Mendelova univerzita v Brně
Provozně ekonomická fakulta

Webová mobilní aplikace pro Investiční aukce s.r.o.

Bakalářská práce

Vedoucí práce:
Ing. David Schubert

Roman Jakl

Brno 2017

Děkuji svému vedoucímu práce Ing. Davidu Schubertovi za věnovaný čas, trpělivost, podporu a cenné připomínky při tvorbě této práce.

Čestné prohlášení

Prohlašuji, že jsem tuto práci: **Webová mobilní aplikace pro Investiční aukce s.r.o.**
vypracoval samostatně a veškeré použité prameny a informace jsou uvedeny v seznamu použité literatury. Souhlasím, aby moje práce byla zveřejněna v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách ve znění pozdějších předpisů, a v souladu s platnou *Směrnicí o zveřejňování vysokoškolských závěrečných prací.*
Jsem si vědom, že se na moji práci vztahuje zákon č. 121/2000 Sb., autorský zákon, a že Mendelova univerzita v Brně má právo na uzavření licenční smlouvy a užití této práce jako školního díla podle § 60 odst. 1 Autorského zákona.
Dále se zavazuji, že před sepsáním licenční smlouvy o využití díla jinou osobou (subjektem) si vyžádám písemné stanovisko univerzity o tom, že předmětná licenční smlouva není v rozporu s oprávněnými zájmy univerzity, a zavazuji se uhradit případný příspěvek na úhradu nákladů spojených se vznikem díla, a to až do jejich skutečné výše.

V Brně dne 22. května 2017 .....

Abstract

Jakl. R. Web mobile application for Investiční aukce s.r.o. Bachelor thesis. Brno, 2016

The bachelor thesis deals with the design and implementation of hybrid application for the company Investiční Aukce s.r.o. The teoretical part describes current technology and frameworks for hybrid application development purposes. The practical part deals with implementation and application design according to assignment from company.

Keywords

AngularJS, Cordova, Hybrid application, Single Page Application, REST API

Abstrakt

Jakl, R. Webová mobilní aplikace pro Investiční aukce s.r.o. Bakalářská práce. Brno, 2016

Bakalářská práce se zabývá návrhem a implementací hybridní aplikace pro firmu Investiční Aukce s.r.o. Teoretická část pojednává o současných technologiích a frameworcích určených k vývoji hybridních aplikací. Praktická část se zabývá návrhem a implementací aplikace dle zadání firmy.

Klíčová slova

AngularJS, Cordova, Hybridní aplikace, Single Page Aplikace, REST API

Obsah

1 Úvod a cíl práce

Tato bakalářská práce se zabývá problematikou vývoje hybridních aplikací pro mobilní zařízení. Na základě nastudování dané problematiky bude navržena hybridní aplikace určená pro přenosná zařízení pro investorskou klientelu firmy Investiční aukce s.r.o. Firma se zabývá dražbou faktur. Aukci vyhrává investor, který nabídne nejnižší úrok, za který je ochotný faktury financovat.

1.1 Úvod

Mobilní telefony dnes už neslouží pouze k volání a posílání zpráv. Moderní telefony jsou vybavené dotykovou obrazovkou a různými senzory jako například GPS nebo čtečka otisků prstů. Výrobci se předhánějí, kdo nabídne výkonnější zařízení a nové funkce. Dotyková obrazovka moderních chytrých telefonů má úhlopříčně až pět palců. Pro náročnější uživatele vzniklo také zařízení, které nese název tablet. Jedná se o přenosný počítač bez klávesnice využívající jako standardní vstup právě dotykovou obrazovku. Rozměry úhlopříčky obrazovky u tabletů jsou od sedmi palců až přes třináct palců. Úhlopříčku obrazovky od pěti do sedmi palců mají podobná zařízení – tzv. phablety. Phablet se označuje zařízení, které kombinuje principy tabletu a telefonu.

Pokrok v této oblasti také otevírá další trh, a to s aplikacemi pro tato chytrá zařízení. Statistický portál NetMarketShare (2016) tvrdí, že platforma Google Android v říjnu 2016 zaujímá 68,54% podíl trhu s přenosnými zařízeními. Platforma iOS od firmy Apple pak zaujímá 25,78% podíl. Portál NetMarketShare poskytuje statistiky týkající se informačních technologií. Tyhle dvě platformy dominují trhu. Každá platforma pak má svůj vlastní obchod s aplikacemi od různých vývojářů.

Podobná oblast pro vývojáře, která se nově rozšiřuje, jsou chytré televize. Dnešní chytré televize disponují připojením k internetu a mají vlastní internetový prohlížeč. Uživatel má lepší uživatelský zážitek z nainstalované aplikace, než když je nucen zadávat adresu webové stránky do prohlížeče na televizi. První chytré televize nabízely k dispozici uzavřené vývojové platformy, v roce 2012 se však vývoj aplikací pro chytré televize začal přesouvat k webovým technologiím (Gasston, 2015, s. 194).

1.2 Cíl práce

Cílem této bakalářské práce je návrh a implementace hybridní aplikace pro firmu Investiční aukce s.r.o. Součástí návrhu je také API, které bude na základě schváleného návrhu implementováno softwarovými inženýry firmy.

Aplikace bude zobrazovat všechny běžící aukce, ukončené aukce, vyhrané aukce nebo běžící aukce podle nastaveného uživatelského filtru. Uživatel bude mít možnost přihazovat svoji nabídku na aukce a zobrazovat dokumenty k aukcím. Dále bude aplikace upozorňovat uživatele notifikací na spuštění nové aukce, blížící se konec aukce, či lepší nabídku jiného investora u aukce, které se účastní. Aplikace získá

potřebná data o probíhajících aukcích ve formátu JSON přes RESTové API rozhraní, které bude navrženo na základě požadavků na aplikaci. Pro notifikace pak bude využita služba, která nabízí odesílání a obsluhu push notifikací na mobilní zařízení. Aplikace bude cílená na zařízení na platformě iOS a Android.

2 Současný stav zkoumané problematiky

V následující kapitole jsou objasněny možnosti vývoje aplikací určených pro tablety a mobilní telefony. Autor srovnává výhody a nevýhody implementace nativních aplikací, webových stránek určených pro tato zařízení a hybridních aplikací. Dále se autor zabývá stavem trhu s podobnými typy aplikací.

2.1 Možnosti vývoje mobilních aplikací

Pro tvorbu mobilních aplikací existují tři přístupy. Vývojář může implementovat nativní aplikaci, webové stránky přizpůsobené mobilním zařízením, nebo hybridní aplikaci.

2.1.1 Nativní aplikace

Nativní aplikace jsou vyvíjeny přímo pro specifickou platformu a mohou využít všechen dostupný výkon zařízení. Jak uvádí Castledine (2013, s. 23), nativní aplikace jsou optimalizovány a kompilovány po konkrétní platformy a zařízení. Tyto aplikace mohou mít přístup k celé sadě funkcí a vestavěných zařízení, jako je například fotoaparát, mikrofon nebo kontakty uživatele. Tento typ aplikací obecně není závislý na internetovém připojení.

Pro každou platformu se však taková aplikace musí programovat zvlášť. Pro vývoj aplikací na platformu Android slouží programovací jazyk Java, pro platformu iOS zase programovací jazyk Objective-C, nebo novější Swift. Pokud bychom tedy chtěli nativní aplikaci pro každou platformu, museli bychom vyvíjet aplikaci v rozdílném prostředí zvlášť pro Android, iOS i ostatní platformy. V praxi to znamená větší náklady z důvodu potřeby více programátorů a složitější údržbu aplikací.

2.1.2 Mobilní webové stránky

Další možností tvorby aplikace pro přenosná zařízení je vytvořit optimalizovanou webovou stránku pro tato zařízení. Tento typ aplikace nevyžaduje výkonné zařízení. Kód takové aplikace je čten a překládán průběžně webovým prohlížečem (Castledine, 2013, s. 23). Výhoda webových stránek je nezávislost na platformě a jejich centralizovaná údržba.

Nevýhoda webových stránek spočívá v nutnosti optimalizovat webové stránky pro mobilní zařízení, nebo takové stránky vytvořit. V takovém případě je nutno při požadavku na webovou stránku rozeznat, zda se jedná o přístup z mobilního zařízení. Při rozeznání se požadavek přepošle na sekundární webovou stránku, obvykle s prefixem mobile, případně m. Přizpůsobení vzhledu webové stránky podle rozlišení displeje lze docílit pomocí technologie nazývané responzivní design.

Pro přístup na webovou stránku z prohlížeče je potřeba internetové připojení. Webová stránka také nemá přístup k funkcím zařízení. Tyto nevýhody řeší takzvané hybridní mobilní aplikace.

2.1.3 Hybridní aplikace

Tento typ aplikací využívá technologie HTML a CSS, stejně jako webové stránky, aplikace však má pomocí skriptovacího jazyka Javascript přístup k nativním funkcím přenosného zařízení (například zjištění zeměpisné polohy, akcelerometr nebo změna orientace zařízení).

Programátor implementuje webovou stránku pomocí HTML a CSS, která bude tvořit grafické uživatelské rozhraní. Podle typu aplikace se přes Javascript případně zajistí komunikace se serverem a zobrazení získaných dat. Poté programátor využije určitého prostředí (např. PhoneGap), které poskytuje „přemosťující“ API rozhraní jazyka Javascript (Castledine, 2013, s. 233). Takové rozhraní pak umožňuje pracovat s hardwarem zařízení. Prostředí PhoneGap navíc podporuje množství různých platforem (PhoneGap, 2013). Taková aplikace se spouští v internetovém prohlížeči mobilního zařízení. Hybridní aplikace jsou schopné běžet i bez internetového připojení, pokud jej nevyžadují například z důvodu získání dat.

Obrovská výhoda hybridních aplikací spočívá tedy v maximalizaci společné části kódu pro všechny cílové platformy a rychlosti vývoje aplikace. Zákazníkovi tak odpadnou náklady za vývoj aplikací pro každou platformu zvlášť.

Vzhledem k požadavkům na aplikaci je bakalářská práce zaměřena právě na problematiku vývoje hybridních aplikací.

2.2 Konkurenční produkty

V České republice i v zahraničí existuje několik firem nabízejících skupování nebo prodej faktur. Z českých firem je to například Acema, Zalep.to, Bibby, Smart Credit, ze zahraničních pak například anglická firma Invoice market nebo americká BlueVine. Autor práce však nenarazil na žádnou veřejně dostupnou mobilní aplikaci určenou přímo pro nákup faktur. Mezi podobné produkty by se však daly zařadit aplikace pro obchodování na burze. Google Play nabízí hned několik aplikací pro nákup akcií.

Takové aplikace mají několik technologických podobností s aplikací, která je cílem této práce. Aplikace potřebují layout s rozložením funkčních prvků. Aplikace také potřebují komunikovat se serverem přes API, díky kterému získají aktuální data o akciích, nebo přes které aplikace odešle informaci o zájmu uživatele o nákup či prodej akcií. Uživatel, který projeví zájem o akcie, musí být autorizovaný. K aplikacím určených k obchodování na burze patří například Degiro, nebo Investing.com. Tyto aplikace umožňují sledování historického vývoje ceny akcie. Stejně jako aplikace, která je cílem této práce, bude zobrazovat historii příhozů.

Uvedené aplikace pro obchodování s akciemi dále umožňují vyhledávání podle klíčových slov a ukládání oblíbených akcií. Obdoba v autorově práci se dá najít v rozdělení aukcí na filtrované aukce a aukce s příhozem aktuálně přihlášeného investora.

3 Metodika práce

Nejprve je potřeba vytvořit návrh obsahu aplikace a rozložení funkčních prvků. K tomu se používají tzv. wireframy. Důležitou částí prvního kroku je konzultace uživatelských scénářů a wireframů se zadavatelem. Z wireframů pak vyplynou funkční požadavky. Dobrý návrh aplikace ušetří později spoustu problémů a přepisování kódu při vývoji.

Z funkčních požadavků pak získáme požadavky na API. Dalším krokem tedy bude návrh RESTového API rozhraní. API rozhraní musí umožnit vrátit informace o aukcích, zaslat nabídku investora k běžící aukci a další. Dále je potřeba aukce filtrovat podle několika kritérií. K návrhu API rozhranní slouží například online mockovací nástroj Apiary.io od firmy Oracle. Nástroj Apiary.io bude sloužit jako server pro získávání dat do doby, než firma API implementuje a spustí. Systém této firmy běží na webovém frameworku Web2py. Informace o aukcích se pak budou vracet ve formátu JSON.

Vzhledem k faktu, že firma nezadala požadavek na konkrétní framework pro aplikaci, bude potřeba udělat analýzu současných frameworků a technologií. Porovnání proběhne na základě ceny, podpory platforem, kvality dokumentace, data vydání poslední stabilní verze a používanosti.

Po splnění předchozích kroků se dostaneme k samotné implementaci aplikace. Protože webová aplikace bude využívat server jako zdroj a úložiště dat, bude se jednat o tzv. single page aplikaci.

Posledním krokem je otestování naimplementované aplikace. Vývojová prostředí jako XCode a Android Studio nabízejí simulátory různých zařízení, ale jak uvádí Castledine (2013, s. 270), simulátor iPhonu má tendenci běžet rychleji, než skutečný iPhone. Naopak simulátor zařízení Android běží pomaleji, než fyzická zařízení. Proto testování proběhne i na reálných zařízeních. Po odladění aplikace je možné ji umístit ke stažení.

4 Frameworky a technologie

V této kapitole jsou popsány frameworky a technologie používané pro vývoj webových stránek a hybridních aplikací. Dále autor provedl analýzu frameworků určených k vývoji hybridních mobilních aplikací.

4.1 AJAX

AJAX (Asynchronous Javascript And XML) je technika používaná pro přístup k webovým serverům z webové stránky. Název AJAX se objevil poprvé v únoru v roce 2005 v článku Jesse Jamese Garretta nazvaném Ajax: A New Approach to Web Applications (Ajax: Nový přístup k webovým aplikacím). Technologie samotná však existuje již od roku 1999, kdy Microsoft vydal pátou verzi internetového prohlížeče Internet Explorer (Ajax v kostce. Inteval.cz, 2015).

AJAX umožňuje asynchronně aktualizovat webové stránky výměnou dat se serverem. Po načtení stránky je možno poslat dotaz na server a získat z něj data. To znamená, že uživatel získá možnost aktualizovat část stránky bez nutnosti obnovení celé stránky (AJAX Introduction. W3Schools.com, 2017). Díky tomu se šetří množství přenášených dat a zvyšuje plynulost práce na webové stránce.

AJAX využívá kombinaci objektu XMLHttpRequest k získání dat z webového serveru a Javascriptu s HTML DOM (Document Object Model) pro zobrazení dat a práci s daty. Komunikace se serverem nemusí probíhat pouze přes dokumenty XML, ale AJAX pracuje i s daty v plain textu a JSONu. Dnes se nejvíce používá přenos dat právě ve formátu JSON.

4.2 AngularJS

AngularJS je oblíbený volně dostupný MVC framework pro tvorbu dynamických HTML dokumentů. Verze 1.0 byla zveřejněna v roce 2012. Zaměstnanec Firmy Google Miško Hevery začal na AngularJS pracovat v roce 2009. Společnosti Google se projekt zalíbil a je touto společností oficiálně podporován (AngularJS tutorial. W3Schools, 2017).

AngularJS nabízí možnost přizpůsobení každé funkce podle potřeb programátora. Díky metodě Data-binding může automaticky aktualizovat pole na View stránce, kdykoliv se změní data v modelu, stejně tak může aktualizovat data v modelu, kdykoliv uživatel změní pole na View stránce. AngularJS nabízí také unikátní možnost tvorby vlastních HTML elementů (AngularJS, 2017).

4.3 Cordova/Phonegap

Apache Cordova je open source framework pro vývoj mobilních aplikací (Overview. Apache Cordova, 2015). Vývojář sestaví aplikaci za použití standardních webových

technologií jako HTML5, CSS3 a Javascript. Za použití nativního obalu, který funguje jako vrstva mezi webovou aplikací a zařízením, pak umožňuje webovou aplikaci transformovat na nativní. Programátor tedy nemusí znát nativní programovací jazyk pro danou platformu. Takový obal zpřístupňuje napojení na API rozhranní fyzického zařízení (Gasston, 2015, s. 132). Přes Cordova pluginy může aplikace přistupovat k akcelerometru, geolokaci, kameře, kontaktům a tak dále.

PhoneGap je distribuce frameworku Apache Cordova od firmy Adobe. Společnosti spolupracují od října roku 2012 (About. PhoneGap, 2016). Cordova slouží jako engine, který pohání PhoneGap. PhoneGap pro kompilaci nabízí cloudovou službu nazvanou PhoneGap Build.

4.4 Intel XDK

Jedná se o SDK od Intelu a nabízí kompletní balíček nástrojů pro vývoj, emulaci, testování, debugování a publikování aplikací založených na technologiích HTML, CSS a Javascript.

Intel nabízí tento SDK zdarma k dispozici pro operační systémy Windows, Mac OS a Linux. První verze Intel XDK vyšla v prosinci 2013. Pro zjednodušené sestavení layoutů má Intel XDK k dispozici Drag-n-Drop Builder. Intel XDK podporuje platformy Android, iOS a Windows Phone (Indel XDK details, 2017).

4.5 Push Notifikace

Jeden z požadavků na aplikaci je přijímat notifikace s oznámením o nově spuštěné aukci a dalších událostech. Aplikace by mohla na pozadí kontrolovat změny v běžících aukcích, toto řešení však není efektivní z důvodu zvýšení množství přijímaných a odesílaných dat a složité implementace. Naštěstí existuje spousta online služeb, které nabízejí řešení v podobě push notifikací. Push notifikace je zpráva, která se zobrazí v horní liště mobilního zařízení. Kliknutí na tuto zprávu otevře příslušnou aplikaci. Mezi nejznámější poskytovatele této služby patří Google s projektem Firebase.

Google Firebase je soubor nástrojů pro vývoj mobilních aplikací. Podporuje zařízení Apple iOS, Android a webové aplikace (Documentation. Firebase, 2017). Kromě push notifikací nabízí pro vytvořený projekt také databázi, úložiště, webhosting a další. Google nabízí tuto službu zdarma s limitem přenesených dat. Pokud uživatel potřebuje přenést větší množství dat, musí zaplatit. Google Firebase využívá vlastní řešení přeposílání zpráv nazvané Firebase Cloud Messaging.

Kromě Google Firebase existují také další webové aplikace poskytující posílání notifikací. Obecně platí, že každá služba má API pro posílání notifikací, každá notifikace má spoustu různých parametrů, dále je možné nastavit template zprávy a čas jejího odeslání. Každá služba pak poskytuje analýzu dat. Autor pro aplikaci vybral službu OneSignal.

OneSignal je služba poskytující pouze posílání notifikací. Oproti službě Google Firebase je však zcela zdarma a navíc podporuje platformy Windows Phone 8.0 a 8.1, Mac OS X a Amazon Fire (Product Overview. OneSignal, 2017). Pro zařízení Apple iOS a Android však využívá právě Google Firebase. OneSignal navíc nabízí plugin pro hybridní aplikace vytvořené frameworkem Cordova a dalšími. Díky těmto pluginům lze do aplikace velmi jednoduše přidat funkcionalitu push notifikací. Všechny tutoriály služby OneSignal jsou přehledně zpracované.

4.6 REST API

REST API je velmi rozšířený datově orientovaný způsob pro získávání dat ze serveru a pro odesílání dat na server. Komunikace nejčastěji probíhá pomocí dokumentů ve formátu JSON. REST využívá HTTP dotazy GET, PUT, POST a DELETE, které odpovídají databázovým dotazům SELECT, UPDATE, CREATE a DELETE. API rozhraní implementované na serveru má jednoznačně definované adresy, formát dokumentu, jak vrací data, či v jakém formátu přijímá data. Specifikace adres, formátování a dostupných metod se nachází v dokumentaci konkrétního API. Za zkratkou REST stojí Representational State Transfer.

RESTové služby se neřídí přesnými standardy (kromě HTTP). I přesto návrh a tvorbu REST API doprovází několik doporučených pravidel a best practices, která by se měla dodržovat, má-li být API čisté, přehledné a jednoduše použitelné. Na internetu existuje k dispozici spousta článků a videí například od zaměstnanců firmy Apigee, zabývajících se problematikou navrhování API.

Základní pravidlo je v URL nepoužívat slovesa (Jauker, 2014) a používat konkrétní podstatná jména jednotně buď v množném, nebo jednotném čísle. Sloveso pak nahrazuje metoda dotazu (GET, POST, PUT a DELETE). Mají-li být například vráceny data o aukcích, pak má být HTTP požadavek směřován na adresu /api/auctions, nikoliv na /api/getAllAuctions. Pokud klient žádá data o dokumentech k aukci s ID 123, pak posílá dotaz na adresu /api/auctions/123/documents. GET požadavky by neměly měnit stav dat.

URL v takovémto tvaru pak může přebírat různé parametry, které se umísťují na konec za URL. Pro stránkování se používají parametry limit a offset. Parametr limit značí kolik položek má API vrátit a parametr offset značí, kolik položek od začátku se má přeskočit. Další parametry pak mohou značit omezení vrácených sloupců, různé filtrování a seřazování dat. Firma si nepřeje atribut pro omezení sloupců a vrácená data budou defaultně seřazena podle času.

Pro lepší informování koncového uživatele o chybách na serveru je také možno vždy odesílat HTTP odpovědi se stavovým kódem 200 a následně posílat skutečný stavový kód a zprávu o chybách v těle HTTP odpovědi. Tato možnost byla však zavržena z důvodu dostatečně informující normy stavových kódů HTTP odpovědí.

V těle zprávy budou také informace (metadata) o počtu celkově nalezených záznamů, a kolik záznamů je momentálně odesláno (aktuální limit a offset). S daty v těle odpovědi také souvisí tzv. HATEOAS, zkratka pro Hypermedia as the Engi-

ne of application state. Jedná se o princip komunikace serveru s klientem tak, aby klient mohl procházet API bez předchozích znalostí. Prakticky tento princip funguje posíláním navigačních linků jako součást těla odpovědi serveru. Firma si tuto možnost ale také nepřeje.

Kvůli budoucím změnám se rovněž doporučuje používat verzování API. Pokud v API nastane změna, klientská aplikace by pak mohla mít problém číst data. K tomu se nejčastěji používá písmenko „v“ a číslo verze. Na příkladu aukcí by tak dotaz byl směřován na adresu /api/v1/auctions.

4.7 Analýza současně používaných frameworků

Firma Investiční aukce s.r.o. nezadala požadavek na konkrétní framework, ve kterém má být aplikace implementována. Výběr frameworku proběhne na základě porovnání cen, podpory mobilních platforem, kvality a přehlednosti dokumentace, data vydání poslední stabilní verze a používanosti.

4.7.1 Appcelerator Titanium

Titaninum je vývojové prostředí od firmy Appcelerator pro vývoj nativních aplikací pro různé platformy za pomocí javascriptu. První verze byla vydána v prosinci roku 2008 ve spolupráci s Adobe AIR (Taft K. Darryl, 2008). Appcelerator Titanium nabízí drag-adn-drop designer aplikací, vlastní vestavěný systém push notifikací, analytické nástroje, cloudové úložiště a nástroj pro tvorbu api rozhraní (Appcelerator, 2016).

Appcelerator nabízí čtyři různé placené balíčky produktu Titanium, trial verzi a ceny startují na 36 dolarech za měsíc. Appcelerator má také open source verzi nazvanou Titanium Mobile. Toto vývojové prostředí využívá MVC Framework Alloy od vývojářu z Appcelerator. První verze Alloy vyšla v roce 2012 (Alloy readme. GitHub, 2015).

Dokumentace obsahuje pouze komentovaný kód s minimem obrázků. Celkově webové stránky platformy Appcelerator jsou nepřehledné a cokoliv se na nich špatně hledá, je potřeba se „proklikat“ mnoha stránkami. Titanium podporuje zařízení na platformě iOS, Android, Windows a Blackberry (Appcelerator Open Source, 2015). Oblíbenost na githubu má 2 200 hvězd. Poslední verze vyšla 22. března 2017.

4.7.2 Bootstrap

Bootstrap je velmi rozšířený HTML, CSS a Javascript framework pro vývoj responsivních webových aplikací stylem „z mobilního zařízení nahoru“ (Bootstrap, 2017). Myšlenka vývoje z mobilního zařízení nahoru souvisí s CSS mediálními dotazy. Mediální dotazy mohou pomocí prohlížeče zjistit, zdali je nějaký výraz pravdivý (například jestli má obrazovka na šířku více, než 320 pixelů). Pokud je layout webové aplikace nejdříve navrhnut pro mobilní zařízení a postupně se bude pomocí mediálních dotazů přizpůsobovat větším obrazovkám, programátor se vyhne potížím

s podporou mediálních dotazů (Kadlec, 2014, s. 73). Také se tím zredukuje celková složitost CSS kódu. Opačný postup se nazývá „z desktopu dolů“.

Bootstrap vznikl v roce 2010 pod názvem Twitter Blueprint jako open source framework zaměstnanci Twitteru (About. Bootstrap, 2017). Bootstrap podporuje prohlížeče platforem Android, iOS a Windows 10 mobile (Getting started. Bootstrap, 2017). Oblíbenost na GitHubu má téměř 109 000 hvězd. Poslední stabilní verze Bootstrap vyšla 25. července 2016 a verze 4 alpha vyšla 6. ledna 2017.

4.7.3 Framework7

Za zmínku stojí i open source framework pro vývoj hybridních aplikací, který nese název Framework7. Ten se snaží kopírovat vzhled, funkce a uživatelský požitek, který uživatelé znají ze zařízení iPhone. Toho dosahuje vzhledem komponent, ikon, posuvníků a „swipe“ akcemi jako swipe back, pull to refresh a tak dále. Framework7 nabízí podporu pro zařízení na platformě Android a iOS (Franework7, 2016). Dokumentace frameworku je přehledná a interaktivní.

Za tímto frameworkem stojí vývojář Vladimir Kharlampidi z vývojového týmu iDangero.us (Franework7, 2016). Poslední verze byla vydána 13. března 2017 a oblíbenost na GitHubu má přes 9 300 hvězd.

4.7.4 Ionic

Jedná se o jeden z nejznámějších open source frameworků pro vývoj hybridních aplikací v HTML5. Ionic se zaměřuje na vzhled aplikace a uživatelský požitek srovnatelný s nativními aplikacemi (Core concepts. Ionic Framework, 2017). Firma byla založena v roce 2012 s myšlenkou vytvořit pro webové vývojáře lepší cestu k tvorbě aplikací za použití jejich současných znalostí (All about Ionic. Ionic, 2017). Pro sestavení a nasazení aplikace používá framework Cordova.

Komunitní verze je k dispozici zdarma. Pokud si uživatel připlatí za verzi Indie, dostane možnost tvorby prototypu aplikací přes Ionic Creator, hosting pro progresivní webové aplikace a cloudové sestavení nativních aplikací. Bussiness balíček pak navíc nabízí základní analýzu aplikace, e-mailovou podporu, přímou aktualizaci aplikací na trzích s aplikacemi a další.

Ionic má hezky zpracovanou a interaktivní dokumentaci. Uživatel si kód u jednotlivých tutoriálů může vyzkoušet přímo na virtuálním zařízení Android, iOS nebo Windows (Ionic Documentation Overview. Ionic, 2016). Popularita na GitHubu činí téměř 29 000 hvězd. Poslední verzi Ionic vydal 8. března 2017.

Ionic také nabízí různé pluginy například pro přístup a práci s kamerou, kontakty, fotogalerii, geolokaci a další. Nevýhodou je fakt, že ne všechny pluginy fungují na všech mobilních platformách.

4.7.5 jQuery

JQuery je populární a jednoduchý framework zjednodušující práci s javascriptem. Slouží k jednoduchému vyhledávání a práci s elementy, manipulaci s CSS, umožňuje tvorbu efektů, animací, reakcí na události a další. Kořeny jQuery sahají do roku 2005 a první verze byla vydána v lednu roku 2006 (JQuery foundation history, 2017). Autoři na stránkách uvádějí, že díky všestrannosti framework jQuery změnil způsob, jakým miliony lidí píší javascript (JQuery, 2017). Vývoj javascriptových knihoven pro jQuery je podporován korporacemi jako IBM, WordPress a dalšími. JQuery má pěkně zpracovanou a interaktivní dokumentaci.

Pro tvorbu responzivních webových stránek určených pro dotyková zařízení se jQuery také nabízí verzi jQuery mobile. JQuery mobile nabízí podporu AJAXu, reakci na události vyvolané dotykem, různé widgety a download builder sloužící ke stažení pouze potřebných komponent. Pro tvorbu vlastních stylů je také k dispozici nástroj ThemeRoller (JQuery mobile, 2017).

JQuery mobile má pěkně zpracovanou a interaktivní dokumentaci. Popularita na GutHubu přesahuje 10 000 hvězd. Poslední stabilní verze vyšla 31. října 2014, od ledna 2017 je k dispozici alpha verze 1.5.0. JQuery mobile má širokou podporu platforem a prohlížečů. Podporované platformy jsou Android, Blackberry, iOS

4.7.6 Kendo UI

Kendo UI je další knihovna pro vývoj v HTML a Javascriptu od společnosti Progress Telerik. Na webových stránkách nabízí placenou verzi pro jQuery a Angular JS s možností vyzkoušení trial verze. V rámci placených verzí pro větší projekty nabízí k dispozici online podporu, přes 70 komponent uživatelského vstupu a export dat do formátů PNG, PDF a Excel. Tvůrci frameworku na webových stránkách uvádí, že knihovna Kendo UI je vhodn8 pro rychlý vývoj aplikací s blížícím se odevzdáním projektu (Kendo UI, 2017).

Kendo UI má však také open source verzi zdarma, nazývá se Kendo UI Core. Ta nenabízí online podporu a má přes 40 komponent založených na jQuery. Kendo UI Core podporuje mobilní platformy Android, Blackberry , iOS a Windows Phone (Download Kendo UI Core, 2017).

Dokumentace Kendo UI není nijak hezky zpracovaná. K dispozici se nabízí demo, kde každá komponenta zobrazí výsledek a pod ní se nachází kód bez komentáře. Kendo UI Core má na GitHubu popularitu přes 1 600 hvězd a poslední verze byla vydaná 18. ledna 2017.

4.7.7 Mobile Angular UI

Jedná se o volně stažitelný framework pro vývoj HTML5 aplikací, který kombinuje Angular JS a Bootstrap. Mobile angular UI vyvinul soukromý vývojář (Mobile Angular UI, 2016). Na oficiálních stránkách je k dispozici interaktivní demo s ukázkou kódů a fórum. Fórum však obsahuje málo aktivních přispěvatelů. Celkově jsou

stránky frameworku chudé. Vývojář pracuje i na jiných projektech, proto framework nabízí málo funkcí a verze vycházejí v dlouhých intervalech.

Oblíbenost na GitHubu má 2800 hvězd, poslední verze vyšla 7. září 2016 a dokumentace je přehledná. Mobile Angular UI uvádí podporu prohlížečů na platformách Android, Blackberry, iOS a Windows (Getting Started. Mobile Angular UI, 2016).

4.7.8 Onsen UI

Onsen je open source framework založený na frameworcích jQuery a Angular JS. Jedná se o konkurenci Ionicu od vývojářů vývojového prostředí Monaca. První verze vyšla v roce 2014 (Introducing Onsen UI 1.0. Onsen blog, 2014). Onsen UI se také snaží maximalizovat uživatelský zážitek. Kromě toho se také pyšní optimalizací výkonu pro mobilní zařízení (Onsen UI, 2017). Framework Onsen UI je zdarma, Monaca však nabízí ročně placené balíčky s online podporou, vývojovými nástroji jako plugin pro Visual Studio, Monaca CLI a online úložiště.

Onsen UI také nabízí pěkně zpracovanou interaktivní dokumentaci. Kód z dokumentace se zobrazuje na virtuálním zařízení a uživatel má možnost vyzkoušet jeho funkcionalitu. Onsen UI podporuje platformy iOS a Android. Na GitHubu má popularitu přes 4 500 hvězd a poslední verze vyšla 28. března 2017.

4.7.9 React Native

React Native vznikl jako interní hackaton projekt od vývojářů z Facebooku v roce 2013 (Sekulić, 2016). Jedná se o open source framework od pro vývoj nativních aplikací a to za použití Javascritpu, programovacích jazyků Java, Objective-C nebo Swift a Javascriptové knihovny React. Tuto knihovnu také vydal Facebook (React Native, 2017).

React Native má přehlednou a okomentovanou dokumentaci s animacemi kompilovaného kódu. Některé komponenty lze vyzkoušet na virtuálním zařízení přímo na webových stránkách frameworku. Framework cílí na platformy iOS a Android. Oblíbenost na GitHubu je 46 500 hvězd a poslední verze vyšla 3. dubna 2017.

4.7.10 Sencha Touch

Sencha Touch je populární framework od firmy Sencha založený na architektuře MVC. Firma byla založena v roce 2007 (About. Sencha, 2017) a první stabilní verze frameworku Sencha Touch vyšla listopadu roku 2010 (Sencha Touch overview, 2017).

Stejně jako Ionic se zaměřuje na uživatelský prožitek a pocit nativní aplikace. Nabízí přes 50 zabudovaných komponent uživatelského rozhraní a nativní vzhled všech majoritních mobilních platforem. Framework disponuje enginem pro adaptivní layout, animace, hladké posouvání obsahu a také nabízí výkonově optimalizované widgety jako seznamy, toolbary, formuláře a podobně. Dále je k dispozici řazení a filtrování uživatelských kolekcí dat a vykreslování různých grafů.

Sencha Touch má hezky zpracovanou a interaktivní dokumentaci. Sencha Touch podporuje zařízení na platformách Android, Blackberry, iOS Windows Phone a Tizen. Od roku 2015 byl vývoj Sencha Touch spojen s vývojovými nástroji Sencha Ext JS (Sencha Touch - Overview. Tutorialspoint, 2017). Licence Sencha Ext JS je placená. Poslední verze produktu Touch vyšla 22. října 2014 (Release Notes for Sencha Touch 2.4.1. Sencha, 2014). Produkt Touch není na GitHubu a bohužel chybí i informace o počtu stažení.

4.7.11 Vyhodnocení

Dnes je k dispozici ke stažení a použití nespočetné množství frameworků. Frameworky popsané výše patří k těm nejznámějším, co se týká vývoje hybridních aplikací. Samotný popis a detailnější porovnání nejrůznějších frameworků by mohlo vydat na celou závěrečnou práci.

V tabulce 1 na následující straně jsou popsány základní informace o výše uvedených frameworcích. Data vydání poslední verze jsou uvedeny ke dni zpracování tohoto textu. Vzhledem k faktu, že všechny frameworky mají minimálně jednu verzi, která je k dispozici zdarma, není v tabulce uvedená cena. Z důvodu velké obliby mezi vývojáři, přehledné dokumentace, podpoře zařízení a udržovanosti byl vybraný framework Bootstrap. Díky své jednoduchosti použití se programátor může s tímto frameworkem velice rychle naučit pracovat. Bootstrap oproti ostatním frameworkům vyniká v počtu dostupných funkcí, také nabízí možnost přizpůsobení dle potřeb programátora. Uživatel si při stažení frameworku může nastavit, které pluginy a komponenty chce stáhnout. Vzhled každé komponenty je také možno přizpůsobit.

Tabulka 1: Srovnání frameworků

Název	**Typ**	**Dokumentace**	**Poslední vydání**	**GitHub oblíbenost**	**Podpora**
Appcelerator Titanium	framework, toolkit	statická, nepřehledná	22.3.2017	2 200	Android, Blackberry iOS, Windows phone
Bootstrap	framework	interaktivní s komentářem	25.7.2016	109 000	Android, iOS, Windows phone
Framework 7	framework	interaktivní s komentářem	13.3.2017	9 300	Android, iOS
Ionic	framework, toolkit	interaktivní s komentářem	22.3.2017	28 800	Android, iOS, Windows phone
jQuery mobile	framework	interaktivní s komentářem	31.10.2014	10 000	Android, Blackberry, iOS, Windows phone
Kendo UI core	framework	interaktivní bez komentáře	18.1.2017	1 600	Android, Blackberry, iOS, Windows phone
Mobile Angular UI	framework	interaktivní demo s komentářem	7.9.2016	2 800	Android, Blackberry, iOS, Windows phone
Onsen UI	framework, toolkit	interaktivní demo s komentářem	28.3.2017	4 500	Android, iOS
React native	framework	interaktivní s komentářem	3.4.2017	46 400	Android, iOS
Sencha Touch	framework	interaktivní s komentářem	22.10.2014	není na Git	Android, Blackberry, iOS, Tizen, Windows phone

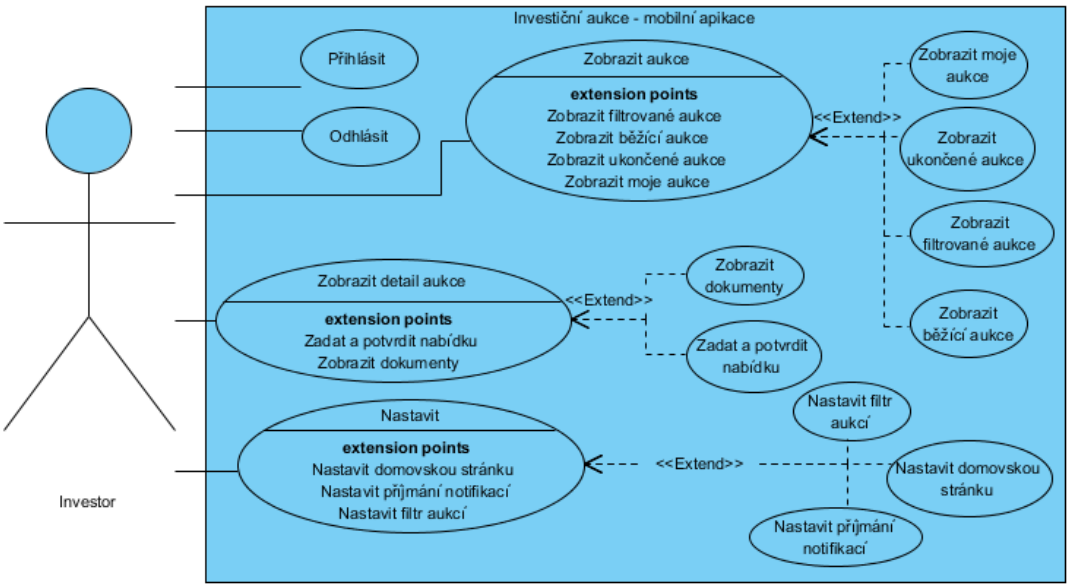
5 Vlastní práce

V následující kapitole autor popisuje, jak bylo postupováno při návrhu a implementaci aplikace a s jakými problémy se autor v průběhu tvorby této bakalářské práce potýkal.

5.1 Návrh aplikace

Firma Investiční Aukce s.r.o. má dvě webové platformy určené zvlášť pro investory (InvestAukce) a zvlášť pro klienty (InstantPlatba). Klienti nabízejí faktury před splatností, ze kterých je po schválení vytvořena aukce. Investoři na aukce posílají své nabídky, za jaký nejnižší úrok jsou ochotni faktury financovat. Firma požaduje multiplatformní aplikaci cílenou právě na investory. Aplikace tedy bude poskytovat stejné informace a stejnou funkcionalitu jako firemní platforma InvestAukce.

V první řadě byl vytvořen use case model podle funkcionality platformy InvestAukce a dále doplněn o funkcionalitu nastavení domovské stránky aplikace, nastavení notifikací a nastavení uživatelského filtru. Autorem vytvořený use case diagram je vidět na obrázku 1. Klientská aplikace bude přebírat a odesílat data prostřednictvím REST API naimplentovaného pro již hotový systém. Získaná data se zobrazí v aplikaci, nebo se z uživatelského vstupu získají data k odeslání. Aplikace je však určená pouze pro registrované investory, nikoliv pro správu ze strany zaměstnanců firmy. Proto je use case velmi jednoduchý a obsahuje pouze jednoho actora a to investora.



Obrázek 1: Use case diagram

V další fázi byl vytvořen tzv. requirements definition dokument, zkráceně RDD. Jedná se o dokument využívaný při vývoji nového softwaru, informačního systému

a k analýze uživatelských požadavků. RDD říká, co bude výsledný produkt dělat. RDD může obsahovat wireframy, funkční a nefunkční požadavky, use case diagramy, podniková pravidla, bezpečnostní politiku, diagramy aktivit, diagramy tříd a další. Po vytvoření RDD je možno jej konzultovat se zákazníkem a podle potřeby RDD modifikovat.

První RDD obsahoval vytvořený use case a prvotní návrh designu aplikace. Design obsahoval záhlaví stránky a obsah. Stránky se seznamem aukcí a detailem aukce byly převzaty z tehdy aktuálních webových stránek platformy InvestAukce. Takový design je však nevhodný pro mobilní aplikaci, z důvodu nevhodného uspořádání prvků a zobrazení velkého počtu informací. Design tedy bylo nutno předělat a dohodnout se, jak a co aplikace bude zobrazovat. Padla také myšlenka, že by aplikace mohla být defaultně zobrazovaná na šířku z důvodu velkého množství zobrazených informací, bez možnosti překlopení. Z důvodu špatného uživatelského požitku z aplikace byla však tato myšlenka zavrhnuta. Dále byla také zavrhnuta stránka zobrazující náhledy dokumentů.

Při konzultaci designu také přibyly požadavky na barevné rozlišení vypořádaných aukcí, aukcí po splatnosti, u běžících aukcí pak také barevné rozlišení aukcí s příhozem přihlášeného investora a aukcí, kde je nabídka přihlášeného investora momentálně nejlepší. U seznamu aukcí také přibyla informace o tom, jestli je daná aukce pojištěná. Také padl návrh, že by investor zadával potvrzovací kód při potvrzování nabídky u aukcí z důvodu dodatečné autentizace uživatele. Tento nápad byl však při konzultaci druhé verze RDD zavržen z důvodu nutnosti modifikace databáze, generování potvrzovacího kódu pro každého investora a jejich rozeslání investorům. Navíc přibyl požadavek možnost uživatelského nastavení úvodní obrazovky

Dále bylo pojednáváno o zobrazení menu. Rozhodovalo se mezi tlačítkem, které zobrazí rollup s odkazy na stránky, pop-up oknem s dlaždicemi a přepínáním panelů. Zadávající firma rozhodla pro pop-up okno.

Po konzultaci designu aplikace v první verzi RDD bylo však možno již začít s implementací layoutu aplikace díky představě o rozmístění prvků a vytvářet tak její prototyp.

RDD tedy byl upraven podle domluvených požadavků a vznikla tak jeho druhá verze. Následovala dlouhá odmlka ze strany firmy, než si zadavatel našel čas projít si druhou verzi RDD. Při konzultaci upraveného RDD bylo zavrženo vyskakovací pop-up menu a pop-up potvrzení nabídky investora. Detail aukce pak pořád obsahoval příliš informací pro zobrazení na mobilních zařízeních a bylo potřeba opět změnit jeho design. Také vyšlo najevo, že firma si objednala nové webové stránky a je nutné celkově přizpůsobit design podle nových stránek. Rozhodnutí o zobrazení menu jako pop-up okno bylo také zavrženo, protože tento přístup se pro nativní aplikace nepoužívá. Autorův návrh upraveného zobrazení aukcí v aplikaci je vidět na obrázku 2. K třetí verzi RDD byl vytvořen prototyp výsledné aplikace a řešily se detaily týkající se vzhledu aplikace.

Obrázek 2: Návrh zobrazení aukcí v aplikaci

5.1.1 Funkční požadavky

Z domluveného designu aplikace se však již dají vyčíst požadavky na funkcionalitu aplikace. Z návrhu tedy vycházejí následující požadavky:

- Autentizace investora
- Zobrazení informace o aktuálním limitu vyhraných aukcí
- Zobrazení běžících aukcí a běžících aukcí podle nastaveného uživatelského filtru
- Zobrazení aukcí s příhozem přihlášeného investora a jejich rozdělení na vyhrané, nevyhrané a běžící
- U vyhraných aukcí barevné rozlišení vypořádaných aukcí a aukcí po splatnosti 20 dní
- U běžících aukcí barevné rozlišení, pokud nabídka investora je nebo není nejlepší

- Push notifikace při vypsání nové aukce, nové aukce podle filtru, když jiný investor nabídne lepší nabídku, a aukce bez příhozu, která končí za 1 hodinu
- Možnost přihodit na aukci
- Zobrazení detailů aukce, historie příhozu a dokumentů k aukci
- Nastavení uživatelského filtru
- Nastavení úvodní stránky
- Nastavení posílání notifikací

5.2 Návrh API

Z požadavků na aplikaci lze vyčíst požadavky na API. Je tedy potřeba, aby API umožnilo vracet informace o běžících aukcích, o skončených aukcích, dále o běžících aukcích podle nastaveného filtru, informace o limitu vyhraných aukcí, u každé aukce informaci o tom, jestli se investor aukce účastní a případně, jestli je jeho nabídka nejlepší. API také musí umožnit přihození uživatelské nabídky na běžící aukci. U ukončených aukcí má API vrátit informaci o stavu, jestli je vypořádaná, či po splatnosti. Další požadavek ke každé jednotlivé aukci je pak vracet informace o jednotlivých příhozech zúčastněných investorů a vracet přiložené dokumenty.

Z důvodu požadavku stejného nastavení uživatelského filtru na webových stránkách je také nutno vytvořit rozhraní pro uživatelský filtr, aby bylo možné jej synchronizovat se serverem a aplikací. Dále aplikace musí získat data o možných oborech, či lépe řečeno průmyslech aukce, z důvodu možných změn v těchto průmyslech v budoucnu. Seznam oborů tak zůstane aktuální a nemusí být staticky nastavený v aplikaci. V těle HTTP odpovědi požadavku na aukci bude informace o průmyslu aukce pouze ve formě jeho aktuálního ID.

Jak již bylo napsáno, parametry pro filtrování se doplňují na konec URL. Aukce mají mnoho parametrů, kterými lze aukci filtrovat. Mezi parametry patří minimální a maximální hodnoty financované a nominální hodnoty, aktuálního a maximálního příhozu, ekonomického ratingu, platební morálky, data konce a splatnosti aukce a také podle oboru a stavu aukce. To je celkem 18 parametrů jen na uživatelský filtr, s tím že parametry obor a stav aukce mohouu nabývat několika hodnot. Pokud by se všechny tyto informace měly posílat v URL, byla by tato URL velmi dlouhá a nepřehledná. Proto se uvažovalo o posílání parametrů filtru v těle metody. Posílat data v těle GET metody se však nedoporučuje z důvodu možného odmítnutí GET požadavku na proxy, nebo zahození těla požadavku. Nabízí se tedy možnost posílat data v těle metodou POST. Tato metoda by však měla být používána pro vkládání dat, nikoliv pro získávání dat. Z důvodu konzistence byly tyto možnosti zavrženy a pro GET požadavek bylo 18 parametrů nahrazeno jedním parametrem datového typu boolean i přes to, že aplikace bude umožňovat nastavení pouze několika z nich (na přání zadávající firmy).

Tento atribut říká serveru, jestli má aukce filtrovat. Server totiž zná ID uživatele za předpokladu, že je uživatel autentizovaný. Proto není potřeba odesílat informaci o ID uživatele a stačí parametr datového typu boolean. Server může tak parametry uživatelsky nastaveného filtru hledat v databázi a klientská aplikace nemusí získávat a odesílat data o nastaveném filtru. Tím se ušetří množství odesílaných dat.

K parametru filtr pak přibudou dva parametry, ze kterých se server dozví, jestli má vracet běžící nebo již ukončené aukce, a také jestli mezi aukcemi má být aukce s příhozem přihlášeného investora. Adresa dle syntaxe Apiary tak vypadá následovně:

```
/auctions{?user_filter}{?my_bid}{?status}{?limit}{?offset}
```

Parametry s podtržítkem jsou použity kvůli konzistenci s programovacími konvencemi programátorů ve firmě. Parametry „user_filter“ a „my_bid“ očekávají datový typ boolean, limit a offset jsou pak celočíselné. Původní parametr „running“, který měl být také datového typu boolean značil, jestli má server vrátit běžící nebo ukončené aukce. Vzhledem k udržení flexibility a k faktu, že ukončené aukce se mohou nacházet v jednom z 16 dalších stavů, byl tento parametr nahrazen parametrem „status“. Ten pak očekává pole řetězců značících stav aukce. Pro rozlišení běžících a ukončených pak parametr bude nabývat hodnot „RUNNING“ a „ALL_ENDED“. Stav aukce psaný kapitálkami je opět v souladu s implementační konvencí firmy. „ALL_ENDED“ pak nahrazuje zmiňovaných 16 stavů ukončených aukcí.

Pro zobrazení základních informací o aukci v aplikaci jsou vyžadována následující data: id aukce, název aukce, stav aukce, protistrana, financovaná a nominální částka, detailní popis aukce, měna, ekonomický rating, platební morálka protistrany, data splatnosti, začátku a konce aukce, id oboru aukce, aktuální a maximální úrok, a zdali je aukce pojištěná. K těmto datům pak přibudou hodnoty ID uživatelů, kteří se aukce účastní a ID investora s nejlepší nabídkou. Z těchto dat získáme informaci pro barevné rozlišení aukcí, které přihlášený investor vyhrává, nevyhrává, nebo kterých se neúčastní.

Aukce mají daný životní cyklus a stav aukce říká, ve které fázi životního cyklu se momentálně nacházejí. Stav aukce po splatnosti se mění v závislosti na počtu dní. Kvůli požadavku na barevné rozlišení aukcí 20 dní po splatnosti a vypořádaných aukcí byla původní autorova myšlenka přidat tuto informaci k datům o aukci. Přibyly by tak dvě hodnoty typu boolean (vypořádaná – ano/ne, 20 dní po splatnosti – ano/ne). Zadávající firma je však ujištěná, že životní cyklus aukcí je dostatečně stálý a tyto parametry si nepřeje předávat. Tento nápad byl tedy opuštěn a toto barevné rozlišení bude implementováno v aplikaci porovnáním stavu aukce.

Všechny částky se posílají nezformátované s desetinnou tečkou a kalendářní data konce a začátku aukce se posílají ve standardním formátu s časovou zónou podle specifikace ISO 8601. V aplikaci proto není možné tyto informace včetně stavu aukce zobrazovat přímo, nýbrž bude potřeba tyto informace zformátovat do uživatelsky přívětivého stavu.

GET požadavek na adresu /auctions tedy vrátí tělo odpovědi v následujícím tvaru:

```
{
    "data":[
        {"id": 173152,
        "status": "RUNNING",
        "counterparty": "Ahold",
        "price": 500000.00,
        "currency": "CZK",
        "winning_user": 6,
        "actual_bid": 11.6,
        "max_interest": 15,
        "bid_users": [6],
        "economical_rating": "A+",
        "due_date": "2017-08-30",
        "end_date": "2017-04-27T23:15:00Z",
        "insurance": true,
        "industry_id": 1,
        "name": "Aukce 173152",
        "start_date": "2016-04-18T13:10:00Z",
        "description": "Lorem ipsum.. ",
        "face_value": 700000.00,
        "payment_rating": "B++" },
        { …}
    ],
    meta": {
        "status":"ok",
        "total_auctions":30,
        "limit" : 10,
        "offset" :20
        }
}
```

Adresa s příponou /auctions/{id} pak vrátí stejné informace pouze o jedné aukci. Zadávající firma požaduje zobrazit přiložené dokumenty a historii příhozů k aukci při zobrazení detailu aukce. Tyto informace také vrátí API. Dokumenty se získají z adresy /auctions/{id}/documents a historii příhozů zase vrátí adresa /auctions/{id}/bids. U dokumentů API vrací typ dokumentu a odkaz na požadovaný dokument z důvodu šetření posílaných dat. Zobrazení náhledu dokumentu bylo zavrženo při konzultaci RDD. U příhozů k aukci je pak posíláno datum příhozu, ID investora a výše jeho nabídky.

Pro přihození nové nabídky se pošle na adresu /auctions/{id}/bids požadavek POST. V těle požadavku pak stačí informace o výši příhozu. Server zná aktuální

čas a ID přihlášeného investora. ID aukce pak server získá z URL.

Firma si přála v aplikaci funkční nastavení filtru pro nastavení následujících parametrů aukcí: maximální a minimální hodnota aukce, měna a minimální úrok. Z adresy /user_filter je možné tato data získat, nebo je metodou PUT aktualizovat. Pro získání dat o průmyslech slouží adresa /industries. Ta vrátí pole dat s ID průmyslu a jeho názvu.

5.3 Implementace aplikace

Aplikaci tvoří přihlašovací stránka s logem firmy, po přihlášení se zobrazí layout tvořený záhlavím s logem, tlačítkem pro zobrazení menu a dynamicky se měnícím obsahem. Aplikace využívá server jako zdroj a úložiště dat za pomoci REST API. Jedná se tedy o single page aplikaci. Single page aplikace se jednou načte a její obsah se dynamicky mění za pomocí javascriptu. Toho je možné docílit Javascriptovým frameworkem AngularJS.

5.3.1 Layout

V první řadě je tedy potřeba vytvořit soubor s layoutem stránky. Vytvoříme HTML dokument, ve kterém nastavíme titulek a kódování. Dále nastavíme viewport.

Viewport představuje šířku prohlížeče. U mobilních zařízení však nastává problém. I přes menší obrazovku oproti desktopovým počítačům se mobilní zařízení snaží zobrazit web v kompletní podobě. CSS pixely se pak netýkají obrazovky, nýbrž viditelné oblasti v prohlížeči. Například na displejích Retina zařízení iPhone jsou dva pixely zařízení rovny jednomu pixelu CSS (Kadlec, 2014, s. 58). Viewport layoutu je pak dvojnásobně široký oproti vizuálnímu viewportu (šířce zařízení). Tento problém řeší meta tag viewport. Díky němu je možné změnit měřítko a viewport layoutu zařízení. Atribut content nastavíme na width=device-width, tím nastavíme viewport layoutu obrazovky roven viewportu zařízení. Direktivou initial-scale pak nastavíme počáteční přiblížení. Celý tag v záhlaví pak bude vypadat následovně:

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Dále importujeme kaskádové styly a javascriptové knihovny. Knihovny frameworků včetně použitých fontů jsou stažené jako součást aplikace, aby jej uživatelská aplikace nemusela stahovat po každém startu aplikace a tím se ušetřilo množství stahovaných dat

Tělo layoutu stránky pak tvoří dva oddíly. První oddíl tvoří layout obrazovky pro přihlášení, druhý oddíl tvoří layout aplikace po přihlášení. Oddíly se zobrazí nebo schovají za pomocí direktivy ng-if, která sleduje hodnotu „logged“ datového typu boolean nastavené v $rootScope. Při změně hodnoty se přepne zobrazení obou oddílů. Login obrazovku pak tvoří kulaté logo firmy, vstupní pole pro uživatelský e-mail, heslo a tlačítko pro odeslání. Na tlačítku je nastavena funkce, která ověří, že uživatel vyplnil obě pole. Funkce vyšle autentizační požadavek na API. Podle stavu

odpovědi pak buď zobrazí informaci o špatném jménu nebo heslu, pokud jsou však údaje správné, funkce schová tento oddíl a zobrazí druhý oddíl.

Druhý oddíl tvoří záhlaví, menu a oddíl, který dynamicky vykresluje obsah. Menu tvoří odkazy, nadpis „Menu“ a křížek pro zavření. Tento křížek je vytvořen za pomocí znaku ×. Tento znak je na rozdíl od písmene „X“ zarovnaný na střed řádku a vypadá lépe. Při kliknutí na křížek a na všechny odkazy nastavíme zavolání javascriptové metody closeMenu(). Menu také obsahuje nastylovaný nadpis s ID menuHeader. Odkazy v menu obsahují direktivu ng-href začínající posloupností znaků #!/. Tím se hodnota za touto posloupností předá routeProvideru frameworku AngularJS. Ten dosadí cestu k požadované stránce, která se má načíst do layoutu a nastaví pro ni kontroller. Pokud by atribut ng-href neobsahoval zmiňovanou posloupnost znaků, prohlížeč by se snažil provést přesměrování do neexistující složky s názvem v tomto atributu. Odkazy nastavíme na provolání metody pro zavření menu při kliknutí.

```
<div id="menu">
    <a id="closebtn" ng-click="closeMenu()">&times;</a>
    <a id="menuHeader">Menu</a>
    <a ng-href="#!/runningAuctions" ng-click ="closeMenu()">
        Běžící aukce
    </a>
    …
</div>
```

Menu nastylujeme v souboru style.css. Nastavení height: 100% roztáhne panel na celou výšku obrazovky. Width: 0 a overflow-x: hidden panel defaultně skryjí. Position: fixed, top: 0 a left: 0 zajistí, že panel bude fixovaný na levou stranu. Transition nastavený v sekundách zajistí pěkný plynulý přechod při zobrazení menu. Díky animovanému přechodu menu „vyjede“ z levé strany. Tato změna šířky bude volána javascriptem při kliknutí na ikonku menu. Z-index pak nahrazuje imaginární osu Z. Prvek s nízkou hodnotou z-indexu překryjí všechny prvky, které mají tuto hodnotu nastavenou vyšší.

```
#menu {
    height: 100%;
    overflow-x: hidden;
    width: 0;
    position: fixed;
    top: 0;
    left: 0;
    background-color: #111;
    transition: 0.5s;
    padding-top: 60px;
    z-index: 5;
```

```
}
```

Pro menší obrazovky pak menu nastavíme maximální šířku zobrazení a zmenšíme velikost fontů. Menu by jinak mohlo přetéct mimo obrazovku.

```
@media screen and (max-width: 250px) {
    #menu {max-width: 170px}
    #menu a~{font-size: 1.125em;} /* 18/16 */
    #menu #menuHeader {font-size: 2em;} /* 32/16 */
  }
```

Dále css soubor obsahuje napozicování prvků v menu a nastavení písma. Do layoutu pak přidáme klasický bootstrap header. Do něj přidáme glyphiconku značící menu a logo firmy. Umístěním loga do tagu s glyphiconkou docílíme zobrazení menu i při kliknutí na logo. Na pravou stranu headeru přidáme také tlačítko pro obnovení stránky.

```
<nav class="navbar navbar-inverse">
    <div class="navbar-header">
        <span class="navbar-brand glyphicon glyphicon-list"
              id="hamburger"
              onclick="openMenu()">
            <img src="css/logo.png"/>
        </span>
    </div>
    <span class="pull-right glyphicon glyphicon-refresh"
          id="refresh"
          ng-click="refreshPage()">
    </span>
</nav>
```

Zde však narazíme na problém. Logo se totiž zobrazuje pod glyphiconkou. Vytvoříme tedy CSS blok pro logo. Logo má poměrně velké rozlišení, a tak nastavíme výšku a šířku automaticky podle nadřazeného prvku. Dále nastavíme vertikální zarovnání a display na hodnotu inline-block. Tím zajistíme, že se glyphiconka s logem zobrazí vedle sebe.

```
.navbar-brand img{
    width: auto;
    height: 100%;
    vertical-align: top;
    display: inline-block;
}
```

Do druhého bloku přidáme také prázdný div element, kterým překryjeme celý viewport. Tomuto elementu přiřadíme atribut ng-click=”closeMenu()” a zobrazíme

jej při zavolání menu. Tím docílíme ztmavení celého obsahu, což bude soustředit pozornost uživatele na zobrazené menu. Pokud uživatel klikne na tento element, menu se zavře a element zmizí. Díky nastavenému z-indexu tento element nepřekryje zobrazené menu. Podobné překrytí přidáme i pro oba bloky s rotujícím spinnerem značícím načítání dat.

```
#overlay {
    background: rgba(0,0,0,0.6);
    display: none;
    position: fixed;
    top: 0;
    right: 0;
    bottom: 0;
    left: 0;
    z-index: 4;
}
```

Nakonec do požadovaného místa vložíme div element s direktivou ng-view, do kterého budeme dosazovat požadovaný obsah.

5.3.2 Zobrazení aukcí

Framework Bootstrap má k dispozici širokou nabídku již hotových komponent připravených k použití. Aukce je potřeba zobrazit po řádcích a počet informací zobrazit podle šířky viewportu. Bootstrap má k dispozici komponentu „list-group“ sloužící právě k zobrazení entit po řádcích. Pro zobrazení aukcí se však více hodí komponenta „accordion“. Tato komponenta zobrazí entity po řádcích, každý řádek je tvořen nadpisem a tělem. Při kliknutí na nadpis se zobrazí nebo skryje tělo. To se hodí pro zobrazení měnícího se počtu informací o aukci v závislosti na šířce viewportu v nadpisu řádku, v těle řádku se pak zobrazí informace, které se na viewport nevešly. Pokud je zobrazeno tělo jednoho řádku a uživatel klikne na jiný řádek, původní zobrazené tělo řádku se opět skryje.

Při implementaci autor této práce narazil dva problémy. Protože aplikace používá framework AngularJS s direktivou ngRoute a „accordion“ pro zobrazení využívá tag <a> s nastaveným atributem href=”#collapseid”, kliknutí na nadpis direktiva ngRoute vyhodnotila jako přesměrování a stránku překreslila. Problém lze vyřešit nastavením parametru onclick=”return false;” v tagu <a>.

Druhý problém spočíval v defaultní implementaci komponenty „accordion“. Zobrazení těla řádku proběhne pouze při kliknutí na text nadpisu, nikoliv při kliknutí na celý panel. V aplikaci spuštěné na mobilním zařízení je taková funkcionalita uživatelsky nevhodná. Problém je možné vyřešit roztažením odkazu na celý blok. V tagu je pak tag s atributem class=”row” pro zobrazení informací o aukci v jednom řádku.

```
.panel-heading {
    display: block;
    cursor: pointer;
}
```

Pod listem aukcí se pak nachází stránkování. Autor použil BootstrapUI direktivu „uib-pagination“. BoostrapUI je Angular nádstavba Boostrapu. Pro direktivu je pak možno nastavit aktuální stránku, celkový počet záznamů, počet záznamů na stránku a další. Stránkování lze vidět na straně 23 na obrázku 2.

Množství zobrazených dat v řádku se řídí podle šířky viewportu. Díky využití frameworku Bootstrap není nutné manuálně nastavovat breakpointy za použití mediálních dotazů. Podle nastavení třídy dle syntaxe Bootstrapu u konkrétního HTML prvku je možno řídit zobrazení tohoto prvku. Aplikace pak obsahuje stažený framework Bootstrap s nastavenými breakpointy od 350 do 760 pixelů. Breakpointy jsou autorem nastavené tak, aby layout zobrazil největší možný počet informací o aukci na řádek.

Pro zamezení kopírování kódu je tento layout seznamu aukcí využit pro zobrazení běžících aukcí, ukončených aukcí, aukcí s příhozem přihlášeného investora a filtrovaných aukcí. Drobné změny mezi každou stránkou jako zobrazení tlačítka pro zapnutí filtru a barevné rozlišení aukcí se pak řídí pomocí proměnných ve $scope.

5.3.3 Javascript

Pro obsluhu aplikace javascriptem je potřeba vytvořit instanci aplikace. Dosadíme název z atributu tagu <html> v layoutu a direktivu ngRoute. Ta právě umožní vkládání obsahu podle nastavené cesty a zajistí funkčnost fyzického tlačítka pro přechod zpět.

```
var investAukce = angular.module('investAukce', ['ngRoute']);
```

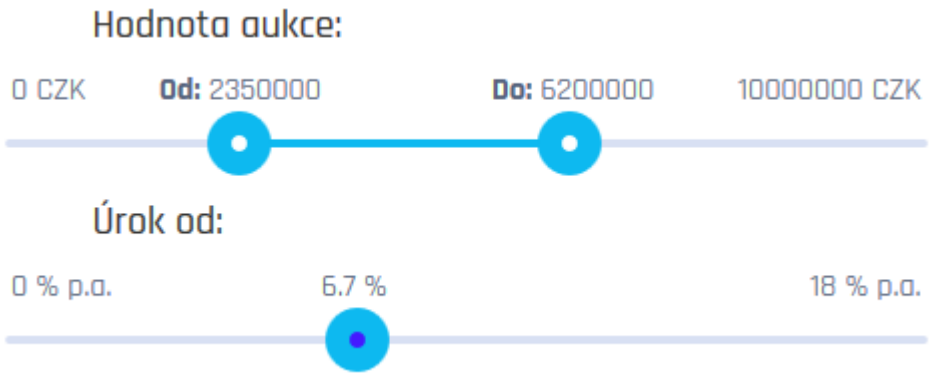
Dále nastavíme ovládání cest skrz aplikaci. Na instanci aplikace zavoláme metodu .config a jako parametr předáme funkci přebírající proměnnou routeProvider. Na ní pak pro každou stránku zavoláme funkci .when. Jako parametry zadáme cestu pro zavolání odkazem a slovník s hodnotami pro skutečnou cestu k souboru s layoutem a název kontrolleru, který bude nastavovat proměnné pro tento soubor, nebo volat různé funkce.

```
investAukce.config(function($routeProvider) {
    $routeProvider
        .when('/running', {
            templateUrl : 'pages/runningAuctions.html',
            controller : 'runningAuctionsController'
         })
        .when(…);
}
```

Jak bylo zmiňováno v sekci pojednávající o API, je potřeba z API načíst obory aukcí, či lépe řečeno průmysl. Není však efektivní posílat HTTP request při každém zobrazení aukcí nebo jedné aukce. Aby aplikace nepoužívala globální proměnné, autor použil AngularJS funkci factory. Tato funkce vytvoří objekt, pro který můžeme nadefinovat soukromé proměnné a funkce. Funkce factory přebírá název továrny a funkci, ve které budeme definovat chování objektu. Tento objekt můžeme volat ve všech kontrollerech a předávat tak mezi kontrollery data.

Autor vytvořil továrnu, která obsluhuje komunikaci s API. Továrna disponuje funkcemi pro získání běžících aukcí, skončených aukcí, filtrovaných aukcí, aukcí s příhozem přihlášeného investora, průmyslů, uživatelského filtru a další. Po zavolání tyto funkce zkonstruují požadovanou URL a odešlou patřičný HTTP požadavek. Z důvodu šetření přenesených dat se odpovědi serveru cachují do slovníku, který používá jako klíč sestavenou URL. Pro každou metodu je vytvořená cache zvlášť. Metoda tedy před zavoláním HTTP requestu zkontroluje, jestli požadovaná data nemá již uložená. V případě požadavku na obnovení stránky se patřičná cache vyprázdní a zavolá se HTTP request s nastavenými parametry limit a offset dle aktuální stránky. Pro zobrazení detailu aukce se neposílá nový HTTP request na API, ale data o konkrétní aukci se předají patřičnému kontrolleru v jiné továrně pro ukládání objektu. Aplikace pouze pošle GET request pro získání dokumentů a historii příhozů k aukci.

Protože framework Bootstrap nedisponuje komponentou pro zobrazení slideru, byla v nastavení filtru pro omezení hodnoty aukce a maximálního úroku využita direktiva AngularJS slider. Sliderům aplikace nastaví hodnoty z dat získaných z API na adrese /user_filter. Výsledný vzhled slideru je vidět na obrázku 3. Nastavení domovské stránky po přihlášení se ukládá do místního úložiště jazyka HTML5. Z tohoto úložiště se pak po přihlášení načítá a provede se přesměrování na patřičný template a kontroller.



Obrázek 3: Ukázka slideru použitého v aplikaci

Součástí aplikace jsou autorem naimplementované javascriptové funkce obsluhující formátování kalendářních dat, částek, zobrazení a schování menu, zobrazení

a schování spinneru informujícího uživatele o načítání dat, nastavení funkcí do $scope a zobrazení chyb HTTP odpovědí. Tyto funkce jsou odděleně v souboru, dále je také pro čitelnost kódu oddělena logika obsluhující přihlašování, komunikaci s API a AngularJS kontroléry obsluhující vkládání stránek do hlavního layoutu.

V rámci implementace bylo také vyzkoušeno vývojové prostředí Intel XDK. Při sestavení jednoduché testovací aplikace pro platformu Android Intel XDK navrátí odkaz na server ke stažení. Stažená aplikace měla přibližně dvacetinásobnou velikost oproti sestavené aplikaci za použití frameworku Cordova.

5.4 Notifikace

Pro odesílání push notifikací na zařízení Android a Apple iOS přes OneSignal je potřeba mít vytvořený projekt v Google Firebase. Aplikaci ve službě OneSignal je pak nutno nakonfigurovat pro každou platformu zvlášť. Pro Android je třeba vyplnit klíč Google Server API a číslo Google projektu. Do Cordova aplikace pak stačí přidat OnesSignal Cordova plugin a přidat javascriptový kód, který naslouchá události deviceready. Po zachycení události pak kód inicializuje OneSignal plugin. Kód je k dispozici na stránkách s dokumentací tohoto pluginu a je nutné do něj vyplnit ID OneSignal aplikace. Také je nezbytné tento tento skript spustit jako první, aby byl schopen zachytit událost deviceready. Ve vývojovém prostředí Android Studio se pak ujistíme, že jsou v SDK Manager nainstalovány repozitáře Google a Android Support.

Pro zařízení Apple iOS je potřeba vygenerovat iOS Push certifikát. Ten je možné vygenerovat poměrně složitě přes Keychain Access v systému Mac OS X a vývojářskou stránku Apple, vývojář však také může využít služby OneSignal Provisionator. Vygenerovaný certifikát se pak vloží do naší aplikace služby OneSignal a do naimplementované aplikace ve vývojového prostředí XCode.

Služba OneSignal ve webovém rozhraní zaznamená jednotlivá zařízení, která mají nainstalovanou naší aplikaci a přidělí jim unikátní ID. Zařízením pak můžeme udělovat tagy a dělit je do segmentů. Podle tohoto nastavení lze notifikace posílat jen určitým zařízením. Plugin OneSignal Cordova pak disponuje rozhraním pro nastavení tagů přímo v aplikaci. Přes OneSignal API je možné posílat notifikace s různým nastavením nebo měnit tagy zařízení.

5.5 Testování aplikace

Hotová aplikace byla otestována na zařízení Honor 4C s operačním systémem Android 6.0. Díky robustnosti frameworku Bootstrap se GUI prvky na mobilním zařízení zobrazují stejně jako v prohlížeči na stolním PC a všechny funkce fungují stejně jako v prohlížeči na PC. Notifikace fungují v pořádku při zapnuté aplikaci. Bohužel většina zařízení nemůže přijímat push notifikace, pokud aplikace neběží alespoň na pozadí.

Pro testování push notifikací služby OneSignal na platformě Apple iOS je nutno vygenerovat iOS push certifikát a použít fyzické zařízení Apple iPhone. Simulátor zařízení ve vývojovém prostředí XCode tyto notifikace nepodporuje. Autor práce nedisponuje zařízením iPhone a XCode nainstalovaný na školních počítačích nepodporoval spolužákův iPhone 4. Z tohoto důvodu nebyla aplikace vyzkoušena na fyzickém zařízení iPhone. V iPhone simulátoru aplikace funguje plynule bez chyb.

V rámci testování aplikace bylo autorem práce napsáno několik jednotkových testů za použití Javascriptového frameworku Jasmine. Testy ověřují správnost funkcí pro formátování částek, kalendářních dat, výpočtů konce aukce a dalších.

5.6 Distribuce aplikace

Sestavenou aplikaci pro platformu Android můžeme umístit přímo ke stažení na webových stránkách, nebo ji distribuovat na Google Play. Distribuce aplikace na tržiště Google Play vyžaduje registraci jako Android vývojář a zaplacení registračního poplatku ve výši 25 dolarů. Novou aplikaci pro tržiště je možné vytvořit pomocí Google Play Console. Nově vytvořená aplikace v Play Console vyžaduje jméno a implicitní jazyk. Nahraná aplikace s příponou apk musí mít nastavenou verzi v souboru manifest a také musí být podepsaná certifikátem. Použít lze však i certifikát podepsaný autorem. Dále Play Console vyžaduje nastavení krátkého a dlouhého popisu aplikace, vložení alespoň dvou screenshotů, ikony v rozlišení 512×512 pixelů a obrázek v rozlišení 1024×500 pro zobrazení jako úvodní obrázek na stránce s aplikací. Potom je potřeba vyplnit informace o aplikaci a kategorii aplikace, z těchto informací umístěná aplikace získá content rating. Na další stránce vyplníme cenu za stažení aplikace a země, ve kterých chceme aplikaci distribuovat. Po potvrzení distribuce trvá několik hodin, než je aplikace umístěná na tržišti. Google Play Console nabízí vydávání i alpha a beta verzí aplikace.

Pro nasazení iOS aplikace na Apple App Store je potřeba být platícím vývojářem Apple. Členství programu Apple Developer stojí pro jednotlivce 99 dolarů za rok. Pro nasazení také potřebujeme vytvořit zřizovací profil, díky kterému získáme distribuční certifikát. V XCode se přihlásíme k profilu Apple, aplikaci podepíšeme tímto certifikátem a vytvoříme archiv pro distribuci. V Apple iTunes Connect pak vytvoříme novou aplikaci a vyplníme jazyk aplikace, její název a námi zvolený SKU identifikátor pro naši vlastní identifikaci aplikace. Na další stránce doplníme cenu za stažení a datum kdy chceme aplikaci publikovat. Ve vývojovém prostředí XCode vyplníme k aplikaci metadata, kontakt, informace o verzi aplikace a další a pomocí tohoto prostředí aplikaci nahrajeme do iTunes Connect. V iTunes Connect aplikaci pošleme k přezkoumání. To zabere několik dní až týden. Po přezkoumání aplikace trvá zhruba 24 hodin, než bude aplikace dostupná na App Store. Pokud aplikace v pořádku projde přezkoumáním, je připravená k umístění na App Store. Celý proces je poměrně složitý oproti distribuci na Google Play.

6 Závěr

Cílem této práce bylo navrhnout a implementovat multiplatformní aplikaci pro firmu Investiční Aukce s.r.o. určenou pro platformy Android a iOS. Hlavní funkcionalita aplikace spočívá v zobrazení informací o aukcích a možnost zasílání nabídky financování. Součástí práce bylo také navrhnout API. Aplikace momentálně komunikuje s prototypem API, které poskytuje služba Apiary. Softwaroví inženýři firmy naimplementují navržené API a zpřístupní aplikaci investorům.

V úvodu práce byl popsán současný stav trhu týkající se mobilních zařízení. Následovně autor navrhnul postup práce. Dále autor popsal tři možnosti vývoje mobilních aplikací. Nejvhodnější pro zadanou práci byl vývoj hybridní aplikace z důvodu maximalizace společné části kódu a uživatelského požitku. V další kapitole se autor zabýval technologiemi pro vývoj hybridních aplikací a provedl analýzu současně používaných frameworků. Na základě zvolených kritérií autor vybral framework Bootstrap. Následně autor popsal postup při návrhu aplikace, API a následnou implementaci aplikace. Na závěr byla aplikace otestována.

Aplikace Investing.com oproti autorově práci umožňuje změnit její vzhled ze světlého na tmavý. Aplikace Degiro zase disponuje pohodlnějším přihlášením za použití pětimístného kódu.

Cíle stanovené v úvodu práce byly splněny a autorova aplikace funguje bez problémů. Pouze při šířce viewportu menším než 240 pixelů se překrývají některé prvky. Aplikace není přizpůsobená takto malým obrazovkám. Aplikace by mohla být rozšířena o zobrazení současného kurzu eura a aktuálního firemního newsletteru. Dále by bylo vhodné naimplementovat funkcionalitu, která by umožnila aktualizovat aukce za určitý časový interval, nebo aukce aktualizovat automaticky. Nehrozilo by pak zahlcení serveru HTTP požadavky.

7 Reference

About. Bootstrap [online]. 2017 [cit. 2017-04-05]. Dostupné z: http://getbootstrap.com/about/.

About. PhoneGap [online]. 2016[cit. 2017-03-13]. Dostupné z: http://phonegap.com/about/.

About. Sencha [online]. 2017 [cit. 2017-03-12]. Dostupné z: https://www.sencha.com/company/.

All about Ionic. Ionic [online]. 2017 [cit. 2017-03-12]. Dostupné z: http://ionic.io/about.

Alloy readme. GitHub [online]. 2015 [cit. 2017-04-05]. Dostupné z: https://github.com/appcelerator/alloy/blob/master/README.md.

Appcelerator [online]. 2016 [cit. 2017-04-05]. Dostupné z: https://www.appcelerator.com/.

Appcelerator Open Source [online]. 2015 [cit. 2017-04-05]. Dostupné z: http://www.appcelerator.org/.

AJAX Introduction. W3Schools.com [online]. 2017 [cit. 2017-03-09]. Dostupné z: https://www.w3schools.com/xml/ajax_intro.asp.

Ajax v kostce. Inteval.cz [online]. 2015 [cit. 2017-03-12]. Dostupné z: https://www.interval.cz/clanky/ajax-v-kostce/.

AngularJS [online]. 2017 [cit. 2017-03-09]. Dostupné z: https://angularjs.org/.

AngularJS tutorial. W3Schools [online]. 2017 [cit. 2017-03-10]. Dostupné z: https://www.w3schools.com/angular/.

Bootstrap [online]. 2017 [cit. 2017-04-05]. Dostupné z: http://getbootstrap.com/.

CASTLEDINE, Earle, Myles EFTOS a Max WHEELER. *Vytváříme mobilní web a aplikace pro chytré telefony a tablety.* Brno: Computer Press, 2013. ISBN 978-80-251-3763-5..

Core concepts. Ionic Framework [online]. 2017 [cit. 2017-03-03]. Dostupné z: http://ionicframework.com/docs/v2/intro/concepts/.

Documentation. Firebase [online]. 2017 [cit. 2017-05-15]. Dostupné z: https://firebase.google.com/docs/.

Download Kendo UI Core [online]. 2017 [cit. 2017-03-08]. Dostupné z: http://www.telerik.com/download/kendo-ui-core.

Franework7 [online]. 2016 [cit. 2017-04-04]. Dostupné z: http://framework7.io/.

GASSTON, Peter. *Moderní web.* Brno: Computer Press, 2015. ISBN 978-80-251-4345-2..

Getting Started. Mobile Angular UI [online]. 2016 [cit. 2017-04-05]. Dostupné z: http://mobileangularui.com/docs/getting-started/.

Getting started. Bootstrap [online]. 2017 [cit. 2017-04-05]. Dostupné z: https://v4-alpha.getbootstrap.com/getting-started/browsers-devices/.

Indel XDK details [online]. 2017 [cit. 2017-03-06]. Dostupné z: https://software.intel.com/en-us/intel-xdk/details.

Introducing Onsen UI 1.0. Onsen blog [online]. 2014 [cit. 2017-04-04]. Dostupné z: https://onsen.io/blog/introducing-onsen-ui-1-0/.

Ionic Documentation Overview. Ionic [online]. 2016 [cit. 2017-04-05]. Dostupné z: http://ionicframework.com/docs/v1/overview/browser-support.

JAUKER, Stefan. *10 Best Practices for Better RESTful API. M-way solutions* [online]. 2014 [cit. 2017-05-15]. Dostupné z: https://blog.mwaysolutions.com/2014/06/05/10-best-practices-for-better-restful-api/.

JQuery [online]. 2017 [cit. 2017-03-08]. Dostupné z: http://jquery.com/.

JQuery mobile [online]. 2017 [cit. 2017-03-12]. Dostupné z: https://jquerymobile.com/ .

JQuery foundation history [online]. 2017 [cit. 2017-03-12]. Dostupné z: https://jquery.org/history.

KADLEC, Tim. *Responzivní design profesionálně.* Brno: Zoner Press, 2014. Encyklopedie Zoner Press. ISBN 978-80-7413-280-3..

Kendo UI [online]. 2017 [cit. 2017-03-08]. Dostupné z: http://www.telerik.com/kendo-ui .

Mobile Angular UI [online]. 2016 [cit. 2017-04-05]. Dostupné z: http://mobileangularui.com/.

Mobile frameworks comparison chart [online]. 2017 [cit. 2017-03-05]. Dostupné z: http://mobile-frameworks-comparison-chart.com/.

NetMarketShare [online]. 2016 [cit. 2016-11-19]. Dostupné z: https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=9qpcustomb=1.

Onsen UI [online]. 2017 [cit. 2017-03-05]. Dostupné z: https://onsen.io/ .

Overview. Apache Cordova [online]. 2015 [cit. 2017-03-13]. Dostupné z: https://cordova.apache.org/docs/en/latest/guide/overview/index.html.

PhoneGap [online]. 2013 [cit. 2016-11-20]. Dostupné z: https://build.phonegap.com/.

Product Overview. OneSignal[online]. 2017 [cit. 2017-05-15]. Dostupné z: https://documentation.onesignal.com/docs.

React Native [online]. 2017 [cit. 2017-04-04]. Dostupné z: https://facebook.github.io/react-native/.

Release Notes for Sencha Touch 2.4.1. Sencha [online]. 2014 [cit. 2017-04-05]. Dostupné z: http://cdn.sencha.com/touch/sencha-touch-2.4.1/release-notes.html.

SEKULIĆ, Robert. *A brief history of React Native* [online]. 2016 [cit. 2017-04-04]. Dostupné z: https://medium.com/react-native-development/a-brief-history-of-react-native-aae11f4ca39.

Sencha Touch overview [online]. 2017 [cit. 2017-03-06]. Dostupné z: https://www.sencha.com/products/touch/overview .

Sencha Touch - Overview. Tutorialspoint [online]. 2017 [cit. 2017-03-12]. Dostupné z: https://www.tutorialspoint.com/sencha_touch/sencha_touch_overview.htm.

TAFT, Darryl K., *Appcelerator Takes On Adobe AIR with Titanium. EWeek* [online]. 2008 [cit. 2017-04-05]. Dostupné z: http://www.eweek.com/development/appcelerator-takes-on-adobe-air-with-titanium.