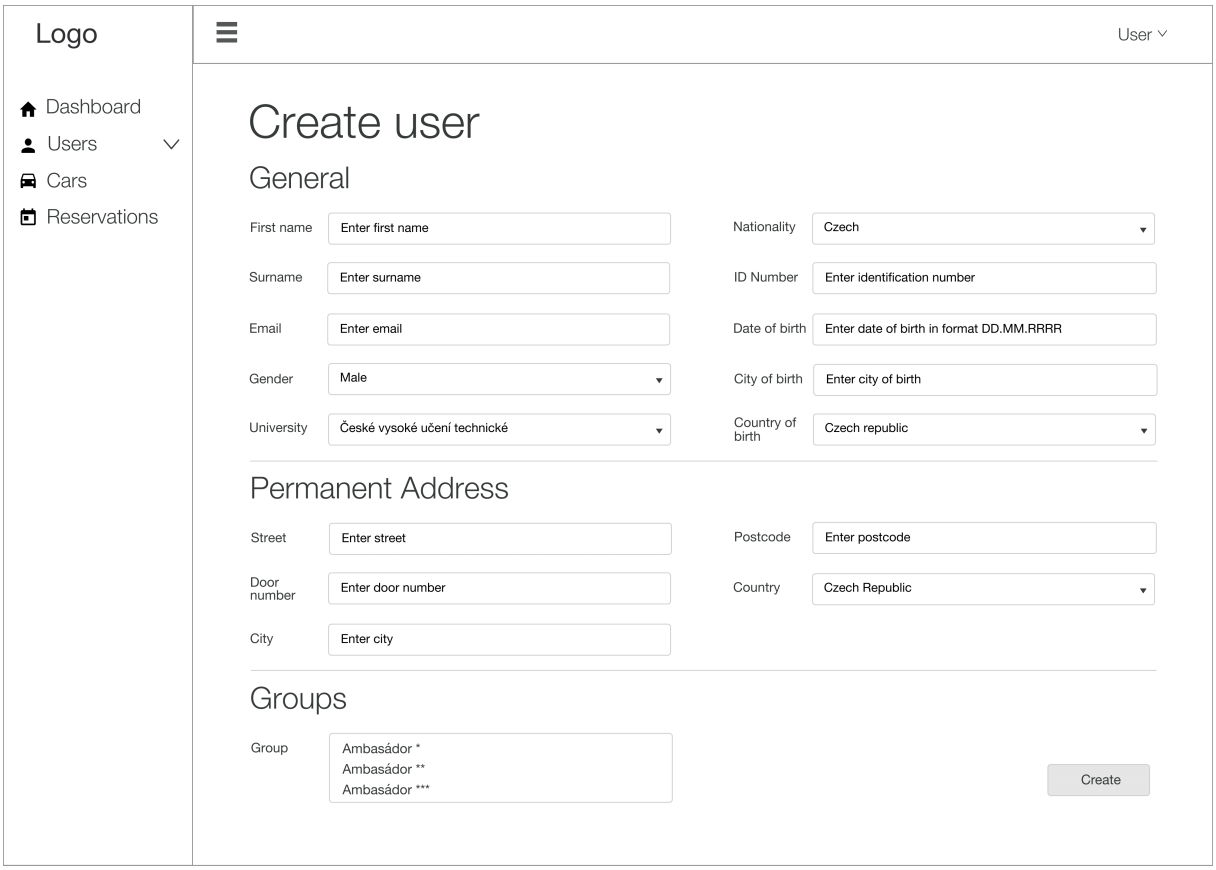
Obr. 3.3: Návrh obrazovky zobrazujúcej rezervácie

Táto obrazovka vznikla na základe funkčnej požiadavky na Zobrazenie všetkých rezervácií a jej návrh je na obr. 3.3. Obrazovka sa zobrazí po kliknutí na odkaz rezervácie (Reservations) v ľavom menu. Administrátor vidí celkový počet rezervácii, počet aktívnych i počet zrušených rezervácii. V tabuľke sa prehľadne zobrazujú všetky rezervácie. Pri každej rezervácií je uvedené kto rezerváciu vytvoril, ktoré auto si zarezervoval, kedy bola rezervácia vytvorená, ako dlho trvá rezervácia a jej stav. Administrátor má možnosť pomocou tohto užívateľského rozhrania zrušiť rezerváciu, podľa funkčnej požiadavky na Zrušenie rezervácie.



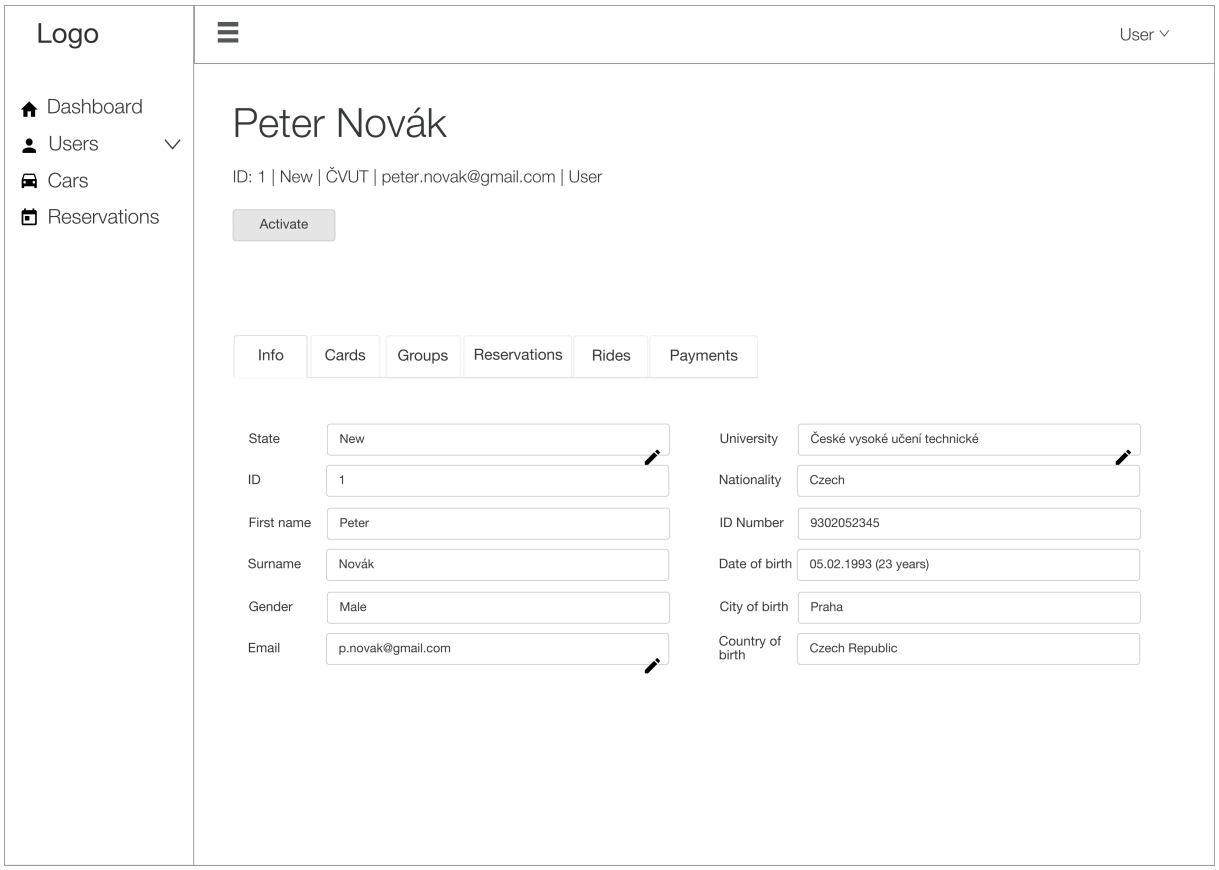
Obr. 3.4: Návrh obrazovky zobrazujúcej autá

Na obrazovke, ktorej návrh je na obr.3.4 má administrátor dostupné informácie o vozovom parku, podľa funkčnej požiadavky na Zobrazenie všetkých áut. Toto zobrazenie je dostupné po kliknutí na odkaz autá (Cars) v ľavom menu. Obrazovka zobrazuje celkový počet áut, koľko áut je momentálne požičaných, koľko je rezervovaných a koľko dostupných. Administrátor vidí aktuálnu polohu áut na Google Mape. Užívateľské rozhranie prehľadne v tabuľke zobrazuje taktiež informácie o každom aute, konkrétne jeho meno, štátnu poznávaciu značku, model, palivo v nádrži v percentách a stav automobilu.



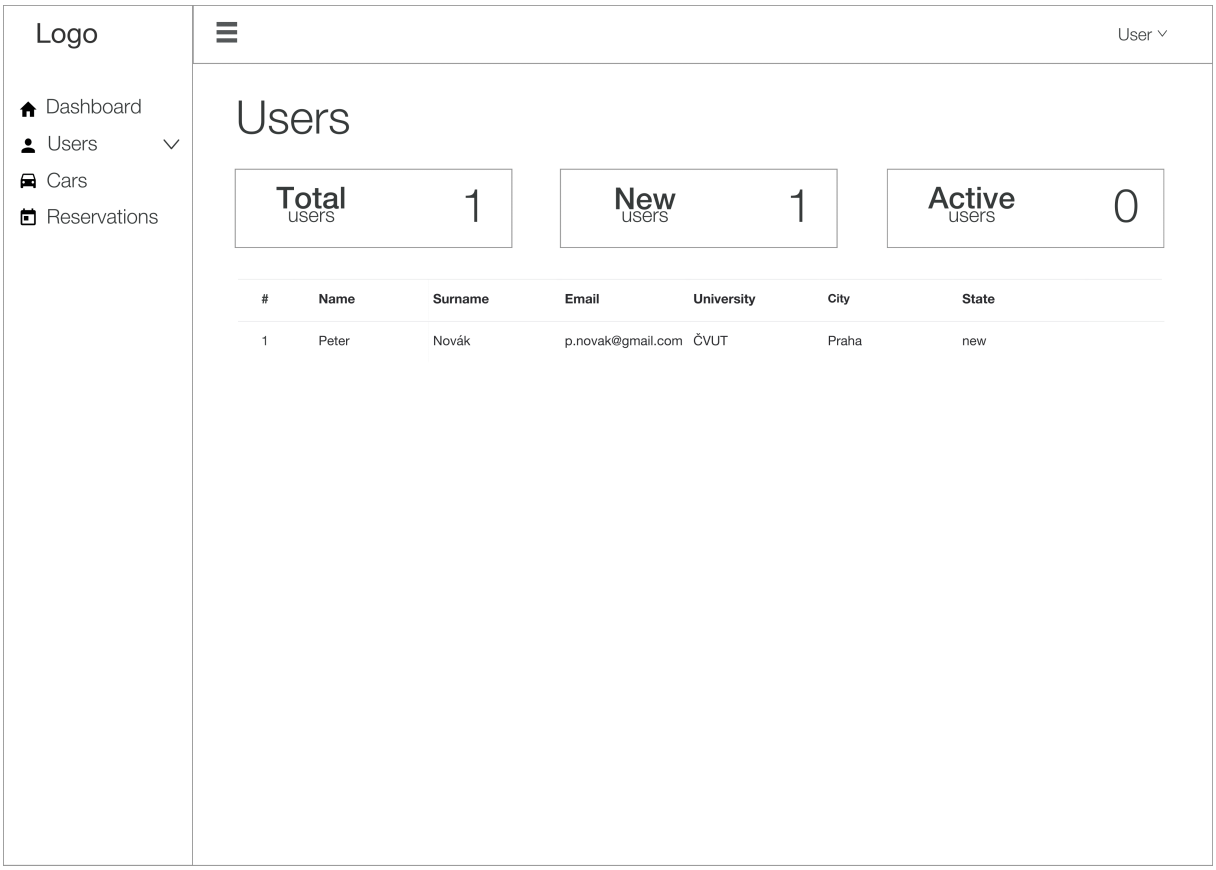
Obr. 3.5: Návrh obrazovky pre vytvorenie užívateľa

Podľa funkčnej požiadavky na vytvorenie užívateľa bolo vytvorené toto užívateľské rozhranie, ktoré ponúka administrátorovi formulár pre vyplnenie a odoslanie údajov o užívateľovi. Návrh tejto obrazovky je na obr. 3.5. Obrazovka je dostupná po kliknutí v ľavom menu na šípku pri odkaze užívatelia (Users). Po kliknutí sa zobrazí rozbaľovacie menu a v ňom administrátor zvolí odkaz vytvoriť užívateľa (Create users). Formulár obsahuje polia meno, priezvisko, email, pohlavie, univerzita, národnosť, rodné číslo, dátum narodenia, mesto narodenia, krajina narodenia a údaje o trvalej adrese. Užívateľa je možné priradiť k istej skupine užívateľov. Kliknutím na tlačidlo vytvoriť (Create) je prihlásený užívateľ notifikovaný o vytvorení a presmerovaný na obrazovku so všetkými užívateľmi.



Obr. 3.6: Návrh obrazovky zobrazujúcej detail užívateľa

Obrazovka ponúka detailné informácie o užívateľovi a jej návrh je na obr. 3.6. Je dostupná po kliknutí na konkrétneho užívateľa v tabuľke všetkých užívateľov, ktorá sa nachádza v obrazovke Zobrazenie všetkých užívateľov. Vznikla na základe funkčnej požiadavky na zobrazenie detailných informácií o užívateľovi. V hornej časti sa zobrazuje meno a priezvisko užívateľa, jeho identifikačné číslo, stav, univerzita kde študuje a jeho mail. Po stlačení tlačidla aktivovať (Activate) prejde užívateľ do stavu aktivovaný, avšak len v prípade, že má vyplnené všetky údaje, podľa funkčnej požiadavky na aktivovanie užívateľa. Ak nemá vyplnené všetky povinné údaje rozhranie upozorní administrátora na údaje, ktoré mu chýbajú. V strednej časti je možné prepínať na detailnejšie informácie o užívateľovi, ktoré sú zoradené v kategóriách. Základné kategórie sú doplňujúce informácie o užívateľovi, preukazy priradené k užívateľovi, skupiny, v ktorých je užívateľ členom, jeho rezervácie, jazdy a platby. Prostredníctvom tejto obrazovky je taktiež možné editovať užívateľa kliknutím na ikonu ceruzky v príslušnom formulárovom okne, podľa funkčnej požiadavky na editáciu užívateľa.



Obr. 3.7: Návrh obrazovky zobrazujúcej užívateľov

Toto užívateľské rozhranie zobrazuje informácie o všetkých užívateľoch, podľa funkčnej požiadavky na zobrazenie všetkých užívateľoch. Návrh tejto obrazovky je na obr. 3.7. Obrazovka je dostupná po kliknutí na odkaz užívatelia (Users) v ľavom menu. Administrátorovi zobrazuje celkový počet užívateľov, počet užívateľov v stave nový a aktívny. Prehľadne v tabuľke je možné vidieť základné informácie o každom užívateľovi, a to jeho meno, priezvisko, email, univerzitu kde študuje, v ktorom meste študuje a stav užívateľa.

3.5 Funkcionalita aplikácie

Webová správcovská aplikácia pre správu vozového parku je naprogramovaná využitím frameworku Angular 2, viď. kapitola 2.4.1. Jej funkcionalita pokrýva všetky funkčné požiadavky, ktoré sú na aplikáciu definované v kapitole 3.1.2. Využitím frameworku Angular 2 aplikácia beží vo webovom prehliadači, vďaka tomu systém spĺňa aj nefunkčnú požiadavku na multiplatformnosť. Dáta, ku ktorým zatiaľ nie je definované aplikačné rozhranie (napríklad autá), boli na serveri vložené do databázy priamo.

3.5.1 Komponenty a servisy systému

Základným stavebným blokom Angular aplikácie je komponenta, viď. kapitola 2.4.1. Webová správcovská aplikácia obsahuje spolu 13 komponent a 6 servís zabezpečujúcich beh a funkcionalitu aplikácie. Komponenty využívajú servisy systému, a to predovšetkým na odoslanie a prijímanie dat.

Komponenty webovej správcovskej aplikácie sú:

- **AppComponent komponenta** - je základnou komponentou systému, ktorá sa spúšťa ako prvá. Je definovaná v súbore app.component.ts.
- **LoginComponent komponenta** - komponenta zaobaľujúca prihlasovací formulár a jeho frontendovú validáciu. Je definovaná v login.component.ts. Využíva auth.service.ts na odoslanie užívateľom zadaných prihlasovacích údajov na server.
- **TopMenuComponent komponenta** - horné menu aplikácie, definované v top-menu.component.ts.
- **LeftMenuComponent komponenta** - ľavé menu aplikácie slúžiace pre navigáciu na stránke. Komponenta je definovaná v left-menu.component.ts.
- **DashboardComponent komponenta** - komponenta slúži pre zobrazenie nástenky s prehľadom informácií, pracuje s modelom Car a Reservation a využíva servisy pre komunikáciu so serverom. Je definovaná v dashboard.component.ts.
- **UserComponent komponenta** - komponenta pre zobrazenie informácií o užívateľoch. Je definovaná v user.component.ts, využívate model User a servisu user.service.ts.
- **UserCreateComponent komponenta** - komponenta pre pridanie užívateľa pracujúca s modelom User a využívajúca servisu user.service.ts pre odoslanie dát. Je definovaná v user-create.component.ts.
- **UserDetailComponent komponenta** - zobrazuje detailné informácie o užívateĺovi, využíva model User a servisu user.servise.ts. Je definovaná v súbore user-detail.component.ts.
- **CarComponent komponenta** - prezenčná vrstva komponenty zobrazuje autá spolu s informáciami, logika pracuje s modelom Car a servisou car.service.ts. Komponenta je definovaná v car.component.ts.
- **ReservationComponent komponenta** - stará sa o zobrazenie rezervácií. Je definovaná v reservation.component.ts, využíva Reservation model a servisu reservation.service.ts.
- **PageNotFoundComponent komponenta** - komponenta zobrazujúca informácie o neplatnej URI. Je definovaná v page-not-found.component.ts.

- **HelpComponent komponenta** - zobrazuje stručný užívateľský návod. Je definovaná v help.component.ts.

- **AlertComponent komponenta** - komponenta notifikuje užívateľa pri práci s užívateľským rozhraním. Je definovaná v alert.component.ts, bližšie je popísana v kapitole 3.5.4.

3.5.2 Modely

Modely v aplikácií sú triedy, ktoré definujú entity s ktorými aplikácia pracuje. V modely sú definované atribúty spolu s dátovými typmi. Aplikáciu tvoria modely Car, Position, Status, User, Reservation, University a Address.

3.5.3 Autentifikácia

Autentifikácia užívateľa prebieha odoslaním jeho prístupových údajov na server pri prihlásení. Server overí prijaté údaje voči záznamu v databáze, v prípade neplatných údajov generuje HTTP odpoveď s kódom 401 - Neoprávnený prístup. Aplikácia notifikuje užívateľa o zadaní neplatných údajov a vyžaduje údaje znova. V prípade platných prístupových údajov server odpovedá s kódom 200, čo znamená že údaje boli v poriadku a v odpovedi posiela vygenerovaný JWT token pre daného užívateľa. Tento token sa odosiela s každou požiadavkou na server a slúži ako informácia o užívateľovi, ktorý so serverom komunikuje. Token je generovaný s každou platnou požiadavkou
na server. Užívateľ po odoslaní neplatného tokenu je z aplikácie automaticky odhlásený. Automatické odhlásenie užívateľa môže nastať po jeho nečinnosti dlhšej ako 30 minút alebo po 8 hodinách práce s aplikáciou. Ide o bezpečnostné riešenia pre prevenciu využitia aplikácie neoprávnenými užívateľmi, stanovené doby sa môžu meniť podľa potrieb praxe.

Komunikácia medzi aplikáciou a serverom prebieha na šifrovanom protokole HTTPS. Komunikáciu medzi serverom a aplikáciou popisuje sekvenčný diagram na obr. B.1 v prílohe B.

3.5.4 Navigácia na stránke

Angular Router slúži pre navigáciu na stránke. Užívateľ môže využitím menu zobrazovať rôzne komponenty systému, čím sa mu mení užívateľské rozhranie. Modul app-routing.mogule.ts slúži pre definovanie týchto ciest. Každá cesta má definovanú path, časť URI na ktorú reaguje, napríklad */cars* a komponentu, ktorá sa spustí. Medzi cestami je definovaná aj predvolená cesta, ktorej komponenta sa spustí ak užívateľom definovaná cesta nie je definovaná. V takom prípade sa spustí PageNotFoundComponent komponenta.

3.5.5 Guards

Guards v aplikácií slúžia na ochranu pred nedovoleným prístupovom užívateľa do častí systému, kde nemá prístup. Užívateľ si vhodnou modifikáciou URI môže napríklad zobraziť komponentu s informáciou o vozovom parku ešte pred prihlásením. Zabrániť spusteniu komponent slúžia práve Guards, ktoré definujú validačné pravidlo a sú zapísane v Routri pri konkrétnej ceste.

V aplikácií sú definované dve validačné pravidlá:

1. Pre zabránenie neprihlásenému užívateľovi na prístup ku inej komponente ako LonginComponent komponente, slúžiacej na prihlásenie užívateľa.
2. Pre zabránenie prihlásenému užívateľovi na prístup ku LoginComponent komponente, prihlásený užívateľ sa musí najprv odhlásiť.

3.5.6 Notifikácie

Dôležitým aspektom z hľadiska užívateľskeho rozhrania sú notifikácie pre užívateľa. Užívateľ aplikácie musí vždy dostať adekvátnu odpoveď na svoju interakciu s aplikáciou, napríklad či sa mu užívateľa podarilo úspešne vytvoriť alebo nastala chyba a užívateľ vytvorený nebol.

V aplikácií sú implementované upozornenia, ktoré užívateľa informujú o zmenách. Komponenta AlertCopoment definuje ako notifikácia vyzerá v externom súbore alert.component.html a taktiež na pozadí čaka na prijatie správy z inej komponenty alebo servisy.

3.6 Užívateľské rozhranie

Užívateľské rozhranie webovej aplikácie vychádza z návrhu užívateľského rozhrania v kapitole 3.4. Pri implementácií bol využitý framework Bootstrap pre zaistenie responzivity. Súčasťou aplikácie je aj návod pre užívateľa systému, ktorý zarhňuje popis základnej práce s aplikáciou. Samotné užívateľské rozhranie bolo navrhnuté so zameraním na jednoduchú a intuitívnu prácu. Náhľady užívateľského rozhrania môžete nájsť v prílohe C.

3.7 Nasadenie aplikácie

Pre kompiláciu zdrojových súborov webovej aplikácie sa využíva Angular CLI, konkrétne príkaz **ng build –prod**. Spustením tohto príkazu sa v projekte vytvorí priečinok *dist*, ktorý obsahuje HTML súbor, CSS súbor, JavaScriptové súbory a favicon. Obsah *dist* nakopírujeme do príslušného adresára webového servera.

Kapitola 4

Záver

Výsledkom bakalárskej práce je webová aplikácia pre správu vozového parku pre systém zdieľania aut viacerými užívateľmi. Aplikácia splňuje funkcionalitu definovanú v tejto práci, podporuje vytvorenie užívateľa, editáciu užívateľa, aktiváciu užívateľa, vymazanie užívateľa, zobrazenie všetkých užívateľov, zobrazenie všetkých áut spolu s ich aktuálnou polohou na mape a zobrazenie všetkých rezervácií. Aplikácia beží v rámci webového prehliadača, čím je nezávislá na type zariadenia alebo operačného systému, má intuintívne užívateľské rozhranie a prístup do aplikácie vyžaduje overenie prihlasovacích údajov užívateľa. Návod pre užívateľa je dostupný v aplikácií. Navrhnutá aplikácia slúži ako správcovská aplikácia projektu Uniqway.

V ďalšom vývoji aplikácie navrhujem:

- **Rozšírenie funkcionality aplikácie** - doplniť funkcionalitu podľa požiadaviek uvedených v prílohe D.1. Aplikácia s doplnenou funkcionalitou bude plnohodnotná webová správcovská aplikácia a môže slúžit v rámci testovacej fáze projektu Uniqway.
- **Pridanie užívateľských rolí** - pridať správcovské role do aplikácie. Využitie rolí oddelí prístup rôznych užívateľov do častí systému. Návrh rolí je bližšie popísaný v prílohe D.2.
- **Rozšírenie aplikačného rozhrania na serveri** - pri rozširovaní funkcionality aplikácie je taktiež potrebné myslieť na rozšírenie aplikačného rozhrania na serveri. Aktuálne navrhnuté aplikačné rozhranie bude taktiež doplnené o novopridané atribúty. Aplikačné rozhranie by malo taktiež v budúcnosti podporovať možnosť filtrovania dát v odpovedi zo servera.
- **Pravidla pre heslá** - nastavenie politiky bezpečnosti pre správu hesiel. Definovanie požiadaviek na zložitosť hesla a jeho expiráciu.
- **Vylepšenie grafického rozhrania aplikácie** - grafické rozhranie aplikácie ponúka stále možnosti na vylepšenia, jeho modfikácie je otázkou zmeny prezenčnej vrstvy aplikácie.

Zoznam bibliografických odkazov

[1] IEEE Recommended Practice for Software Requirements Specifications. *IEEE Std 830-1998*. Oct 1998, s. 1–40. doi: 10.1109/IEEESTD.1998.88286.

[2] Systems and software engineering – Vocabulary. *ISO/IEC/IEEE 24765:2010(E)*. Dec 2010, s. 1–418. doi: 10.1109/IEEESTD.2010.5733835.

[3] BIERMAN, G. – ABADI, M. – TORGERSEN, M. *Understanding TypeScript*, s. 257–281. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014. doi: 10.1007/978-3-662-44202-9_11. Dostupné z: `http://dx.doi.org/10.1007/978-3-662-44202-9_11`. ISBN 978-3-662-44202-9.

[4] CHOUDHARY, V. Software as a Service: Implications for Investment in Software Development. In *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on*, s. 209a–209a, Jan 2007. doi: 10.1109/HICSS.2007.493.

[5] FAIN Y, M. A. *Angular 2 Development with TypeScript*. : Manning Publications, 2007. ISBN 9781617293122.

[6] GARRETT, J. J. et al. Ajax: A new approach to web applications. 2005.

[7] MIKOWSKI, M. S. – POWELL, J. C. Single page web applications. *B and W*. 2013.

[8] SANDHU, R. S. et al. Role-based access control models. *Computer*. Feb 1996, 29, 2, s. 38–47. ISSN 0018-9162. doi: 10.1109/2.485845.

[9] SCHMIDT, D. C. Applying patterns and frameworks to develop object-oriented communication software. *Handbook of programming languages*. 1997, 1.

[10] WOOD, L. et al. Document object model (DOM) level 1 specification. *W3C recommendation*. 1998, 1.

Dodatok A

Zoznam použitých skratiek

AJAX Asynchronous JavaScript And XML

API Application programming interface

CLI Command Line Interface

CSS Cascading Style Sheets

DNS Domain Name Server

DOM Document Object Model

HTML Hyper Text Markup Language

HTTP Hyper Text Transfer Protocol

HTTPS Hyper Text Transfer Protocol Secure

JSON JavaScript Object Notation

JWT Java Web Tokens

OP Občiansky preukaz

RFID Radio Frequency IDentification

SPA Single page application

URI Uniform Resource Identifier

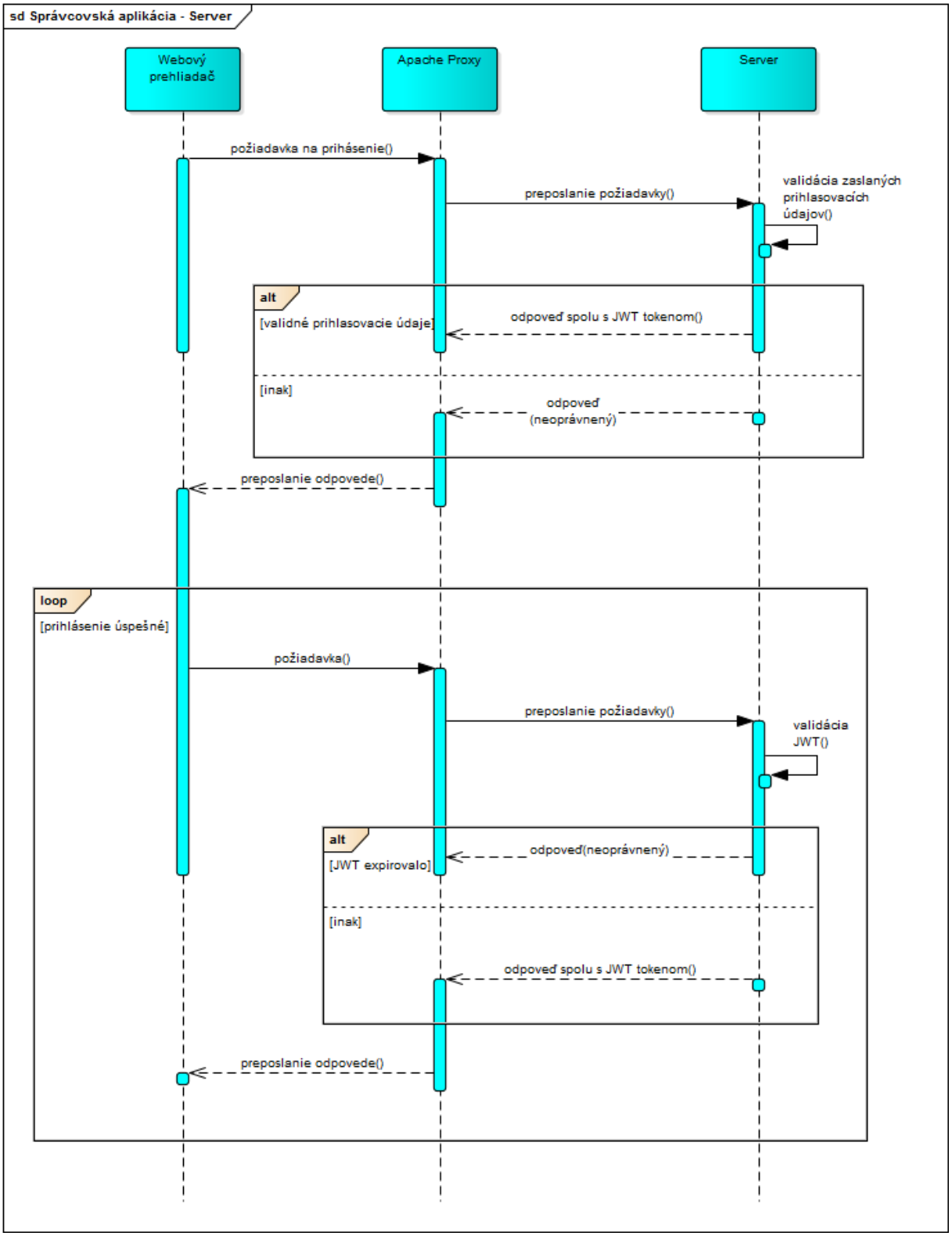
VP Vodičský preukaz

XML eXtensible Markup Language

ŠPZ Štátna poznávacia značka

Dodatok B

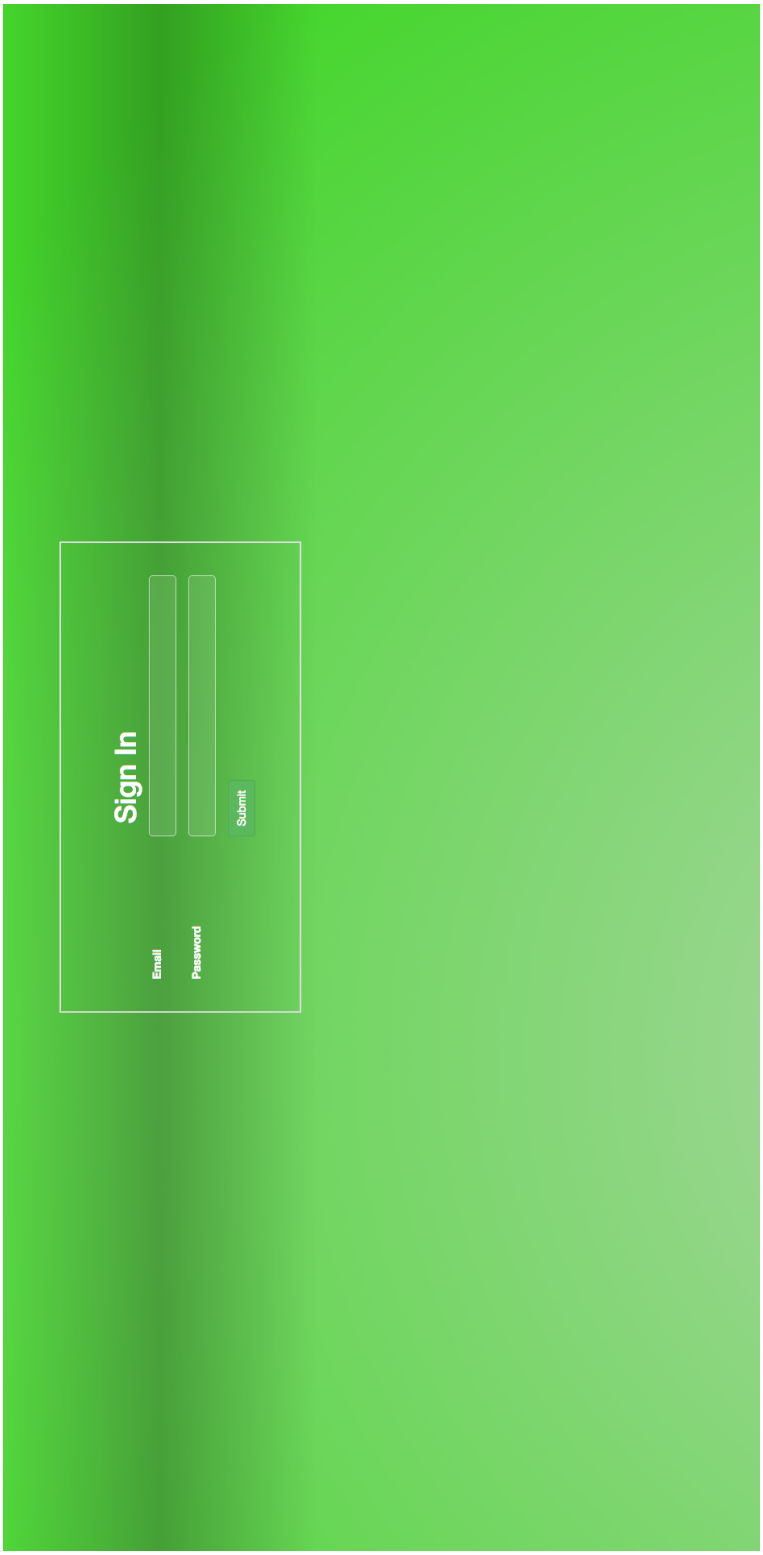
Diagram komunikácie aplikácie so serverom



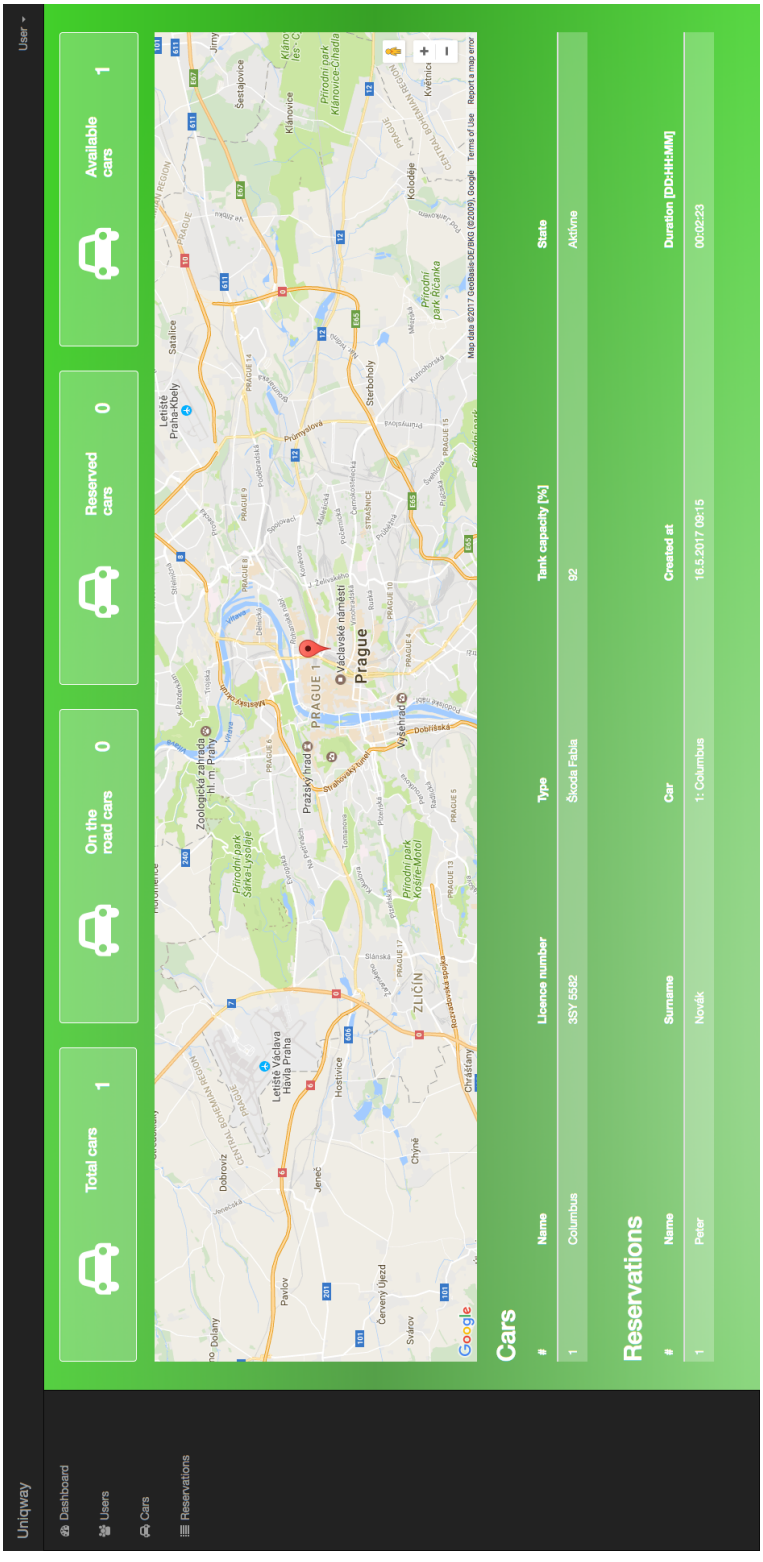
Obr. B.1: Sekvenčný diagram komunikácie správcovskej aplikácie so serverom

Dodatok C

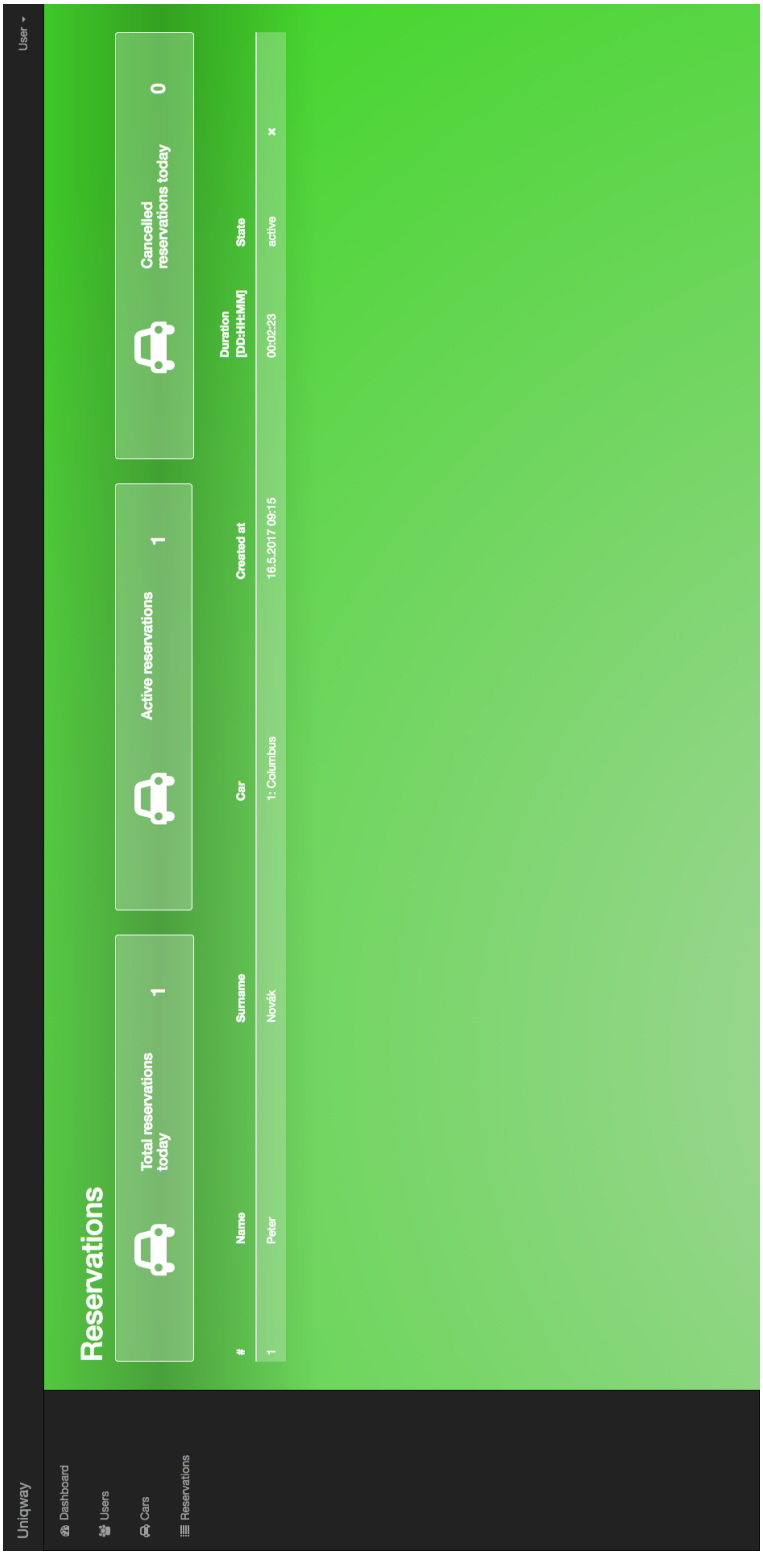
Náhľady užívateľského rozhrania



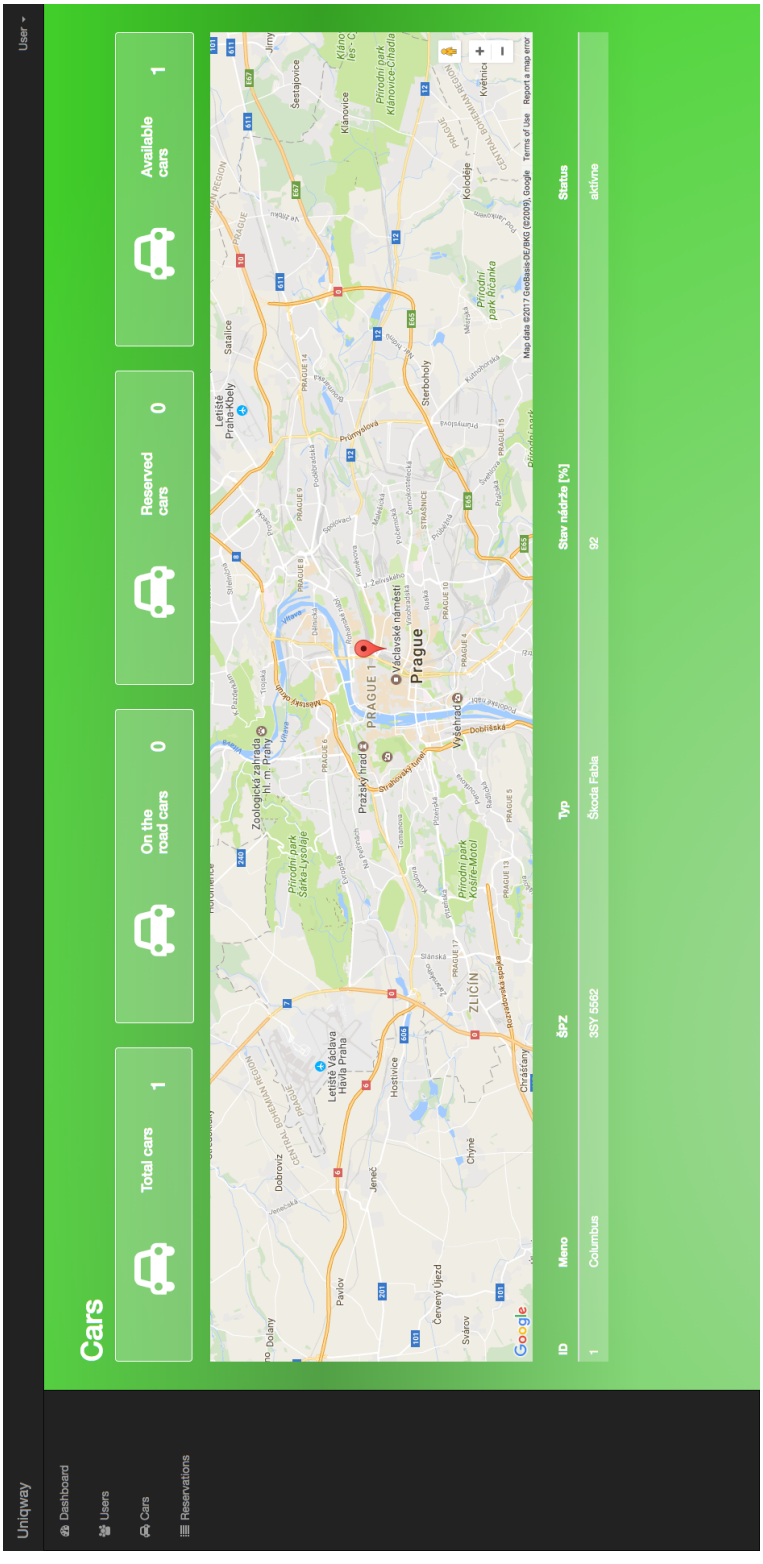
Obr. C.1: Obrazovka pre prihlásenie



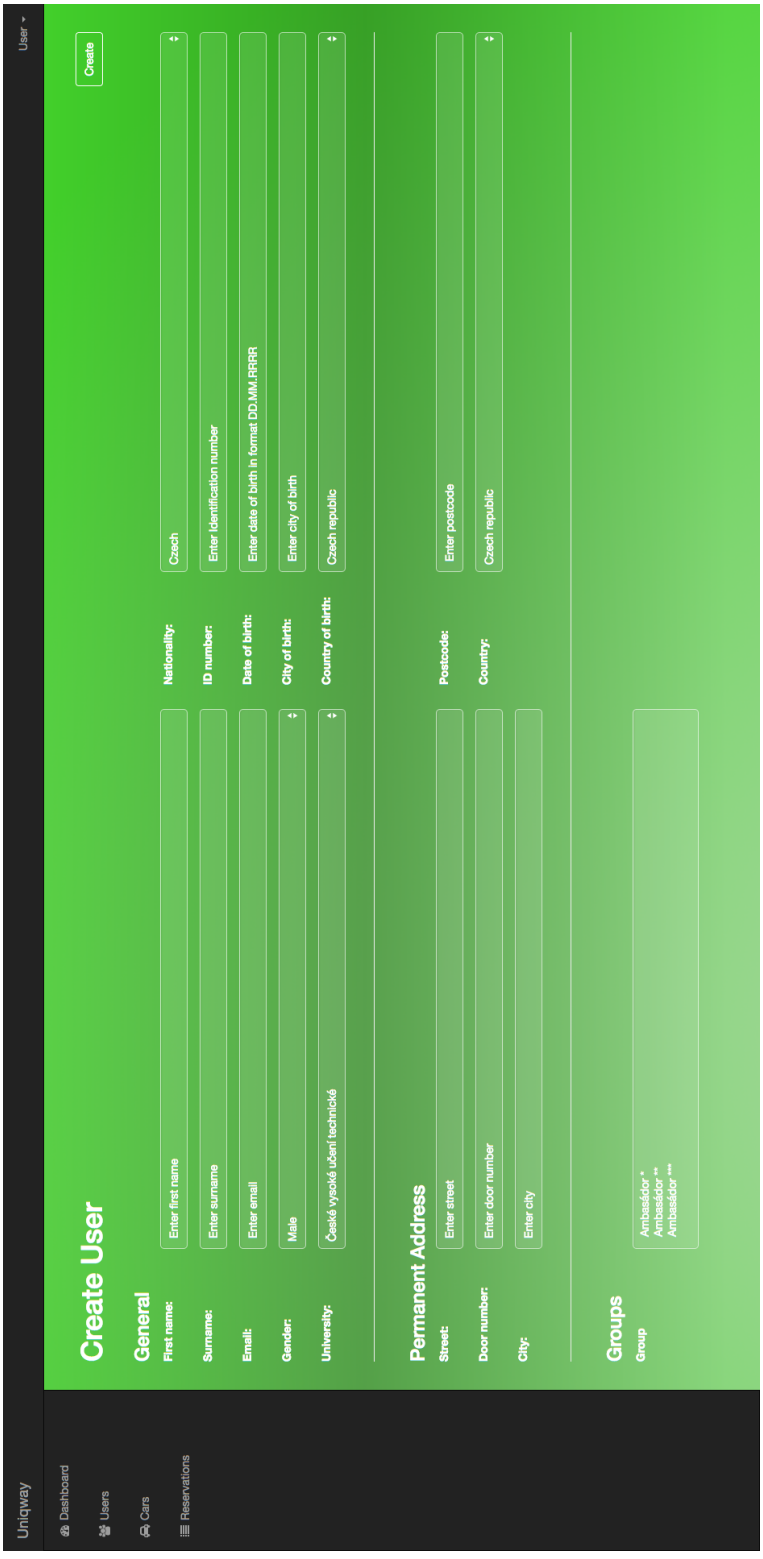
Obr. C.2: Úvodná obrazovka



Obr. C.3: Obrazovka zobrazujúca rezervácie

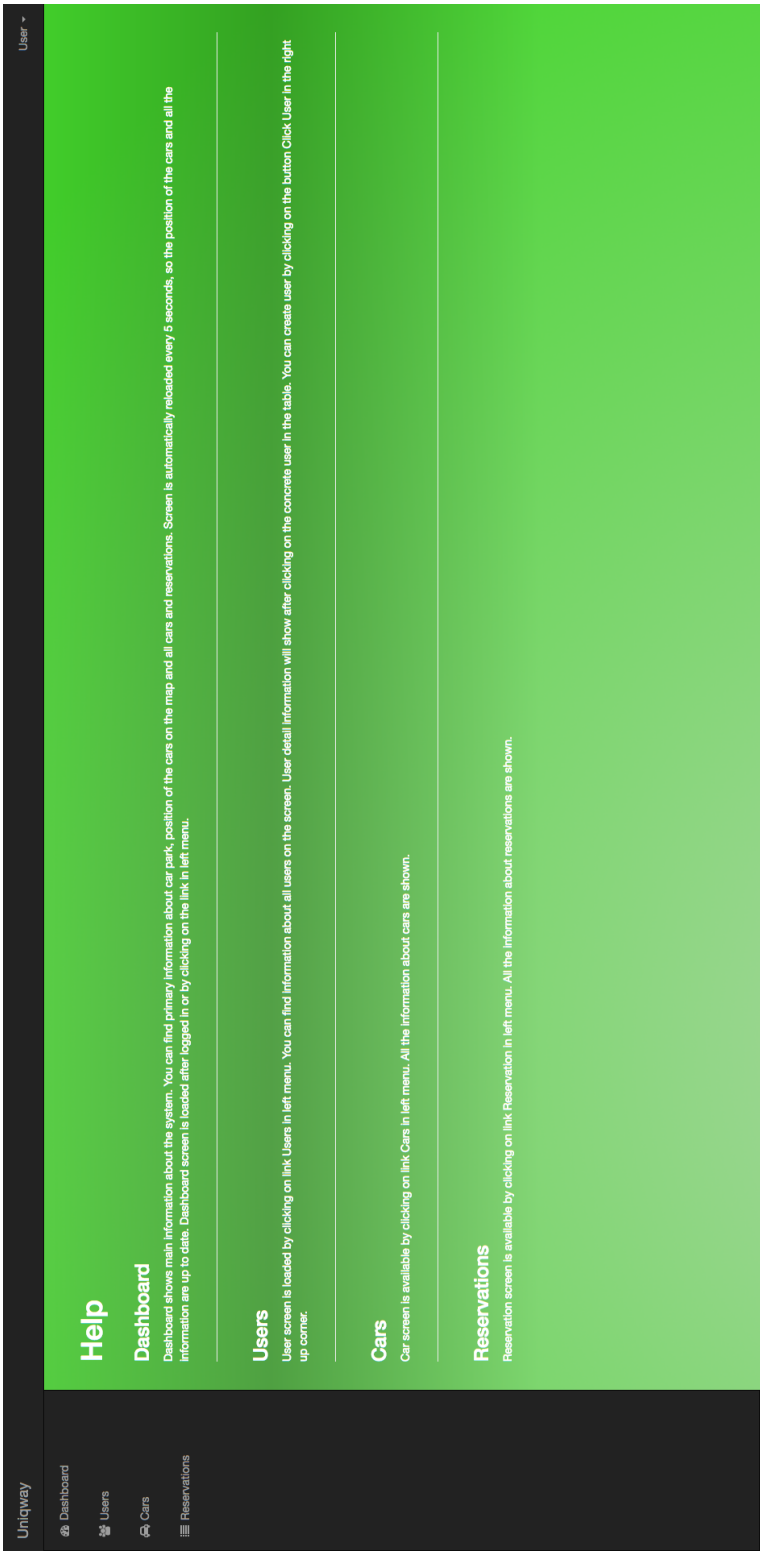


Obr. C.4: Obrazovka zobrazujúca autá



Obr. C.5: Obrazovka pre vytvorenie užívateľa

Obr. C.6: Obrazovka zobrazujúca užívateľov



Obr. C.7: Obrazovka zobrazujúca návod

Dodatok D

Návrh rozšírenia funkcionality a správcovských rolí

D.1 Rozšírenie funkcionality aplikácie

Funkčné požiadavky na rozšírenie funkcionality sú rozdelené do častí Autá, Užívatelia, Správcovia, Rezervácie, Jazdy, Komunikačné jednotky, Palivové platobné karty, Štatistiky a Systém.

D.1.1 Autá

- **Pridanie auta** - systém umožní správcovi vozového parku vytvoriť záznam o aute. Auto bude v databáze obsahovať informácie ako poradové číslo auta, číslo technického preukazu, doba platnosti technického preukazu, majiteľ auta, značka, typ auta, VIN karosérie, farba karosérie, štátna poznávacia značka, počet miest na sedenie, druh paliva/ zdroj energie, objem nádrže v litroch, normovaná spotreba v litroch na 100 km, dojazd v km pri plnej nádrži, dojazd v km pri plne nabitej batérii, druh prevodovky, typ pneumatiky, dátum nasledujúcej výmeny pneumatík, číslo rezervného kľúča od auta, číslo palivovej platobnej karty na nákup PHM, číslo komunikačnej jednotky, dátum nasledujucej kontroly technickej kontroly, dátum nasledujucej emisnej kontroly, stav tachometra pre vykonanie následujúcej záručnej prehliadke, stav tachometra pre vykonanie nasledujúcej výmene oleja, aktuálny stav tachometra auta, stav vozidla.

- **Editácia auta** - systém umožní správcovi vozového parku upravovať informácie o aute.

- **Vymazanie auta** - systém umožní správcovi vozového parku odstrániť auto, presnejšie zmeniť jeho stav na Unavaliable (Nedostupné). V tomto stave sa auto nebude zobrazovať užívateľom v mobilnej aplikácii a ani správcom vozového parku v prehľade aut, či na mape.

- **Zmena stavu auta** - systém umožní správcovi vozového parku zmeniť stav auta. Stavy áut budú Available (Dostupné), Reserved (Rezervované), On the road (Vypožičané), Maintenance (Servis), Unavailable (Nedostupné).

- **Zobrazenie detailných informácií o aute** - systém umožní správcovi vozového parku zobraziť detailné informácie o aute. Správcovi sa po kliknutí na auto v tabuľke áut zobrazí okno s popisom informácií o aute špecifikovanými v požiadavke *pridanie auta* spolu s históriou jázd, informáciách o poškodeniach, priemernou spotrebou a iných finančných nákladov na auto.

- **Zobrazenie informácií o tankovaní** - systém umožní správcovi vozového parku zobraziť informácie o tankovaní. Informácie o tankovaní budú pochádzať z dvoch zdrojov, a to z modulu v aute, ktorý odosiela stav paliva v nádrži a dotazu na API spoločnosti prevádzkujúcej palivovú platobnú kartu priradenú k autu. Dáta o čase a množstve natankovaného paliva sa musia zhodovať, v opačnom prípade je informovaný správca vozového parku.

- **Pridanie záznamu o poškodení vozidla** - Systém umožní správcovi vozového parku zaevidovať poškodenie auta. Informácie o poškodení pochádzajú

z formulára v klientskej aplikácií. Poškodenia môžeme klasifikovať do skupín, a to brzdy, motor, prevodovka, karoséria, osvetlenie, lak, sklá, pneumatiky, prevádzkové kvapaliny (olej, voda v ostrekovači, brzdová kvapalina) a iné.

- **Zobrazenie informácií o poškodeniach** - systém umožní zobraziť správcovi vozového parku informácie o poškodení vozidiel.

- **Pridanie informácie o finančných nákladoch na auto** - systém umožní správcovi vozového parku zaevidovať finančné náklady na konkrétne auto (oprava, údržba, umývanie auta, čistenie interiéru, zákonná poistka, havarijná poistka, dialničná známka ...).

D.1.2 Užívatelia

- **Vyhľadávanie užívateľov** - systém umožní správcovi užívateľov vyhľadávať užívateľov a to podľa priezviska, čísla OP a čísla VP.

- **Pridanie užívateľa na Čiernu listinu** - systém umožní správcovi užívateľov na určité časové obdobie zneprístupniť využívanie carsharingu konkrétnemu užívateľovi. Dôvody pre toto rozhodnutie môžu byť napríklad sústavný neporiadok v odovzdanom aute, neskoré vracanie vypožičaného auta a pod.. Daný užívateľ bude o zaradení na Čiernu listinu informovaný e-mailom.

- **Pridanie užívateľa do skupiny užívateľov** - systém umožní správcovi užívateľov pridať užívateľa do skupiny užívateľov. Skupiny užívateľov môžu tvoriť hierarchickú štruktúru, alebo slúžiť na zvýhodnenie určitej skupiny na predom definované časové obdobie.

D.1.3 Správcovia

- **Pridanie správcu** - systém umožní administrátorovi pridať správcu. Správca má prístup do webovej správcovskej aplikácie. Po vytvorení je správca automaticky v stave Active (Aktívny).

- **Editácia správcu** - systém umožní administrátorovi editovať informácie o správcovi. Jedine administrátor môže meniť správcovi jeho role, čím mu pridáva práva pre prístup do rôznych častí správcovskej aplikácie.

- **Vymazanie správcu** - systém umožní administrátorovi vymazať správcu, presnejšie zmeniť jeho stav na Inactive (Neaktívny). Správca v stave Inactive (Neaktívny) nemá prístup do správcovskej aplikácie.

- **Zobrazenie detailných informácií o správcovi** - systém umožní administrátorovi zobraziť detailné informácie o správcovi. Medzi detailné informácie bude patriť dátum pridania správcu do systému a dátum pridania jednotlivých rolí.

- **Zobrazenie všetkých správcov** - systém v tabuľke zobrazí administrátorovi všetkých správcov. Pri tomto náhľade zobrazí iba základné údaje (poradové

číslo správcu, meno, priezvisko, pridelené správcovské role), po kliknutí na konkrétneho správcu zobrazí jeho detail.

- **Vyhľadávanie správcu** - systém umožní administrátorovi vyhľadať správcu. Vyhľadávať sa bude podľa priezviska osoby, čísla OP.
- **Pridanie správcovskej role správcovi** - Systém umožní administrátorovi pridať rolu správcovi. Správcovská rola zabezpečuje oprávnenie pre prístup do rôznych častí správcovskej aplikácie. Role Dispečer, Ekonóm, Správca užívateľov, Správca vozového parku, Administrátor. Jeden správca môže mať súčasne viac správcovských rolí. Rola Administrátor je nadradená všetkým ostatným rolám.
- **Pridanie záznamu o podpise zmluvy o Ochrane osobných údajov ku správcovi** - systém bude evidovať pri informáciách o správcovi aj záznam o podpise zmluvy o Ochrane osobných údajov. Zmluva musí byť podpísaná každým správcom užívateľov ako aj administrátorom.
- **Zobrazenie upozornení správcovi** - systém umožní všetkým správcom dostávať upozornenia o zmene dôležitých informácií v systéme. Upozornenia môžu byť formou emailu alebo ako samostatné správy v správcovskej aplikácií.
- **Možnosť kontaktovať užívateľa** - systém umožní všetkým správcom kontaktovať užívateľa. Správcovská aplikácia môže odoslať e-mail danému užívateľovi, či mu poslať priamo správu do mobilnej aplikácie.
- **Možnosť kontaktovať skupinu užívateľov** - systém umožní všetkým správcom kontaktovať skupinu užívateľov. Správcovská aplikácia môže odoslať e-mail danej skupine užívateľ, alebo im priamo poslať správu do mobilnej aplikácie.

D.1.4 Rezervácie

- **Zmena stavu rezervácie** - systém umožní dispečerovi zmeniť stav rezervácie. Rôzne stavy rezervácie sú predmetom ďalšej analýzy.

D.1.5 Jazdy

- **Zobrazenie detailných informácií o jazde** - systém prehľadne zobrazí dispečerovi detailné informácie o jazde. Medzi detailné informácie patrí poradové číslo jazdy, užívateľ, celkový čas jazdy HH.MM.SS, celková cena jazdy v CZK, počet najazdených kilometrov, poradové číslo rezervácie, dátum a čas začiatku výpožičky, dátum a čas ukončenia výpožičky, počiatočný stav tachometra v km, koncový stav tachometra v km, počiatočný stav nádrže v litroch, konečný stav nádrže v litroch, tankovanie počas jazdy, prehľad trasy na Google mape.
- **Zobrazenie všetkých jázd** - systém v tabuľke zobrazí dispečerovi všetky jazdy. V tabuľke sa zobrazujú len základné údaje (poradové číslo jazdy, užívateľ, celkový čas jazdy, celková cena jazdy, počet najazdených kilometrov) a detail je dostupný po kliknutí na konkrétnu jazdu.

- **Vyhľadávanie jázd** - systém umožní dispečerovi vyhľadať jazdu. Jazdu je možné vyhľadať podľa: rezervácie, dátumu a času, užívateľa, auta.

D.1.6 Cenník

- **Pridanie cenníka** - systém umožní správcovi s rolou ekonóm pridať cenník. Cenník obsahuje aktuálne ceny, ktoré sa využívajú pri výpočte ceny za jazdu. Cenník je stále platný na určité konkrétne obdobie od DD.MM.YY do DD.MM.YY.
- **Editácia cenníka** - systém umožní správcovi s rolou ekonóm zmeniť informácie v cenníku.
- **Vymazanie cenníka** - systém umožní správcovi s rolou ekonóm vymazať cenník, presnejšie zmeniť jeho stav na "Neplatný". Zmene však musí predchádzať pridanie aktuálneho cenníka a to tak, aby systém mal vždy informácie pre výpočet ceny.
- **Zobrazenie detailných informácií o cenníku** - systém umožní správcovi s rolou ekonóm zobraziť detailné informácie (ceny pre jednotlivé balíčky- pri každom cena za km a hodinu) o cenníku.
- **Zobrazenie všetkých cenníkov** - systém v tabuľke zobrazí správcovi s rolou ekonóm všetky cenníky. Detailné informácie sú dostupné po kliknutí na konkrétny cenník.

D.1.7 Komunikačné jednotky

- **Pridanie komunikačnej jednotky** - systém umožní správcovi vozového parku vytvoriť záznam o komunikačnej jednotku. Pri komunikačnej jednotke bude v databáze uložené poradové číslo komunikačnej jednotky, typ komunikačnej jednotky, výrobné číslo a verzia firmware.
- **Editácia komunikačnej jednotky** - systém umožní správcovi vozového parku zmeniť informácie o komunikačnej jednotke.
- **Vymazanie komunikačnej jednotky** - systém umožní správcovi vozového parku vymazať komunikačnú jednotku, presnejšie zmeniť jej stav na Unavailable (Nedostupná). Informácie z tejto komunikačnej jednotky naďalej ostávajú v systéme, avšak už nie je možné priradiť ju k autu.
- **Zobrazenie detailných informácií o komunikačnej jednotke** - systém zobrazí správcovi vozového parku detailné informácie o komunikačnej jednotke.
- **Zobrazenie všetkých komunikačných jednotiek** - systém v tabuľke zobrazí správcovi vozového parku všetky komunikačné jednotky, ktorých detailné informácie sú dostupné po kliknutí na konkrétnu komunikačnú jednotku.
- **Vyhľadávanie komunikačnej jednotky** - systém umožní správcovi vozového parku vyhľadať komunikačnú jednotku. Vyhľadávanie je možné podľa jej výrobného čísla.

D.1.8 Palivové platobné karty

- **Pridanie platobnej karty** - systém umožní správcovi vozového parku vytvoriť záznam o palivovej platobnej karte. Túto kartu budú využívať užívatelia pri tankovaní na čerpacích staniciach. Systém si pri vytvorení bude ukladať poradové číslo palivovej platobnej karty, číslo palivovej platobnej karty, denný limit na nákup pohonných hmôt v CZK, spoločnosť sprostredkúvavajúca palivovú platobnú kartu.
- **Editácia platobnej karty** - systém umožní správcovi vozového parku editovať platobnú kartu.
- **Vymazanie platobnej karty** - systém umožní správcovi vozového parku vymazať platobnú kartu, presnejšie zmeniť jej stav na Inactive (Neaktívna). Platobná karta v stave Inactive (Neaktívna) už viac nebude zobrazovaná v prehľade všetkých kariet.
- **Zobrazenie detailných informácií o platobnej karte** - systém zobrazí správcovi vozového parku detailné informácie o platobnej karte. Medzi ne patrí aj zoznam transakcií s danou kartou.
- **Zobrazenie všetkých platobných kariet** - systém v tabuľke zobrazí správcovi vozového parku všetky platobné karty. Detailné informácie o karte sú dostupné po kliknutí na konkrétnu kartu.
- **Vyhľadávanie platobnej karty** - systém umožní správcovi vozového parku vyhľadať platobnú kartu. Vyhľadávanie je možné podľa čísla karty.

D.1.9 Štatistiky

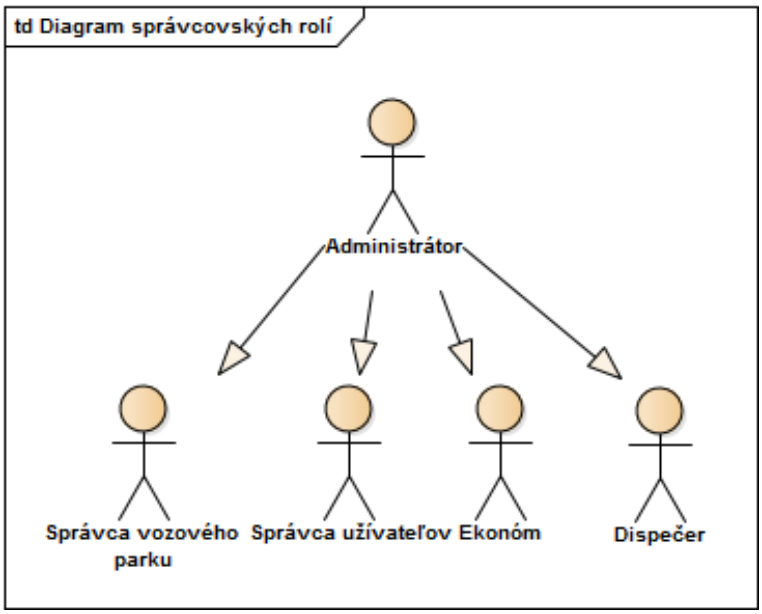
- **Zobrazenie počtu jázd za obdobie** - systém umožní správcovi s rolou ekonóm zobraziť počet jázd za určité obdobie podľa jednotlivých áut.
- **Zobrazenie počtu najazdených kilometrov za obdobie** - systém umožní správcovi s rolou ekonóm zobraziť počet najazdených kilometrov za určité obdobie podľa jednotlivých áut.
- **Zobrazenie počtu aktívnych klientov za obdobie** - systém umožní správcovi s rolou ekonóm zobraziť počet aktívnych klientov za ním zvolené obdobie. Aktívnym klientom sa chápe užívateľ, ktorý v danom časovom období službu využil.
- **Zobrazenie tržby za obdobie** - systém umožní správcovi s rolou ekonóm zobraziť tržby za určité obdobie podľa jednotlivých áut.
- **Zobrazenie nákladov za obdobie** - systém umožní správcovi s rolou ekonóm zobraziť náklady za určité obdobie. Nákladmi za určité obdobie je myslená suma nákladov na jednotlivé autá v danom časovom období.

D.1.10 Systém

- **Export údajov vo formáte CSV** - systém bude podporovať export áut, komunikačných jednotiek a užívateľov vo formáte csv. Tento formát je štandardom a je možné ho importovať do tabuľkových procesorov ako napríklad Excel.
- **Report e-mailom** - systém bude podporovať odoslanie reportu e-mailom administrátorom. Report bude obsahovať informácie ako počet nových klientov, celkový počet najazdených kilometrov, tržby či počet nehôd za posledný mesiac.
- **Reset hesla správcovi** - systém pri prihlásení bude podporovať možnosť resetu hesla správcovi. Nové heslo sa automaticky vygeneruje a zašle správcovi na e-mail.

D.2 Rozšírenie správcovských rolí

Správcovské role boli navrhnuté na základe funkčných požiadaviek uvedených v prílohe D.1. Každá správcovská rola má prispôsobenú Úvodnú obrazovku svojej funkcionalite.



Obr. D.1: Diagram správcovských rolí

Popis funkcionalít správcovských rolí:

- **Správca vozového parku** - má možnosť správy áut, palivových platobných kariet a komunikačných jednotiek v aplikácií.
- **Správca užívateľov** - má možnosť správy užívateľov v aplikácií.

- **Ekonóm** - má prístup ku cenníku a štatistikám systému. Vidí informácie o jazdách, rezerváciách a platbách.
- **Dispečer** - má prístup k aktuálnym informáciám. Vidí jazdy, rezervácie a tankovania.
- **Administrátor** - obsahuje všetky ostatné role. Má možnosť správovať správcov systému.