Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Informační systém pro objednávku a rozvoz jídla

2019

Rostislav Cibulka

Vedoucí práce:
Mgr. Martin Trnečka, Ph.D.

Studijní obor: Aplikovaná informatika,
prezenční forma

Bibliografické údaje

Autor:	Rostislav Cibulka
Název práce:	Informační systém pro objednávku a rozvoz jídla
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2019
Studijní obor:	Aplikovaná informatika, prezenční forma
Vedoucí práce:	Mgr. Martin Trnečka, Ph.D.
Počet stran:	45
Přílohy:	1 CD/DVD
Jazyk práce:	český

Bibliograhic info

Author:	Rostislav Cibulka
Title:	Food delivery management system
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2019
Study field:	Applied Computer Science, full-time form
Supervisor:	Mgr. Martin Trnečka, Ph.D.
Page count:	45
Supplements:	1 CD/DVD
Thesis language:	Czech

Anotace

Práce pojednává o vývoji informačního systému pro objednávání a rozvoz jídel. Skládá se ze dvou dílčích částí. První z nich je webová prezentace, která obsahuje část pro objednávající, kde je možné vytvářet objednávky a připojení se k věrnostnímu programu pomocí registrace a část pro administrátora stránky, kde je možné objednávky dále zpracovávat a vytvářet nabídku jídel a nápojů. Druhá část je mobilní aplikace určená rozvozcům pro snadnější evidenci rozvezených objednávek, která je napojena na první část.

Synopsis

The thesis deals with developing a food delivery management system. The first part of the thesis focuses on a web based front-end allowing customers to make orders and sign up for a rewards program. It also features admin pages where food and beverages menus can be created, and the orders can be easily managed and processed. The second part of the thesis discusses a mobile application simplifying the tracking of the delivery process for a delivery person. Both applications are fully integrated.

Klíčová slova: PHP, Nette framework, Angular, Ionic framework

Keywords: PHP, Nette framework, Angular, Ionic framework

Tímto bych chtěl poděkovat vedoucímu bakalářské práce panu Mgr. Martinu Trnečkovi, Ph.D. za odborné vedení při vytváření této práce a nasměrování v důležitých částech vývoje.

Místopřísežně prohlašuji, že jsem celou práci včetně příloh vypracoval samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

datum odevzdání práce podpis autora

Obsah

Seznam obrázků

1 Úvod

Rozvoz a objednávka jídel je v současné době naprosto běžná záležitost. Dříve byla objednávka jídel domů doménou velkých měst. Dnes už tomu tak není. I menší obce s několika tisíci obyvateli mají většinou alespoň jeden podnik, který nabízí rozvážku jídel. V dnešní době se čím dál více lidí soustředí na kariéru, a tak mnohdy nezbývá čas na vaření. Když si spočítáme kolik času by nám zabralo uvařit nějaké teplé jídlo, často se nám vyplatí si ho objednat. Nemusíme opouštět teplo domova, nemusíme se převlékat do společnosti, nemusíme na nákup.

Mnoho restaurací má vlastní stránky s možností objednání jídla, jiné zase využívají službu Dáme jídlo[1]. Restaurace v menších obcích však Dáme jídlo nebo podobné služby tolik nevyužívají, lidé tak mají možnost telefonické objednávky nebo případně využít jednoduché stránky. A právě na systém, který by umožnil restauraci, ideálně pizzerii, přijímat a spravovat objednávky, jsem se zaměřil.

1.1 Motivace k vytvoření systému

Zhruba po dovršení 18 let a získání řidičského oprávnění jsem usiloval o to, abych se co nejrychleji naučil řídit. Jistě, občasná jízda autem pod dohledem někoho zkušenějšího vás pomalu posouvá vpřed, ale trvá dost dlouho, než se naučíte dobře řídit. Proto jsem se začal poohlížet po nějaké brigádě, kde bych mohl dělat řidiče o víkendech nebo odpoledne po škole. Shodou okolností se mi v tu dobu naskytla příležitost rozvážet v pizzerii v blízkosti mého bydliště. Po několikadenním zaučení jsem začal jezdit sám a rozvážet jídlo po městě a přilehlých obcích.

Velice rychle jsem si začal všímat, že se spousta činností neustále opakuje a trvají velice dlouho. Jedním z nich byla evidence rozvážených objednávek. Probíhalo to tak, že jsem po obdržení objednávky zapsal na blok papíru ulici, kde právě pojedu a částku, kterou jsem vyčetl z účtenky. V jeden den většinou rozváželi dva rozvozci najednou. Velice často se stávalo, že jsme si objednávku zapsali do špatného sloupce, nebo ji tam zapomněli zapsat úplně. Večer, když rozvoz končil, jsme poté museli spočítat tržbu a porovnat s obsluhou v restauraci, která si naše objednávky také zapisovala. Byl v tom chaos a nejeden večer jsem kvůli tomu ztratil třeba i hodinu hledáním objednávky, kterou si někdo zapomněl zapsat.

Takových věcí, které práci ztěžovaly, bylo opravdu dost a to mne právě vedlo k tomu, že si jednou vytvořím vlastní systém, který by celý proces dokázal zrychlit a ulehčit. Při vytváření různých funkcí systému jsem čerpal ze zhruba 3leté zkušenosti, které jsem jako rozvozce získal.

1.2 Obsah systému

Samotný systém se skládá ze 3 hlavních částí. První z nich je webová prezentace smyšleného podniku, kterou jsem nazval Pizza 45, kde je možné vytvořit objed-

[1]Dostupné na https://www.damejidlo.cz/.

návku, zaregistrovat se a získávat body v rámci věrnostního programu. Jako logo jsem využil volně dostupnou ikonu pizzy[2]. Druhá část je administrace stránky, kde je možné objednávky upravovat, vytvářet nabídku jídel, nápojů, nastavovat ceny a podobně. Poslední část je samostatná mobilní aplikace, která je určena pro rozvozce jídel a má za cíl zjednodušit evidenci rozvezených objednávek.

Systém by měl obsahovat většinu nutných funkcionalit, které jsou k rozvozu jídel v restauraci potřeba a dle mého názoru by bez většího zásahu mohl být použit v reálném provozu. Jedné věci jsem se však záměrně vyhnul: jedná se o vytváření účtenek, výpočtu DPH[3] a EET[4]. Důvodů, proč jsem se tomu vyhnul je několik. Jeden z nich je ten, že se zákony okolo daní neustále mění, a to, co by bylo správně dnes, by nemuselo být v pořádku za několik měsíců. Dalším důvodem je fakt, že zavedené restaurace používají nějaký pokladní systém, který se stará o vytváření účtenek a výpočty DPH. Je proto velice pravděpodobné, že by se při nasazení mého systému na reálný provoz vytvářel nějaký exportovací skript do daného pokladního systému.

Pokud by však můj systém měl řešit EET, tak by se musela doprogramovat (nebo využít nějaká dostupná) knihovna, která řeší výpočet DPH a je schopná odesílat požadavky na servery finanční správy kvůli EET.

1.2.1 Webová prezentace

Webová prezentace pro návštěvníky a potencionální objednávající se skládá z jednoduchého primárního hlavního menu a sekundárního submenu, kde se nachází jednotlivé kategorie jídel (pizza, nápoje, přílohy apod.). Nabízí potřebné informace a funkcionalitu, kterou běžný uživatel očekává. Zejména se jedná o možnost upravit pizzu či jídlo. Je zde také možnost registrace a zapojení do věrnostního programu. Hlavní funkce jsou:

1. přidávání jídel a nápojů do košíku,
2. možnost přidávat k jídlu ingredience navíc,
3. registrace a přihlášení zákazníka,
4. zapojení se do věrnostního programu po úspěšné registraci,
5. možnost zopakovat objednávku (u přihlášeného uživatele),
6. možnost vyprázdnit košík.

1.2.2 Administrační část

Jedná se o část stránky, která je určena pouze pro administrátora či nějakého vedoucího provozu (případně číšníka). Je zde možné vytvářet a upravovat nabídku

[2]Dostupné na https://www.flaticon.com/.
[3]Daň z přidané hodnoty.
[4]Elektronická evidence tržeb.

jídel v restauraci, vytvářet kategorie jídel pro webovou prezentaci, potvrzovat nebo stornovat příchozí objednávky. Samotná administrace bude popsána v dalších částech práce. Hlavní funkce jsou:

1. vytváření jídel a nápojů,
2. správa příchozích objednávek,
3. odeslání emailu s rekapitulací objednávky a předpokládaným časem dovozu,
4. správa rozvozců,
5. nastavení webové části.

1.2.3 Mobilní aplikace

Mobilní aplikace je určená rozvozcům. Její hlavní funkcí je správa potvrzených objednávek. Rozvozce může rezervovat objednávku, kterou bude rozvážet, získávat informace, kde má objednávku dovézt a měnit její stav. Mezi hlavní funkce patří:

1. správa potvrzených objednávek,
2. evidence rozvezených objednávek a tržby jednotlivého rozvozce,
3. možnost evidovat spropitné v daném dni,
4. spustit navigaci k adrese uvedené v objednávce (za předpokladu, že uživatel vyplnil správně adresu v objednávce),
5. odeslat pomocí tlačítka SMS objednávajícímu, že je jeho jídlo na cestě.

2 Podobné aplikace

Spousta restaurací má vlastní stránky, kde je možné vytvořit objednávku. Takových příkladů by se dalo uvést mnoho. Z toho, co jsem zjistil, se většinou jedná o jednoduché aplikace, často postavené na nějakém redakčním systému (CMS[5]), obvykle se jedná o Wordpress[6]. Zaměřím se proto nyní na trochu více obecná řešení. Jedním z nich jsou franšízy, které mají vlastní systém pro správu objednávek. Díky nim je možné založit restauraci pod jejich jménem, ale odvádíte smluvené poplatky. Dále existuje již zmiňované Dáme jídlo, které seskupuje mnoho restaurací dohromady a nabízí jednotné uživatelské rozhraní pro objednávku napříč restauracemi.

2.1 Dáme jídlo

V poslední době si všímám jistého trendu, a sice, že vznikají stránky, které seskupují určité odvětví služeb dohromady a nabízejí jednotné rozhraní. Rozhodně věřím, že tyto aplikace mají své místo na trhu a pro mnoho restaurací to může být velkým přínosem. Co se týče gastronomie, tak se na českém trhu nejvíce ujala služba Dáme jídlo. Restaurace dostane možnost nabídnout své jídlo, nemusí platit velké peníze na vývoj vlastního systému a je velmi pravděpodobné, že si jí lidé všimnou. Služba je velice výhodná, když restaurace začíná a ještě nemá stálou klientelu. I to má ovšem své nevýhody. Restaurace je závislá na službě, která může kdykoliv změnit obchodní podmínky, musí platit poplatky a podobně.

2.2 Franšízy

Co se týče franšíz, tak těch existuje celá řada. Ve zkratce se jedná o to, že nějaká zastřešující firma nabídne novým podnikatelům možnost otevřít si pobočku pod jejich jménem. Nový vedoucí pobočky získá recepty, zaměstnanci jsou zaškoleni a rozjetí pobočky je jednodušší. Obdrží hotové řešení, nemusí se starat o vývoj nějakého systému.

Nevýhodou je, že je pobočka závislá na nabídce jídel, kterou se franšíza prezentuje. Pobočka musí platit poplatky z tržby v řádech procent. Není zde příliš „volná ruka“ v tom, co zákazníkům nabídnout. Tomuto faktu se ovšem nelze divit, jelikož snaha franšíz je, aby zákazník dostal na každé pobočce stejný produkt. Asi největší nevýhodu však vidím v tom, že si restaurace nemůže budovat vlastní značku.

2.2.1 Telepizza

Na první pohled spatřuji největší podobnost mého projektu ve franšíze Telepizza[7]. Zaměřuje se na rozvoz jídel, zejména pizz.

[5] Zkr. pro Content management system.
[6] Dostupné na https://wordpress.com/.
[7] Dostupné na https://www.telepizza.cz/.

2.3 Srovnání

Jakkoli nechci zpochybňovat užitečnost franšíz nebo služby Dáme jídlo, tak věřím, že je pro spoustu restaurací stále výhodnější mít vlastní řešení, které jim umožní být nezávislé na podobných službách. Můj projekt je zaměřený na to, aby dokázal fungovat jako základ pro nějakou zavedenou restauraci, která nabízí rozvoz jídel.

Vzhled webové části, dostupný běžným klientům, by se dal velice snadno změnit a přizpůsobit, tím pádem je zde prostor budovat vlastní značku. Dokázal bych si představit, že bych tento projekt nasadil do více restaurací nezávisle na sobě a upravoval jednotlivé instance podle potřeb dané restaurace.

3 Použité technologie

Technologií, pomocí kterých se dají vytvářet webové aplikace, je celá řada. Osobně raději využívám již zavedené technologie, na kterých byly případně postavené úspěšné projekty, než se učit něco, co teprve vzniklo a není jisté, že se technologie uchytí. Dalším kritériem pro mne také je, jak velká komunita kolem dané technologie existuje. Pokud bych totiž používal něco, co používá jen několik nadšenců, tak je dost pravděpodobné, že si budu muset spoustu knihoven napsat sám, s případnými problémy mi pravděpodobně nikdo nepomůže, a tím se následný vývoj podstatně prodlužuje. Také se snažím vyhnout tomu, abych kvůli nějakému efektu stáhnul celou knihovnu, kterou jinak takřka nevyužiji. Níže budu popisovat použité technologie a důvody, proč jsem je použil.

3.1 PHP: Hypertext Preprocessor

PHP[1] je dynamicky typovaný skriptovací jazyk, který je nejvíce rozšířený na poli webových stránek. Má velkou aktivní komunitu a bylo na něm postaveno mnoho úspěšných projektů, mezi které se řadí například i sociální síť Facebook. Jazyk obsahuje velké množství funkcí, které usnadňují práci s textem. PHP se obvykle používá pro generování dynamických stránek. Je velice spjatý s databázovou technologií MySQL, ale má podporu i pro ostatní technologie využívající jazyk SQL (Oracle, Postgres apod.).

3.2 MySQL

MySQL[2] je databázový server využívaný při vývoji malých i velkých aplikací. Velmi často se používá právě v kombinaci s jazykem PHP. Vkládání, výběr a úprava dat nad databází se provádí pomocí SQL[8] dotazů.

3.3 HTML5

HTML[3] (HyperText Markup Language) je označován jako základní stavební kámen pro vývoj webu. Udává strukturu stránky a význam obsahu, který se na ní nachází. Zdrojové kódy mají příponu .html. Jedná se o značkovací jazyk, který je tvořen značkami, kterým se říká tagy. Jednotlivé tagy mohou být párové nebo nepárové a obsahovat nejrůznější atributy, které jsou společné buď pro všechny, anebo se vážou pouze k určitému tagu. HTML5 je poslední verzí standardu, který definuje W3C[9]. Samotné HTML nebylo zamýšleno k tomu, aby jeho tagy určovaly, jak bude výsledný dokument vypadat, o to se stará CSS (více 3.4). HTML se velmi často používá v kombinaci s jazykem JavaScript.

[8] Zkr. pro Structured Query Language.
[9] World Wide Web Consortium (https://www.w3.org/).

3.4 JavaScript

JavaScript[4] je interpretovaný objektově orientovaný jazyk, který je nejvíce využívaný v prostředí prohlížečů. Podporuje objektově orientované, imperativní i funkcionální programování. Umožňuje manipulovat s elementy na stránce, reagovat na různé události, například kliknutí na nějaké tlačítko a podobně. JavaScript je implementací ECMAScriptu, který se neustále vyvíjí. Při vytváření skriptů je nutné pamatovat na to, že uživatelé mohou mít starší verzi prohlížeče, která ještě nemusí podporovat nejnovější funkcionalitu. Využití JavaScriptu neslouží jen pro úpravu obsahu u klienta, může také komunikovat se serverem anebo dokonce běžet na straně serveru.

3.5 CSS

CSS[5] (Cascading Style Sheets) je jazyk, který popisuje podobu HTML dokumentu. Popisuje, jak by měly být HTML elementy zobrazeny na obrazovkách, při tisku na papír a dalších médiích. CSS vzniká pod záštitou W3C. CSS kód může být přítomen přímo v tagu za použití atributu style, dále pak za použití elementu <style>, který musí být uvnitř <head> elementu, avšak nejčastěji je uložen externě jako samostatný soubor s příponou .css, na který HTML stránka odkazuje a prohlížeč tyto styly následně načítá. CSS styly se zapisují pomocí pravidel, které se skládají ze selektoru (ten určuje, které elementy v dokumentu tímto pravidlem budou ovlivněny) a poté dvojic vlastností a hodnot.

3.6 Nette Framework

Nette [6] je známý český framework, který je postavený na Model-View-Controller (MVC) architektuře, která rozděluje aplikaci do 3 základních stavebních kamenů. Prvním z nich je model, který se stará o logiku celé aplikace, stará se například o přihlašování uživatelů, změny vnitřního stavu entit (článek, objednávka apod.), komunikaci s databází. Dále je zde view (česky pohled), který se stará o zobrazení výsledku požadavku. Posledním prvkem je controller (v Nette označován jako presenter), který je něco jako prostředník mezi těmito vrstvami. Controller zpracovává požadavky uživatele, následně zavolá aplikační logiku (model) a poté si nechá pomocí view vykreslit data.

Na Nette frameworku je postavená celá serverová část aplikace. Pro použití tohoto frameworku jsem se rozhodl z několika důvodů. Předně se jedná o framework, který je v České republice velice oblíbený, má velkou komunitu, která dokáže pomoct s případnými probémy. Další věcí je, že jednotlivé části frameworku se dají většinou použít samostatně, dají se rozšiřovat a upravovat pro vlastní potřeby aplikace. Nette framework se stará o spoustu věcí, které se týkají bezpečnosti aplikace. Šablonovací systém latte automaticky escapuje zobrazovaný obsah, a tak tedy účinně předchází Cross Site Scripting útokům. Má také výbornou tvorbu formulářů, pomocí kterých lze nadefinovat různá omezení (délka vstupů, rozsah hodnot a podobně.), která jsou poté automaticky

kontrolovány na straně serveru a není tak potřeba při zpracování formuláře dále kontrolovat správnost dat (za předpokladu, že omezení definovaná ve formuláři jsou dostačující, například při registraci uživatele bylo potřeba provést, i přes úspěšné zavolání formuláře, kontrolu, zda zadaný email již není registrovaný).

3.7 Apache Cordova

Apache Cordova[7] je open-source framework, který umožňuje vytvářet mobilní aplikace pomocí technologií HTML, CSS a JavaScriptu. Je vytvořen skeleton pro jednotlivé platformy (iOS, Android), skrze který je následně samotná aplikace spuštěna. Framework dále obsahuje pluginy, pomocí kterých se přistupuje k nativním funkcím dané platformy pomocí volání funkcí JavaScriptu. Pluginy umožňují přístup například k fotoaparátu, lokálnímu úložišti, různým senzorům a podobně.

3.8 Ionic Framework

Ionic Framework[8] je open source projekt pro tvorbu uživatelského mobilního i desktopového rozhraní, který využívá technologie HTML, CSS a JavaScript. Cílem tohoto frameworku je vytvářet multiplatformní aplikace pro iOS, Android a desktop. Cílem je tedy napsat kód jednou a spustit aplikaci na zařízeních s různým OS[10], kdy je navíc možnost zachovat zvyklosti tvorby uživatelského rozhraní pro daný OS (ikony například vypadají různě u Android a iOS apliakce). Ionic Framework má oficiální integraci s Angular[11] frameworkem (lze ho však používat i bez něj nebo využít jiný JS framework).

[10] Zkr. pro operační systém.
[11] Velmi oblíbený JS framework, více zde: https://angular.io/.

4 Programátorská dokumentace

V této sekci popíši stručně rozdělení serverové části aplikace a databázovou strukturu.

4.1 Serverová část

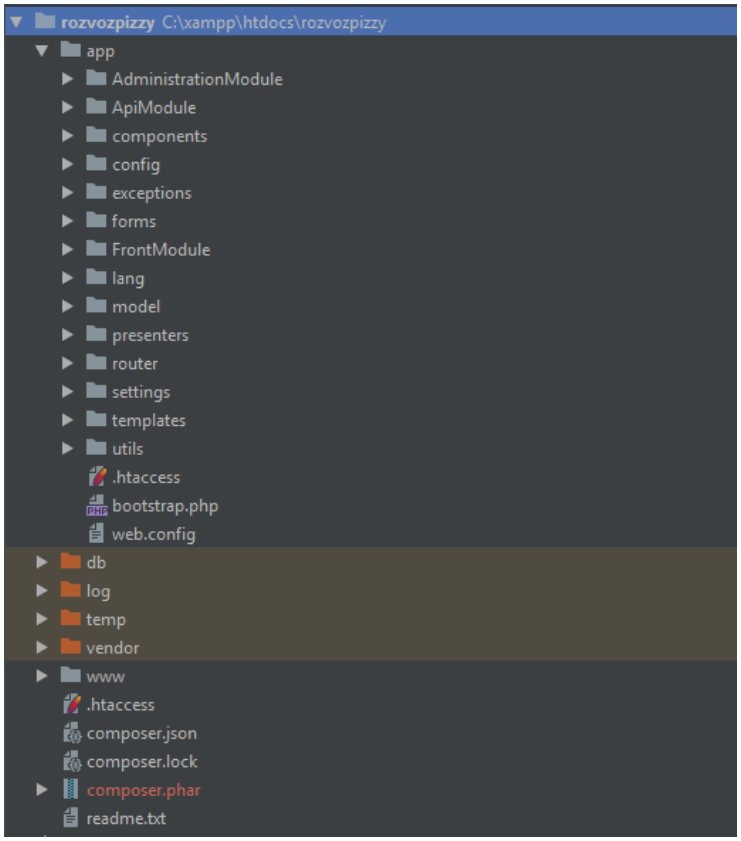
Jak už jsem se výše zmiňoval, pro serverovou část aplikace jsem využil Nette Framework, jeho struktura je odvozena ze sandboxu[12] pro verzi Nette 2.4[13]. Struktura (viz obrázek 1) se skládá z několika složek, z nichž nejdůležitější je `app/` složka, ve které je veškerá funcionalita tohoto projektu (popsáno níže).

Dále zde nalezneme složky `db/` (obsahuje .sql soubory potřebné při instalaci), `log/` (zde se ukládají chybové zprávy způsobené v produkčním režimu), `temp/` (cachované soubory, které Nette vytváří při prvním requestu), `vendor/` (samotný Nette Framework + knihovny, které využívá a případně i další knihovny třetích stran) a složka `www/`.

Ve složce `www/` se nacházejí všechny veřejně dostupné soubory (obrázky, styly, JS skripty) a je zde také přítomen index.php soubor, což je inicializační skript pro všechny dotazy na server (vyjma zmiňovaných obrázků, stylů a JS skriptů, ke kterým se přistupuje přímo).

[12] Dostupné z https://github.com/nette/sandbox/tree/v2.4.

[13] V době psaní této práce už proběhl release Nette 3.0, avšak v době vývoje aplikace byla nejnovější verze právě 2.4.



Obrázek 1: Struktura výsledného projektu

4.1.1 App složka

V `app/` složce se nacházejí 3 samostatné moduly (AdministrationModule, FrontModule, ApiModule), které mají určitou úlohu v rámci aplikace. Je zde také několik dalších složek, ty stěžejní se nyní níže pokusím popsat.

4.1.1.1 Model

Model je část aplikace, kde se nachází veškerá logika a provádí se operace nad databází. `App/model/` složka, obsahuje třídy, které jsou společné pro všechny moduly, jednotlivé moduly mají poté dále svou vlastní `model/` složku, specifickou pro daný modul a mnohdy využívá právě logiku z obecného modelu.

Při vývoji jsem využil databázovou vrstvu Database Explorer[14], která je přítomna v základu Nette Frameworku, a která umožňuje pohodlnější práci s databází. Jednou z jejích předností je například přístup z aktivního řádku tabulky k přidruženému záznamu z jiné tabulky, který je definován pomocí cizího klíče,[15]

[14] Vice informací zde: https://doc.nette.org/cs/2.4/database-explorer.

[15] Více informací například zde: https://dev.mysql.com/doc/refman/5.5/en/innodb-foreign-key-constraints.html.

aniž bychom museli ručně vytvářet další SQL dotaz.

Postupem vývoje, kdy začínaly přibývat další tabulky a logika aplikace se začala více komplikovat, jsem se dostal do stavu, kdy byl už další vývoj mnohem obtížnější a narážel jsem na duplicitní kód. Proto jsem se rozhodl napsat vlastní, velice jednoduchý pseudo ORM[16] framework, který využívá zmíněný Database Explorer. Skládá se ze 2 abstraktních tříd:

- **BaseEntity** - jedná se o abstraktní třídu, která slouží jako přepravka na data, která jsou získána z databáze. Třída (entita), která tuto abstraktní třídu rozšiřuje, má nadefinované proměnné, které se naplňují a získávají pomocí get a set metod, takže je možné při změně nějaké proměnné provádět různá omezení. V mém případě entita odpovídá vždy jednomu záznamu z nějaké tabulky.

- **BaseRepository** - jedná se o abstraktní třídu, jejíž třídy, které ji rozšiřují, musí definovat, k jaké tabulce se váže a jaké entity mají vytvářet (pomocí anotací nad definicí třídy). Jednotlivé repositáře pak mají společné rozhraní, které volá metody z Database Explorer objektu a následně vytváří příslušné entity, které poté vrací. Umožňuje také entity ukládat, vytvářet anebo mazat.

Než jsem se pustil do napsání této knihovny, zvažoval jsem, jestli nevyužít nějaký existující ORM framework (některé z nich jsem si samozřejmě vyzkoušel), avšak nakonec jsem dospěl k rozhodnutí, že bych si pro studijní účely rád vyzkoušel naprogramovat něco vlastního. Ovšem napsat univerzální a plnohodnotný ORM framework by nebyla jen několikadenní záležitost, ale spíše by to trvalo několik měsíců, ne-li let, proto to označuji jako pseudo ORM framework.

Při využití ORM frameworku se také musí myslet na to, že ačkoliv to značně zjednodušuje práci s databází a logika aplikace se jednodušeji spravuje, tak její využití přináší určitou režii navíc, jelikož se neustále vytváří entity. Vzhledem k předpokladu, že má aplikace je určená pro menší restauraci s rozvozem, neočekává se příliš velká návštěvnost, a proto jsem přesvědčen, že režie navíc je v tomto případě spíše zanedbatelná a výhody plynoucí z jejího využití převažují nad negativními aspekty.

V model složce se mimo entit a repositářů dále nacházejí Service a Manager třídy, které používají repositáře a slouží jako nejvyšší modelová vrstva, jejíž metody se volají, když chce například návštěvník vložit, upravit či odebrat jídlo z košíku a podobně.

4.1.1.2 FrontModule

Ve FrontModule je část aplikace, která je určená pro návštěvníky stránky. Nachází se zde složka `app/FrontModule/model/`, specifická pro tento modul.

[16] Zkr. pro objektově relační mapování.

Jsou zde také presentery, starající se o určitou část na stránce, například ShoppingCartPresenter se stará o věci spojené s nákupním košíkem a odesláním objednávky. Dále jsou zde šablony, které se vážou vždy k nějakému presenteru, ty mají příponu .latte.

V tomto případě jsem se snažil o to, aby šablony generovaly sémanticky správný HTML kód, jelikož se jedná o část aplikace, ke které mají přístup všichni. Samotný design stránky je vytvořen pomocí čistého CSS, bez využití knihovny, jako je například Bootstrap framework[17].

4.1.1.3 AdministrationModule

AdministrationModule je část aplikace určená pro administrátory stránky. Je zde logika pro zpracování objednávek, přidávání obsahu na web (nové jídla, nápoje apod.) či možnost různého nastavení stránky.

Design této části vychází z volně dostupné šablony SB Admin,[18] postavené na zmíněném Bootstrap frameworku.

4.1.1.4 ApiModule

ApiModule je modul, který slouží jako REST[19] API pro požadavky z mobilní aplikace. Z velké části využívá logiku z AdministrationModule, kterou většinou pouze rozšiřuje o nějakou kontrolní funkcionalitu navíc.

Samotné endpointy na API jsou generované pomocí rozšíření ApiRouter[20] pro Nette Framework. REST API v podstatě umí pouze vracet objednávky (v JSON[21] formátu) a případně jim měnit stav. Přihlašování a zabezpečení se věnuji níže v sekci 4.1.2.

4.1.1.5 Config

V config složce se nacházejí konfigurační soubory, kde se provádí nejrůznější nastavení stránky, jako jsou přístupové údaje k databázi, registrace presenterů, komponent a mnoho další.[22]

4.1.1.6 Ostatní složky

V projektu se nachází další složky, které hrají určitou úlohu v rámci aplikace. Například ve složce `app/router/` se nachází třída RouterFactory, starající se o nasměrování požadavků do správného modulu a zavolání příslušné akce požadovaného presenteru [10]. V `app/components/` a `app/forms/` složkách

[17] Dostupné z https://getbootstrap.com/.
[18] Dostupné z https://startbootstrap.com/templates/sb-admin/.
[19] Zkr. pro Representational State Transfer.
[20] Dostupné z https://ublaboo.org/api-router/.
[21] Zkr. pro JavaScript Object Notation, více např. zde: https://www.zdrojak.cz/clanky/json-jednotny-format-pro-vymenu-dat/
[22] Více zde: https://doc.nette.org/cs/2.4/configuring

se nachází znovupoužitelné komponenty, které obsahují určitou funkcionalitu, tu lze poté jednoduše použít na více místech v aplikaci, což přispívá k zabránění vytvoření duplicitního kódu [11].

4.1.2 Přihlášení a zabezpečení

Při přihlašování k návštěvnické sekci a do administrace se využívá třída Passwords[23], která využívá algoritmus bcrypt, pro vytvoření otisku z hesla, které uživatel zadá ve formuláři, jenž je následně porovnán s otiskem uloženým v databázi. V aplikaci je také definována statická ACL[24], kde jsou definované tyto role:

- **Guest** - tuto roli má každý nepřihlášený uživatel,
- **Member** - přihlášený návštěvník stránky,
- **Admin** - administrátor stránky,
- **Driver** - řidič, kterému admin udělil práva z member role.

V konfiguračních souborech je poté nastaveno, k jakým částem aplikace mají dané role přístup. Samotná identita přihlášeného uživatele se ukládá v sessions na serveru, o to se ale stará framework[12].

Mobilní aplikace využívá pro kontrolu přístupu k API JSON Web Token (dále už jen JWT[25]), který se předává v Authorization hlavičce requestu. Získání tokenu probíhá při zavolání POST requestu na endpointu api/v1/login/authenticate, kdy jsou v případě úspěchu vrácena tato data (viz také obrázek 2):

- **AccessToken** - token, který se používá pro přístup k objednávkám, změně stavů a podobně, jeho platnost je nastavena na 1 hodinu,
- **RefreshToken** - token, který se používá pro získávání nového accessTokenu, platnost nastavena na 7[26] dní, poté je nutné se znovu přihlásit,
- **User** - obsahuje informace o přihlášeném řidiči.

[23] Dokumentace ke třídě zde: https://api.nette.org/2.4/Nette.Security.Passwords.html.
[24] Zkr. pro Access Control List.
[25] Více informací zde: https://jwt.io/.
[26] Platnost obou tokenů lze změnit v konfiguračním souboru.

```
{
  "accessToken": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzUxMiJ9.eyJpc3MiOiJyb3p2b3pwaXp6eSIsImlhdCI6MTU1NTkyMzg5NSwiZXhwIjoiMTU1NTkyNzQ5NSIsImFwaUlkIjoiYWZ3MWx6NGs3MiIsInR5cGUiOiJhY2Nlc3MifQ.mBpputKDGnKH3eQLgSlDSEEuMiP3CIF3_s36Oyui72Bj1cwco5SdGNBELwR3f8W8E6CM3BC_cyHihdHdfp5Y-g",
  "refreshToken": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzUxMiJ9.eyJpc3MiOiJyb3p2b3pwaXp6eSIsImlhdCI6MTU1NTkyMzg5NSwiZXhwIjoiMTU1NjUyODY5NSIsImFwaUlkIjoiYWZ3MWx6NGs3MiIsInR5cGUiOiJyZWZyZXNoIn0.gBbNe_hZFT5Y1bBs7ODYuL0ydleDE7ytN_8reE6KLsGkiBi24-grQona_3tZqUsCdczJsl7-U91ekv-7X_RO_w",
  "user": {
    "id": 1117,
    "username": "cibulkarostislav@gmail.com",
    "role": "driver",
    "person": {
      "id": 88,
      "detail": {
        "id": 88,
        "firstName": "Rostislav",
        "lastName": "Cibulka",
        "phone": "+420222333333",
        "secondaryPhone": "",
        "email": "cibulkarostislav@gmail.com"
      }
    }
  }
}
```


Obrázek 2: Odpověď serveru po úspěšném přihlášení přes API

4.1.3 Struktura databáze

Návrhu databáze jsem věnoval poměrně dost času, jelikož jsem chtěl, aby aplikace byla do budoucna rozšiřitelná a dalo se z ní snadno získávat potřebná data. Nakonec jsem vytvořil celkem 22 tabulek, proto nebudu popisovat každou z nich, ale pokusím se je rozdělit do 3 základních kategorií:

- **Uživatel** - tabulky, které se vážou k uživatelům, jako je například adresa, osobní údaje nebo věrnostní program,
- **Objednávka** - tabulky, vytvářející výsledný nákupní košík,
- **Produkt** - tabulky, kde je uložena nabídka, z jaké si mohou návštěvníci vybírat. Patří zde tabulky pro nastavování cen jídel a nápojů, zařazení jídel do kategorií nebo tabulka pro definici, z jakých ingrediencí se jídlo skládá.

4.2 Mobilní aplikace

Mobilní aplikace je postavená na Ionic frameworku, o kterém jsem se zmiňoval v použitých technologiích, aplikace je napsaná v Angularu, který má s Ionic frameworkem oficiální integraci. Samotná aplikace není nikterak rozsáhlá a obsahuje nejnutnější funkcionalitu, kterou z mého pohledu rozvozce využije.

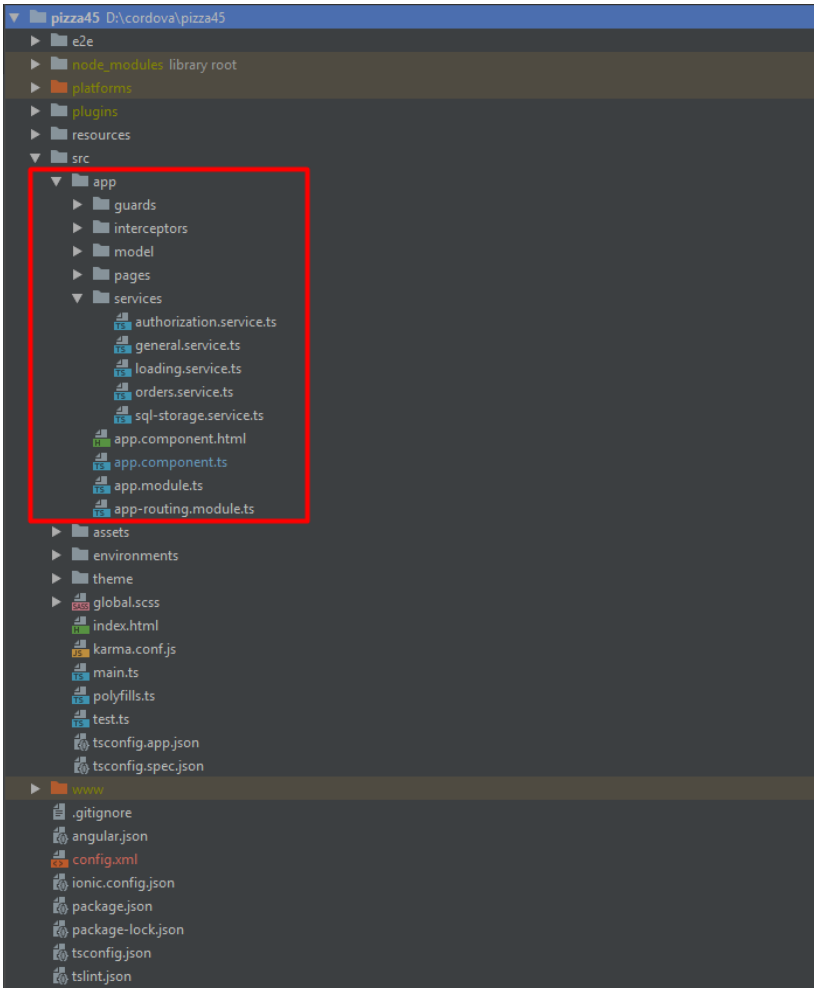
4.2.1 Struktura

Struktura aplikace vychází z vygenerovaného projektu pomocí Ionic CLI[27]. Výsledná struktura je zobrazena na obrázku 3. Samotné zdrojové kódy aplikace jsou ve složce `src/app/`, ostatní složky mimo `src/` jsou de facto automaticky vygenerované.

Logika aplikace se nachází ve složce `src/app/services/`, kterou jednotlivé části aplikace používají. Níže jsou popsány nejdůležitější třídy.

- **SqlStorageService** - používá SQLite objekt, který umožňuje uložit persistentně data v úložišti telefonu. Ostatní třídy ve složce services většinou tuto třídu využívají pro persistentní ukládání dat.
- **AuthorizationService** - využívá se k přihlašení, odhlášení uživatele a pro získání nového přístupového tokenu (accessToken), kontroluje tak, jestli je uživatel přihlášen.
- **OrderService** - odesílá požadavky na server pro získání nových objednávek, případně mění jejich stav.

[27] Dostupné zde: https://ionicframework.com/docs/cli.



Obrázek 3: Struktura výsledné mobilní aplikace

4.2.2 Offline režim

Aplikace se umí v případě ztráty internetového připojení přepnout do offline režimu. Tento režim se od normálního liší pouze v tom, že zobrazuje data uložená v lokálním úložišti a dokud se nepřepne v nastavení na online režim (za předpokladu, že je internetové připojení k dispozici), nemůže rozvozce získávat nová data.

Samotné ukládání objednávek v úložišti telefonu je řešeno pomocí jednoduché tabulky, která má 2 sloupce - key a value. Všechny objednávky se tak ukládají pod klíčem order-id (například order-34). Ve value sloupci už je naparsovaný JSON celé objednávky, kterou vrací server. Při inicializaci aplikace se pak načtou všechny objednávky z lokálního úložiště telefonu (to se děje v OrderService).

Důvodem, proč jsem se rozhodl k takto jednoduchému způsobu ukládání objednávek je fakt, že rozvozce nebude potřebovat nějak složitě vyhledávat v objednávkách podle složitých kritérií. V telefonu budou totiž uloženy vždy jen nové

objednávky a ty, které rozvozce v daném dni rozvezl (případně několik stornovaných). Najednou tedy v telefonu bude uloženo maximálně několik desítek objednávek (z praxe jsem vypozoroval, že i při 12hodinovém rozvážení v jednom dni jsem byl schopen rozvézt maximálně okolo 40 objednávek).

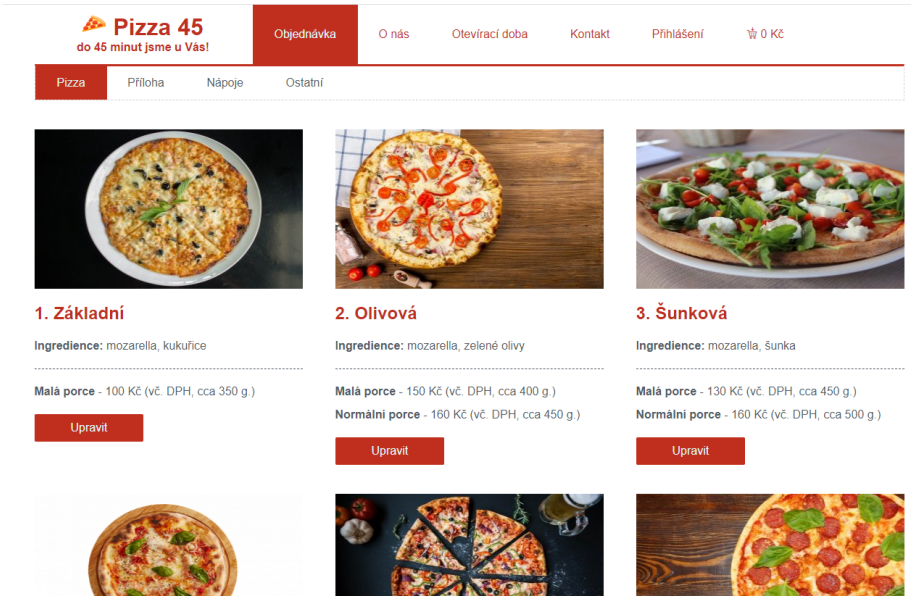
5 Uživatelská dokumentace

V této části se pokusím popsat funkcionalitu jednotlivých částí projektu z pohledu uživatele.

5.1 Část pro návštěvníky

5.1.1 Úvodní stránka

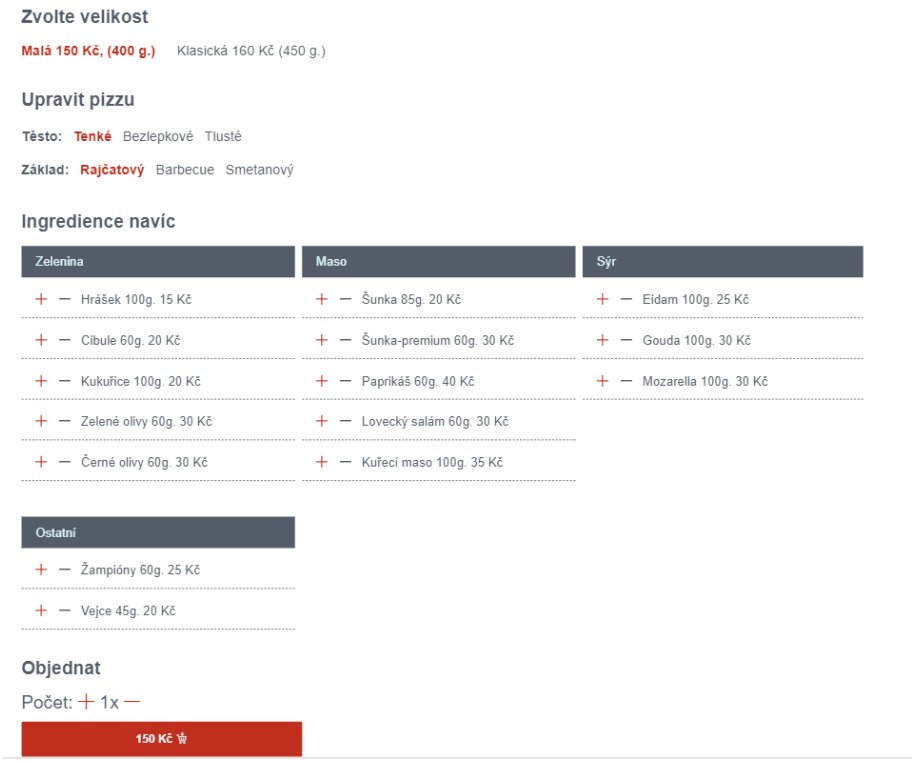
Úvodní stránka začíná rovnou v sekci, kde je seznam jídel z první kategorie v navigaci (viz obrázek 4). Mělo by se jednat o jídla, která jsou pro danou restauraci ta nejdůležitější. Pod obrázkem a názvem jednotlivých jídel je také uvedeno jejich složení (seznam ingrediencí). Pro získání bližších informací může návštěvník na jídlo kliknout, čímž dojde k přesměrování na detail daného jídla.



Obrázek 4: Úvodní stránka v návštěvnicé sekci

5.1.2 Detail jídla

Detail vybraného jídla obsahuje možnost nastavení si daného jídla (viz obrázek 5), například přidání ingredience navíc, výběr velikosti a v případě pizzy také výběr těsta a základu. Tlačítko vespod potom zobrazuje výslednou cenu a po kliknutí se jídlo vloží do košíku a uživatel je nakonec přesměrován opět do výpisu kategorie, kde byl naposledy.



Obrázek 5: Příklad detailu pizzy

5.1.3 Vytvoření objednávky

Jakmile se uživatel rozhodne, že má v košíku vše, co chce, je potřeba projít procesem objednávky, který se skládá z několika kroků:

1. **Kontrola nákupního košíku** - během vytváření nákupního košíku se může stát, že se nějaké jídlo nebo ingredience stane nedostupnými, proto uživatel v přehledu objednávky vidí, které produkty musí upravit, aby mohl objednávku vytvořit (viz obrázek 6). Uživatel však i přesto může pokračovat v dalších krocích a vyplnit údaje a k této části se vrátit až na samotném konci.

2. **Vybídnutí k registraci** - jakmile uživatel klikne na tlačítko pro pokračování, objeví se vybídnutí k tomu, aby se přihlásil. Pokud nemá vytvořen účet, má možnost se registrovat. Tento krok není povinný a může kliknout na odkaz „pokračovat bez registrace“.[28]

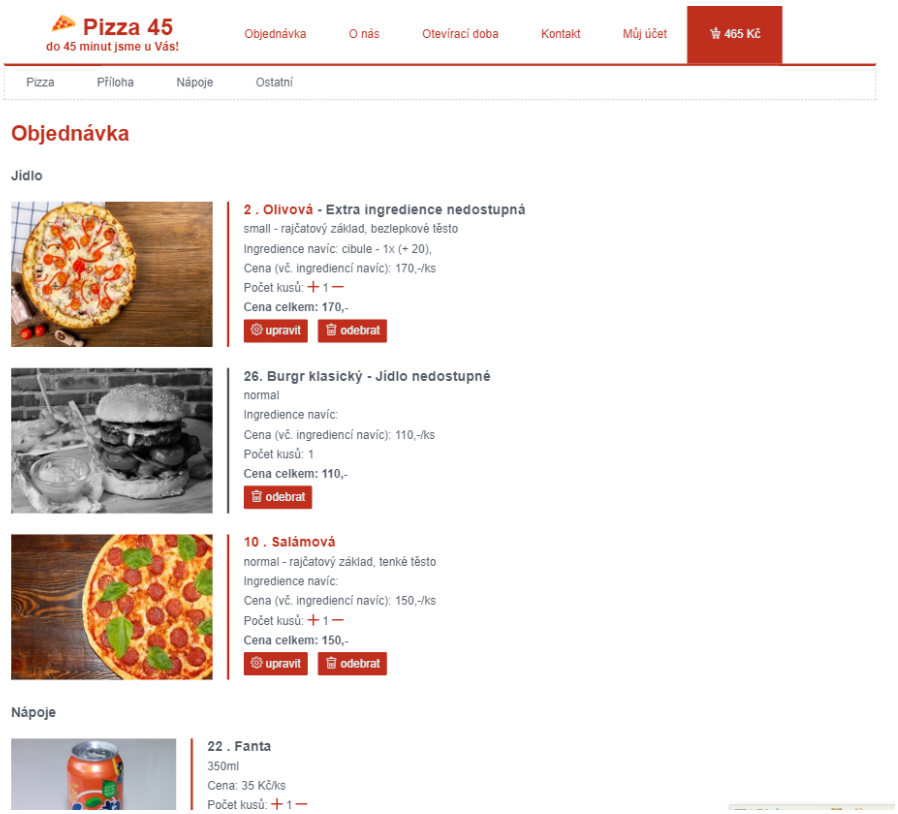
3. **Osobní údaje** - v tomto kroku[29] je uživateli zobrazen formulář, kde se po něm požaduje zadat tyto údaje: jméno, příjmení, e-mail a telefon.

[28] Tento krok je přeskočen, je-li uživatel již přihlášený.
[29] V případě, že je uživatel přihlášen, je tento krok přeskočen.

4. **Adresa** - v dalším kroku[30] uživatel zadává údaje na místo určení, kde se objednávka bude rozvážet. Skládá se z povinných i nepovinných položek (viz obrázek 7). Jednotlivé formuláře mají obvykle nadefinované nějaké omezení pro vstupy: například při zadání příliš velkého čísla nebo špatně zapsané emailové adresy je uživatel upozorněn červeným textem pod daným vstupem.

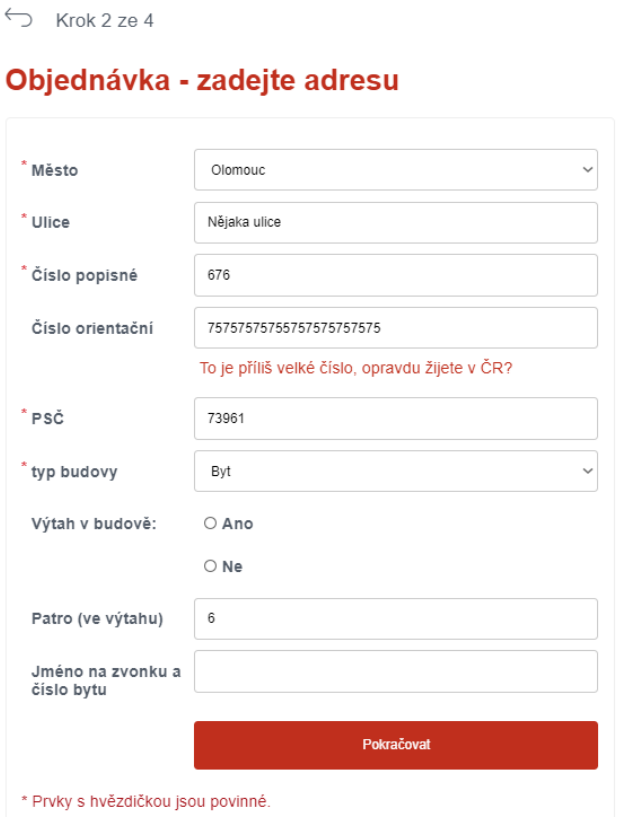
5. **Rekapitulace objednávky** - v předposledním kroku je zobrazena rekapitulace objednávky s výčtem všech jídel a nápojů, spolu s osobními údaji a dodací adresou[31].

6. **Potvrzení** - v posledním kroku může uživatel ještě vybrat typ platby[32] a napsat případné poznámky k objednávce. V tomto kroku také dochází ke kontrole, zda-li jsou všechny produkty v košíku dostupné, pokud ano, objednávka se odešle, pokud ne, je uživatel upozorněn na nutnost úpravy košíku.



Obrázek 6: Ukázka nákupního košíku s nedostupnými položkami

[30]Přihlášený uživatel má tyto informace předvyplněné podle poslední objednávky.
[31]Přihlášený uživatel ještě může aplikovat kredity získané ve věrnostním programu.
[32]V současné době pouze hotovost, platba kartou je zde jako příklad dalšího možného rozšíření a není možné tuto volbu vybrat.



Obrázek 7: Ukázka formuláře při zadávání adresy

5.1.4 Část pro registrované

Registrace byla vytvořena za účelem urychlit návštěvníkům opakované objednávání jídel. Do této sekce se dostaneme přes odkaz „Přihlášení“ v hlavní navigaci, v textu se pak nachází odkaz pro registraci. Také je zde odkaz pro obnovu hesla.

Samotná registrace vyžaduje vyplnit email, heslo, jméno, příjmení a telefon[33]. Po úspěšném vyplnění formuláře je uživateli odeslán potvrzovací email, kde je vygenerovaný odkaz, pomocí kterého lze účet potvrdit a až poté je umožněno se pod zadanými údaji přihlásit.

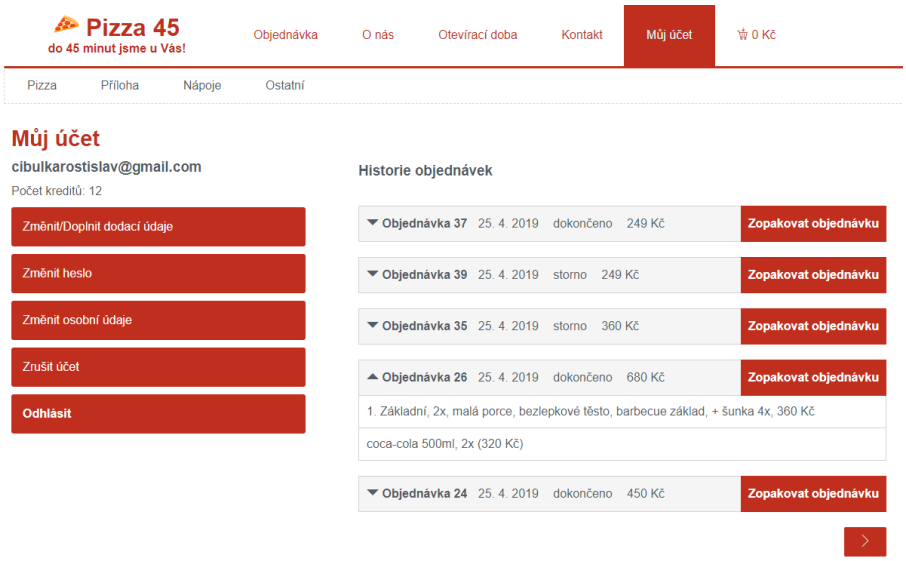
5.1.4.1 Přihlášený uživatel

Po přihlášení je uživatel přesměrován na stránku (viz obrázek 8), kde může měnit údaje na svém účtu, případně se podívat na historii objednávek[34]. Velkou výhodou přihlášeného uživatele je, že nemusí při opakované objednávce vyplňovat znova dodací údaje, tyto jsou totiž v jednotlivých formulářích předvyplněné.

[33] Email a telefon může být zaregistrovaný pouze jednou.
[34] U objednávky je vždy tlačítko pro její zopakování.

5.1.4.2 Věrnostní program

Jakmile se uživatel zaregistruje, automaticky se stává členem věrnostního programu. Jeho podstata tkví v tom, že za každou objednávku uživatel obdrží určité procento[35] z výsledné částky zpět, a sice v kreditech. Ty lze poté využít (po dosažení určitého počtu) u další objednávky ke snížení ceny nebo jejímu úplnému zaplacení.



Obrázek 8: Ukázka detailu účtu přihlášeného uživatele

5.1.5 Ostatní odkazy v navigaci

V navigaci se dále nachází odkazy na stránky „O nás“, „Otevírací doba“ a „Kontakt“. Jejich obsah odpovídá jejich názvu a jedná se de facto o statické stránky (s výjimkou otevírací doby, kterou lze v administraci stránky upravit), které mají sloužit jen jako možný příklad.

5.2 Administrace

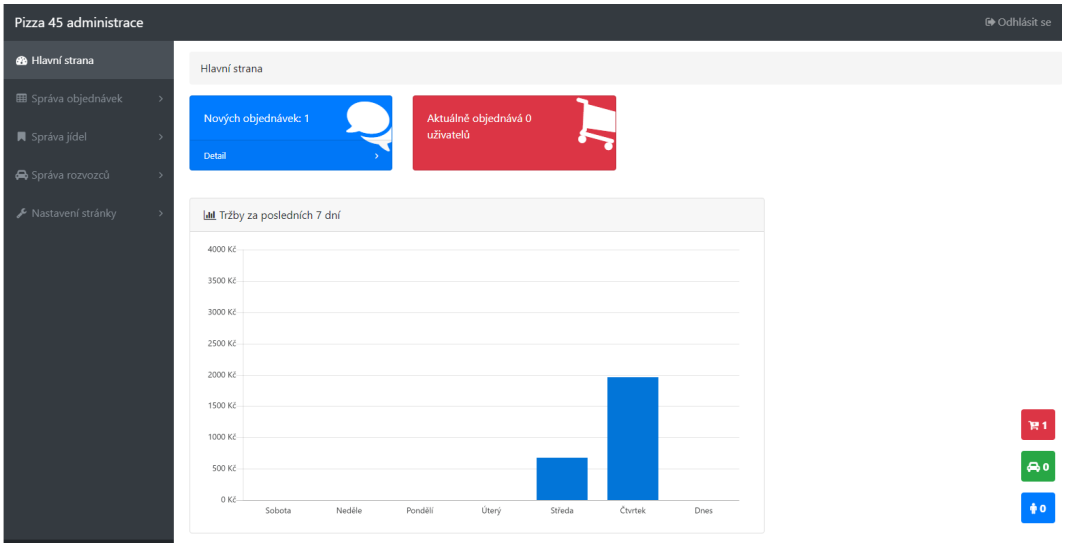
Administrace představuje druhou stěžejní část projektu, níže se pokusím zmínit nejdůležitější prvky, které se zde nachází. Ačkoliv je šablona administrace postavená na Bootstrap 4 frameworku, nesnažil jsem se optimalizovat zobrazení stránky na malém zařízení. Důvod je ten, že by samotná práce s nastavováním stránky (vyplňování formulářů, nahrávání obrázků k produktům apod.) byla nepohodlná na tabletu nebo mobilním telefonu.

[35] Nastavuje se v administraci stránky.

5.2.1 Úvodní stránka

Po přihlášení do administrace se dostaneme na úvodní stránku (viz obrázek 9), kde se nachází horizontální menu na levé straně, tam je možné přepínat mezi jednotlivými částmi administrace. Na každé stránce v administraci se vpravo dole nachází 3 malé informační bloky, které se v pravidelném intervalu obnovují a zobrazují tyto údaje:

- **Počet nových objednávek** - za novou objednávku se zde považuje objednávka, která byla odeslána zákazníkem a ještě se nijak dále neměnil její stav.
- **Počet potvrzených objednávek** - jedná se o objednávky, které byly v administraci potvrzeny, zkontrolovány a čeká se na jejich dokončení (tedy jejich rozvezení).
- **Počet aktuálně vytvářených objednávek** - velice užitečný údaj, který může orientačně napovědět, zda-li se v nejbližší době odešlou nějaké objednávky. Například v situaci, kdy najednou objednává větší množství lidí se může rozvozce rozhodnout nákup potravin[36] odložit na později. V opačném případě může na nákup vyjet hned anebo si dát třeba oběd.



Obrázek 9: Ukázka hlavní strany po přihlášení do administrace stránky

5.2.2 Správa objednávek

Nová objednávka, zaslána návštěvníkem, prochází pak určitým vývojem. Tento proces se odehrává v části navigace s názvem „Správa objednávek“, pokusím se ho nyní stručně popsat:

[36] V restauraci, kde jsem pracoval musel rozvozce téměř každý den zajet na nákup některých surovin.

- **Nová objednávka** - je to objednávka ve stavu, v jakém se ji zákazníkovi podařilo odeslat. V tuto chvíli má administrátorem proběhnout kontrola, kdy se může objednávce ještě měnit obsah. Jakmile jsou všechny úpravy hotové, tak se může změnit její stav na „potvrzená“[37].

- **Potvrzená objednávka** - v tuto chvíli už nelze provádět úpravy jídel nebo nápojů v objednávce. Můžeme objednávku dále označit jako zaplacenou, ale vzhledem k tomu, že se objednávka pravděpodobně bude rozvážet klientovi, tak se očekává, že si objednávku rezervuje rozvozce, který ji bude rozvážet (přes mobilní aplikaci).

- **Rezervovaná objednávka** - tento stav objednávka získává, jakmile si objednávku rezervuje rozvozce (to se děje v mobilní aplikaci, o které píšu v 5.3).

- **Zaplacená objednávka** - jakmile je objednávka zavezena a zaplacená, její stav se nyní může změnit na „zaplacená“ (stav lze změnit jak v administraci, tak mobilní aplikaci). Tímto se objednávka dostala do ideálního stavu.

- **Zaplacená objednávka - odevzdáno** - zaplacená objednávka ještě může získat příznak, že peníze, které za ni rozvozce získal, byly odevzdány.

- **Stornovaná objednávka** - čas od času se stane, že je potřeba objednávku stornovat, na tento stav se může objednávka změnit během kteréhokoliv zmíněného stavu výše[38].

5.2.3 Správa jídel

Správa jídel je část, kde se vytváří nabídka jídel a nápojů na stránce. Můžeme zde přidávat jídla, nápoje a vytvářet ingredience. Dále můžeme tyto produkty zařazovat do jednotlivých kategorií na stránce. Jídlům můžeme vyplňovat ingredience, z jakých jsou vytvořená. Nastavování jídel a nápojů funguje ve dvou režimech:

- **Omezený** - omezený režim nastává v případě, kdy stránka jede v produkčním módu (lze změnit v nastavení stránky). V tuto chvíli nemůžeme měnit ceny nebo jakékoliv jiné informace, ovšem můžeme měnit dostupnost jídel, nápojů či ingrediencí[39].

- **Bez omezení** - probíhá tehdy, kdy se přepneme na režim údržby. Můžeme vytvářet, upravovat jídla a nápoje.

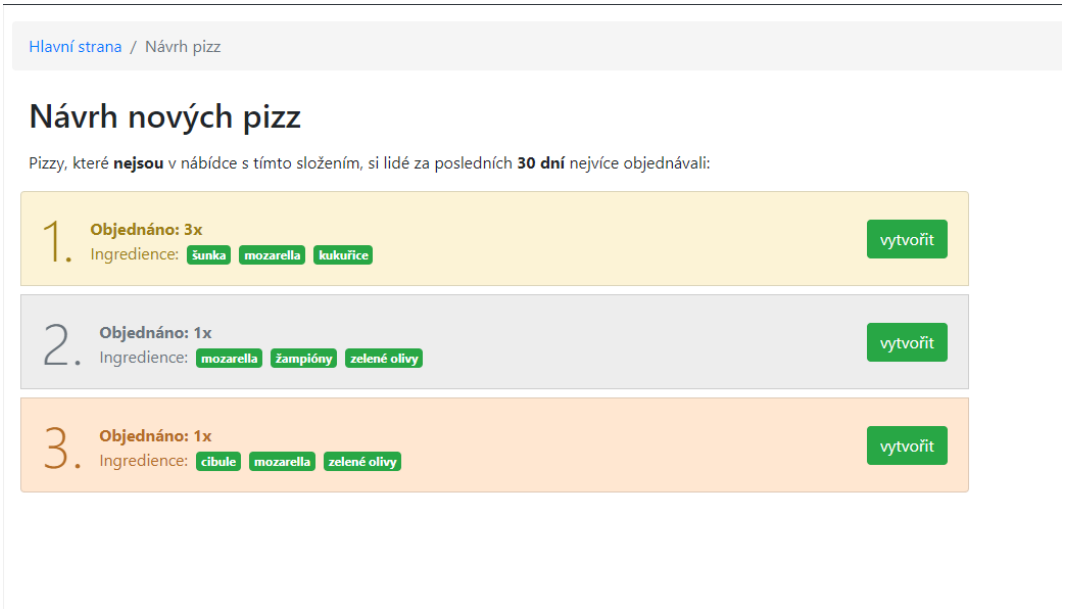
[37] Ještě před samotným potvrzením můžeme odeslat zákazníkovi email s rekapitulací objednávky a odhadovanou dobou doručení.

[38] Stornovanou objednávku už nelze obnovit.

[39] Pokud například změníme dostupnost nějaké ingredience, všechna jídla s obsahem této ingredience se také změní na nedostupná a nepůjde je objednat.

5.2.3.1 Návrh nových pizz

Tato funkcionalita byla vytvořena s cílem ukázat, jakým způsobem se dá využít získaná data ke zlepšení nabídky jídel v restauraci. Jako rozvozce jsem vypozoroval, že pizzerie má určitou nabídku pizz s přiřazenými ingrediencemi. Mnoho pizz se objednávalo sporadicky, jiné naopak často. Také se stávalo, že si lidé objednali pizzu, na kterou přidali určitou ingredienci navíc. Proto jsem se rozhodl implementovat funkcionalitu (viz obrázek 10), která zobrazí tři nejčastější kombinace pizzy, které byly za posledních 30 dní nejvíce objednávané, a které ještě nejsou v nabídce.



Obrázek 10: Ukázka návrhu nových pizz

5.2.4 Správa rozvozců

Ve správě rozvozců se nachází jednoduchý seznam uživatelů, kteří mají udělena práva rozvozce. Práva se udělují registrovaným uživatelům z klientské části aplikace.

5.2.5 Nastavení stránky

V nastavení stránky se provádí nejrůznější nastavení, které ovlivňují, do jisté míry, chování celé aplikace. Patří mezi ně obecná nastavení, kde lze například nastavit minimální cenu vytvářené objednávky, aby se nestalo, že přijde objednávka za 20 korun, která by se restauraci nevyplatila.

Dále je zde možné měnit otevírací dobu, která se vypisuje v klientské části. Poslední nejdůležitější možností z nastavení je režim stránky, který níže popíšu.

5.2.5.1 Změna režimu stránky

Aplikace funguje v následujících třech režimech:

- **Produkční režim** - produkční režim je ten, v němž je umožněno vytváření objednávek zákazníkům, avšak není možné vytvářet nebo upravovat nabídku jídel a nápojů. Může se však měnit jejich dostupnost.
- **Režim nepřijímání objednávek** - tento se od produkčního liší pouze v tom, že nebude možné odeslat objednávku v klientské části. Režim je vhodný ve chvíli, kdy dojde například k nějaké krátkodobé události, která znemožní rozvoz jídel, nebo v časech, kdy má restaurace zavřeno.
- **Režim údržby** - v tomto režimu je umožněno aktualizovat ceny jídel a nápojů nebo vytvářet novou nabídku. Tento režim zároveň zruší všechny rozpracované objednávky všem uživatelům[40], proto by se do tohoto režimu mělo přepínat co nejméně, ideálně po skončení otevírací doby.

5.3 Mobilní aplikace

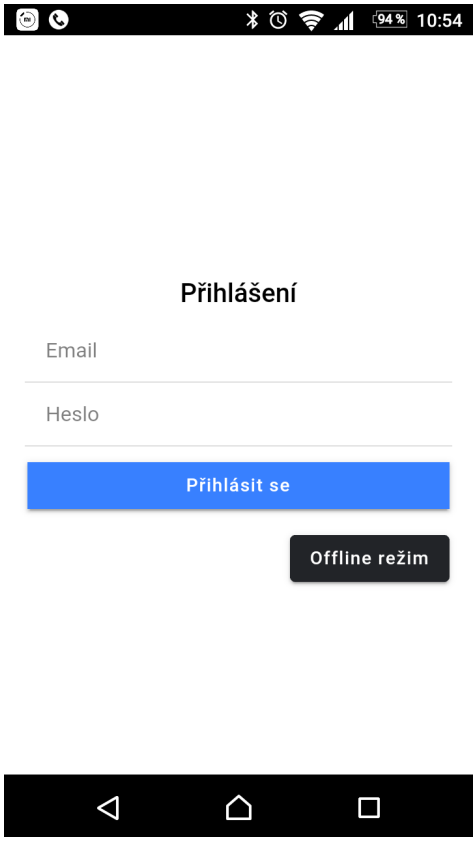
Mobilní aplikace má především usnadnit správu rozvezených objednávek. Je výhradně určená pro rozvozce.

5.3.1 Úvodní stránka

Na úvodní stránce (viz obrázek 11) je uživatel vyzván k přihlášení, případně se může přepnout do offline režimu.

Po přihlášení je uživatel přesměrován na stránku s potvrzenými objednávkami. Vespod se nachází navigace se čtyřmi tlačítky, kterými je možné přepínat mezi jednotlivými částmi aplikace.

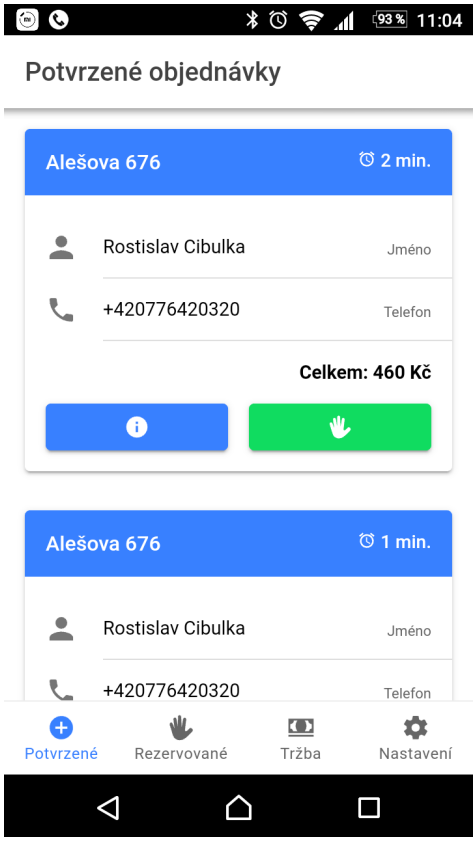
[40] Jejich status je nastaven na „zombie“, prozatím pro tyto objednávky v aplikaci není využití, avšak mohly by se využít při analýze.



Obrázek 11: Ukázka úvodní stránky aplikace

5.3.2 Potvrzené objednávky

Potvrzené objednávky (viz obrázek 12) jsou ty, které administrátor potvrdil. Je zde možné kliknout na tlačítko rezervace (ikona ruky), kdy se následně přihlášenému rozvozci objednávka přiřadí anebo ikona „info“ pro zobrazení detailu objednávky.



Obrázek 12: Ukázka seznamu potvrzených objednávek v aplikaci

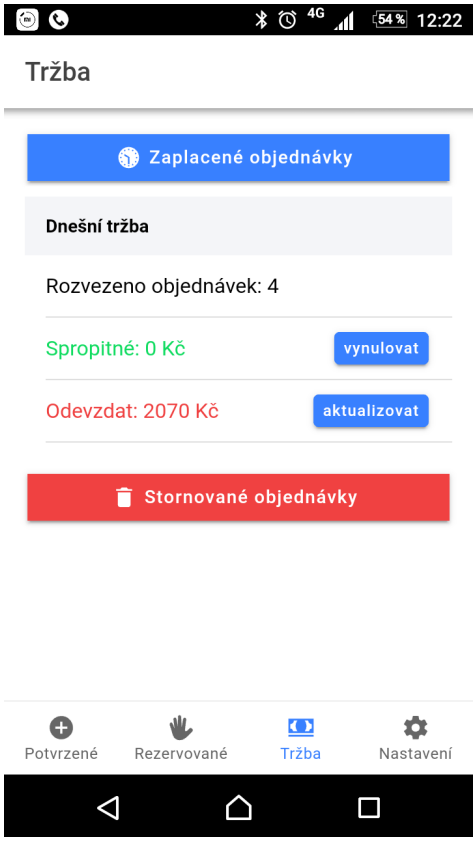
5.3.3 Rezervované objednávky

Část rezervovaných objednávek je takřka stejná jako „Potvrzené objednávky“, změna je především v tom, že místo ikony ruky pro rezervaci je zde ikona bankovky pro nastavení objednávky jako zaplacené. Po stisknutí tohoto tlačítka ještě může uživatel zadat spropitné, které případně získá od zákazníka.

5.3.4 Tržba

Tržba (viz obrázek 13) nabízí přehled o počtu rozvezených objednávek, informace, kolik bude muset na konci pracovního dne rozvozce odevzdat peněz, a také jak velké spropitné během dne získal. Mimo to může kliknout na tlačítko „Zaplacené objednávky“, kde se mu zobrazí seznam zaplacených objednávek, které rozvezl. Podobně si také může prohlédnout stornované objednávky[41] za posledních 12 hodin.

[41] Jsou zde pouze ty stornované objednávky, které byly v administraci potvrzené.



Obrázek 13: Ukázka stránky s názvem „Tržba“

5.3.5 Nastavení

V nastavení je možné uživatele odhlásit a přejít tak na úvodní stránku anebo přepínat mezi online/offline režimem.

5.3.6 Offline režim

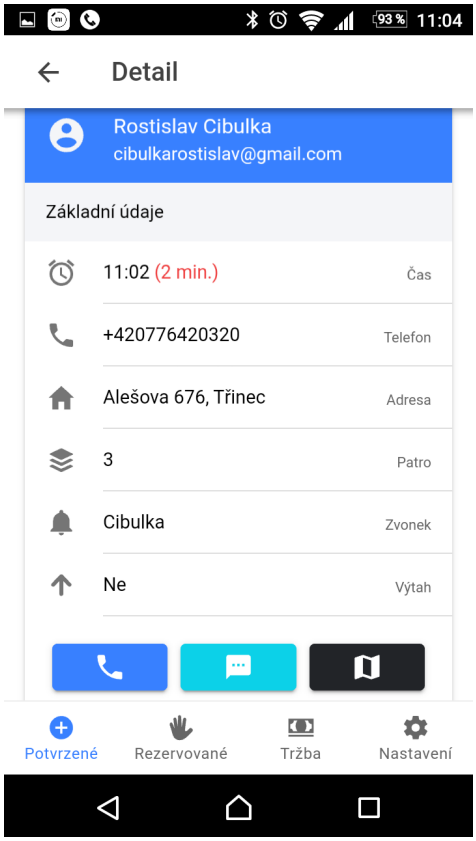
Do offline režimu se aplikace přepíná v nastavení, o kterém se píše výše, anebo se do něj aplikace přepne automaticky v momentě, kdy dojde ke ztrátě připojení k internetu. V tomto režimu se uživateli zobrazují na jednotlivých stránkách objednávky, které jsou uložené v paměti telefonu. Není možné však objednávkám měnit jejich stav (provádět rezervaci, nastavit jako zaplacenou apod.).

5.3.7 Detail objednávky

Detail objednávky (viz obrázek 14) obsahuje bližší informace o objednávce (informace o zákazníkovi, objednaných jídel a nápojů), ale navíc také tři tlačítka:

- **Tlačítko s ikonou telefonu** - pomocí tohoto tlačítka se spustí aplikace pro vytáčení daného telefonního čísla.

- **Tlačítko s ikonou zprávy** - pomocí tohoto tlačítka je možné zaslat SMS zprávu, jíž je zákazník upozorněn, že je rozvozce na cestě a objednávka bude v nejbližších minutách přivezena.
- **Tlačítko s ikonou mapy** - toto tlačítko spustí aplikaci navigace, kde se přednastaví adresa z objednávky[42].



Obrázek 14: Ukázka detailu objednávky

[42]Navigovat na adresu z objednávky bude možné za předpokladu, že zákazník vyplnil adresu, které bude navigace rozumět.

6 Rozšíření aplikace

Myslím si, že na tomto projektu se nabízí mnoho možností, jak by se dalo některé věci vylepšit, jedná se například o:

- **Párování jídel s přílohou** - jistě by bylo užitečné nabídnout uživateli možnost spárovat například smažený sýr s hranolkama.
- **Modul pro vytváření akcí** - ze získaných dat se také nabízí vytvořit funkcionalitu, díky které by se daly vytvářet nějaké akce (2+1, nápoj zdarma a podobně).
- **Vylepšit uživatelské rozhraní v klientské části** - osobně se nepovažuji za odborníka přes uživatelské rozhraní a návrh designu stránek není oblast, které bych se v budoucnosti chtěl více věnovat, proto si myslím, že v případě reálného použití by stálo za to zaplatit odborníka, který by poukázal na nedostatky, jež by mohly mít vliv na odchod návštěvníka, který by z důvodu nepochopení uživatelského rozhraní stránku opustil.
- **Více funkcionality v mobilní aplikaci** - u mobilní aplikace by se určitě nabízela možnost úpravy jídel a nápojů v objednávkách a provádět tak potvrzování objednávek přímo z telefonu, nikoliv přes administraci na webu. Dále by bylo užitečné, aby aplikace notifikovala rozvozce v momentě vzniku nové objednávky.
- **Jazykové mutace** - jazykové mutace, především tedy překlad do anglického jazyka, by určitě posunulo projekt o kousek dál.

Závěr

Výsledkem této práce je poměrně komplexní systém pro vytváření objednávek jídel se zaměřením na rozvoz pizzy. Při vývoji jsem kladl velký důraz na to, aby se aplikace dala v budoucnu dále rozšiřovat. Z mého pohledu aplikace obsahuje potřebnou funkcionalitu, aby mohla být použitá v reálném provozu bez větších zásahů.

Myslím si, že mě práce posunula o velký kus dál, jelikož se jedná o první větší projekt, který jsem realizoval, a který by mohl mít reálné využití. Musel jsem se seznámit se spoustou nových technologií, strávil jsem mnoho hodin komunikace na různých fórech a čtením dokumentace. Jsem přesvědčen, že zkušenosti, které jsem tímto získal mi určitě pomohou při řešení dalších výzev.

Conclusions

The practical outcome of the thesis is a complex pizza delivery management system, although the system as such is not restricted to the pizza business at all. I have been very meticulous in ensuring the software's extensibility and I am convinced that it is therefore production ready.

The thesis has proven to be a source of much personal growth, as it is my first major project with a potential to win a real user community. I have explored many technologies, spent countless hours reading technical documentation, as well as connecting with other developers on various websites. The experience gained hereby will become invaluable in taking on new challenges in the future.

A Instalace a spuštění serveru

Pro testovací účely byla aplikace umístěna na adresu:
https://rozvoz.rostislavcibulka.cz.

A.1 Požadavky

Serverová část požaduje pro spuštění:

- Apache server (vyvíjeno na verzi 2.4.37),
- MySQL databázi (vyvíjeno na MariaDB v. 10.1.37),
- PHP 7.1 a vyšší (vyzkoušeno také na verzi 7.2 a 7.3).

Pro spuštění serveru lokálně je nutné splnit požadavky webového prostředí pro Nette 2.4 (viz https://doc.nette.org/cs/2.4/requirements). Aplikace byla vyvíjena za použití LAMPP balíčku, v mém případě XAMPP verze 7.1.25[43].

A.2 Instalace

Instalace se skládá z těchto kroků:

- vytvořit novou databázi,
- importovat `src/server/install/install.sql` soubor nacházející se v přiloženém CD,
- volitelně možno importovat soubor `data.sql` ze stejného adresáře (obsahuje ingredience, jídla, nápoje apod.),
- do webového adresáře překopírovat soubory ze složky `src/server/` (složku `install/` je možno vynechat),
- ve webovém adresáři následně upravit přístup k databázi v souboru `app/config/config.local.neon`.

Rozšíření ApiRouter pro nette, pomocí kterého je vytvořeno REST API, vyžaduje pro správné fungování v lokálním prostředí nastavit Virtual Host na Apache serveru, který bude směřovat do webového adresáře projektu, v opačném případě nebude API fungovat[44] (klientská část a administrace však bude plně funkční).

A.3 Spuštění

Klientská část a administrace by poté měla být přístupna v prohlížeči na adrese **`localhost/webovy-adresar-projektu`**.

[43] Dostupný z https://www.apachefriends.org/download.html.

[44] O tomto problému se diskutovalo zde https://github.com/contributte/api-router/issues/12.

B Instalace a spuštění aplikace

Pro instalaci a spuštění aplikace na OS Android je potřeba do telefonu přesunout soubor `app/pizza45.apk`[45] a nainstalovat (pravděpodobně bude potřeba povolit instalaci z neznámých zdrojů).

Server, na který aplikace odesílá požadavky, je nastaven na adresu https://rozvoz.rostislavcibulka.cz. Pokud by bylo nutné adresu změnit, je potřeba:

- upravit ve složce `src/aplikace/src/app/services/` soubory **`orders.service.ts`** a **`authorization.service.ts`**, u obou se nachází proměnná baseUrl, kde je možné změnit adresu, na kterou se budou následně odesílat požadavky,
- nainstalovat Node.js server[46] (při vývoji byla použita verze 10.15.0),
- mít nainstalovaný správce balíčků npm (součást instalace Node.js),
- nainstalovat Ionic CLI pomocí příkazu `npm install -g ionic`,
- překopírovat obsah ze složky `src/aplikace/` na lokální úložiště a v lokálním adresáři provést příkaz `npm install` (nainstalují se potřebné závislosti),
- následně lze pomocí příkazu `ionic serve` spustit aplikaci na lokálním serveru v prohlížeči, ovšem nebude zde fungovat funkcionalita spojená s nativními funkcemi telefonu (odesílání SMS apod.).

Aby bylo možné vytvořit instalační soubor pro OS Android, je potřeba splnit požadavky uvedené zde https://ionicframework.com/docs/installation/android, poté stačí zadat příkaz `ionic cordova build android`, který vytvoří .apk soubor, jehož umístění by mělo být vypsané po skončení příkazu v příkazové řádce.

Vzhledem k tomu, že Ionic Framework má za cíl vytvářet cross-platform aplikace, neměl by být problém udělat build aplikace také pro iOS[47], avšak na tuto platformu testování neproběhlo, a tak není možné zaručit bezproblémový chod.

[45] Případně stáhnout na adrese https://rozvoz.rostislavcibulka.cz/pizza45.apk.
[46] Dostupný na adrese https://nodejs.org/en/download/.
[47] Požadavky zde https://ionicframework.com/docs/installation/ios.

C Obsah přiloženého CD/DVD

app/
: Složka obsahuje instalační soubor mobilní aplikace pro OS Android.

src/
: V této složce se nachází veškeré zdrojové kódy pro zprovoznění webové prezentace a mobilní aplikace.

doc/
: Obsahuje text práce ve formátu PDF a soubory nutné pro jeho vygenerování.

readme.txt
: Soubor obsahuje informace pro zprovoznění serveru, aplikace a také přístupové údaje do administrace.

Literatura

[1] **Webová stránka w3schools.com:** PHP 7 Introduction
Dostupné z https://www.w3schools.com/php7/php7_intro.asp

[2] **Webová stránka w3schools.com:** PHP: MySQL database
Dostupné z https://www.w3schools.com/php/php_mysql_intro.asp

[3] **Webová stránka developer.mozilla.org:** HTML | MDN [online]. [cit. 2019-04-12].
Dostupné z https://developer.mozilla.org/en-US/docs/Web/HTML

[4] **Webová stránka developer.mozilla.org:** JavaScript | MDN [online]. [cit. 2019-04-12].
Dostupné z https://developer.mozilla.org/en-US/docs/Web/JavaScript

[5] **Webová stránka w3schools.com:** CSS Introduction
Dostupné z https://www.w3schools.com/css/css_intro.asp

[6] **Webová stránka nette.org:** Nette | MVC aplikace & presentery [online]. [cit. 2019-04-12].
Dostupné z https://doc.nette.org/cs/2.4/presenters

[7] **Webová stránka cordova.apache.org:** Apache Cordova | Architectural overview of Cordova platform - Apache Cordova [online]. [cit. 2019-04-12].
Dostupné z https://cordova.apache.org/docs/en/9.x/guide/overview/index.html

[8] **Webová stránka ionicframework.com:** Ionic | What is Ionic Framework [online]. [cit. 2019-04-12].
Dostupné z https://ionicframework.com/docs/intro

[9] **Webová stránka ionicframework.com:** Ionic Framework | Ionic Documentation [online].
Dostupné z https://ionicframework.com/docs

[10] **Webová stránka nette.org:** Nette | Routování URL [online]. [cit. 2019-04-21].
Dostupné z https://doc.nette.org/cs/2.4/routing

[11] **Webová stránka nette.org:** Nette | Komponenty & ovládací prvky [online]. [cit. 2019-04-21].
Dostupné z https://doc.nette.org/cs/2.4/access-control

[12] **Webová stránka nette.org:** Nette | Přihlášené a oprávnění uživatelů [online]. [cit. 2019-04-21].
Dostupné z https://doc.nette.org/cs/2.4/routing

[13] **Webová stránka blog.florimondmanca.com:** RESTful API Design: 13 Best Practices to Make Your Users Happy [online].
Dostupné z https://doc.nette.org/cs/2.4/routing

[14] **Webová stránka devdavtic.com:** Protecting Your App With Ionic Auth Guards [online].
Dostupné z https://devdactic.com/ionic-auth-guards/

[15] **Webová stránka itnetwork.cz:** Základy Nette frameworku [online].
Dostupné z https://www.itnetwork.cz/php/nette/zaklady

[16] **Webová stránka angular.io:** Angular [online].
Dostupné z https://angular.io/docs

[17] **Webová stránka nodejs.org:** Node.js [online].
Dostupné z https://nodejs.org/

[18] **Webová stránka php.net:** PHP:Hypertext Preprocessor [online].
Dostupné z https://php.net/