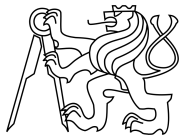
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ



ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Modulární rezervační systém pro restaurační zařízení
Student:	Pavel Beran
Vedoucí:	Ing. Petra Pavlíčková, Ph.D.
Studijní program:	Informatika
Studijní obor:	Web a multimédia
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2017/18

Pokyny pro vypracování

1) Seznamte se s aktuálními možnostmi rezervací v restauračních zařízení a zanalyzujte jejich výhody a nevýhody.
2) Zanalyzujte možnosti generování univerzálně použitelného HTML kódu.
3) Na tomto základě navrhněte, implementujte a následně otestujte nový rezervační systém, který bude umožňovat všechny potřebné procesy pro vytváření a správu rezervací restauračních zařízení (registrace zařízení, vygenerování HTML formuláře na stránky, API pro zpracování rezervací z formulářů, administrační rozhraní pro jednotlivá restaurační zařízení, e-mailové notifikace, ...)
4) Implementaci zhodnoťte a navrhněte další doporučení/vývoj.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 24. ledna 2017

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Bakalářská práce

Modulární rezervační systém pro restaurační zařízení

Pavel Beran

Vedoucí práce: Ing. Petra Pavlíčková, Ph.D.

10. května 2017

Poděkování

Rád bych poděkoval vedoucí práce Ing. Petře Pavlíčkové, Ph.D. za odpovědi na mé dotazy ohledně práce. Zároveň bych chtěl poděkovat rodině, přítelkyni a přátelům za mentální podporu během studia, především v jeho závěru.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 10. května 2017 .....

Odkaz na tuto práci

Beran, Pavel. *Modulární rezervační systém pro restaurační zařízení.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2017.

Abstrakt

Tato práce se zabývá vývojem nového rezervačního systému pro restaurační zařízení. Hlavní myšlenkou systému je vygenerování rezervančího formuláře, jehož vizuální podobu si bude moci restaurace sama nastavit a zároveň nasazení formuláře na stránky bude jednoduché a celý proces časově nenáročný. Za účelem vývoje byly prozkoumány a zhodnoceny aktuálně dostupná řešení a možnosti vytvoření univerzálního kódu, který, vložen na stránky, nenaruší strukturu a zároveň graficky zapadne. Výstupem práce je samotná otestovaná aplikace splňující uvedené požadavky a doporučení pro její další vývoj.

Klíčová slova rezervační systém, webová aplikace, generování kódu, php, material design

Abstract

The aim of this thesis is the creation of a new reservation system for restaurants. The new system should generate reservation form, which visual can be set by restaurant. The whole process should be fast and the insertion of the form in the webpages should be easily done, while the form will not destroy the document structure and will fit graphically there. The output of the thesis

is the application that fulfills the above requirements and recommendations for its further development.

Keywords reservation system, web application, code generating, php, material design

Obsah

Seznam obrázků

Úvod

Internet se za posledních deset let rozšířil mezi širokou veřejnost. Roku 2005 používala internet méně než třetina české populace starší 16ti let. V roce 2015 již počet uživatelů přerostl tři čtvrtiny a nyní v roce 2017, podle posledních dostupných údajů, je v České republice pouze 1,5 milionu lidí, kteří nikdy internet nepoužili. Zde tedy mluvíme o méně než 15procentní menšině. [1]

V současné době, kdy je, podle výše uvedených dat, internet stále více využíván, je pro restaurační zařízení kvalitní webová prezentace prakticky nutnou součástí podnikání. Část populace již jinde než na internetu stravovací zařízení nehledá. Přesto spousta restaurací stále výhody dostupné technologie přehlíží. Především mluvíme-li o možnostech on-line rezervací. Spousta strávníků dá přednost jednoduchému vyplnění formuláře na stránce restaurace, oproti nutnosti psát e-mail, v horším případě muset do restaurace přímo telefonovat. Obě varianty jsou pro mnoho lidí zdlouhavé a složité. Při telefonování bývá linka obsazena, nebo obsluha telefon nebere a zároveň hovor stojí peníze. U e-mailu je zase člověk nucen sepsat alespoň částečně formální text, což může být pro některé lidi obtížné. Na potvrzení je pak někdy potenciální strávník nucen čekat až do druhého dne, a dlouho tedy neví, jestli rezervace opravdu úspěšně proběhne. Oproti tomu vyplnění rezervačního formuláře umožňuje uživateli jednoduchou cestou, pomocí vyplnění pár polí, odeslat svou rezervaci, na kterou může obratem dostat potvrzení, v horším případě zamítnutí požadované rezervace.

Důvod proč stále velká část restaurací nenabízí zákazníkům možnost rezervace přímo na svých stránkách, ale nemusí být pouze v tom, že by snad provozovatelé byli laxní v přístupu k novým možnostem, jak přitáhnout zákazníky, ale spíše v tom, že aktuálně dostupná řešení jsou často nevhodná. Z tohoto důvodu jsem se rozhodl na problematiku zaměřit a přijít s novým řešením, které umožní provozovatelům restauračních zařízení, zařídit si na své stránky jednoduše a rychle on-line rezervační formulář, aniž by museli složitě předělávat samotné stránky.

Kapitola 1

Cíle a metodika

1.1 Cíle

Hlavním cílem práce je vytvoření nového rezervačního systému, který bude provozovatelům restaurace umožňovat jednoduše a rychle získat rezervační formulář, který si budou moci, bez znalosti HTML, zprovoznit na svých webových stránkách a zároveň jim umožňovat správu rezervací takto vytvořených.

K dosažení tohoto hlavního cíle bude potřeba splnit několik průběžných cílů. Prvním z nich je prozkoumání aktuálně dostupných řešení, analyzovat jejich jednotlivé plusy a minusy z pohledu provozovatele restauračního zařízení a na základě těchto dat určit klíčové funkcionality a jejich provedení.

Dalším cílem je zaměřit se na možnosti generování HTML kódu (včetně kaskádových stylů a JavaScriptu), který je možné vložit na stránky co nejjednodušeji. Zanalyzovat tedy, jak je možné kód do stránky zapsat a vybrat nejlepší možný způsob pro to, aby byla kód na stránky schopná vložit i osoba bez znalosti dané technologie a zároveň aby daný kus stránky vypadal tak, že je součástí celku a vizuálně zapadal.

Třetím dílčím cílem je vytvořit analýzu nového systému. Vytvořit případy užití, vybrat vhodné technologie a navrhnout celý systém. Tento cíl je pak blízce spjat s posledním cílem, jehož obsahem je samotná implementace systému a jeho následné otestování a ověření funkčnosti.

1.2 Metodika

Metodika bakalářské práce jde ruku v ruce s cíli popsanými v předchozí sekci. Jde tedy o analýzu již existujících řešení a důkladný rozbor jednotlivých kladů a záporů. Dalším základním kamenem metodiky je studium a zpracování dostupných zdrojů, které se zabývají danou problematikou, v tomto případě především HTML a CSS a jejich implementace do již existujících dokumentů. Na základě těchto analýz budou vytvořeny případy užití nově vznikající aplikace a navržena architektura, na základě čehož potom implementací jednot-

livých případů užití vznikne výsledný produkt - nový rezervační systém. Ten bude posléze testován a v závěru budou sepsány doporučení pro další vývoj a případné úpravy aplikace.

Kapitola 2

Rešeršní část

2.1 Dostupná řešení

V této kapitole se budu zabývat rezervačními systémy určenými přímo pro, nebo vhodnými pro restaurační zařízení. Budu se zabývat pouze českými systémy, jelikož zahraniční řešení jsou téměř výhradně pro českého restauratéra nedostupná, ať už jazykově, nebo finančně. Je tedy téměř jisté, že když bude restauratér hledat rezervační systém, zvolí jeden z tuzemských. I přes to má aktuálně provozovatel poměrně širokou nabídku ze které může vybírat. Ne všechny jsou ale vhodné pro restaurační zařízení. Na trhu je spoustu rezervačních systémů, které mimo jiné tvrdí, že jsou využitelná i pro rezervace míst v restauraci, avšak ve skutečnosti je řešení téměř nepoužitelné. Stejně tak se vyhnu řešením, které nestojí za zmínku ať už z toho důvodu, že jsou neúspěšná a nepoužívaná a nebo z toho, že jejich řešení je nevhodné a spíše na škodu než k užitku. Budu zde tedy řešit pouze čtyři systémy, které opravdu stojí za zmínku, jelikož jsou buď pro restaurační zařízení přímo určené, anebo jsou tak použitelné.

2.1.1 Restu.cz

Restu.cz je aktuálně v České republice nejznámějším a nejpoužívanějším katalogem restaurací vůbec. Ke dni 21.3.2017 je v katalogu registrováno 17 136 restaurací, z nichž více než polovina používá i zde dostupný rezervační systém. [2]

Strávníkovi Restu.cz nabízí příjemné webové rozhraní, kde může uživatel vyhledávat restaurace na základě lokality, recenzí a mnoho dalšího. Restu samotné i rozdává ceny, kterými hodnotí restaurace. Zákazník si tak může jednoduše například vyfiltrovat restaurace, které mají nejlepší hamburgery a podobně. Zákazník si pak u restaurací, které si službu platí, může rovnou vytvořit rezervaci.

Samotné řešení pro provozovatele restaurací je také poměrně kvalitní. Zde

jsou nabízeny tři různé tarify. V prvním tarifu nazvaném Start je umožněno restauraci založit si na restu profil a na něm publikovat menu a případně akce a slevy. Tento tarif stojí restauraci 3 000 Kč ročně. Druhý, Gold tarif je prakticky vše, co provozovatel potřebuje. Restaurace tak získá přístup k možnostem rezervací, statistikám, exportu kontaktů, hodnocení od zákazníků a dalším službám. Hlavním problémem tohoto tarifu je pak cena. Paušálně platí restaurace 4 980 Kč za rok, tedy 415 Kč měsíčně, zároveň ale platí 25 Kč za jednoho usazeného hosta přes stránky Restu.cz (5 Kč mezi 11:00 a 15:00), každá rezervace provedená přes formulář na stránkách restaurace, je zároveň zpoplatněna 3 Kč. Posledním tarifem je tarif Platinum za 9 900 Kč ročně. Nabízí stejné služby jako tarif Gold, pouze s tím, že je lehce snížená cena za usazené hosty a provedené rezervace a zároveň má provozovatel přístup k elektronické rezervační knize. Aplikaci na počítač nebo tablet, kde může provozovatel spravovat rezervace. [3]

Rezervační formuláře jsou poměrně uživatelsky přívětivě navržené. Jediným problémem je nasazení na stránky. Provozovatel restaurace dostane od Restu.cz tlačítko, které zákazníkovi otevře novou záložku v prohlížeči, kde se načte rezervační formulář na doméně restu.cz. Provozovatel si sice může nastavit obrázek na pozadí pod formulář na této stránce a zákazník si tak teoreticky nemusí uvědomit, že již není na stránkách restauračního zařízení. Stále je však klientsky přívětivější zůstat přímo na původní stránce. Co je dalším problémem u nasazení, je to, že tlačítko je potřeba někam na stránky umístit, k čemuž je potřeba alespoň základní znalost HTML. Tlačítko má ještě k tomu pouze jeden design, tedy zelené tlačítko v barvě Restu.cz se zaoblenými rohy a občas tak do stránek vizuálně nezapadá. Restu.cz samozřejmě nabízí služby, kdy za úplatu pomůže s nasazením, případně i nabízí tři služby výroby webu. Základní zadarmo s poplatkem 550 Kč měsíčně, kdy restaurace dostane jednu jednostránkovou šablonu a v CMS si může změnit barvy a přidat logo. Za 19 500 Kč + 550 Kč měsíčně pak restaurace dostane na výběr z pěti šablony a možnosti vytvořit více stránek. Jako poslední za 34 900 Kč + 550 Kč měsíčně Restu.cz vyrobí web na míru. V takovýchto případech je samozřejmě jednoduché nasadit rezervační formulář i pro osobu bez znalosti HTML, opět ale značně stoupá cena za provoz. Zároveň velká část restaurací je se svojí webovou prezentací spokojená a pouze hledá rezervační systém. Zde je pak Restu.cz vhodné pouze částečně.

2.1.2 RezervujStůl

Rezervační systém RezervujStůl je poměrně komplexní rezervační řešení. Velkou výhodou pro zákazníky restauračního zařízení, využívajícího tohoto systému, je fakt, že zákazník si dělá rezervaci přímo k určitému stolu, který vidí na interaktivní mapě restaurace. S tímto je ovšem spojena velká nevýhoda pro restaurační zařízení. Ačkoliv provozovatel do instalačního poplatku za 3 000 Kč zahrnuje i grafické přizpůsobení webovým stránkám klienta, jedná

se spíše o barevné úpravy jednotlivých komponent mapy. Po grafické stránce je tedy řešení poměrně zaostalé, a je tedy téměř nemožné kombinovat ho s moderním designem webových stránek. Provozovatel sice v rámci úprav umožňuje i odebrání výběru stolů, čímž se grafika trochu vylepší, ale zbytek formuláře stále zůstává graficky několik let pozadu. Rozhraní zároveň není responzivní, a tak je velice obtížné provést rezervaci na mobilním zařízení s menší obrazovkou, protože jednotlivá tlačítka jsou moc malá na to, aby se do nich mohl klient jednoduše trefit prstem. Na některých stránkách je provedení rezervace na mobilním zařízení prakticky nemožné, jelikož formulář zmizí a je ho vidět například pouze horních 10%. Na první pohled působí řešení i trochu chaoticky, a klient tak může mít v prvním okamžiku problémy zorientovat se a uvědomit si kam je potřeba kliknout, aby rezervaci založil.

Cenově se RezervujStůl pohybuje v podobných hladinách jako výše uváděné Restu.cz. Na rozdíl od Restu.cz je však zákazníkovi ještě účtován poplatek za instalaci a cenové rozdíly jsou založené hlavně na tom, kolik stolů a místností má restaurace. Instalace je pro restauraci jednoduchá, za povinný poplatek se o ni postará provozovatel rezervačního systému. Z toho důvodu ale zprovoznění trvá delší dobu, jelikož restauratér je závislý na jiné osobě, která musí provést potřebné úpravy a výsledný systém zprovoznit. [4]

2.1.3 BookioPro

BookioPro je rezervační systém pro restaurační zařízení, který založil spolumajitel společnosti Creative web - Petr Paška. V roce 2004 založil portál obedovat.sk, který se následně spojil s českým lunchtime.cz. Oba portály byly následně koupeny indickou společností Zomato za 65 milionů korun. Ještě před tím založil Petr Paška rezervační systém bookio.cz, z něj následně vzniklo BookioPro. Tento rezervační systém aktuálně využívá například i restaurace Jamies's Italian, jejímž majitelem je mediálně známý kuchař Jamie Oliver. [5]

BookioPro aktuálně využívá přes 126 restaurací. V nabídce služby jsou tři základní tarify. Basic, který uživateli umožňuje, za 49 euro měsíčně, přijímat rezervace, ty spravovat v elektronické knize rezervací, automatické potvrzení rezervací a sms potvrzení. U sms potvrzení je každá sms zpoplatněna. Tarif Standard za 59 euro navíc provozovateli dává přístup k seznamu hostů, hodnocení od hostů a ke statistikám. Tyto hodnocení jsou na rozdíl od Restu.cz dostupné vždy pouze provozovateli restaurace a mohou tak být využity ke zlepšení služeb. Nejdražší tarif Professional za 89 euro měsíčně ještě přidává do elektronické knihy mapu stolů na které obsluha vidí živě obsazenost stolů a zároveň čekací listinu, ta umožňuje obsluze přiřadit zákazníky na obsazený stůl jako čekající a posléze je případně informovat o tom, že stůl se uvolnil. [6]

Implementaci rezervačního formuláře v tomto případě ve většině případů provádí provozovatel rezervačního systému. Restauratér je povinen do deseti dní od registrace poskytnout BookioPro přístup ke svým stránkám a součinnost v nasazení formuláře. Samotný formulář je na stránce vložen jako iframe,

který na dané místo vygeneruje formulář, jako jeden z atributů url v iframe je vložen odkaz na css soubor uložený na serveru restaurace, kde jsou definovány styly formuláře. Výsledný formulář tedy vhodně graficky zapadá do vzhledu stránky. BookioPro je obecně vzato z pohledu restaurací dražší varianta Restu.cz. Funkcionálně umožňuje téměř totožné věci, pouze asi za tří násobně vyšší cenu. Velikou výhodou je ovšem výsledný vizuální dojem z formuláře přímo na stránce restauračního zařízení. Vzhledem k poměrně vysoké ceně je ale řešení dostupné spíše pro úspěšnější a luxusnější restaurace.

2.1.4 Reservanto

Reservanto není rezervační systém určený čistě pro restaurační zařízení. Jedná se o velmi komplexní rezervační systém, umožňující klientovi vybrat si zaměření jeho podnikání a na základě toho vytvořit rezervační formulář. Administrační rozhraní pro zprávu rezervací je poměrně složité a zprovoznit si rezervační formulář na stránkách tedy není otázka pár kliknutí. Výhodou je však fakt, že pokud se restauratér spokojí s naprostým minimem, kdy nemá e-mailové potvrzení, nemůže si jakkoliv upravit rezervační formulář, nemůže upravit styl formuláře a opravdu jediné co může, je spravovat klientské rezervace, pak nemusí zaplatit ani korunu. Základní služba je totiž zdarma. Poměrně užitečná začíná být služba až od třetího tarifu, tedy tarifu Premium, za který restaurace zaplatí 599 Kč měsíčně. Tento tarif již umožňuje odesílat zákazníkům e-mailové notifikace, nastavovat si vlastní pole ve formuláři, ale především umožňuje alespoň částečně měnit styl formuláře pomocí změny barev. [7]

Nasazení na stránky restaurace musí v tomto případě provést přímo provozovatel dané restaurace nebo správce stránek. Reservanto pouze vygeneruje kód, který pak má být na stránky vložen. Jedná se o tlačítko, které vyvolá formulář. U tohoto tlačítka je možné vybrat jeden z šesti základních stylů, tedy kombinaci barvy tlačítka a textu. Dále je možné nastavit text tlačítka. Další případné grafické úpravy tlačítka si musí provést restauratér sám pomocí svých stylů. U formuláře je možné nastavit barvu záhlaví, barvu textu v záhlaví a barvu potvrzovacího tlačítka. Zde je již na výběr celá barevná paleta. Zároveň je možné si do formuláře nahrát vlastní logo a případně i zakázat zobrazování loga systému Reservanto.

Tento rezervační systém obecně není příliš vhodný pro restaurační zařízení. Řešení není intuitivní ani pro restauratéra, ani pro jeho zákazníka. Restaurace má zároveň velmi omezené možnosti nastavení grafických vlastností prvků, které je nucena si na stránky vložit. V neposlední řadě samotné nasazení vyžaduje určitou znalost HTML, jelikož je potřeba tlačítko umístit na správnou pozici ve stránce, což velké části restaurací pouze zvýší náklady na zprovoznění tohoto systému.

2.2 Generování HTML

2.2.1 HTML

HTML - HyperText Markup Language vznikl v roce 1990 a první dokument poté v roce 1991. Důvodem vzniku byla potřeba překonat problém s nelineárností textu na síti a tedy s potřebou odkazovat mezi jednotlivými textovými dokumenty. Potřeba vznikla při vývoji systému ENQUIRE v laboratořích organizace CERN, který se stal předchůdcem systému světové sítě World Wide Web, tak jak ji známe dnes. Jedná se o značkovací jazyk, který pomocí takzvaných elementů definuje základní vzhled a strukturu dokumentu tak, aby webový prohlížeč byl schopný dokument správně vykreslit. První verze HTML měla ovšem pouze základních 18 elementů. [8]

Nejnovější, již široce využívanou verzí, je HTML5 (nejnovější verzí je 5.1, jedná se však o technologii zatím s malou podporou ze strany prohlížečů a prvky se tedy ještě moc nevyužívají). Tato verze opět zjednodušuje tvorbu dokumentu pomocí nových elementů, které ještě více podporují práci s multimédii, formuláři a dalšími technologiemi, z kterých se začínají postupně stávat webové standardy. HTML5 zároveň zjednodušuje práci s grafikou a optimalizací stránek, například pomocí možnosti nastavit pro různá rozlišení obrazovky různé zdroje, z kterých se stahuje obrázek v dokumentu. [9]

Samotné HTML je však stále pouze prostředek k definování základní struktury dokumentu a úplně základního stylu. Proto je HTML využíváno v kombinaci s dalšími technologiemi, především CSS a Javascriptem. Tyto tři jazyky ve spolupráci vytváří vizuální a funkcionální podobu webových stránek tak, jak je známe.

2.2.2 CSS

CSS - Cascading Style Sheets, v češtině Kaskádové Styly, mají kořeny, stejně jako HTML v organizaci CERN a to v roce 1994. Hlavním důvodem diskuzí na toto téma byla potřeba stylování HTML dokumentů. Dokument byl v té době jednoduše černý text. Nešla změnit ani barva a font. Fakt, že potřeba umožnit stylovat dokument byla opravdu velká, dokazuje například e-mail zaslaný do www-list, jehož autorem je Marc Anderseen, budoucí spoluzakladatel Netscape.

„*In fact, it has been a constant source of delight for me over the past year to get to continually tell hordes (literally) of people who want to – strap yourselves in, here it comes – control what their documents look like in ways that would be trivial in TeX, Microsoft Word, and every other common text processing environment: "Sorry, you're screwed."*“ [10]

V uvedené komunikaci je vidět, že lidé, zvyklí na možnost jednoduše změnit font například z TeXu nebo programu Microsoft Word očekávali, že stejnou možnost budou mít i v dokumentu napsaném v HTML. Ten ovšem danou

funkcionalitu neumožňoval. Ještě téhož roku tedy vznikl dokument, který navrhoval a nastiňoval využití CSS. V roce 2005 na World Wide Web konferenci byly již prezentovány, na ukázku, dokumenty nastylované pomocí CSS. Prezentovány byly v prohlížeči Arena, který byl upraven tak, aby podporoval využití kaskádových stylů. Prezentace zažehla diskuze a posléze se začala připravovat specifikace. Roku 1995 se začalo naplno rozvíjet konsorcium World Wide Web Consortium, pro mnohé nejspíše známé spíše pod zkratkou W3C. To se začalo o CSS velmi zajímat a podporovat jeho rozvoj. Na konci roku 1996 bylo CSS vyhlášeno, jako technologie doporučená W3C. Společně s tím začaly CSS podporovat prohlížeče Internet Explorer a Netscape. CSS se začalo ve velkém využívat a již roku 1998 byla W3C doporučená technologie CSS2. [11]

Aktuálně jsou již používány moduly z CSS3 a stále se dodělávají nové. Pomocí CSS je autor dokumentu schopný jednotlivým elementům přiřadit určité styly. Ať už přímo elementu, nebo vytvořit třídu, té nastavit jednotlivé hodnoty a tu pak přiřadit patřičným elementům. Velkou výhodou je také to, že definice třídy je uchovávána na jednom místě a tím pádem také z jednoho místa přenastavitelná Autor tedy není nucen procházet dokumenty a měnit nastavení například všech nadpisů, ale pouze změnit styl třídy, která nadpisy definuje. [8]

2.2.3 JavaScript

JavaScript je objektově orientovaný, skriptovací jazyk, který hraje v moderním webovém vývoji velmi silnou roli. Při pohledu na nové, moderní a hlavně hojně využívané knihovny a frameworky postavené na JavaScriptu, jako jsou AngularJS, nebo React.js, by si člověk těžko pomyslel, že JavaScript byl vytvořen během přibližně pouhých deseti dnů.

Základním hybatelem, který uvedl do pohybu vývoj JavaScriptu, byla potřeba udělat stránky interaktivními. Hlavní společností co se o vznik JavaScriptu postarala byl Netscape. Cílem bylo vytvořit skriptovací jazyk, který umožní interakci, animaci a živou reakci na uživatele na stránkách, aniž by bylo potřeba stránku vždy načíst znovu. Vznikající konkurencí však byla Java, která chystala Java applety. Aby nevznikala přímá konkurence, bylo rozhodnuto, že JavaScript nebude určen tolik pro vývojáře, jako spíše pro designéry, tedy pro někoho, kdo nemusí mít nutně zkušenost s programováním. Vzhledem k blížícímu se termínu kdy měla být uzavřena dohoda ohledně podpory Javy v Netscapu, byl vyvíjen velký tlak na zprovoznění funkčního prototypu JavaScriptu. Během několika dní byl tedy vyvinut funkční prototyp, který byl zprovozněn v prohlížeči Netscape. V době, kdy všechny dosavadní stránky byly statické, byl samozřejmě zájem o JavaScript obrovský. Zájem měla i asociace ECMA, která se zabývá standardizací v oblasti informačních a komunikačních technologií. Přejala JavaScript jako standard a vznikl takzvaný ECMAscript.

Ten je dále rozvíjen a na jeho základě jsou implementovány jazyky jako JavaScript, nebo například JScript, které jsou pak používány prohlížeči. [12]

JavaScript je nejčastěji spouštěný na straně klienta, po stažení HTML dokumentu, kde je také definován. Výhodou je tak kromě možnosti udělat interaktivní webovou stránku i možnost vytvořit aplikaci využívající takzvaně „thick client“ architekturu, která přenáší velkou část výpočetní zátěže ze serveru na klienta. Nově už vznikají i řešení umožňující použití JavaScriptu i na straně serveru, jako je například framework Node.js.

2.2.4 Propojení HTML, CSS a JavaScriptu

Řekněme, že máme hotový HTML dokument a potřebujeme ho nastylovat pomocí CSS a také umožnit interakci pomocí JavaScriptu. Musíme tedy propojit všechny tři jazyky a zakomponovat je do dokumentu.

Možnosti jak do HTML dokumentu dostat CSS stylování jsou tři. Nejjednodušší, avšak také nejhůře udržovatelné a nejméně přehledné, je využít atribut style přímo v elementu, který chceme stylovat. Do hodnoty tohoto atributu potom můžeme zapsat jednotlivé CSS vlastnosti a jejich hodnoty. Druhou, o poznání lepší možností, je nadefinovat CSS styly přímo do HTML mezi otevírací a zavírací tag elementu <style> v hlavičce dokumentu, tedy mezi <style> a </style>. Zde pak můžeme nadefinovat jednotlivé třídy, nebo styly jednotlivých elementů. Výhodou je větší udržovatelnost a přehlednost, jelikož styly máme na jednom místě a změnou definice stylu pro jeden element se nám změní všechny výskyty daného elementu v dokumentu. Poslední a nejvhodnější, pokud stylujeme webovou stránku složenou z více dokumentů, je použití externího stylesheetu (do češtiny občas překládáno jako „stylopis“, což je důvod proč dále budu používat originální anglický výraz). V tomto dokumentu s koncovkou .css jsou styly nadefinovány stejně jako v elementu <style>. Stylesheet je potřeba importovat do HTML dokumentu pomocí elementu <link>. Tato metoda je vhodná při stylování více stránek, jelikož zatímco při použití elementu <style> bychom museli pro každý jednotlivý dokument definovat styly znovu, čímž by nám vznikal redundantní kód, takto máme styly definované jednou a do každého dokumentu je pouze importujeme. [13]

Propojení JavaScriptu s HTML je založené na stejném principu, jako při propojování s CSS. Máme opět dvě metody, kterými můžeme kód do HTML dokumentu dostat. Oba dva používají element <script> </script>. Kód můžeme buď vepsat mezi otevírací a zavírací tag, nebo pokud máme kód v externím souboru s koncovkou .js, můžeme jej nalinkovat tím, že cestu k souboru dáme jako hodnotu atributu src do elementu <script>. Opět je lepší používat druhou metodu, jelikož můžeme tentýž soubor využívat ve více dokumentech a tím opět zabránit zbytečně redundantnímu kódu. Na co je však potřeba si dát pozor v případě JavaScriptu je čas, kdy se daný kód provádí. Dokument se načítá odshora dolů, jakmile narazí na element <script>, přestane načítat stránku a místo toho pustí skript. Jakmile skript dokončí, pokračuje teprve

v načítání stránky. Pokud se skript na stránku načítá z externího zdroje, můžeme použít atribut async, nebo atribut defer. Async říká, že se má skript spustit asynchronně, jakmile to bude možné, neblokuje tedy načítání stránky. Defer říká, že skript se má spustit až na konci, poté co je celá stránka načtena. Z důvodu blokace načítání se proto skripty většinou dávají až na konec dokumentu, kdy už je zbytek stránky načtený. [14]

2.2.5 Generování univerzálně použitelného kódu

Definicí univerzálně použitelného kódu je v rámci této práce myšlen kód, který při vložení na stránky graficky zapadne a zároveň nebude narušovat stávající vzhled stránky. V kontextu k předmětu práce je jako kód brán formulář. Problémů je zde hned několik. Prvním problémem je již samotné vložení na stránky. Pokud by vložení prováděl člověk neznalý HTML, došlo by s největší pravděpodobností k rozpadu stránky, jelikož by byl kód vložen někam, kde by zasahoval do jiného elementu, a tím by rušil jeho vzhled. Univerzálně použitelný formulář je tedy potřeba udělat tak, aby při vložení téměř na jakékoliv místo ve stránce nedošlo ke znehodnocení stávajícího obsahu stránky. Druhým problémem je samotný styl formuláře, tedy barvy pozadí, barvy textu a dalších vlastností. CSS sice obsahuje dědičnost, takže každý element dědí vlastnosti rodičovského elementu, ale nemůžeme si být jistí tím, co je v již existujících stylech na stránce definované a hlavně jak.

Nejlepším možným řešením, jak nastylovat v době generování formulář správně, je tedy jednoduše dát možnost přizpůsobit si graficky formulář zákazníkem, pro kterého je formulář generovaný. A na základě tohoto nastavení následně vygenerovat styly. Nelze se spoléhat na dědičnost a to, že formulář náhodou zdědí zrovna dobré styly, nebo že jsou styly pro formulář na stránkách již předdefinované.

Druhý problém, ohledně toho, kam na stránce formulář vložit a jak zajistit to, že vkládající tím, že umístí kód na špatné místo, neponičí dosavadní vzhled stránky je nejlépe řešitelný tím, že formulář nebude umístěn do hlavního okna. Pokud využijeme modálního okna vytvořeného pomocí CSS a volaného JavaScriptem, nebo stránky vnořené pomocí elementu <iframe>, vystoupí formulář v samostatném okně do popředí na žádost uživatele. Zabráníme tedy tomu, že by formulář kazil původní stránku, jelikož od ní bude odstíněn. Některé prohlížeče však element <iframe> nepodporují, je tedy potřeba udělat obyčejné modalní okno pomocí HTML a CSS3. Spouštěčem, který má vyvolat formulář musí být tlačítko. Zde už není možné tlačítko zobrazit v modálním okně. Je potřeba, aby bylo zobrazeno na původní stránce, zároveň ale tak, aby nebylo potřeba vložit tlačítko přímo na určité místo. Toho můžeme docílit CSS vlastností position a její hodnotou fixed. Tímto můžeme tlačítko pevně „přilepit“ ke kraji obrazovky, nehledě na to, do jaké části kódu je celý nový kód vložen.

Optimálním řešením, respektive v současné době nejlepším možným, je tedy nechat zákazníka nastavit styl formuláře pomocí určitého rozhraní a následně mu vygenerovat kód, který mu doporučit vložit nejlépe před uzavírací tag těla stránky. Tento kód bude obsahovat jak CSS definice formuláře, tak JavaScript, který bude formulář ovládat. V neposlední řadě potom bude obsahovat samotný formulář, řešený jako modální okno s tlačítkem, které je nastaveno na fixní pozici, kterou si, v optimálním případě, zvolil zákazník.

2.3 Shrnutí kapitoly

Konkurence na poli rezervačních systémů stále narůstá. Na trhu je již několik opravdu dobře propracovaných možností, ze kterých si může provozovatel restaurace vybrat. Všechna řešení jsou však většinou příliš robustní, a na základě toho i poměrně drahá. Stejně tak jejich zprovoznění na stránkách restaurace může být problémem, ať už v důsledku obtížnosti vložení kódu na stránky, nebo velké časové náročnosti. Stále tedy na trhu chybí systém, který by umožňoval jednoduše a rychle získat a zprovoznit rezervační formulář tak, aby zároveň graficky do stránek zapadal.

Takovýto formulář je řešitelný pomocí modálního okna, které tak nenarušuje strukturu stránek. Okno by mělo být vyvolávané tlačítkem, které má absolutní pozici na vrchu a tím pádem nehýbe se zbylým obsahem stránky. Vzhled formuláře a tlačítka by měl být zároveň pro restauraci jednoduše nastavitelný tak, aby se docílilo co nejmenšího pocitu, že formulář není přirozenou součástí stránek restaurace.

Kapitola 3

Analýza a návrh

3.1 Problémy dostupných řešení

V rešeršní části jsou rozebírány jednotlivá aktuálně dostupná řešení. Vynechány byly ty, které byly vyloženě nevyhovující. Na dalších několika řádcích bych tedy rád vyjmenoval hlavní problémy, které při analýze dostupných řešení vyvstaly.

3.1.1 Složitost

Jedním z hlavním problémů je celková složitost dostupných řešení. Uživatel často musí projít poměrně zdlouhavým registračním procesem, kde musí vyplnit všechny možné údaje, aby se následně dostal do administračního rozhraní, kde má opět velký počet různých zbytečných funkcionalit, které mohou provozovateli restaurace připadat složité a někdy i zbytečné. V poslední řadě je často dalším problémem zprovoznění rezervačního rozhraní na straně restaurace, kdy je potřeba umístit určité tlačítko, ne-li formulář přímo do stránek a je tedy potřeba, využít osoby s alespoň částečnou znalostí HTML.

3.1.2 Časová náročnost zprovoznění

Tento problém je blízce spojený se složitostí řešení. Když si chce restaurace zprovoznit rezervační systém, chce ho mít co nejdříve. Po samotné složité registraci je ale ještě postavena před otázku, jak registrační rozhraní dostat na své stránky. Některé rezervační systémy vygenerují provozovateli restaurace kód, který si má sám na stránky vložit. Ať už se jedná o jedno tlačítko, nebo se jedná o celý formulář, pokud restauratér sám není znalý HTML, je nucen si najít externího spolupracovníka na nasazení daného kódu, případně s žádostí kontaktovat svého administrátora. Tak jako tak, proces nasazení se značně prodlužuje. Některé rezervační systémy přímo ani možnost vlastního nasazení neumožňují, ale nasazují kód samy, v tomto případě ale taky může nasazení

trvat několik dní. Některé systémy mají dokonce ve smlouvě lhůtu deseti dnů od podání žádosti. V obou případech čas od chvíle, kdy se restaurace začala u služby registrovat a kdy si zákazníci mohou začít provádět rezervace on-line je ve většině případů poměrně dlouhý.

3.1.3 Nedostatečná přizpůsobitelnost

Velká část rezervačních systémů nabízí pouze základní možnost, jak si v nastavení přizpůsobit tlačítko nebo formulář. Některé sice umožní restauraci, vybrat si ze 4 základních motivů a barev, ale ty se často míjí se stylem webové stránky restaurace, kam má prvek zapadnout. Jedinou možností je pak úprava přímo stylů, ale ty vyžadují za prvé znalost CSS a za druhé často ani nejsou možné, protože se například formulář stahuje ze serveru provozovatele systému a tak je styl neovlivnitelný. Restaurace jsou pak často nuceny vkládat si na své stránky tlačítka, která mají nevhodnou barvu, což potom kazí celý dojem z webové prezentace.

3.1.4 Cena

Tak jako v každém jiném odvětví, i v restauratérství jsou peníze vždy jedním z hlavních faktorů. Systémy jsou ale často poměrně drahé, a z tohoto důvodu mohou být pro menší restaurace nedosažitelné. Pouze malý počet systémů nabízí základní tarif s omezenými funkcionalitami zdarma. Nejlevnějším tarifem jsou často již objemné balíky funkcionalit, z nichž menší restauraci by stačilo 20% a zároveň cena bývá již od několika set korun a více. Malé restaurace, které by pak rády systém využily, jsou pak často postaveny před volbu, jestli chtějí zaplatit větší peníze, i když většinu funkcí ani nepotřebují, anebo raději zůstanou u současného řešení, které je ale zdarma.

3.2 Požadavky na aplikaci

Základní požadavky na aplikaci vychází přímo z minulé sekce, tedy z problémů aktuálně dostupných řešení. Cílem je dané problémy vyřešit a vytvořit tak klientsky příjemnější službu.

3.2.1 Rychlost

Provozovatel restaurace musí mít možnost projít celým procesem od začátku registrace do samotného nasazení formuláře na své stránky během pár minut. Hlavní výhodou služby má být její nenáročnost a okamžitost. Celý proces tedy musí být pohodlný, jednoduchý a bez zbytečného zdržování. Je tedy snaha omezit veškeré nutné věci, které jsou od uživatele na začátku požadovány na naprosté minimum.

3.2.2 Konfigurovatelnost

Dalším z důležitých bodů je nastavitelnost prvků, které si bude restaurace dávat na své stránky. Snahou je, aby koncový uživatel, tedy potenciální strávník, měl pocit, že vše je součástí webové prezentace restauračního zařízení. Důležité tedy bude, aby si restaurace mohla nastavit, jak má vypadat a kde bude tlačítko, které se bude zobrazovat na jejích stránkách a stejně tak, aby si mohli nastavit vizuální podobu okna s rezervačním formulářem.

3.2.3 Cenová dostupnost

Dalším a možná jedním z nejdůležitějších požadavků je to, aby základní použitelný tarif byl opravdu zdarma. Restaurace nepotřebují často hluboké statistiky a potvrzovací sms. Potřebují levnou náhradu za svůj dosavadní systém, kterým je často telefon, propiska a kniha. To je vše, co je v základní verzi potřeba, a to vše musí být pro restaurace dostupné zdarma.

3.3 Architektura aplikace

Aplikace bude vystavěna na konceptu třívrstvé architektury. Třívrstvá architektura je jednou z nejpoužívanějších případů vícevrstvé architektury. Jedná se o server-klient architekturu, ve které jsou tři oddělené vrstvy. Datová, která se stará o práci s daty. Prezentační, která zobrazuje výstup uživateli a Aplikační, která zpracovává výstup z vrstvy datové a předává je vrstvě prezentační. [15]

3.3.1 Datová vrstva

Datová vrstva bude v tomto případě řešena pomocí databáze, společně se souborovým systémem serveru a třídami, které se budou starat o komunikaci s příslušnými úložišti.

3.3.2 Aplikační vrstva

Aplikační vrstva je v podobě skriptů reprezentujících jednotlivé funkcionality, respektive stránky aplikace.

3.3.3 Prezentační vrstva

Prezentační vrstva bude řešena pomocí šablonovacího systému.

3.3.4 Komponenty

Aplikace bude rozdělena na dvě hlavní komponenty. První bude administrační rozhraní, které bude umožňovat veškeré potřebné funkcionality pro správu rezervací a vygenerování rezervačního formuláře. Druhou komponentou je potom

API, v prvotní verzi řešeno jedním skriptem, které bude využito pro komunikaci s rezervačním formulářem na stránce restauračního zařízení.

3.4 Funkcionality

3.4.1 Registrace

Uživatel, v tomto případě restaurační zařízení, se bude muset nejprve do systému registrovat. V současném stavu, kdy nebude služba zpoplatněná, není důvod chtít po uživateli při registraci nic jiného, než e-mail a heslo. Uživateli bude na zadanou adresu následně poslán potvrzovací e-mail, kde bude muset kliknout na odkaz, který uživatele ověří. Tím dojde k dokončení registrace.

3.4.2 Nastavení stolů

V administrační sekci bude moci uživatel spravovat své stoly. Přidávat nové typy stolů (typ stolu je definován počtem osob). U jednotlivých vytvořených typů potom spravovat, kolik stolů se v restauraci nachází. Zároveň bude moci nastavit, mohou-li se případně stoly spojit k sobě.

3.4.3 Generování rezervačního formuláře

Tato funkcionalita bude umožňovat uživateli vygenerovat si rezervační formulář vhodný ke vložení na jeho stránky. Formulář bude umožňovat nastavení barvy, textu a stylu tlačítka vyvolávajícího formulář, stejně jako jeho fixní umístění na stránce. Dále si uživatel bude moci graficky upravit formulář a modální okno, především barevně. Také bude moci nahrát logo své restaurace, které se případně bude zobrazovat nad formulářem. Po nastavení bude uživateli vypsán příslušný kód s instrukcemi, kam by ho měl na stránky optimálně vložit.

3.4.4 Nastavení otvírací doby

Uživatel bude muset pro správnou funkčnost nastavit otvírací čas pro jednotlivé dny.

3.4.5 Správa rezervací

Uživatel má v rozhraní možnost spravovat rezervace. Rezervace může procházet a případně přijmout, nebo zamítnout. Rezervace by měly být řazeny podle data začátku rezervace a zároveň by měly být první řazeny ty, které ještě čekají na rozhodnutí ze strany restaurace.

3.4.6 Vytvoření rezervace

Vytvoření rezervace provádí koncový uživatel, tedy návštěvník stránky restauračního zařízení. Vyplní formulář, kde zadá e-mail, jméno a datum a čas rezervace a obratem je ihned zpraven o výsledku v rámci zprávy pod formulářem.

3.5 Databáze

3.5.1 restaurant

Tabulka restaurant obsahuje data o restauraci. Přihlašovací údaje, token do rezervačního formuláře, otevírací dobu, název souboru na serveru obsahujícího logo restaurace a další data, především spjatá s obecným nastavením restaurace. Privátním klíčem tabulky je sloupec id.

3.5.2 seat

Tabulka seat obsahuje informace o jednotlivých stolech v restauraci. Primárním klíčem je id. Tabulka zároveň v kolonce capacity uchovává informaci o počtu osob, které stůl pojme a v kolonce restaurant_id informaci, do jaké restaurace stůl spadá.

3.5.3 reservation

Tabulka obsahuje údaje o rezervaci, její id, e-mail a jméno osoby, datum a čas od kdy a do kdy rezervace je a zároveň jestli byla schválena, zamítnuta, nebo se čeká na rozhodnutí.

3.5.4 seat_reservation

Jedná se o spojovací tabulku pro m:n vztah tabulek seat a reservation. Obsahuje id stolu z tabulky seat a id rezervace.

3.5.5 form_style

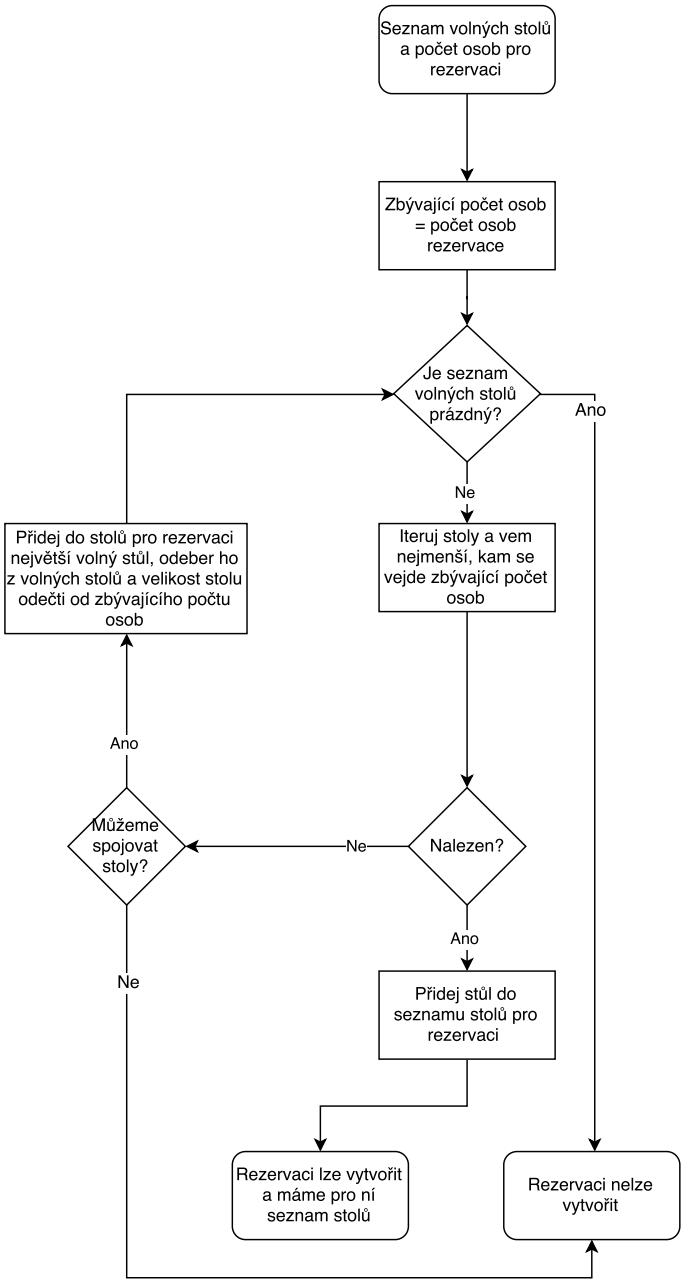
Zde se uchovávají informace o stylování formuláře. Vzhledem k faktu, že vztah mezi tabulkami restaurant a form_style je vztah 1:1, je primárním klíčem sloupec id, který odpovídá id restaurace.

3.5.6 button_style

Tabulka stejné funkce jako tabulka form_style uchovávající údaje o nastavení tlačítka na vyvolání rezervace.

3.6 Algoritmus vytvoření rezervace

Při vytvoření rezervace dostává systém informaci o dni, času začátku a konce rezervace, o počtu osob a token k určení restaurace, do které se má rezervace vytvářet. Systém je na základě těchto dat nucen určit, je-li možné na daný čas restauraci vůbec vytvořit a případně určit stoly, které budou k rezervaci využity. Stoly jsou sice v databázi reprezentovány pomocí id, ale nejedná se o žádnou spojitost s realitou. Důležité pouze je, aby obsluha věděla, že určitá rezervace má být u spojení například tří stolů. Na diagramu na následující stránce je uveden algoritmus, který určí, je-li možné rezervaci vytvořit a případě z jakých stolů. Na jeho začátku je seznam volných stolů seřazených podle velikosti od nejmenšího a počet osob, pro které se rezervace vytváří. Na výstupu je seznam stolů, které budou využity pro rezervaci, anebo informace o tom, že rezervaci není možné s daným počtem volných stolů splnit.

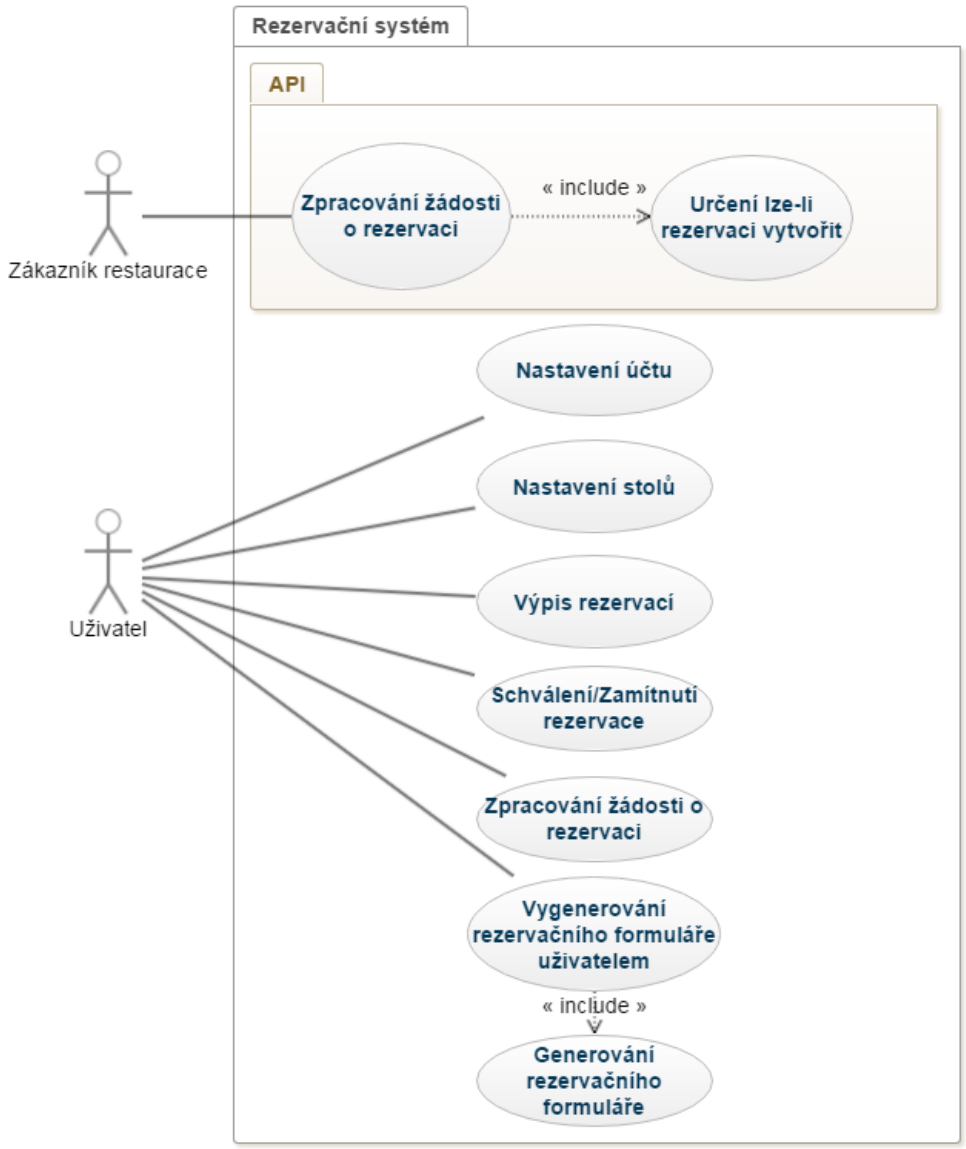


Obrázek 3.1: Algoritmus výběru stolů pro rezervaci

3.7 Případy užití

Případy užití jsou forma, kterou lze definovat interakce mezi rolí a systémem. Obvykle jsou používány jako prostředek k popisu jednotlivých funkcionalit

pro vývojáře. Přestože případy užití lze vytvářet i formou diagramu, za lepší formát je považován čistý text. Martin Fowler, jeden z předních světových specialistů na návrh a design software, říká, že případy užití nemohou mít pevně specifikovanou formu, zároveň však i na svém blogu [16] zmiňuje knihu Writing Effective Use Cases [17] od Alistaira Cockburna. Ten nadefinoval poněkud pevnější strukturu pro případy užití, podle které také budu případy definovat.



Obrázek 3.2: Diagram případů užití

3.7.1 Nastavení účtu

Hlavní účastník: Uživatel
Rozsah: Komponenta
Úroveň: Uživatelský cíl
Zúčastněné strany a zájmy:

- Uživatel - Změnit nastavení restaurace.
- Navrhovaný systém - Upravit nastavení restaurace v databázi.
- Databáze

Předpoklady:

- Funkční připojení k databázi.
- Uživatel je přihlášen.

Minimální plnění: Uživatel získá nazpět informaci o výsledku.
Úspěšné plnění: Uživatel získá nazpět informaci o výsledku a data v databázi jsou změněna.
Spouštěč: Kliknutí na tlačítko uložit na stránce nastavení.
Scénář úspěšného plnění:

1. Uživatel v nastavení klikne na tlačítko uložit.
2. Systém zkontroluje, zdali je uživatel přihlášen.
3. Systém zkontroluje vyplněné hodnoty.
4. Systém dané hodnoty změní v databázi.
5. Systém uživateli vypíše výsledek pokusu o změnu.

Alternativní scénář:

- **Některá z odeslaných hodnot nebyla validní:** Uživateli je nazpět předána informace o tom, které hodnoty nebyly validní.

Výjimky:

- **Uživatel není přihlášen:** Uživatel je s informací, že není přihlášen, přesměrován na přihlašovací stránku.
- **Nepodařilo se navázat spojení s databází:** Uživateli je vrácena informace o tom, že se nepodařilo navázat spojení s databází.

3.7.2 Nastavení stolů

Hlavní účastník: Uživatel
Rozsah: Komponenta
Úroveň: Uživatelský cíl
Zúčastněné strany a zájmy:

- Uživatel - Změnit nastavení stolů v restauraci.
- Navrhovaný systém - Upravit nastavení stolů restaurace v databázi.
- Databáze

Předpoklady:

- Funkční připojení k databázi.
- Uživatel je přihlášen.

Minimální plnění: Uživatel získá nazpět informaci o výsledku.
Úspěšné plnění: Uživatel získá nazpět informaci o výsledku a data v databázi jsou změněna.
Spouštěč: Kliknutí na tlačítko uložit na stránce nastavení stolů.
Scénář úspěšného plnění:

1. Uživatel změní hodnoty stolů a klikne na tlačítko uložit.
2. Systém zkontroluje, zdali je uživatel přihlášen.
3. Systém zpracuje údaje o stolech a změní data v databázi.
4. Systém uživateli vypíše výsledek pokusu o změnu.

Výjimky:

- **Uživatel není přihlášen:** Uživatel je s informací, že není přihlášen přesměrován na přihlašovací stránku.
- **Nepodařilo se navázat spojení s databází:** Uživateli je vrácena informace o tom, že se nepodařilo navázat spojení s databází.

3.7.3 Výpis rezervací

Hlavní účastník: Uživatel
Rozsah: Komponenta
Úroveň: Funkce
Zúčastněné strany a zájmy:

- Uživatel - Získat seznam rezervací.

- Navrhovaný systém - Dodat uživateli seznam rezervací.
- Databáze

Předpoklady:

- Funkční připojení k databázi.
- Uživatel je přihlášen.

Minimální plnění: Uživatel získá nazpět nějaká data.
Úspěšné plnění: Uživatel získá nazpět seznam rezervací odpovídající zadaným filtrům.
Spouštěč: Vstup na stránku s rezervacemi, nebo spuštění filtrace/řazení.
Scénář úspěšného plnění:

1. Uživatel vstoupí na stránku s výpisem, nebo spustí filtr/řazení.
2. Systém zkontroluje, zdali je uživatel přihlášen.
3. Systém z databáze získá potřebná data o rezervacích a vypíše je uživateli.

Alternativní scénář:

- **Na uživatelovu restauraci nejsou žádné rezervace:** Uživateli je nazpět předána informace o tom, že žádné rezervace neexistují.
- **Vybranému filtru neodpovídají žádné rezervace:** Uživateli je nazpět předána informace o tom, že žádné rezervace pro zadaný filtr neexistují.

Výjimky:

- **Uživatel není přihlášen:** Uživatel je s informací, že není přihlášen přesměrován na přihlašovací stránku.
- **Nepodařilo se navázat spojení s databází:** Uživateli je vrácena informace o tom, že se nepodařilo navázat spojení s databází.

3.7.4 Schválení/Zamítnutí rezervace

Hlavní účastník: Uživatel
Rozsah: Komponenta
Úroveň: Uživatelský cíl
Zúčastněné strany a zájmy:

- Uživatel - Schválit/Zamítnout rezervaci.
- Navrhovaný systém - Zpracovat schválení/zamítnutí rezervace.

- Databáze

Předpoklady:

- Funkční připojení k databázi
- Uživatel je přihlášen

Minimální plnění: Uživateli je vypsán výsledek pokusu.
Úspěšné plnění: Uživateli je vypsán výsledek a stav rezervace je změněn v databázi.
Spouštěč: Kliknutí na tlačítko schválit/zamítnou.
Scénář úspěšného plnění:

1. Uživatel u vybrané rezervace klikne na jedno ze dvou tlačítek schválit/-zamítnout.
2. Systém zkontroluje, zdali je uživatel přihlášen a zdali je daná rezervace jeho.
3. Systém provede změnu v databázi a vrátí uživateli informaci o provedené změně.

Výjimky:

- **Uživatel není přihlášen:** Uživatel je s informací, že není přihlášen přesměrován na přihlašovací stránku.
- **Rezervace není do uživatelovy restaurace:** Uživateli je vrácena informace o tom, že daná rezervace neexistuje.
- **Nepodařilo se navázat spojení s databází:** Uživateli je vrácena informace o tom, že se nepodařilo navázat spojení s databází.

3.7.5 Určení lze-li vytvořit rezervaci

Hlavní účastník: Navrhovaný systém - API
Rozsah: Komponenta
Úroveň: Funkce
Zúčastněné strany a zájmy:

- Navrhovaný systém - API - Zpracovat rezervaci a vrátit výsledek.
- Databáze

Předpoklady:

- Požadavek na API obsahuje potřebná data z formuláře.
- Funkční připojení k databázi.

Minimální plnění: API vrátí návratovou hodnotu.
Úspěšné plnění: API vrátí návratovou hodnotu obsahující informaci o výsledku zakládání rezervace.
Spouštěč: Přijetí požadavku v API.
Scénář úspěšného plnění:

1. API zpracuje a zkontroluje zaslaná data.
2. API si z databáze získá data volných stolech a jejich nastavení v restauraci.
3. API pomocí algoritmu na určení stolů k rezervaci získá seznam stolů, které budou k rezervaci využity.
4. API uloží data o rezervaci do databáze a vrátí informaci o založení rezervace.

Alternativní scénář:

- **Na daný termín není možné rezervaci vytvořit:** API rezervaci nezakládá a vrací informaci o nemožnosti vytvořit rezervaci na daný termín.

Výjimky:

- **Součástí požadavku na API není token restaurace:** API vrací chybovou hlášku o chybějícím tokenu.
- **Nevalidní přijatá data:** API vrací informaci o tom, která data ve formuláři nebyla validní.
- **Nepodařilo se navázat spojení s databází:** API vrací informaci o tom, že se nepodařilo připojit k databázovému serveru.

3.7.6 Zpracování žádosti o rezervaci

Hlavní účastník: Koncový uživatel využívající rezervačního formuláře na stránce restaurace (dále: Strávník)
Rozsah: Komponenta
Úroveň: Uživatelský cíl
Zúčastněné strany a zájmy:

- Strávník - Vytvořit rezervaci do restaurace.
- Formulář - Odeslat data a zobrazit výsledek.
- Navrhovaný systém - API - Zpracovat rezervaci a vrátit výsledek.
- Databáze

Předpoklady:

- Požadavek na API obsahuje potřebná data z formuláře.
- Funkční připojení k databázi.

Minimální plnění: Formulář vypíše výsledek pokusu o rezervaci.
Úspěšné plnění: Formulář vypíše výsledek pokusu o rezervaci a rezervace je zpracována pomocí API.
Spouštěč: Odeslání formuláře strávníkem.
Scénář úspěšného plnění:

1. Strávník vyplní a odešle rezervační formulář na stránce restaurace.
2. API přijme data z formuláře a na jejich základě určí možnost vytvoření rezervace.
3. Rezervace je založena a API vrací výsledek o založené rezervaci čekající na schválení.
4. Formulář výsledek zpracuje a zobrazí strávníkovi.

Alternativní scénář:

- **Na daný termín není možné rezervaci vytvořit:** API rezervaci nezakládá a vrací informaci o nemožnosti vytvořit rezervaci na daný termín.

Výjimky:

- **Součástí požadavku na API není token restaurace:** API vrací chybovou hlášku o chybějícím tokenu, ta je zpracována a zobrazena formulářem.
- **Nevalidní data ve formuláři:** API vrací informaci o tom, která data ve formuláři nebyla validní, ta je zpracována a zobrazena formulářem.
- **Nepodařilo se navázat spojení s databází:** API vrací informaci o tom, že se nepodařilo připojit k databázovému serveru, ta je zpracována a zobrazena formulářem.

3.7.7 Generování rezervačního formuláře

Hlavní účastník: Navrhovaný systém
Rozsah: Komponenta
Úroveň: Funkce
Zúčastněné strany a zájmy:

- Navrhovaný systém - Vygenerovat uživateli rezervační formulář.

- Databáze

Předpoklady:

- Funkční připojení k databázi

Úspěšné plnění: Je vygenerován rezervační formulář.
Spouštěč: Kliknutí na tlačítko generovat.
Scénář úspěšného plnění:

1. Systém získá data s nastaveními z formuláře.
2. Systém získá z databáze uživatelův token.
3. Systém vytvoří prázdný řetězec a začne do něj sestavovat kód.
   a) Do řetězce jsou vloženy definice stylů s využitím hodnot z formuláře s nastaveními.
   b) Do řetězce je vložen odkaz na skript ovládající formulář.
   c) Nakonec je do řetězce vložen ještě samotný kód tlačítka a formuláře.
4. Řetězec s kódem je vypsán uživateli.

Alternativní scénář:

- **Nepodařilo s navázat spojení s databází:** Kód je vygenerován, ale data nejsou uložena do databáze, o čemž je uživatel spraven.

3.7.8 Vygenerování rezervačního formuláře uživatelem

Hlavní účastník: Uživatel
Rozsah: Komponenta
Úroveň: Uživatelský cíl
Zúčastněné strany a zájmy:

- Uživatel - Získat kód rezervačního formuláře.
- Navrhovaný systém - Vygenerovat uživateli rezervační formulář.
- Databáze

Předpoklady:

- Funkční připojení k databázi
- Uživatel je přihlášen

Minimální plnění: Uživatel je spraven o výsledku generování.
Úspěšné plnění: Uživateli je vypsán kód rezervačního formuláře.
Spouštěč: Kliknutí na tlačítko generovat.
Scénář úspěšného plnění:

1. Uživatel nastaví vizuální podobu výsledného formuláře, přičemž živě vidí náhled daného formuláře.
2. Uživatel spustí generování kódu pomocí kliknutí na tlačítko generovat.
3. Data z nastavovacího formuláře jsou zpracována a uložena do databáze.
4. Na základě zadaných dat je vygenerován a uživateli zobrazen kód rezervačního formuláře společně s instrukcemi k jeho nasazení.

Alternativní scénář:

- **Nepodařilo s navázat spojení s databází:** Kód je vygenerován, ale data nejsou uložena do databáze, o čemž je uživatel spraven.

Výjimky:

- **Uživatel není přihlášen:** Uživatel je s informací, že není přihlášen přesměrován na přihlašovací stránku.

3.8 Shrnutí kapitoly

Na začátku kapitoly byly probrány nedostatky aktuálně existujících řešení a na jejich základě určeny primární požadavky na aplikaci. Ta musí být jednoduchá, tedy jednoduše použitelná a nemít zbytečně moc složitých funkcionalit. Zároveň musí být proces zprovoznění formuláře rychlý a zvládnutelný za pár minut. Proto je i potřeba, aby kód, který je generován byl jednoduše nasaditelný na stránky i bez znalosti HTML. Styl výsledného formuláře musí být zároveň nastavitelný tak, aby ve výsledku do vizuální podoby stránek restaurace zapadal.

Bylo určeno použití třívrstvé architektury a popsány základní funkce aplikace, tedy nastavení stylu formuláře a tlačítka, vytvoření rezervace, správa rezervací, vygenerování formuláře a nastavení restaurace. Tyto funkcionality byly následně popsány pomocí případů užití, které budou využity jako podklad pro implementaci.

Kapitola 4

Implementace

4.1 Dostupné a použité technologie

Jelikož systém je koncipován jako webová aplikace, je třeba vyřešit především, které technologie budou využívány na straně serveru, dále na straně klienta a v neposlední řadě systém ukládání a získávání dat.

4.1.1 Serverové technologie

Aktuálně je dostupné velké množství jazyků, potažmo technologií, které mohou běžet na serveru a jsou využívány. Z jazyků, které jsou obecně více používány již téměř každý určitou cestou umožňuje výstavbu webové aplikace. Ne všechny jsou však vhodné právě na tento typ aplikace. Především z důvodu, že aplikace je plně synchronní, všechny funkcionality jsou postaveny na principu požadavku a odpovědi, bez potřeby asynchronních volání na další služby v mezičase. Z tohoto důvodu nejsou tedy na použití vhodné technologie, jako například knihovna Node.js, která je zaměřena především na aplikace, které potřebují zpracovávat data v reálném čase.

Jako vhodný jazyk pro stavbu back-endu aplikace jsem proto zvolil PHP, přesněji poměrně nové PHP7. Tento jazyk je stavěn téměř výhradně jako synchronní, zároveň je objektový a především je velmi často využívaný a má tedy velkou uživatelskou základnu, s čímž je spjatá velká internetová podpora a zároveň dostupnost velkého množství knihoven a doplňků. PHP7 zároveň přineslo novinky, které z PHP dělají velmi efektivní a vhodný jazyk. PHP7 přineslo možnost definovat datové typy vstupních a výstupních parametrů a především zásadně zvýšilo rychlost přechodem na nový virtuální stroj Zend Engine 3. Tímto se průměrný počet dotazů za sekundu a samotné načítání stránek zrychlilo až o 50%. [18]

4.1.2 Klientské technologie

Pro zjednodušení zde budu za klientskou technologii považovat i šablonovací systémy, ačkoliv ty se ve skutečnosti provozují a využívají už na straně serveru, v podstatě jsou ale tím co připraví celý dokument, který se posílá v odpovědi a proto je zde budu probírat ve front-endové sekci.

Vytvořit výsledný dokument, který je předán klientovi, je možné několika způsoby. Jelikož aplikace je dynamická, nelze využít čistých statických HTML stránek. Dynamicky tedy můžeme kód vytvářet již v PHP skriptu a to jednoduše postupného kombinování PHP výstupu a html kódu. Tento způsob je však nevhodný, především kvůli špatnému oddělení výpočetní a vykreslovací části aplikace. Druhou možností je naopak převést velkou část přímo na klienta, například využítím knihovny React, která velkou část výpočetního procesu přenáší na klienta a se serverem komunikuje pouze pomocí připraveného rozhraní. Veškeré vykreslování pak probíhá až přímo na klientovi. Tato možnost v tomto případě také není vhodná, jelikož jak už bylo řečeno, aplikace je převážně synchronní a na klientovi, až na výjimky, především statická, proto je zbytečně používat robustní řešení, jako je například React. Jako nejlepší, a také využité řešení se proto jeví využití výše zmíněných šablonovacích systémů. Jedná se o knihovny, které dostanou data ze skriptu a vykreslí výsledný dokument za použití vestavěných funkcí, které například umožňují iterovat nad poli a tak dynamicky vytvořit dokument ze zadaných dat. V tomto případě byla zvolena knihovna Smarty.

Stránky budou tvořeny HTML5 v kombinaci s CSS3, které jsou v současné době standardem pro tvorbu vzhledu webových aplikací. Ačkoliv samotné stránky jsou především statické, je několik stránek, které potřebují provádět živé změny na základě vstupu od uživatele. K tomuto bude použit JavaScript.

4.1.3 Ukládání dat

Pro ukládání dat byla zvolena MySQL databáze. Jedním z důvodu byla její rozšířenost a kvalitní podpora, dalším potom fakt, že v menším měřítku je velmi efektivní i bez potřeby speciálních úprav. Tato databáze je vhodná k ukládání především textových dat. Loga firem tedy budou ukládána přímo v souborovém systému aplikace ve složce pro to určené. V databázi pak bude uchována pouze informace o názvu souboru. Stejně tak budou ukládány i CSS soubory s definicemi stylů formulářů jednotlivých restaurací.

4.2 Back-end

Back-end aplikace je rozdělen na dvě hlavní části. První částí jsou třídy, jejichž hlavním účelem je komunikace s databází. Druhou jsou pak samotné skripty, které reprezentují jednotlivé stránky. Tyto skripty zpracují požada-

vek od klienta, předají a následně zpracují data z potřebných tříd, ty potom předají šablonovacímu systému, který vrátí HTML dokument, který je odeslán klientovi v odpovědi.

4.2.1 Konfigurace

Konfigurace aplikace je řešena pomocí asociovaného pole, které se vytváří v souboru config.php. Toto pole má dva klíče, z nichž každý má jako hodnotu další pole. Přes klíč „database“ se aplikace dostane k poli, které obsahuje hodnoty potřebné pro připojení k databázi. Klíč „app“ potom ukazuje na pole, obsahující další informace, jako například e-mailové adresy, absolutní cestu ke kořenovému adresáři aplikace a další.

4.2.2 Databázový ovladač

Aplikace využívá databázový ovladač PDO. PHP7 odstranilo rozšíření ext/mysql, především kvůli bezpečnosti. PDO umožňuje vytváření předpřipravených dotazů, které za prvé chrání databázi před útokem typu sql injection, a za druhé zvyšuje efektivitu dotazování tím, že dotazy jsou automaticky optimalizovány, tedy pokud aplikace, například v rámci jednoho skriptu, několikrát vkládá do stejné tabulky, pouze různá data, tak PDO na konci odešle pouze jeden dotaz na vložení.

4.2.3 Inicializace

Vzhledem k tomu, že PHP je synchronní, znamená každé volání to, že se celá aplikace znovu zapíná. Z tohoto důvodu je na začátku v každém skriptu volán skript init.php. Tento skript v první řadě vytvoří, případně obnoví session. Následně inicializuje šablonovací systém, nastaví proměnné, jako id přihlášeného uživatele a příznak o přihlášení a nakonec vytvoří připojení k databázi.

4.2.4 Registrace a přihlášení

Při registraci uživatel zadá e-mail, který slouží k následnému přihlášení a zároveň heslo, pro kontrolu dvakrát. Po registraci je uživateli odeslán potvrzovací e-mail, ve kterém je zároveň obsažen odkaz na potvrzení registrace. Tento odkaz je složen z tokenu, který je při registraci vygenerován pomocí zavolání hashovací funkce MD5 na číslo, které vzniklo z funkce rand(). Kliknutím na daný odkaz uživatel ověří, že se e-mail shoduje s účtem. V databázi se zároveň token smaže a tím je účet ověřen.

Přihlášení probíhá tak, že uživatel zadá e-mail a heslo. Heslo je v databázi uloženo zahashované pomocí funkce password_hash s volbou algorytmu PASSWORD_BCRYPT, heslo je tedy při přihlášení zkontrolováno funkcí password_verify() a zároveň je zkontrolováno, že je uživatel již ověřen, tedy, že token v databázi je prázdný. Pokud je uživatel úspěšně přihlášen, pak je

v PHP session nastaveno userid na id přihlášeného uživatele a logged na true. Pomocí těchto příznaků v session je poté vždy kontrolováno, v uzavřené sekci, zdali je uživatel přihlášen.

4.2.5 Databázové třídy

Jak bylo výše uvedeno, komunikaci s databází zařizují výhradně třídy k tomu určené. Tímto způsobem je komunikace s databází centralizovaná a vhodně strukturovaná, což napomáhá snadnému udržování kódu.

Hlavní databázovou třídou je třída crud. Zde jsou definovány čtyři základní metody: „create“, „read“, „update“ a „delete“. Tedy metody na vložení, čtení, úpravu a smazání řádku v tabulce. Od této třídy dědí všechny další databázové třídy. Každá reprezentuje jednu z hlavních tabulek v databázi. Třídy uchovávají v privátní proměnné informaci o tom, jaký název nese tabulka, kterou reprezentují. V třídách jsou pak definovány další rozšiřující veřejné a privátní metody pro komunikaci s databází nad rámec vkládání a získávání dat.

4.2.6 Vytváření rezervace

Vytváření rezervace probíhá ve skriptu make_reservation.php. Tento skript je volán rezervačním formulářem a jsou mu předány potřebná data k vytvoření rezervace. Skript nejprve zkontroluje, jestli jsou zadaná data validní, následně zkontroluje, jestli je validní token označující restauraci. Pokud ano, provede algoritmus popsaný v případu užití 3.7.5. Na základě tohoto algoritmu zjistí, zdali je možné rezervaci vytvořit, případně ji vytvoří a odešle e-mail na adresu zadanou v rezervaci. Nazpět je poté vrácena informace o vytvoření rezervace, případně informace o tom, že restaurace je v požadované době zavřená, nebo nemá volná místa. V případě neplatných přijatých dat z formuláře, je nazpět vrácena hláška o špatném údaji, společně s HTTP stavovým kódem 409.

4.2.7 Výjimky

Metody a funkce, které by mohly skončit chybou jsou uzavřeny v bloku na odchycení výjimek. Chyba je následně zpracována, vložena do logu a prezentována uživateli. K logování se nevyužívá vlastní tabulka v databázi, ale je využíván chybový log serveru. Na následujících řádcích je vidět tělo funkce catch:

```
catch(Exception $e){
if($userid == null){$userid = 'NULL';}
error_log( date('d.m.Y␣:␣H:i:s', time())
          . ':button.php:Generate:'
          . $e->getMessage()
          . ':userid(' . $userid . ')');
```

```
$error = 'error message';
}
```

Jak je vidět v kódu, do chybového výpisu se v první řadě zaznamená datum a čas chyby, následně název skriptu a funkce, případně funkcionalita, která chybu způsobila, následně samotná chybová hláška a nakonec všechny důležité hodnoty a proměnné, které do funkce vstupovaly. Společně s tím je vytvořena chybová hláška, která je následně prezentována uživateli v šabloně.

4.2.8 Instalace

Instalace aplikace je řešená velmi jednoduše. Zdrojové soubory aplikace se nahrají do kořenové složky webového serveru. V dalším kroku je potřeba ručně změnit hodnoty v souboru /config/config.php. Jedná se o hodnoty, jako jsou připojení k databázi, e-mailové adresy a další. Po nastavení a uložení daných změn je potom potřeba spustit skript /install/install.php, který vytvoří potřebné složky pro chod aplikace a zároveň vytvoří tabulky v databázi. Tímto je instalace hotová. Je doporučeno po úspěšné instalaci smazat složku /install.

4.3 Front-end

4.3.1 Celkový vzhled

Celkový vzhled samotné aplikace, tedy bez rezervačních formulářů, byl vytvářen podle principů a doporučení Material designu. Jedná se o designový jazyk vytvořený společností Google, který propojuje principy a styly klasického designu s moderními technologiemi. Google sám začal tento design implementovat prakticky ve všech svých aplikacích. Hlavní myšlenkou je použití stínů, které by v reálném životě mělo člověku dávat pocit o hmatatelnosti daného objektu. Převedeno tedy do samotné aplikace, je hlavním konstrukčním prvkem karta, která má ostré rohy a pod sebou lehký stín, který v uživateli vzbuzuje pocit, že se jedná o prvek s kterým by měl provést určitou interakci [19]. K sestavení grafického stylu jsem využil šablonu Material Design Lite, která byla vytvořena Googlem. Šablona je stále však velmi mladá a nabízí poměrně omezený seznam prvků. Zároveň často prvky nefungují tak, jak by měly a bylo tedy potřeba vytvořit vlastní styly, které fungují v kombinaci s danou šablonou.

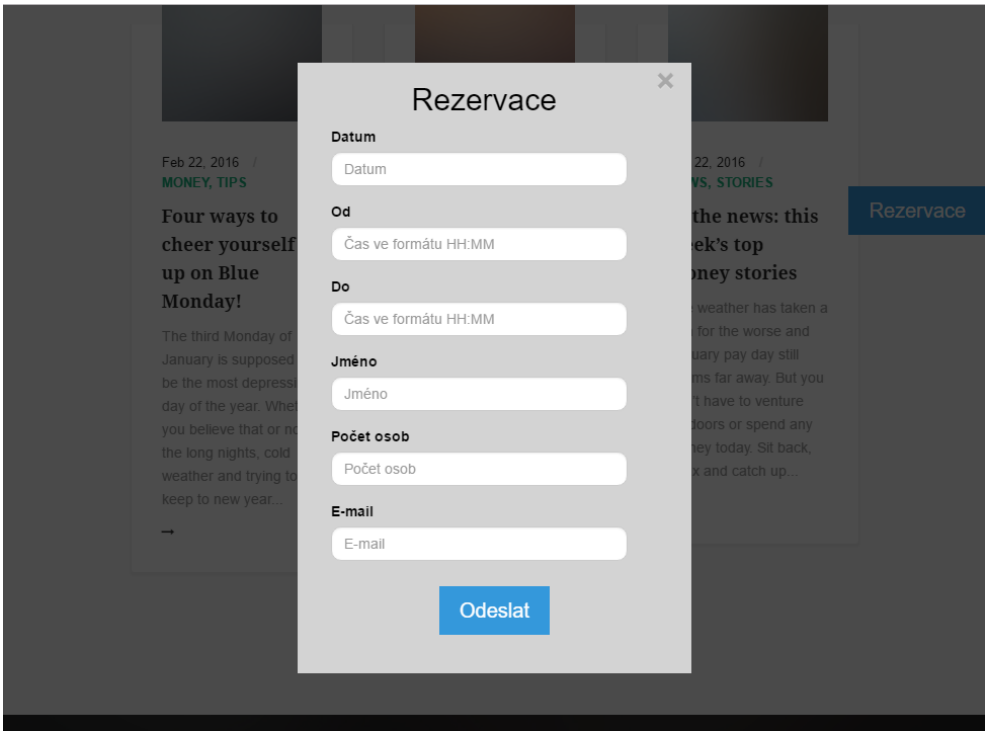
4.3.2 Rezervační formulář

Rezervační formulář, který si restaurace vkládají na své webové stránky je postaven na knihovně Bootstrap 3. Z důvodu velikosti byly vybrány pouze nejnutnější prvky, tedy pouze input a grid systém. Není tedy potřeba, aby restaurace stahovaly ještě Bootstrap JavaScript, čímž se sníží zatížení stránky, které s sebou přinese formulář. Z tohoto důvodu také není využívána knihovna

jQuery. Je však potřeba kvůli poli s datem stahovat knihovny Pikaday a Moment.js, obě jsou ale poměrně malé a nejedná se tedy o tak velký problém. V poslední řadě musí stránka stáhnout JavaSript na ovládání formuláře a styly formuláře.

Než si restaurace může kód formuláře vygenerovat, musí na příslušné stránce nastavit jeho vizuální podobu. Stejně tak musí nastavit podobu tlačítka, které pak formulář vyvolává. Jelikož každá restaurace má styl formuláře jiný, ukládá se na server pro každou restauraci vlastní CSS soubor s definicemi stylů, který je pak stránkou restaurace stahován.

Formulář je ovládán funkcí v JavaScriptovém souboru form.js, který je stránkou restaurace stažen. Tato funkce nejprve ověří, jestli jsou zadaná data validní a pokud ano, vytvoří XMLHttpRequest a ten odešle aplikaci. Ta zpracuje požadavek, jak bylo popsáno v předchozí kapitole a vrátí výsledek. Ten je následně ve formuláři zobrazen uživateli.



Obrázek 4.1: Vzhled modálního okna s formulářem

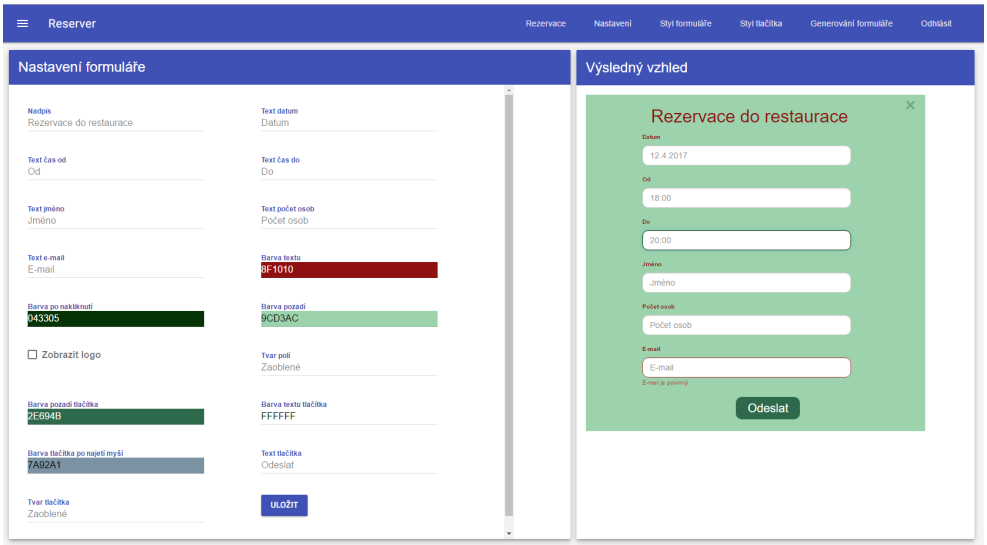
Formulář má podobu modálního okna, které je vyvoláváno stisknutím příslušného tlačítka a zavřeno buď kliknutím na X v rohu okna, nebo kliknutím mimo okno. Tlačítko a okno mají dvě podoby závislé na šířce obrazovky, na které je stránka načtena. Pokud je šířka větší než 680 px, okno zabírá 40% šířky a 84% výšky zařízení, vyvolávací tlačítko je zároveň nastaveno fixně ke straně obrazovky, podle toho jakou pozici si zvolila restaurace v nastavení.

Pokud je šířka obrazovky menší než 680 px, okno zabírá celou obrazovku a tlačítko je ukotveno ke spodní části obrazovky, zabírá 10% její výšky a je roztaženo na celou šířku obrazovky. Rozhodování o stylu je řešeno pomocí CSS3 @media pravidla.

4.3.3 Nastavení vzhledu

Nastavení vzhledu formuláře a tlačítka, které formulář vyvolává je řešeno pomocí dvou stránek. Každá stránka má skript psaný v JavaScriptu, který ovládá nastavovací formulář a na základě dat v něm mění živě vizuální podobu rezervačního formuláře nebo tlačítka. Ke sledování změn jednotlivých vstupů se používá knihovna jQuery a především její události .change() a .keyup(). V těle funkcí, které se volají při nastání těchto událostí se následně upravovaným elementům přiřadí, nebo změní, příslušná CSS definice stylu. Další důležitou používanou knihovnou v těchto formulářích je jscolor, knihovna, která umožňuje textovému vstupu přidat funkcionalitu barevné palety a uživateli tak umožňuje vybírat z barevného spektra. Tato knihovna je využívána k tomu, aby uživatel mohl volit barvu pozadí formuláře, tlačítka a textů.

V nastavení formuláře může uživatel nastavit texty jednotlivých popisků vstupů, barvu textu, barvu okraje vstupu po jeho nakliknutí, tvar vstupních polí, tedy jestli má hranaté nebo zaoblené okraje, zdali se má zobrazovat nahoře pod nadpisem logo restaurace a také styl a text odesílacího tlačítka. V nastavení vyvolávacího tlačítka můžou být nastaveny barvy tlačítka, jeho text, tvar a ukotvení, zde může uživatel nastavit, má-li být tlačítko ukotveno vlevo nebo vpravo a v jaké výšce.



Obrázek 4.2: Stránka k nastavení stylu rezervačního formuláře

4.3.4 Chybové stránky

V aplikaci jsou vytvořeny dvě statické HTML stránky, které nevyužívají šablonovací systém. Jedná se o stránky 404.html a 500.html. Tyto stránky jsou uživateli zobrazeny v případě HTTP chyb 404 a 500. Tyto dva chybové kódy jsou ty se kterými se uživatel setká nejčastěji a je tedy dobré mít pro tyto chyby vlastní stránky. Samotné přesměrování na tyto stránky v případě chyby je nastaveno v souboru .htaccess.

4.4 Shrnutí kapitoly

Pro implementaci byl zvolen na straně serveru jazyk PHP ve verzi PHP7 se šablonovacím systémem Smarty. Databáze je řešena pomocí MySQL. Klientská část je vytvořena kombinací HTML5, CSS3 a JavaScriptu. Vizuální podoba je postavena na principech Material designu a knihovně Material Design Lite od společnosti Google. K této knihovně byly dodělávány vlastní styly. Pro samotný rezervační formulář byla využita knihovna Bootstrap 3 v omezené velikosti a vlastní JavaScript pro jeho ovládání. Pro ovládání formulářů nastavujících vzhled formuláře a tlačítka byla využita ještě knihovna jQuery, především kvůli jednoduššímu nastaví posluchačů na události vstupů. Dále byly využity knihovny jscolor na výběr barev, Pikaday na výběr data a Moment.js jako podpora Pikaday.

Implementací byly pokryty všechny případy užití, ačkoliv některé případy užití byly implementací upraveny. Jaké a jak je popsáno v další kapitole.

Kapitola 5

Ověření řešení

Implementací vznikla aplikace, která pokrývá všechny výše uvedené případy užití. Při implementaci však vyšly najevo otázky a problémy, s kterými nebylo v analýze počítáno a některé případy užití byly implementovány částečně jinak, než byly v analýze navrženy.

Mezi upravenými případy užití je 3.7.2 Nastavení stolů. Zde bylo původně navrženo hromadné nastavení stolů, které je následně uloženo. Při implementaci však vyšlo najevo, že vhodnější je funkcionalitu vyřešit pomocí dvou polí s typem stolu a počtem daných stolů a tlačítek přidat a odebrat. Druhým případem užití, který byl implementován jinak, je 3.7.8 Vygenerování rezervačního formuláře uživatelem. Tato funkcionalita byla v analýze přímo spojena s nastavením vzhledu formuláře. Tedy jakmile uživatel nastavil vzhled formuláře, klikl na tlačítko, které mu vygenerovalo kód formuláře. Tato funkcionalita byla nakonec rozdělena na dvě. Na dvou stránkách si uživatel nastaví styl formuláře a tlačítka na jeho vyvolání a na třetí stránce pak dostane vygenerovaný kód s informacemi, jak ho umístit na stránky jeho restaurace. Ke změně došlo především kvůli nemožnosti jakkoliv tyto funkcionality vložit na jednu společnou stránku, jelikož se jedná o rozsáhle grafické prvky, které zabírají spoustu místa a vše na jedné stránce by bylo pro uživatele naprosto nepřehledné.

Vzhledem k podstatě aplikace a faktu, že back-end obsahuje pouze několik základních funkcionalit, byla aplikace testována přímo s uživateli. Testy ukázaly, že nastavení probíhají bez problémů, až na fakt, že občas není dostatečně popsáno, jak nastavit určité hodnoty, především u otvírací doby. Co se samotného rezervačního formuláře týče, byla aplikace testována na několika fiktivních stránkách a zkoušeno kam se dá formulář vložit tak, aby fungoval jak samotný formulář, tak aby nebyla narušena vizuální podoba stránky. V tomto případně byla aplikace poměrně úspěšná a v případě, že byl kód vložen kamkoliv do <body>, tedy prakticky do většiny stránky, fungoval formulář až na výjimky dobře. Problém nastával ve výjimečných okamžicích, kdy se kód vložil například do , který fungoval jako slider. V tomto případě formulář byl špatně zobrazen, nezavíral se a vznikaly i další vizuální a funkční problémy.

V aplikaci je však doporučeno, aby byl kód umístěn těsně před zavírací tag </head> a v takovém případě formulář funguje ve 100% případů.

Aplikace byla zároveň testována na několika rozhraních. Mezi prohlížeči na Google Chrome, Mozilla Firefox, Microsoft Edge a Internet Explorer a zároveň byla testována použitelnost na mobilních zařízení různých rozlišení. Aplikace by tedy měla být použitelná, jak na různých platformách, tak na různých zařízeních. Podstata aplikace však zabraňuje pohodlnému užívání na mobilních zařízeních, především z důvodu stránek na nastavení vzhledu formulářů, kde je nutné zobrazovat jak samotný formulář na nastavení, tak výsledný rezervační formulář. Je tedy možné aplikaci plnohodnotně využívat na mobilních zařízeních, avšak používání je tak méně pohodlné, než na větších obrazovkách.

Kapitola 6

Doporučení dalšího postupu

6.1 Další vývoj

Jelikož se jedná pouze o funkční beta verzi aplikace, jsou ještě rezervy, které by bylo vhodné dodělat, či přidělat.

6.1.1 Uživatelské rozhraní

Jak již bylo řečeno v předchozí kapitole, ovládání aplikace může být místy neintuitivní. Bylo by v rámci dalších kroků dobré provést uživatelské testy v laboratoři použitelnosti a na jejich základě upravit stránky tak, aby bylo pro uživatele jednodušší pochopit, jak provádět nastavení.

Zároveň by bylo dobré lépe navrhnout grafickou podobu aplikace pro mobilní zařízení. V tuto chvíli je řešení dostačující a aplikace je z mobilního zařízení ovladatelná, avšak například stránka na spravování rezervací je ovladatelná poměrně špatně. V tuto chvíli se jedná tedy o minimum, které na mobilních zařízeních funguje tak, aby veškeré funkcionality šly použít. Avšak bylo by dobré vytvořit lepší grafický návrh na zařízení menšího rozlišení a ten následně převést do HTML šablony.

6.1.2 Funkcionality

Stav všech funkcionalit je v současném stavu řešení postačující k tomu, aby byla aplikace reálně využitelná. Bylo by však dobré vylepšit určité funkcionality tak, aby byly buď jednodušeji použitelné, nebo lépe pochopitelné. Stejně tak by bylo vhodné doplnit aplikaci o několik dalších funkcí, které by ulehčily práci jak provozovatelům restaurací, tak jejich zákazníkům.

6.1.2.1 Správa rezervací

Zde by bylo dobré v budoucnu umožnit filtrování a vlastní řazení nad tabulkou rezervací, které dovolí provozovateli, trochu lépe se zorientovat v rezervacích.

Zároveň by měla mít restaurace i možnost nechat si odesílat upozornění o nově vznikajících restauracích na e-mail. Stejně tak by měl v budoucnu uživatel vidět, které stoly jsou k rezervaci potřeba, aby nebylo nutné informace o stolech vést samostatně ručně.

6.1.2.2 Nastavení restaurace

Nastavení otevírací doby je v tuto chvíli málo intuitivní. Uživatel je nucen zadávat otevírací dobu ručně a to pouze ve formátu HH:MM. Co se týče nastavení, které dny je úplně zavřeno, v současné chvíli je za den, kdy je restaurace zavřená, považovaný takový, který má otevírací i zavírací čas nastaven na 00:00, toto ale nikde není explicitně řečeno a tak může být systém nastavení poněkud matoucí. Do budoucna by tedy bylo dobré nasadit do formuláře knihovnu na vybírání času a zároveň nějak dát uživateli vědět, jak může nastavit, že je některý den zavřeno. Vhodné by zároveň bylo přidělat možnost, automaticky potvrzovat rezervace bez nutnosti ručního potvrzení obsluhou restaurace.

6.1.2.3 Vytváření rezervace

Zde se jedná o stejnou situaci s časy jako u nastavení otevírací doby. Uživatel je nucen zadat čas přesně v určeném formátu. Zde ale pracuje kontrolní skript, který v případě špatného formátu uživatele upozorní. I tak by ale knihovna na volbu času uživatelům práci ulehčila.

Větším problémem je ale fakt, že aplikace v tuto chvíli neumí formuláři zadat časy, kdy je restaurace zavřená a uživatel tedy může odeslat jakoukoliv žádost, na kterou dostane odpověď o neúspěchu kvůli otevírací době až po zpracování na serveru. Zde by bylo dobré, aby se při inicializaci formuláře dotázal inicializační skript serveru na otevírací dobu v jednotlivé dny a podle ní potom živě měnil možnost výběru času podle toho, jaký si zvolil uživatel den. Tím by uživatel okamžitě věděl, kdy je v daný den restaurace otevřená a nebyl nucen vždy nejprve formulář odeslat.

Zároveň je aktuálně trochu problém s tím, že aplikace neumí odlišit, zdali je restaurace v danou chvíli obsazena, nebo nemá vůbec dostatečnou kapacitu, nebo zdali má restaurace nastaveno, že stoly se nespojují a zároveň nemá tak velký stůl, aby daný počet lidí usadila. V těchto všech případech je vždy uživatel pouze zpraven o tom, že restaurace je v tu dobu obsazená. Výsledek tedy není přímo jasný a mohou nastávat určité nedorozumění. Tato situace by byla možná vyřešit vylepšením algoritmu, který se stará o výběr stolů tak, že by kromě prázdného pole se stoly vracel i příznak, proč je pole prázdné a na základě toho by se odeslala do formuláře příslušná zpráva.

6.1.2.4 Úvodní strana

V tuto chvíli je úvodní stranou aplikace strana pouze s kartou pro přihlášení a menu, které umožňuje dostat se ještě na stranu pro rezervaci. Bylo by dobré upravit danou stránku graficky a udělat z ní tak jednostránkovou vizuální prezentaci aplikace, kde by se potenciální uživatel mohl seznámit s tím, co využíváním systému může získat a v čem jsou jeho výhody.

6.2 Shrnutí kapitoly

Jelikož se jedná o beta verzi, doporučení na další vývoj je poměrně hodně. Aplikace je v tuto chvíli plnohodnotně použitelná, avšak v některých místech může být ovládání méně intuitivní. Proto by bylo vhodné v první řadě udělat uživatelské testování v laboratoři použitelnosti, na jehož výsledcích by se dala aplikace upravit, především po vizuální stránce tak, aby bylo použití pro uživatele příjemnější.

Zároveň ačkoliv je aplikace v této verzi použitelná na mobilních zařízeních a plně responzivní, bylo by potřeba vytvořit návrh designu speciálně pro mobilní zařízení. Systém je totiž sice plně responzivní, ale ovládání na menších rozlišeních není vyloženě příjemné.

Dále byly v kapitole popsány úpravy a dodělávky funkcionalit jako vytváření rezervace, jejich správa a nastavení restaurace.

Závěr

Cílem práce bylo analyzovat, navrhnout a implementovat nový rezervačního systému pro restaurační zařízení.

V první rešeršní části byly představeny dostupná řešení a vyzdvihnuty jejich klady a nedostatky. Tyto získané informace byly posléze využity v analytické části. Rešerše dále obsahuje historii vzniku, použití a spolupráce webových jazyků HTML, CSS a JavaScript, na což dále navazuje poslední kapitola rešerše, zabývající se návrhem HTML formuláře, který je jednoduše zakomponovatelný do stránek tak, aby zároveň co nejméně narušil jejich dosavadní strukturu.

Na základě dat z rešeršní části byl následně v části analytické vytvořen návrh aplikace. V tom se kladl důraz především na možnost nastavení vizuální stránky formuláře a jeho jednoduché nasazení na stránky. Zároveň byl kladen důraz na to, aby byl proces co nejjednodušší a nejrychlejší.

Aplikace byla následně na základě analýzy implementována a její funkčnost ověřena uživatelskými testy. Na základě tohoto testování byly nalezeny a opraveny implementační chyby, ale především bylo ověřeno, že byly splněny požadavky na rychlost vygenerování funkčního formuláře pro restauraci a zároveň jeho následné jednoduché nasazení.

V závěru práce jsou popsány doporučené úpravy a rozšíření aplikace, které by dále měly vést ke zlepšení uživatelnosti a celkové kvalitě výsledného produktu.

Vzniklá aplikace cílí především na restaurační zařízení, které nepotřebují robustní složité řešení. Nabízí tak vhodnou alternativu pro již zavedené velké rezervační systémy. Její výhodou oproti ostatním službám je především nenáročnost zprovoznění a zároveň fakt, že celý proces od registrace po nasazení funkčního formuláře na stránky restaurace zabere pouze pár minut.

Všechny stanovené cíle práce tím byly splněny.

Literatura

[1] Český statisický úřad: Informační společnost v číslech - 2016. [online], duben 2016. Dostupné z: https://www.czso.cz/documents/10180/42790941/061004-16_C.pdf/fde15bda-831c-4f19-a745-3690937e0346?version=1.1

[2] Restu.cz: Největší online průvodce po restauracích | Restu.cz. [online], [cit. 2017-21-3]. Dostupné z: https://www.restu.cz/

[3] Restu.cz: Zapojit svůj podnik. pro.RESTU.cz | Rezervace jako nástroj. [online], [cit. 2017-21-3]. Dostupné z: https://pro.restu.cz/zapojit-podnik/

[4] RezervujStůl: Ceník rezervačního systému pro restaurace | Rezervační systém pro restaurace RezervujStůl. [online], [cit. 2017-21-3]. Dostupné z: http://www.rezervujstul.cz/cenik/

[5] BookioPro: BookioPro | O nás. [online], [cit. 2017-21-3]. Dostupné z: https://www.bookiopro.com/about-us?lang=cs

[6] BookioPro: BookioPro | Ceník. [online], [cit. 2017-21-3]. Dostupné z: https://www.bookiopro.com/price-list/monthly

[7] Reservanto: Porovnání variant | Reservanto.cz. [online], [cit. 2017-21-3]. Dostupné z: https://www.reservanto.cz/PorovnaniVariant

[8] Schafer, S. M.: *HTML, XHTML a CSS - Bible pro tvorbu WWW stránek*. Grada, čtvrté vydání, 2009, ISBN 978-80-247-2850-6.

[9] W3Schools Online Web Tutorials: HTML5 New Elements. [online], [cit. 2017-22-3]. Dostupné z: https://www.w3schools.com/html/html5_new_elements.asp

[10] Marc Andreessen: Indented <MENU>s. [online], osobní komunikace, [cit. 2017-22-3]. Dostupné z: http://1997.webhistory.org/www.lists/www-talk.1994q1/0648.html

[11] World Wide Web Consortium (W3C): A brief history of CSS until 2016. [online], [cit. 2017-22-3]. Dostupné z: https://www.w3.org/Style/CSS20/history.html

[12] Single Sign On & Token Based Authentication - Auth0: A Brief History of JavaScript. [online], [cit. 2017-22-3]. Dostupné z: https://auth0.com/blog/a-brief-history-of-javascript/

[13] Duckett, J.: *HTML & CSS: design and build websites*. Wiley, první vydání, 2011, ISBN 1118008189.

[14] Duckett, J.: *JAVASCRIPT & JQUERY: Interactive Front-End Web Development* . Wiley, první vydání, 2013, ISBN 1118531647.

[15] IBM Knowledge Center: Three-tier architectures. [online], [cit. 2017-23-3]. Dostupné z: https://www.ibm.com/support/knowledgecenter/en/SS7JFU_8.5.5/com.ibm.websphere.express.doc/ae/covr_3-tier.html

[16] Martin Fowler: UseCase. [online], [cit. 2017-23-3]. Dostupné z: https://martinfowler.com/bliki/UseCase.html

[17] Cockburn, A.: *Writing Effective Use Cases*. Addison-Wesley, třetí vydání, 2001, ISBN 0201702258.

[18] Zend Technologies: Get performance insight into the upcoming release of PHP7. [online], [cit. 2017-4-5]. Dostupné z: http://www.zend.com/en/resources/php7_infographicl

[19] Google: Material Design. [online], [cit. 2017-4-5]. Dostupné z: https://material.io/guidelines/material-design/introduction.html

Příloha **A**

Seznam použitých zkratek

HTML HyperText Markup Language - základní jazyk na tvorbu statických webových stránek

CMS Content Management System - systém určený k úpravě obsahu na stránkách

CSS Cascading Style Sheets - jazyk upravující vizuální podobu webových stránek

API Application Programming Interface - rozhraní pro interakci s programem

Příloha **B**

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `install.txt` ..... návod na instalaci aplikace ze src
- `src.zip` ..... zdrojové kódy implementace
- `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `thesis.pdf` ..... text práce ve formátu PDF