Restaurační rezervační systém

David Vajdík

Bakalářská práce
2024

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav informatiky a umělé inteligence

Akademický rok: 2023/2024

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**David Vajdík**
Osobní číslo:	**A21282**
Studijní program:	**B0613A140020 Softwarové inženýrství**
Forma studia:	**Prezenční**
Téma práce:	**Restaurační rezervační systém**
Téma práce anglicky:	**Restaurant Reservation System**

Zásady pro vypracování

1. Vypracujte literární rešerši na dané téma.
2. Specifikujte požadavky na systém.
3. Proveďte funkční a datovou analýzu problémové oblasti.
4. Vytvořte model systému s využitím diagramů UML.
5. Implementujte navrhnutý systém.

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. ARLOW, Jim a NEUSTADT, Ila. UML 2 a unifikovaný proces vývoje aplikací: objektově orientovaná analýza a návrh prakticky. 2., aktualiz. a dopl. vyd. Brno: Computer Press, 2007. ISBN 9788025115039.
2. COCKBURN, Alistair. Use Cases: jak efektivně modelovat aplikace. Brno: CP Books, 2005. ISBN 8025107213.
3. FOWLER, Martin. UML Distilled: A Brief Guide to the Standard Object Modeling Language. 3rd edition. Addison-Wesley Professional, 2003. ISBN 978-0321193681.
4. LARMAN, Craig. Applying UML and patterns: introduction to object-oriented analysis and design and iteractive development. 3rd ed. New Jersey: Prentice-Hall, 2005. ISBN 01-314-8906-2.
5. ROSS, Jeanne W., Peter WEILL a David C. ROBERTSON. Enterprise Architecture As Strategy: Creating a Foundation for Business Execution. Harvard Business Review Press, 2006. ISBN 1591398398.

Vedoucí bakalářské práce: **Ing. Darina Bajusová**
Ústav počítačových a komunikačních systémů

Datum zadání bakalářské práce: **5. listopadu 2023**
Termín odevzdání bakalářské práce: **13. května 2024**

doc. Ing. Jiří Vojtěšek, Ph.D. v.r.
děkan

prof. Mgr. Roman Jašek, Ph.D., DBA v.r.
ředitel ústavu

Ve Zlíně dne 5. ledna 2024

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne 9. 5. 2024

David Vajdík v. r.
podpis studenta

ABSTRAKT

Bakalářská práce se zabývá nejen návrhem, ale především praktickou tvorbou funkčního rezervačního systému, který umožňuje zákazníkům rezervovat si stoly v restauraci podle jejího půdorysu, a to bez nutnosti telefonického hovoru. Tato práce se zaměřuje na zjednodušení a zefektivnění procesu rezervace, kdy zákazníci mohou prostřednictvím online systému snadno a rychle vybrat a rezervovat stoly podle svého přání, aniž by museli kontaktovat restauraci telefonicky. Výsledkem je moderní minimalistický vzhled webové aplikace, jenž umožňuje zákazníkům nejen provádět rezervace, ale i získávat informace o restauraci.

Klíčová slova: Rezervace, Restaurace, UML

ABSTRACT

The bachelor thesis deals not only with the design, but mainly with the practical creation of a functional reservation system that allows customers to reserve tables in a restaurant according to its floor plan, without the need for a phone call. This work focuses on unifying and streamlining the reservation process, where customers can easily and quickly select and reserve tables according to their preferences through an online sys-system without having to contact the restaurant by phone. The result is a modern, minimalist look and feel of the web application that allows customers to not only make reservations but also get information about the restaurant.

Keywords: Reservations, Restaurants, UML

Chtěl bych poděkovat vedoucí práce slečně Ing. Darině Bajusové, za pomoc a správné nasměrování cesty bakalářské práce. Taktéž bych chtěl poděkovat své rodině za podporu při vypracovávání bakalářské práce.

Prohlašuji, že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Prohlašuji, že při tvorbě této práce jsem použil/a nástroj generativního modelu AI ChatGPT; https://chatgpt.com/ za účelem úpravy a ověření správnost kódu. Po použití tohoto nástroje jsem provedl/a kontrolu obsahu a přebírám za něj plnou zodpovědnost.

Prohlašuji, že při tvorbě této práce jsem použil/a nástroj generativního modelu AI Copilot; https://www.bing.com/chat za účelem generování obrázků. Po použití tohoto nástroje jsem provedl/a kontrolu obsahu a přebírám za něj plnou zodpovědnost.

OBSAH

ÚVOD

Rozhodl jsem se vybrat téma bakalářské práce *Restaurační rezervační systém* z několika důvodů. Osobně mám silný zájem o restaurace a často je navštěvuji, což mě motivuje k tomu, abych přinesl inovace do této oblasti. Zároveň jsem chtěl využít příležitosti k prohloubení svých znalostí v HTML, CSS a JavaScriptu a zároveň získat zkušenosti s PHP, Bootstrapem a UML. Tento projekt mi tak umožnil rozvíjet mé dovednosti a zároveň přispět k zlepšení efektivity v restauračním průmyslu.

Restaurační rezervační systém umožňuje zákazníkům snadno a efektivně rezervovat stoly v restauraci podle jejího půdorysu. Cílem této bakalářské práce je vytvořit funkční a intuitivní systém, který zjednoduší proces rezervace a umožní zákazníkům vybrat si stoly podle svých preferencí, a to bez nutnosti telefonických hovorů.

Tato práce je rozdělena na dvě části. První část je teoretická a poskytuje čtenářům přehled o tématu a technologiích použitých při tvorbě projektu. Druhá část je praktická a zaměřuje se na popis mého praktického projektu, včetně jeho vzhledu a funkčnosti.

I. TEORETICKÁ ČÁST

1 LITERÁRNÍ REŠERŠE

Moderní technologický pokrok rychle transformuje způsob, jakým lidé provádějí rezervace a jak podniky v oblasti pohostinství operují. Hotelnictví a pohostinství obzvláště zažívají výrazný vliv těchto inovací. Digitální rezervační systémy otevírají nové příležitosti, včetně zvýšení efektivity, personalizace zážitků zákazníků a reakce na aktuální změny v chování spotřebitelů. Jedním z klíčových trendů je přechod k digitálním rezervacím díky pohodlí a dostupnosti prakticky neustále. Tyto rezervační systémy přinášejí okamžitá potvrzení, aktualizace v reálném čase a globální dosah, což mění paradigma tradičních rezervací. Mobilní integrace hraje klíčovou roli, kde aplikace umožňují pohodlné a přístupné rezervace jediným klepnutím na obrazovku. Personalizace rezervací podle individuálních preferencí zákazníků zvyšuje celkovou spokojenost a vede ke zvýšené loajalitě. [1]

1.1 Historie rezervačních systémů

Rezervační systémy má kořeny v letecké dopravě. První kroky k centralizovanému rezervačnímu systému byly podniknuty v roce 1946. Původním záměrem tohoto systému bylo zajistit efektivnější správu rezervací a dostupnosti letenek, byl totiž vyvinut pro leteckou dopravu. [2]

S nástupem moderních proudových motorů, které začaly nahrazovat pístové motory, došlo ke snížení účinnosti a vznikla potřeba sofistikovanějších systémů. K tomu došlo v roce 1952. Poté rostla poptávka po sofistikovanějších rezervačních systémech. Rezervační systémy se stále více rozšiřují do oblastí jako je železniční doprava, pohostinství a další průmyslová odvětví. V současné době se stávají klíčovým prvkem pro řízení rezervací a řízení dodávek zboží v reálném čase. [2]

1.2 Funkce, výhody a nevýhody

Pracovníci v oblasti pohostinství tráví mnoho času vyřizováním telefonických hovorů a vysvětlováním možností zákazníkům. Tradiční rezervační systémy vyžadují neustálou dostupnost zaměstnance, což může omezovat jejich produktivitu. Online rezervační platformy umožňují úsporu času a administrativních nákladů. Zákazníci mohou provádět rezervace sami přímo na webových stránkách podniku, což zvyšuje pohodlí a eliminuje potřebu telefonických hovorů. Některé restaurace již tuto moderní technologii využívají a zaznamenávají zvýšení efektivity a úspory pracovního času. Další výhody online rezervací zahrnují možnost integrování s webovým prodejem, poskytování lepšího zážitku zákazníkům a

moderní přístup k rezervacím. S rostoucím trendem digitálního nakupování je důležité držet krok s moderními technologiemi a přejít na online rezervace, aby podniky zůstaly konkurenceschopné. Přestože přinášejí mnoho pozitiv. Některé systémy účtují finanční náklady, měli bychom být obezřetní před sliby bezplatných verzí, protože některé mohou mít skryté poplatky za každou transakci nebo měsíční paušál. I přes finanční náklady jsou výhody online rezervačních systémů pro firmy výraznější než tyto malé měsíční poplatky. Další nevýhodou je závislost online rezervačních systémů na připojení k internetu. I když je přístup k internetu běžný, někteří lidé nemusí mít k dispozici internet ve chvíli potřeby. Online rezervační systémy otevírají podnikům různé příležitosti, včetně maximalizace sociálních médií, úspory času zaměstnanců, nabídky doplňkových produktů a dalších funkcí. [3]

1.3 Technologické trendy v rezervačních systémech

Trendy v oblasti umělé inteligence a automatizace umožňují pohostinství poskytovat výjimečné služby zákazníkům. Od zkrácení doby čekání po cílení a personalizaci pomocí umělé inteligence přináší tyto technologie značné výhody. Integrace s dalšími systémy, jako jsou CRM, POS a správa inventáře, minimalizuje provozní náklady a přináší holistický přístup k online rezervacím. Bezkontaktní řešení, včetně QR kódů a bezdotykových plateb, reaguje na nové normy a přináší zvýšenou hygienu a efektivitu procesů. Významným prvkem inovací jsou průkopnické veletrhy, kde se vytvářejí a sdílí nové nápady. Zabezpečení dat se stává nezbytným v oblasti online rezervací s důrazem na bezpečné platební brány, šifrování a certifikáty SSL. V souladu s přísnými vládními nařízeními přichází trend udržitelnosti. Ekologické postupy nejen snižují náklady a zvyšují efektivitu podniků, ale také přitahují ekologicky uvědomělé zákazníky, což přináší konkurenční výhodu. Využívání přehledů v reálném čase umožňuje podnikům okamžité reakce na trendy v rezervacích, úpravy cen a personalizaci komunikace. Analýza dat přináší informovaná rozhodnutí a zvyšuje efektivitu pohostinství. Celkově lze konstatovat, že aktuální trendy v oblasti online rezervací redefinují pohostinství, poskytují nové možnosti a vytvářejí lepší zážitky pro zákazníky. Tyto inovace jsou klíčovým faktorem pro konkurenceschopnost podniků v dnešním digitálním prostředí. [3]

1.4 Bezpečnostní aspekty rezervačních systémů

Bezpečný online rezervační systém představuje klíčový prvek pro uživatele rezervačního softwaru, kteří vyhledávají bezpečné a důvěryhodné možnosti. Zabezpečení tohoto systému

je kritické z hlediska ochrany jak dat, tak i citlivých informací zákazníků, a současně zajištění integrity celého systému. Pro zajištění důvěrnosti tajných zpráv byl zvolen algoritmus AES jako soukromý klíč. Kryptografie s eliptickou křivkou byla implementována pro veřejný klíč, což zabezpečuje autentizaci, integritu a službu nepopírání. Pro udržení integrity zpráv byla zvolena kombinace SHA-256 a digitálního signálu eliptické křivky. Citlivé údaje zákazníků jsou chráněny pomocí 256bitové rezervační stránky SSL, splňující průmyslový standard. Pravidelné skenování a odstraňování škodlivého softwaru s využitím antivirového softwaru pro aktivní ochranu před možnými útoky.[4]

Zavedení těchto bezpečnostních opatření, spolu s certifikátem SSL, poskytuje zákazníkům jistotu, že jejich transakční údaje jsou v bezpečí, což výrazně přispívá k důvěře uživatelů a úspěšnému provozu online rezervačního systému.[4]

1.5 Současné restaurační rezervační systémy

V současné době je předchozí rezervace v některých podnicích nezbytná, zejména kvůli rostoucímu počtu návštěvníků a snaze podniků efektivně spravovat své kapacity. Přesto tato rezervace často nevyhovuje mým osobním preferencím. Mnoho podniků přijímá rezervace pouze telefonicky, což může být pro některé zákazníky nepříjemné anebo časově náročné. [5][6]

Když je online rezervace dostupná, často chybí možnost výběru konkrétního místa k sezení. Pro mě osobně je volba přesného umístění stolu v podniku zásadní, protože může ovlivnit kvalitu mého zážitku z návštěvy. [7]

Z těchto důvodů jsem se rozhodl vypracovat tuto bakalářskou práci, která se zaměřuje na analýzu současného stavu rezervací v podnicích a zkoumá možnosti zlepšení procesu rezervace tak, aby vyhovoval potřebám a preferencím zákazníků, včetně možnosti výběru konkrétního místa k sezení při online rezervacích.

2 ROZBOR PROBLEMATIKY

V teoretické části se podrobně zabývám jazykem UML (Unified Modeling Language) a diagramy znázorněné v tomto jazyce, které budou později využity v praktické části mé práce. V práci se zaměřuji na robor a vysvětlení těchto diagramů, jejich strukturu a účel, stejně jako na způsoby, jak proces návrhu a vývoje softwaru využívá tyto diagramy.

Následně se soustředím na popis technologií v práci, konkrétně na programovací jazyky a nástroje, které jsou důležité pro vývoj webových aplikací. V práci se zabývám PHP, populárním serverovým jazykem, který umožňuje rychlé generování obsahu na webových stránkách. Dále zkoumám JavaScript, který je klíčovým klientským jazykem pro interaktivní prvky a dynamické chování webových stránek.

V práci také popisuji strukturu webových stránek pomocí HTML a design jejich rozhraní s využitím CSS. Kromě toho analyzuji použití frameworku Bootstrap, který usnadňuje vytváření responsivních a moderních webových stránek. V práci budu diskutovat, jak tyto technologie a nástroje přispívají k celkovému řešení problému a jejich vzájemnou integraci.

Tento soubor programovacích jazyků a nástrojů jsem si vybral z důvodu toho, že jsem s některými z nich měl krátkodobé předešlé zkušenosti a chtěl jsem se v nich zdokonalit, a naopak některé z nich jsem měl zájem se naučit.

2.1 Požadavky na systém

Je důležité před započetím práce na projektu jasně specifikovat požadavky na funkčnost systému, protože nejasné zadání požadavků vede k tomu, že se projekt opozdí a z finančního hlediska se stane velmi neefektivním. To je způsobeno hlavně díky změnám, které musí nastat po nejasném nebo neúplném zadání požadavků na systém. Požadavky rozlišujeme na funkční a nefunkční. Porozumění rozdílu mezi těmito požadavky je klíčové nejen pro IT dodavatele, ale také pro zákazníka, protože vede k upřesnění rozsahu práce, optimalizaci nákladů, a to vede ke spokojenosti zákazníka.

2.1.1 Funkční požadavky

Funkční požadavky určují, jaké konkrétní činnosti a schopnosti má systém splňovat pro jeho efektivní průběh. Tyto požadavky jsou hlavním způsobem, jak zákazník komunikuje své požadavky na systém s týmem, který si díky nim může ověřit, zda systém zahrnuje všechny požadované prvky a funkce. Také jsou zodpovědné za udržení směru projektu. Nejasně, nebo

špatně definované funkční požadavky mohou vést ke špatné definici rozsahu daného projektu, což může způsobovat problémy už v samotných začátcích projektu. Pokud systém nesplní funkční požadavek, dochází k selhání, protože nedokáže plnit svou určenou roli a reagovat na očekávání zákazníka. Systém reaguje dle funkčních požadavků na vstupní data a dle toho stanovuje výstupy a tím zajišťuje, že systém bude fungovat správně dle podmínek kvality. Pokud budou důsledně dodržovány požadavky, bude systém funkční. [8]

2.1.2 Nefunkční

Nefunkční požadavky, popisují, jak má systém fungovat a jsou omezením chování pro systém. Tyto požadavky se zaměřují zejména na věci, jako je výkon systému, spolehlivost, bezpečnost anebo rychlost. I když se na první pohled může zdát, že jsou funkční požadavky důležitější než ty nefunkční, tak tomu tak není, protože nefunkční požadavky jsou klíčové pro zajištění správného fungování systému. Při zadání nedostatečných nefunkčních požadavků může dojít ke špatné uživatelské zkušenosti, což může odradit zákazníky od opakovaných užívání. [8]

3 JAZYK UML

Unified Modeling Language dále jen UML je modelovací jazyk je navržený pro specifikaci, vizualizaci, konstrukci a dokumentaci softwarových systémů i obchodních modelů a dalších nesoftwarových systémů. UML je tvořeno integrovanou sadou diagramů, které napomáhají při zobrazování toho, jak se struktury systému chovají. Účelem těchto diagramů je modelování a analýza různých aspektů systému. UML není programovací jazyk, ale modelovací nebo také vizuální jazyk. Tenhle jazyk používá grafické notace k vyjádření návrhů softwarových projektů. Díky této vlastnosti je velmi důležitou součástí objektově orientovaného softwaru a vývojového procesu těchto softwarů. UML nabízí sadu prověřených inženýrských postupů pro modelování rozsáhlých a komplexních systémů, ve kterých poskytuje podporu softwarovým inženýrům, obchodníkům a systémovým architektům při návrhu a analýze. [9][10]

3.1 Historie UML

UML byl přijat jako standardní modelovací jazyk Object Management Group v roce 1997 a mezinárodní organizace pro normalizaci ISO jej uznala jako schválený standard v roce 2005. UML byl navržen s cílem sjednotit nejlepší prvky předchozích notací a poskytnout standardní způsob vizualizace návrhu systému. UML vzniklo sjednocením tří hlavních metod Object Modeling Technique nebo OMT od Jamese Rumbaugha. Tato metoda byla nejlepší pro analýzu a práci s datově náročnými informační systémy. Poté Booch od Gradyho Boocha, tato metoda byla nejlepší pro návrh a implementaci. A v neposlední řadě Object-Oriented Software Engineering od Ivara Jacobsona. Hlavním přínosem této metody byl případy užití. V roce 1994 se Jim Rumbaugh tvůrce OMT spojil s Grady Boochem v softwarové společnosti jménem Rational Corp, aby sloučili jejich myšlenky od jedné jediné sjednocené metody. Tato událost vedla k vytvoření UML. V této práci byly případy užití zahrnuty pro správné pochopení chování celého systému. [9][10]

3.1.1 Případy užití

Případy užití, také známé jako Use Case diagram, jsou v UML nástrojem pro vizualizaci interakcí mezi uživatelem a systémem. Umožňují uživateli dosáhnout konkrétních cílů a poskytují přehled o různých způsobech, jak může uživatel komunikovat se systémem. Rovněž definují kontext a požadavky systému. Komponenty případů užití zahrnují aktéry, případy užití, hranice systému a vazby/vztahy.[11][12]

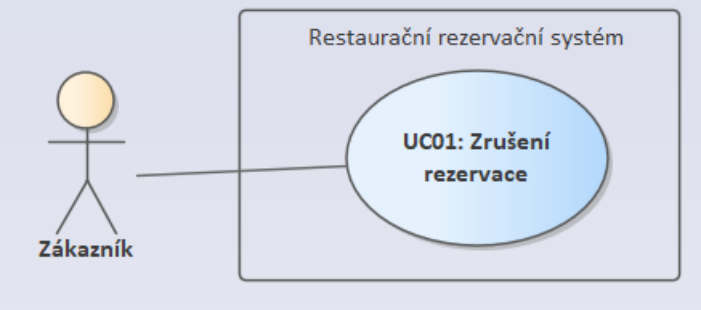
Aktéři jsou externí entity, jako jsou například uživatelé, jiné systémy a zařízení, které komunikují se systémem zvenčí. Uživatel může být jednotlivec nebo skupina lidí, která využívá systém k dosažení konkrétních cílů nebo provádění úkolů. Jiné systémy mohou zahrnovat software nebo hardware, který si vyměňuje data s naším systémem nebo spolupracuje s ním. Externí entity hrají klíčovou roli v interakci se systémem a určují reakce systému na různé vstupy a výstupy.[11][12]

Případy užití představují aktivity, které systém umožňuje provádět, a jsou znázorněny ovály v Use Case diagramu. Tyto ovály představují konkrétní možnosti využití v interakci se systémem.[11][12]

Hranice systému v Use Case diagramu vymezují rozsah systému tím, že určují, které prvky jsou jeho součástí a které jsou mimo něj. Pomáhají tak oddělit interní prvky systému od externích, například uživatelů. Tím se jasně vymezují zodpovědnosti a povinnosti systému a jeho interakce s okolním prostředím.[11][12]

Vztahy v případech užití znázorňují komunikaci mezi aktéry a případy užití. Zobrazují, jak aktéři a případy užití spolupracují na dosažení konkrétních cílů.[11][12]

Při tvorbě Use Case diagramu je důležité zachovat jednoduchost a přehlednost. Použití konzistentního jazyka zajišťuje srozumitelnost pro všechny. Pomocí klíčových hledisek systému lze identifikovat hlavní případy užití. [11][12]



Obrázek 1 Ukázka Případů užití, vlastní tvorba

3.1.2 Matice sledovatelnosti požadavků

Matice sledovatelnosti požadavků neboli Requirement Traceability Matrix (RTM) je nástroj používaný v projektovém řízení a vývoji softwaru k zajištění, zdali požadavky jsou zohledněny a řádně adresovány. RTM je matice jenž sleduje požadavky v rámci projektu od počátku až po dokončení. Umožňuje uživatelům identifikovat změny požadavků a jejich

příčiny. Je důležité, aby bylo zajištěno že všechny požadavky zákazníka jsou pokryty případy užití, a aby se minimalizovalo riziko, že požadavky budou během návrhu opomenuty. [17]

Pomocí základní šablony RTM můžeme vidět jednotlivá ID požadavků, které jsou řazeny řádkově a ID případů užití sloupcově, RTM, jedná se o důležitý nástroj, který slouží pro zajištění určité kvality a úplnosti návrhu. To následně může vést k vyšší spokojenosti zákazníků nebo také zlepšení vývojového procesu. [17]

3.1.3 Diagram tříd

Diagram tříd nebo také „Class“ diagram se využívá v softwarovém inženýrství pro vizuální znázornění struktury a vztahů jednotlivých tříd v systému. Diagram tříd je základním nástrojem pro konstrukci a vizualizaci objektově orientovaných systémů. Usnadňují práci ve fázi návrhu a dokumentace softwarových systémů a zajišťují přehled o návrhu systému.[13][14]

Diagram tříd zobrazuje třídy, jako boxy. V horní části se nachází název třídy, poté v prostřední části třídy se uvádí atributy neboli datové členy to jsou vlastnosti třídy například ID neboli identifikátor. Nakonec ve spodní části se nacházejí metody neboli operace, které může třída poskytovat, například Přihlášení(). Propojení tříd je pomocí čar, které znázorňují například asociace, dědičnost anebo závislost.[13][14]

Asociace představuje vztah mezi třídami, který mají nějakou formu propojení nebo spolupráce. [13][14]

Dědičnost představuje vztah, kdy jedna třída dědí vlastnosti nebo metody od druhé.[13][14]

Závislost představuje vztah mezi třídami, kdy jedno třída využívá atributy nebo funkce druhé třídy.[13][14]

V zobrazení tříd UML je viditelnost atributů a operací v rámci třídy reprezentována symboly. Znaménko plus (+) představuje veřejné prvky, znaménko minus (-) soukromé prvky a mřížka (#) chráněné prvky. Tyto symboly pomáhají definovat přístup k atributům a operacím v rámci třídy.[13][14]

Třídní diagramy hrají klíčovou roli v pochopení a dokumentaci struktury softwarového systému. Také slouží jako návod pro realizaci softwaru umožňujícího vývojářům tvořit kód dle požadavku. Diagram tříd také hraje důležitou roli ve vývoji, dokumentuje a komunikuje organizaci softwaru. [13][14]

3.1.4 Entity relační diagram

Entity relační schéma neboli Entity-Relationship (ER) model, jež je základem pro koncept databázového designu. Používá se k identifikaci entit a jejich atributů, jež následně budou rezervovány v databázi. Také slouží k popisu, jakým způsobem jsou entity vzájemně propojeny. [15]

ER diagramy reprezentují ER model v databázi. Jež je snadno převoditelné do tabulek. Zároveň poskytují vizualizaci objektů, jako jsou například auta, lidé, společnost a vztahy mezi nimi.[15]

Symboly v ER modelu:

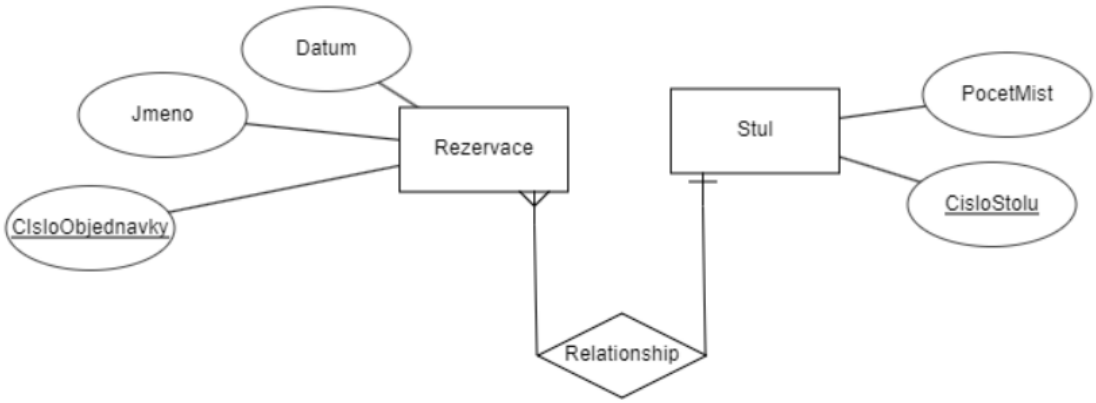
1. Obdélníky reprezentují entity. Entity mohou být fyzické objekty jako třeba dům, zaměstnanec, nebo s konceptuální existencí jako třeba firma a univerzitní předmět. Každá entita má vlastní klíčový atribut, jenž ji jednoznačně identifikuje. Tento klíčový atribut je reprezentován jako primární klíč v databázi.[15]
2. Elipsy reprezentují atributy, jež jsou vlastnostmi entit jako je třeba jméno nebo věk. Entita pak může mít několik atributů zároveň.[15]
3. Diamanty reprezentují vztahy mezi entitami. Tyto vztahy mohou být banální jako třeba: Zaměstnanec pracuje pro firmu. Nebo mohou být složitější: Student se dotazoval na několik předmětů.[15]
4. Čáry reprezentují vztahy mezi entitami a atributy například: Student má atribut „jméno“, jenž je spojen s entitou „student“[15]
5. Dvojité elipsy reprezentují vícehodnotové atributy. Tyto vícehodnotové atributy mohou mít více hodnot například: Telefonní číslo u osoby.[15]
6. Dvojité obdélníky reprezentují takzvané slabé entity. Slabé entity nemají vlastní klíčový atribut a závisí na nějaké jiné entitě.[15]

Kardinalita ER diagramu vyjadřuje počet vztahů nebo interakcí mezi entitami, například jeden k jednomu nebo jeden k mnoha. Zjednodušeně by se dalo říct, že kardinalita je poměr čísel vyjádřen v symbolech (jedna k jedné). Volitelnost u kardinality nám říká, jestli je vztah povinný nebo nepovinný. Jsou vztahy, které jsou na jedná straně povinné na druhé však už ne. Vztah (1:1), tento vztah určuje, že jedna entita má pouze jednu událost sdílenou s jinou entitou. Na příklad osoba a cestovní pas.[16]

Vztah (1:N) to znamená, že entita má událost, která se vyskytuje pouze jednou, avšak druhá entita může mít více než jedno opakování této události. Jeden zaměstnavatel může mít více zaměstnanců.[16]

Nakonec (M:N) tento vztah určuje to, že u obou entit se může stejná událost vyskytnou více, než jednou. Mnoho studentů se může zúčastnit mnoha kurzů a mnoho kurzů může mít mnoho studentů.[16]

ER modely jsou nejlepší pro modelování reálných objektů a jejich vztahů. To je důležité pro návrh databází. [16]



Obrázek 2 Ukázka ERD, vlastní tvorba

4 IMPLEMENTACE WEBU

Rozhodl jsem se vyvinout webové stránky s komplexní kombinací technologií, abych měl možnost rozvíjet více své zdatnosti, které zahrnují HTML, CSS a JavaScript pro strukturování a interaktivitu stránek, a také PHP a framework Bootstrap pro vytvoření sofistikovaného a uživatelsky přívětivého designu. Pro efektivní správu dat a jejich integraci do webové aplikace budu používat jazyk SQL, což mi umožní spolehlivou práci s databází a zabezpečí bezproblémové propojení mezi různými prvky systému.

4.1 HTML

HTML neboli HyperText Markup Language je značkovací jazyk pro tvorbu a strukturalizaci webových stránek. Kombinuje značkovací jazyk pro popis struktury a vzhledu webu s hypertextem pro propojení stránek. HTML umožňuje přidávat různé prvky, jako jsou obrázky, videa, zvukové soubory a hypertext, aby byl web interaktivnější a bohatý. [18][19]

Klíčové body HTML jsou, že je platformě nezávislý a snadný k naučení. Předdefinované značky a prvky se používají k určení vzhledu a struktury obsahu webových stránek. Struktura stránky HTML zahrnuje deklaraci typu dokumentu, kořenový element HTML, elementy head, p a tak dále.[18][19]

Výhody HTML jsou v tom, že lze přidávat obrázky, videa a zvuk do webových stránek. Také umožňuje začlenit hypertext k textu a zároveň je základní součástí webového vývoje a často se používá společně se stylovacími a skriptovacími jazyky.[18][19]

Nevýhodou HTML je, že může být omezený ve flexibilitě a funkčnosti ve srovnání s modernějšími alternativami.[18][19]

Tagy v HTML jinak řečeno značky umožňují definici toho, jak má být obsah na stránce zobrazen. Tyto tagy mohou obsahují otevírací a zavírací značku, ať už se jedná o obrázek, text nebo jiné vložené prvky. V HTML existují také atributy, které poskytují dodatečné informace o elementu a tím pomáhají s jeho následným stylováním anebo manipulací pomocí JavaScriptu.[18][19]

Obecně je jazyk HTML základním nástrojem pro každého, kdo se chce podílet na tvorbě webových stránek, protože tvoří základ pro návrh a strukturu stránek.[18][19]

```
<div class="center">
<a href="rezervace.php" class="btn rezervace-btn">Rezervace</a>
```

Obrázek 3 Ukázka kódu, vlastní tvorba

Na tomto obrázku je ukázka HTML kódu. Vidíme zde prvek `<div>`, který má atribut `class` pro pojmenování a stylování. Prvek `<div>` je kontejnerový prvek, který seskupuje ostatní prvky. Dále vidíme prvek `<a>`, který je odkazem na stránku `rezervace.php`, a také obsahuje atribut `class` pro stylování. [18][19]

4.2 CSS

CSS neboli Cascading Style Sheets je jazyk který hraje důležitou roli v moderním web designu, protože díky jeho schopnostem jsou vývojáři a návrháři schopni vytvářet atraktivní a přizpůsobitelné webové stránky. Používá se k definování prezentace dokumentů napsaných za pomocí jazyka HTML nebo XML.[20][21]

Prezentace je velmi ovlivněna použitím CSS, protože umožňuje ovlivňovat jak rozložení, tak vzhled webových stránek. Příkladem může být třeba změna písma, nebo změna pozice prvku na stránce.[20][21]

CSS též definuje skupiny stylů, které se následně používají na konkrétní prvky nebo skupiny prvků na webové stránce.[20][21]

Jsou tři techniky integrace CSS v HTML kódu.

1. Interní CSS což znamená, že CSS je definováno přímo v HTML dokumentu v sekci ‘<head>’ v rámci prvku ‘<style>’. Všechny styly nadefinované v této sekci budou následně aplikovány na prvky uvnitř daného HTML dokumentu. Toto řešení je vhodné k použití u menších nebo jednorázových stránek.[20][21]

```
<head>
    <style>
        h1 {
            color: blue;
            font-size: 3em;
        }
    </style>
</head>
```

Obrázek 4 Ukázka interního CSS, vlastní tvorba

2. Externí CSS je soubor, který je samostatný s příponou ‘.css’, jenž obsahuje všechny styly. Tyto soubory se následně k HTML kódu připojují za pomocí prvku ‘<link>’ v sekci ‘<head>’. Toto řešení je vhodné pro rozsáhlejší projekty, kde je potřeba oddělovat obsah od stylů a usnadnit si správu a úpravy stylů skrze různé stránky.[20][21]

```
<head>
    <link rel="stylesheet" href="styles.css">
</head>
```

Obrázek 5 Ukázka připojení externího CSS, vlastní tvorba

3. Inline CSS jsou přímo aplikovány na konkrétní prvku v HTML kódu za pomoci atributu ‘style’. Tento přístup se používá zejména, když chceme upravit stylování konkrétního prvku a nechceme použít jednu z předešlých metod.[20][21]

```
<h1 style="color: red; font-size: 2em;">Nadpis s inline stylem</h1>
```

Obrázek 6 Ukázka inline CSS, vlastní tvorba

```
/* Nastavení obrázku jako pozadí */
body {
    background-image: url('obrazky/Restaurace.png');
    background-size: cover;
    background-position: center;
    overflow: hidden;
    font-family: Arial, sans-serif;
}
```

Obrázek 7 Ukázka kódu, vlastní tvorba

Na tomto obrázku lze vidět ukázku CSS kódu, který nastavuje pro prvek `body` obrázek na pozadí pomocí `background-image`. Vlastnost obsahuje `url` s odkazem na obrázek `Restaurace.png`. Dále kód obsahuje `background-size: cover;`, což nastavuje obrázku pokrytí celé stránky, bez ohledu na jeho původní rozměry. `background-position: center;` nastavuje obrázek do středu prvku. Vlastnost `overflow: hidden;` zabraňuje zobrazení obsahu, který přesahuje rozměry prvku. `font-family: Arial, sans-serif;` nastavuje písmo v prvku `body` tak, že použije `Arial` jako hlavní volbu a `sans-serif` jako záložní volbu.

Kaskádování CSS je proces ve který určuje, jak se bude CSS aplikovat na HTML kód, zejména, když jsou definovány v různých místech nebo úrovních. Kaskádování určuje, jaký ze stylů bude mít přednost. [20][21]

Každý selektor má svou vlastní specificitu. Ta určuje to, jak se budou jednotlivé styly prioritizovat. Začíná to symbolem ' #', jež má nevyšší specificitu, což znamená, že „přebije" ostatní styly. Následně je třída 'class' ta má střední úroveň specifity. Poslední a nejnižší je typová specificita, kde stylujeme přímo prvek. [20][21]

CSS je zásadní, pokud chceme webu zajistit respozivitu. To znamená pohodlné zobrazování stánky na různých zařízeních od mobilních telefonů až po stolní počítače. Pomocí CSS lze vytvářet uživatelsky přívětivé a interaktivní webové stránky. Správné použití CSS je zásadní pro úspěšný webový design. [20][21]

4.3 JavaScript

JavaScript je dynamický, objektově orientovaný skriptovací jazyk, jenž se využívá k tvorbě interaktivního a dynamického obsahu na stránkách. Umožňuje využití složitých funkcí jako jsou interaktivní mapy, animace 2D/3D grafika, video a další. JavaScript zaznamenal výrazné změny v roce 2015, když se vydalo ECMAScript 2015, jenž přidalo mnoho nových funkcí.[22][23]

JavaScript je nejen jazyk pro webové stránky, ale používá se také v prostředích, jako je Node.js. Poskytuje programové ovládání hostitelských objektů a obsahuje standardní knihovny objektů, jako jsou Array, Date a Math. JavaScript lze rozšířit o další objekty, které slouží k různým účelům, například pro komunikaci s databází na serveru.[22][23]

Způsoby začlenění JavaScriptu do HTML kódu jsou interní a externí, podobně jako u CSS. Interní začlenění se provádí použitím značky `<script>` přímo v části `<body>` nebo `<head>`. Externí začlenění se provádí připojením externího souboru s příponou `.js` pomocí značky `<script>` v části `<head>` nebo `<body>`.[22][23]

JavaScript umožňuje vytvářet různé interaktivní a dynamické prvky na webových stránkách, jako jsou rozevírací nabídky, modální okna, zobrazení data a času, digitální hodiny, validace na straně klienta a dynamické chování tlačítek a odkazů.[22][23]

JavaScript nabízí mnoho výhod, včetně vytváření dynamických webových stránek a interaktivního uživatelského rozhraní. Je klíčový pro moderní webový vývoj, protože umožňuje vytvářet bohatší uživatelskou zkušenost. Znalost JavaScriptu je zásadní pro každého, kdo se chce věnovat vývoji webu.[22][23]

```
document.querySelector('.navigace a[href="#additionalContent"]').addEventListener('click', function (event) {
    event.preventDefault();
    document.getElementById('additionalContent').style.display = 'block';
    document.getElementById('showAllReservationsBtn').style.display = 'none';
});

document.querySelector('.navigace a[href="#showAllReservationsBtn"]').addEventListener('click', function (event) {
    event.preventDefault();
    document.getElementById('additionalContent').style.display = 'none';
    document.getElementById('showAllReservationsBtn').style.display = 'block';
});
```

Obrázek 8 Ukázka kódu, vlastní tvorba

Kód vyhledává v elementu s třídou „navigace“ odkaz „<a>“ s atributem „href“ nastaveným na „#additionalContent“. Jakmile je na tento odkaz kliknuto, přidá posluchač událostí pro událost „click“ pomocí „addEventListener“. Poté funkce zabraňuje přesměrování pomocí „event.preventDefault()“ a zobrazí prvek s ID „additionalContent“, zatímco skryje prvek s ID „showAllReservationsBtn“. Další část kódu provádí podobnou operaci, ale na odkazu s atributem „href“ nastaveným na „#showAllReservationsBtn“. Když je na tento odkaz kliknuto, funkce zobrazí prvek s ID „showAllReservationsBtn“ a skryje prvek s ID „additionalContent“.

4.4 PHP

PHP neboli Hypertext Preprocessor je skriptovací jazyk na straně serveru, který se používá převážně pro vývoj webových stránek. PHP je otevřený zdrojový kód a lze jej stáhnout a používat zdarma. PHP vytvořil v roce 1994 Rasmus Lerdrf jako Personal Home Page, ale nyní je známý jako Hypertext Preprocessor.[24][25]

Jazyk PHP je univerzální jazyk, který lze použít k vytváření nejrůznějších aplikací. Je multiplatformní a funguje ve všech hlavních operačních systémech. Podporuje také různé databáze, včetně MySQL, PostgreSQL, MS SQL a další. PHP je flexibilní a lze jej využít na straně serveru pro skriptování včetně vývoje dynamických webových stránek a webových aplikací.[24][25]

Hlavní výhodou PHP je jednoduchost. PHP je snadno naučitelné a začínající programátoři se s ním mohou rychle seznámit. Další výhodou PHP je rychlost. Stránky vytvořené pomocí PHP běží velmi rychle, což významně přispívá uživatelskému komfortu. V neposlední řadě je to stabilita.[24][25]

Za pomoci PHP lze vkládat kód přímo do jazyka HTML a vytvářet dynamické webové stránky. Lze jej použít k manipulaci s formuláři, vytváření dynamického obsahu, komunikaci s databází a interakcemi se servery. Za pomoci PHP jsou vytvořeny významné stránky

jako je například Facebook a Yahoo. PHP je dynamický jazyk, takže deklarace datových typů proměnných není potřeba. PHP je schopné pracovat s mnoha platformami a podporuje hlavní protokoly jako je například HTTP (HyperText Transfer Protocol) Basic.[24][25]

PHP je obecně důležitým nástrojem při vývoji webových stránek umožňuje totiž vývojářům vytvářet dynamické a interaktivní webové stránky. [24][25]

4.5 Bootstrap

Bootstrap je oblíbený framework pro tvorbu responzivních webových stránek, které jsou vhodné pro mobilní zařízení. Poskytuje základní koncepty, komponenty a techniky pro tvorbu webových stránek. Bootstrap umožňuje snadné vytváření návrhů, které se přizpůsobí různým zařízením. Použití je snadné stačí si stáhnout Bootstrap, přidat jej do projektu a použít předdefinované komponenty a styly. [26][27]

4.6 MySQL

MySQL je široce používaný open source systém pro správu relačních databází, který je založený na strukturovaném dotazovacím jazyku neboli SQL. MySQL je známý svou spolehlivostí, snadným použitím a škálovatelností. Často se používá k ukládání a zpracování dat ve webových aplikacích a je součástí softwarové sady LAMP, což je softwarový balíček používaný pro webový vývoj. L je pro Linux, což je operační systém. A je pro Apache, což je webový server. M je pro MySQL, což je relační databáze. P je pro PHP. MySQL je open-source, což znamená, že jej lze zdarma využívat a přizpůsobovat dle potřeb uživatele. Dále pak ukládá data do tabulek s řádky a sloupci. Využívá SQL pro dotazy a manipulaci s daty. MySQL je považován za spolehlivý a stabilní a zároveň je vhodný pro malé i velké projekty díky škálovatelnosti.[28][29][30]

MySQL je primární databází pro mnoho aplikací, včetně populárních webů, jakým je třeba YouTube, WordPress nebo Facebook. MySQL je spolehlivá volba pro správu dat a ukládání informací.[28][29]

MySQL umožňuje vytvářet a spravovat databáze, tabulky a záznamy. Podporuje různé datové typy, například textové, číselné, datumové a časové. Mezi pokročilé funkce patří transakce, uložené procedury, spouštěče a pohledy. [28][29]

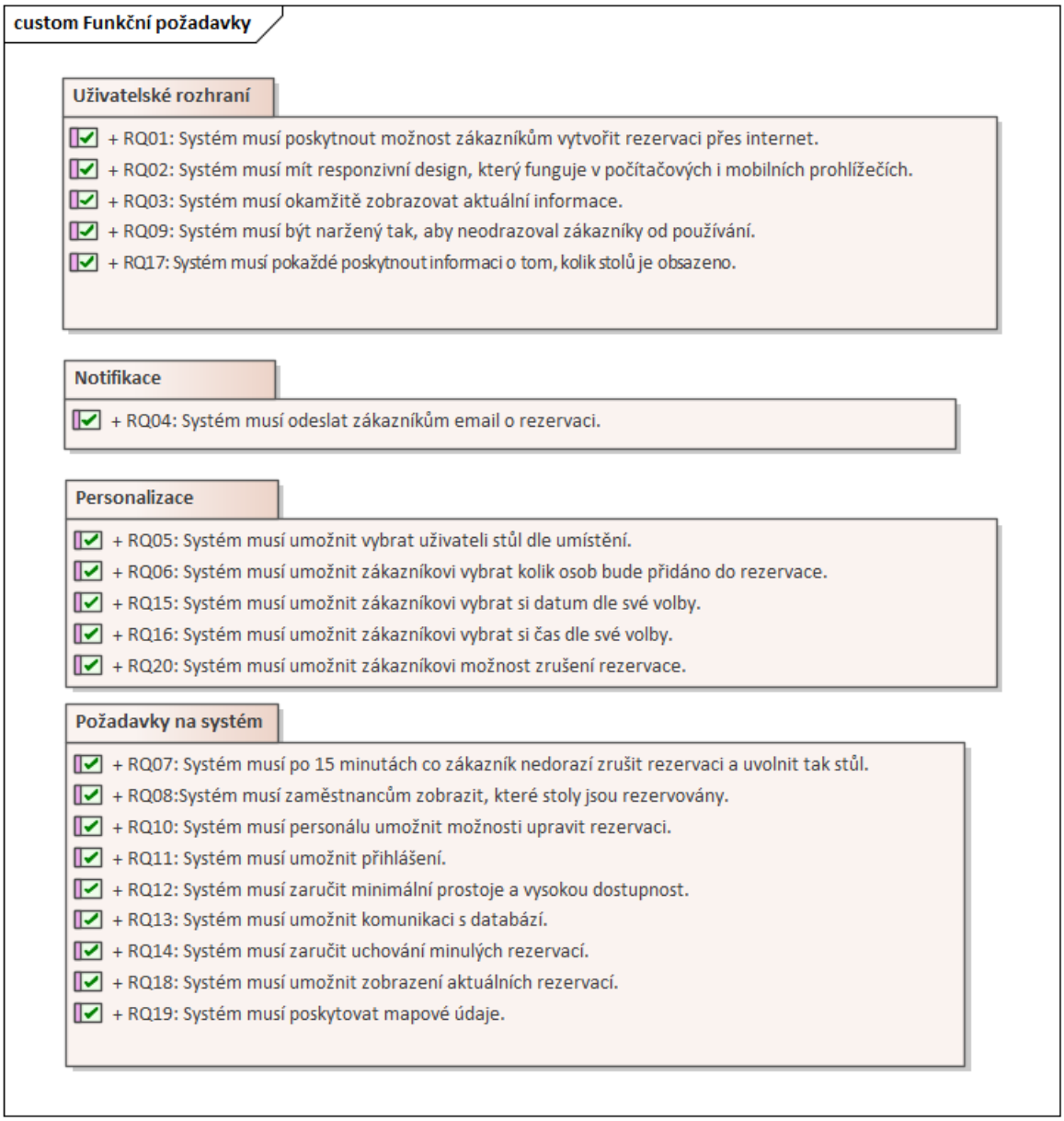
II. PRAKTICKÁ ČÁST

5 NÁVRH

Dnes je téměř nutné si před návštěvou restaurace udělat rezervaci. Mým cílem bylo vytvořit efektivní rezervační systém, který umožňuje zákazníkům snadno rezervovat konkrétní stůl podle jejich preferencí online, aniž by museli telefonovat. Systém také umožňuje zákazníkům zrušit rezervaci, pokud je to nutné, bez nutnosti kontaktovat restauraci telefonicky. Tímto způsobem se sníží počet hovorů, což usnadní práci obsluze a zlepší zákazníkům zážitek z rezervace. Hlavním cílem bylo navrhnout systém, který by zákazníkům umožnil jednoduše vytvářet a rušit rezervace. Dále jsem se zaměřil na to, aby zaměstnanci měli přehled o všech vytvořených rezervacích a mohli je snadno spravovat.

5.1 Funkční požadavky na systém

Nejprve bylo zapotřebí vytvořit funkční a nefunkční požadavky na systém.



Obrázek 9 Funkční požadavky na systém, vlastní tvorba

5.1.1 Uživatelské rozhraní

RQ01: Správné vytvoření rezervace je pro systém klíčová vlastnost, bez které by aplikace nemohla fungovat.

RQ02: Responsivní webdesign je klíčový pro to, aby zákazníci mohli využívat služby webové aplikace nejen na stolních počítačích, ale i na jiných zařízeních.

RQ03: Zobrazování aktuálních informací je pro systém zásadní, protože zákazník tak má neustále přístup k nejnovějším údajům.

RQ09: Systém je navržen intuitivně tak, aby zákazníky neodradil svou složitostí.

RQ17: Systém musí zobrazovat, které stoly jsou aktuálně obsazeny k danému datu a času.

5.1.2 Notifikace

RQ04: Odeslání potvrzení o provedené rezervaci je důležité pro zákazníka, protože tím získá záznam o rezervaci ve své e-mailové schránce.

5.1.3 Personalizace

RQ05: Zákazník musí mít možnost vybrat si stůl podle svého výběru na základě půdorysu restaurace.

RQ06: Zákazník musí mít možnost zvolit kolik osob bude rezervace obsahovat.

RQ15: Zákazník musí mít možnost si vybrat datum dle své volby.

RQ16: Zákazník musí mít možnost si vybrat čas příchodu a odchodu dle své volby.

RQ20: Systém musí umožnit zákazníkovi zrušit rezervaci bez nutnosti telefonického hovoru.

5.1.4 Požadavky na systém

RQ07: Pro plynulý chod je nezbytné, aby systém automaticky zrušil objednávky, pokud zákazník nedorazí.

RQ08: Zaměstnanec musí mít možnost si zobrazit rezervace.

RQ10: Zaměstnanec má možnost upravit rezervaci za pomocí formuláře.

RQ11: Zaměstnanec musí mít možnost se do systému přihlásit.

RQ12: Systém musí být plynulý a dostupný.

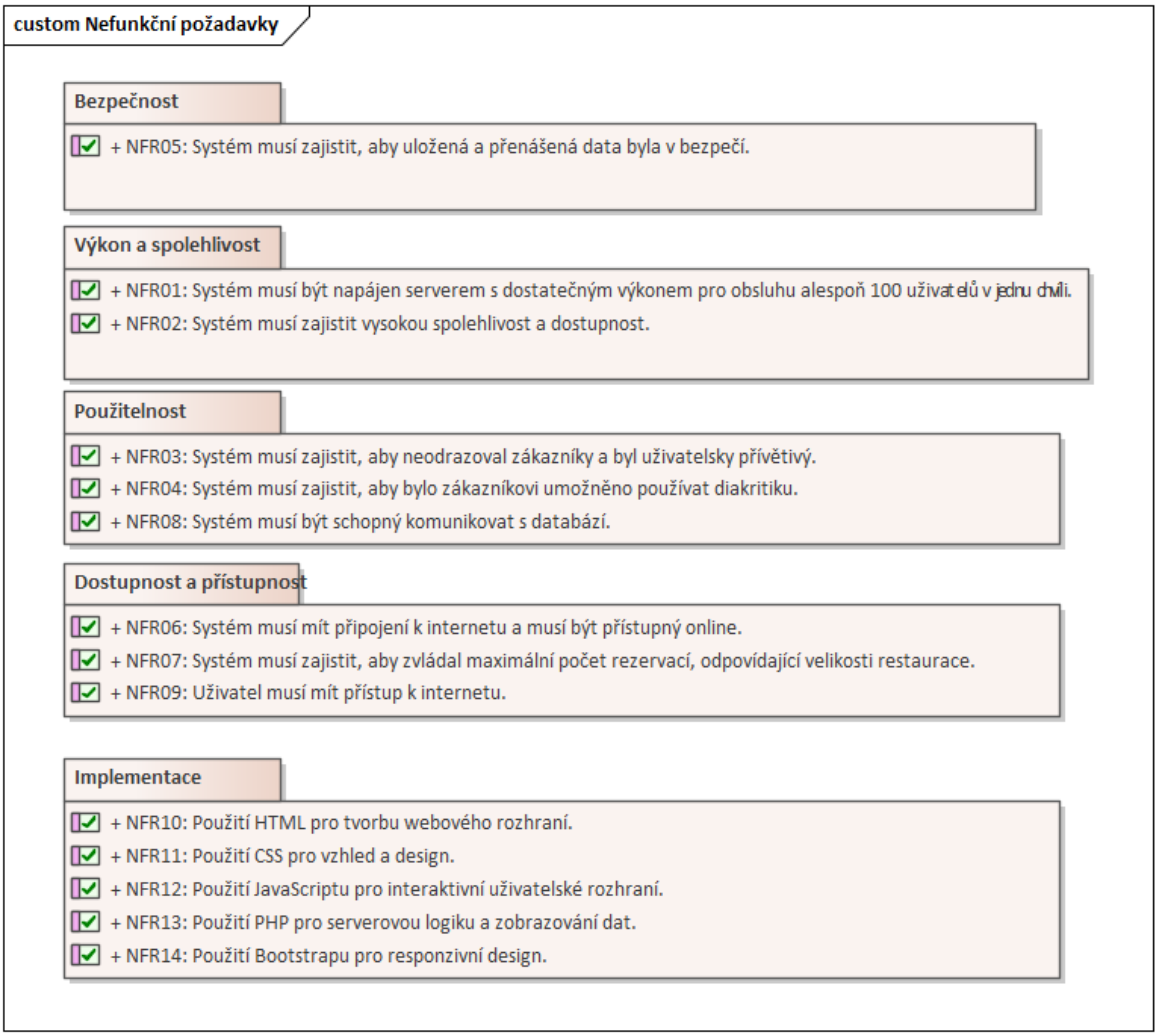
RQ13: Systém musí komunikovat s databází.

RQ14: Systém zobrazuje a uchovává všechny rezervace nejen aktuální.

RQ18: Zákazník musí mít možnost si vybrat stůl dle svého výběru.

RQ19: Systém musí zobrazovat mapu s místem, kde se rezervace nachází.

5.2 Nefunkční požadavky



Obrázek 10 Nefunkční požadavky, vlastní tvorba

5.2.1 Bezpečnost

NFR05: Systém musí zajistit, aby data přenášená na veřejně přístupných stránkách byla bezpečná a řádně ukládána.

5.2.2 Výkon a spolehlivost

NFR01: Systém musí být uložen na serveru s dostatečnou kapacitou.

NFR02: Systém musí být spolehlivý a dostupný všem uživatelům.

5.2.3 Použitelnost

NFR03: Systém musí být intuitivní a jednoduchý k použití i pro nové zákazníky.

NFR04: Systém musí zákazníkovi použít diakritiku při tvoření rezervace.

NFR08: Systém musí načítat a ukládat data do databáze.

5.2.4 Dostupnost a přístupnost

NFR06: Systém musí být přístupný online pro zákazníky.

NFR07: Systém musí zajistit, aby zvládl minimálně 50 rezervací v jeden čas.

NFR09: Uživatel musí být připojen k internetu, aby mohl mít přístup na stránky.

5.2.5 Implementace

NFR10: Systém musí být tvořen pomocí HTML.

NFR11: Systém musí být tvořen pomocí CSS.

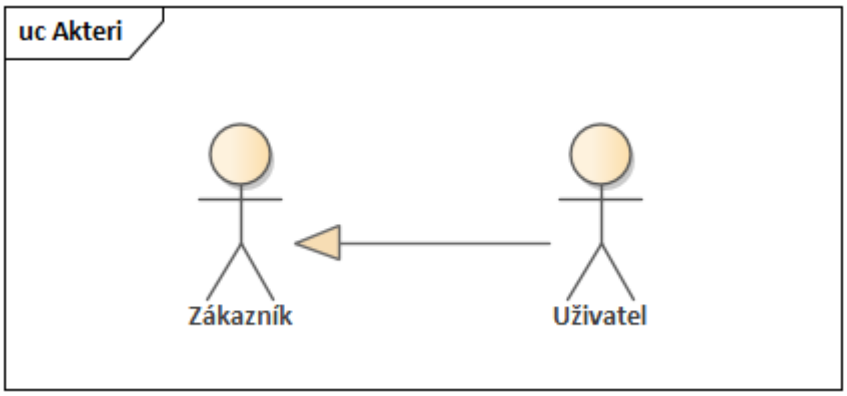
NFR12: Systém musí být tvořen pomocí JavaScriptu.

NFR13: Systém musí být tvořen pomocí PHP.

NFR14: Systém musí být tvořen pomocí Bootstrapu.

5.3 Aktéři

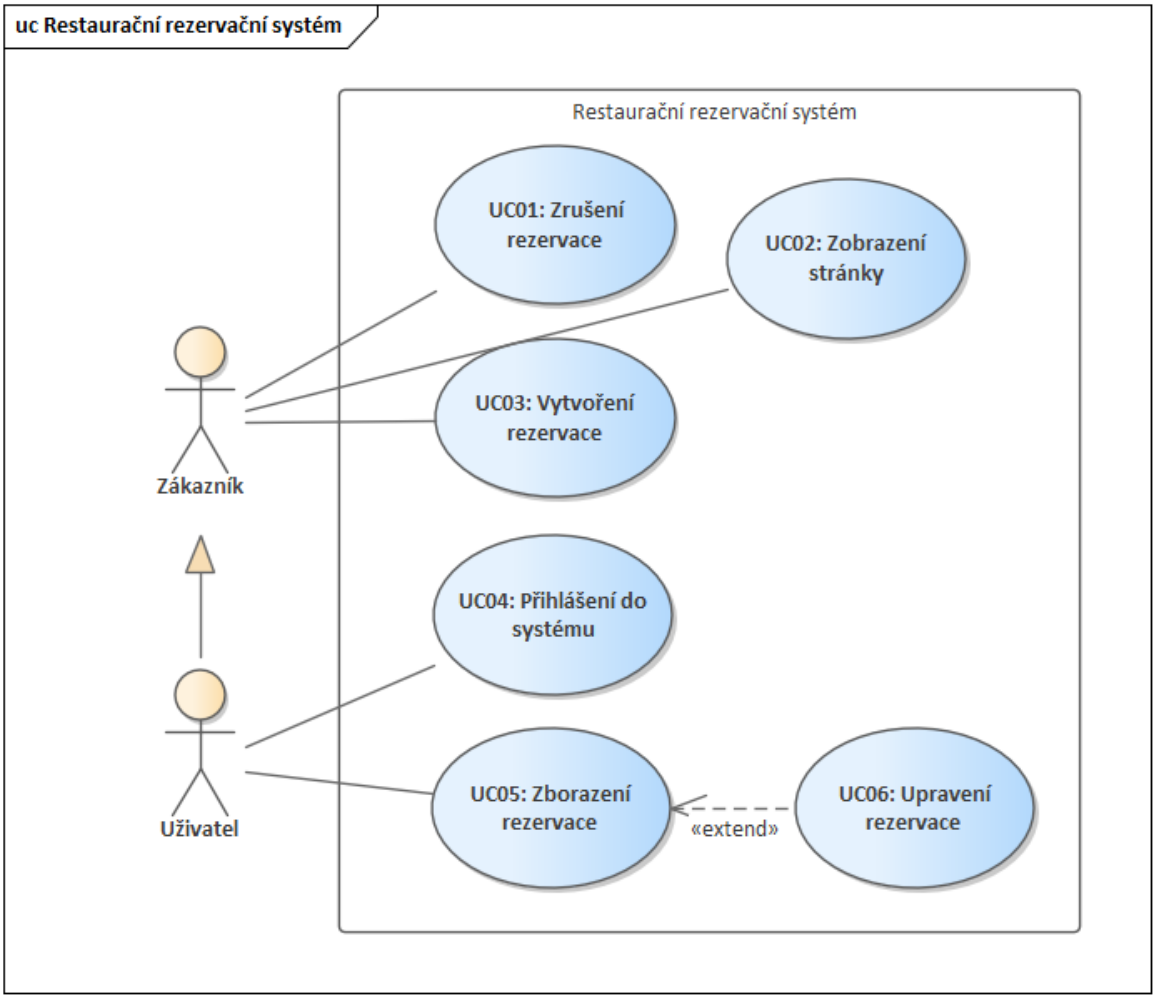
Systém mohou využívat dva typy uživatelů. Prvním z nich je „Zákazník“, který používá webovou aplikaci. Druhým typem je „Uživatel“, ten reprezentuje jakéhokoliv zaměstnance podniku, který je oprávněn využívat systém.



Obrázek 11 Aktéři, vlastní tvorba

5.4 Případy užití

V modelu lze vidět dva aktéry. Aktéra „Zákazník“ a aktéra „Uživatel“ tito aktéři mají přístup k jednotlivým případům užití, které budou specifikovány v podkapitole.



Obrázek 12 Případy užití, vlastní tvorba

5.4.1 Scénáře případů užití

Název: Zrušení rezervace		
ID: UC01		
Charakteristika: Zákazník bude chtít zrušit rezervaci		
Primární aktér: Zákazník		
Hlavní scénář:		
Krok	**Aktér**	**Popis**
1	Zákazník	Zákazník se požaduje zrušit rezervaci.
2	Systém	Systém vyzve zákazníka k zadání čísla rezervace pro zrušení.
3	Zákazník	Zákazník zadá číslo rezervace a potvrdí.
4	Systém	Systém zruší rezervaci.

Tabulka 1 UC01 Zrušení rezervace, vlastní tvorba

Název: Zobrazení stránky		
ID: UC02		
Charakteristika: Zákazník bude chtít zobrazit stránky restaurace		
Primární aktér: Zákazník		
Hlavní scénář:		
Krok	**Aktér**	**Popis**
1	Zákazník	Zákazník přijde na stránku.
2	Systém	Systém zákazníkovi zobrazí stránku.
3	Zákazník	Jestliže zákazník přejde na stránku Jídelníček.
4	Systém	Systém zobrazí stránku Jídelníček.
5	Zákazník	Jestliže zákazník přejde na stránku O restauraci.
6	Systém	Systém zobrazí stránku O restauraci.
7	Zákazník	Jestliže zákazník přejde na stránku Rezervace.
8	Systém	Systém zobrazí stránku Rezervace.
9	Zákazník	Jestliže zákazník přejde na stránku Zrušení rezervace.
10	Systém	Systém zobrazí stránku zrušení rezervace.
11	Zákazník	Jestliže zákazník přejde na stránku Domů.
12	Systém	Systém zobrazí stránku Domů.

Tabulka 2 UC02 Zobrazení stránky, vlastní tvorba

<table>
<tr><td colspan="3">Název: Vytvoření rezervace</td></tr>
<tr><td colspan="3">ID: UC03</td></tr>
<tr><td colspan="3">Charakteristika: Zákazník bude chtít vytvořit rezervaci</td></tr>
<tr><td colspan="3">Primární aktér: Zákazník</td></tr>
<tr><td colspan="3">Hlavní scénář:</td></tr>
<tr><td>Krok</td><td>Aktér</td><td>Popis</td></tr>
<tr><td>1</td><td>Zákazník</td><td>Zákazník klikne na stránku Rezervace</td></tr>
<tr><td>2</td><td>Systém</td><td>Systém zobrazí stránku Rezervace.</td></tr>
<tr><td>3</td><td>Zákazník</td><td>Zákazník vyplní potřebné informace (Jméno, Email, Čas příchodu, Čas odchodu, Datum).</td></tr>
<tr><td>4</td><td>Systém</td><td>Systém vyzve zákazníka k výběru stolu.</td></tr>
<tr><td>5</td><td>Zákazník</td><td>Vybere si stůl a přejde k přehledu rezervace.</td></tr>
<tr><td>6</td><td>Systém</td><td>Systém zobrazí souhrn rezervace.</td></tr>
<tr><td>7</td><td>Zákazník</td><td>Zákazník potvrdí rezervaci.</td></tr>
<tr><td>8</td><td>Systém</td><td>Systém zaregistruje rezervaci a odešle ji emailem.</td></tr>
<tr><td colspan="3">Alternativní scénář: UC03-4A, UC03-4B</td></tr>
</table>

Tabulka 3 UC03 Vytvoření rezervace, vlastní tvorba

Název: Alternativní scénář k UC03 – Zákazník nezadá potřebná data.		
ID: UC03-4A		
Charakteristika: Systém upozorní zákazníka na chybějící dat.		
Alternativní scénář:		
Krok	**Aktér**	**Popis**
1	Systém	Upozorní zákazníka na to, že některá data chybí.
2	Zákazník	Zákazník vyplní chybějící data.

Tabulka 4 UC03-4A Alternativní scénář k UC03-4A, vlastní tvorba

Název: Alternativní scénář k UC03 – Zákazník si uvědomí, že zadal špatná data.		
ID: UC03-4B		
Charakteristika: Uživatel chce opravit špatně zadaná data.		
Alternativní scénář:		
Krok	**Aktér**	**Popis**
1	Zákazník	Chce-li se zákazník vrátit zpátky.
2	Systém	Systém zákazníka přemístí.
3	Zákazník	Zákazník opraví data.
4	Systém	Systém uloží upravená data.

Tabulka 5 UC03-4B Alternativní scénář k UC03-4B, vlastní tvorba

Název: Přihlášení do systému		
ID: UC04		
Charakteristika: Zaměstnanec se bude chtít přihlásit do sytému		
Primární aktér: Zaměstnanec		
Hlavní scénář:		
Krok	**Aktér**	**Popis**
1	Zaměstnanec	Zaměstnanec přejde na stánku s přihlášením.
2	Systém	Systém zobrazí stránku s přihlašovacím formulářem.
3	Zaměstnanec	Zaměstnanec zadá přihlašovací údaje (Jméno, Heslo).
4	Systém	Pokud jsou údaje správné systém přihlásí uživatele
Alternativní scénář: UC04-4A		

Tabulka 6 UC04 Přihlášení do systému, vlastní tvorba

Název: Alternativní scénář k UC04 – Nekorektní vstup dat		
ID: UC04-4A		
Charakteristika: Systém upozorní zaměstnance na nekorektní vstup dat.		
Alternativní scénář:		
Krok	**Aktér**	**Popis**
1	Systém	Upozorní uživatele o neúspěšném pokusu o přihlášení.
2	Zaměstnanec	Zaměstnanec se znovu přihlásí

Tabulka 7 UC04-4A Alternativní scénář k UC04, vlastní tvorba

Název: Zobrazení rezervací		
ID: UC05		
Charakteristika: Uživatel chce zobrazit rezervace		
Primární aktér: Uživatel		
Hlavní scénář:		
Krok	**Aktér**	**Popis**
1	Zaměstnanec	Jestliže zaměstnanec zvolí všechny rezervace.
2	Systém	Systém zobrazí všechny rezervace.
3	Zaměstnanec	Jestliže zaměstnanec zvolí aktuální rezervace.
4	Systém	Systém zobrazí aktuální rezervace.

Tabulka 8 UC05 Zobrazení rezervací, vlastní tvorba

Název: Upravení rezervace		
ID: UC06		
Charakteristika: Uživatel bude chtít upravit rezervaci.		
Primární aktér: Uživatel		
Hlavní scénář:		
Krok	**Aktér**	**Popis**
1	Zaměstnanec	Zaměstnanec zvolí upravit rezervaci.
2	Systém	Systém zobrazí detaily rezervace v editovacím okně.
3	Zaměstnanec	Zaměstnanec provede požadované změny.
4	Systém	Systém změny uloží

Tabulka 9 UC06 Upravení rezervace, vlastní tvorba

5.5 Matice vztahů

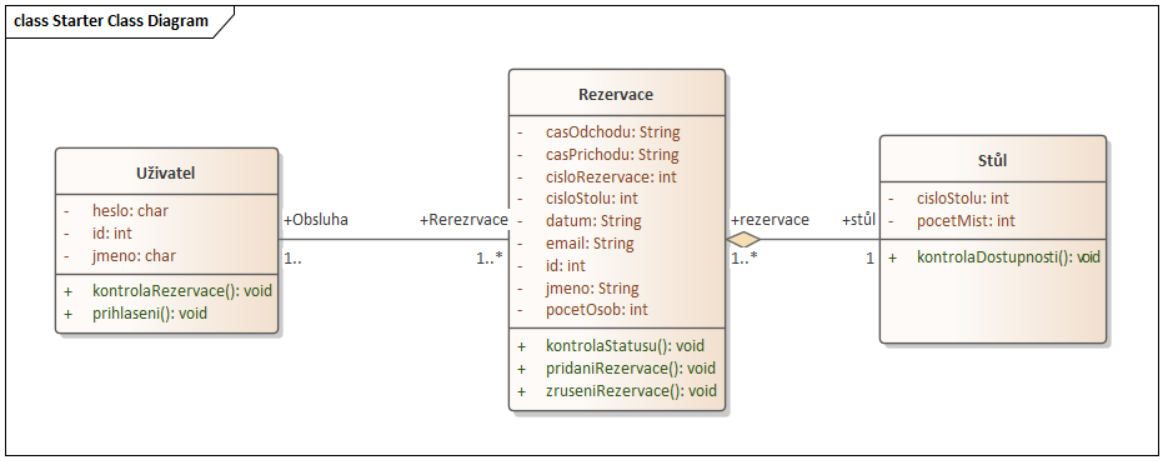
V této matici vidíme nahoře ve sloupcích případy užití a v levém sloupci v řádcích jsou funkční požadavky, každý funkční požadavek musí být řešen alespoň jedním z případů užití.

Source \ Target	UC01: Zrušení rezervace	UC02: Zobrazení stránky	UC03: Vytvoření rezervace	UC04: Přihlášení do systému	UC05: Zobrazení rezervací	UC06: Upravení rezervace
RQ01: Systém musí poskytnout možnost zákazníkům vytvořit rezervaci přes internet.			←			
RQ02: Systém musí mít responzivní design, který funguje v počítačových i mobilních prohlížečích.			←			
RQ03: Systém musí okamžitě zobrazovat aktuální informace.			←			
RQ04: Systém musí odeslat zákazníkům email o rezervaci.			←			
RQ05: Systém musí umožnit vybrat uživateli stůl dle umístění.			←			
RQ06: Systém musí umožnit zákazníkovi vybrat kolik osob bude přidáno do rezervace.			←			
RQ07: Systém musí po 15 minutách co zákazník nedorazí zrušit rezervaci a uvolnit tak stůl.	←					
RQ08:Systém musí zaměstnancům zobrazit, které stoly jsou rezervovány.			←			
RQ09: Systém musí být naržený tak, aby neodrazoval zákazníky od používání.			←			
RQ10: Systém musí personálu umožnit možnosti upravit rezervaci.						←
RQ11: Systém musí umožnit přihlášení.			←			
RQ12: Systém musí zaručit minimální prostoje a vysokou dostupnost.	←	←	←	←		←
RQ13: Systém musí umožnit komunikaci s databází.	←		←	←		←
RQ14: Systém musí zaručit uchování minulých rezervací.					←	
RQ15: Systém musí umožnit zákazníkovi vybrat si datum dle své volby.			←			
RQ16: Systém musí umožnit zákazníkovi vybrat si čas dle své volby.			←			
RQ17: Systém musí pokaždé poskytnout informaci o tom, kolik stolů je obsazeno.			←			
RQ18: Systém musí umožnit zobazení aktuálních rezervací.					←	
RQ19: Systém musí poskytovat mapové údaje.		←				
RQ20: Systém musí umožnit zákazníkovi možnost zrušení rezervace.	←					

Obrázek 13 Matice, vlastní tvorba

5.6 Class diagram

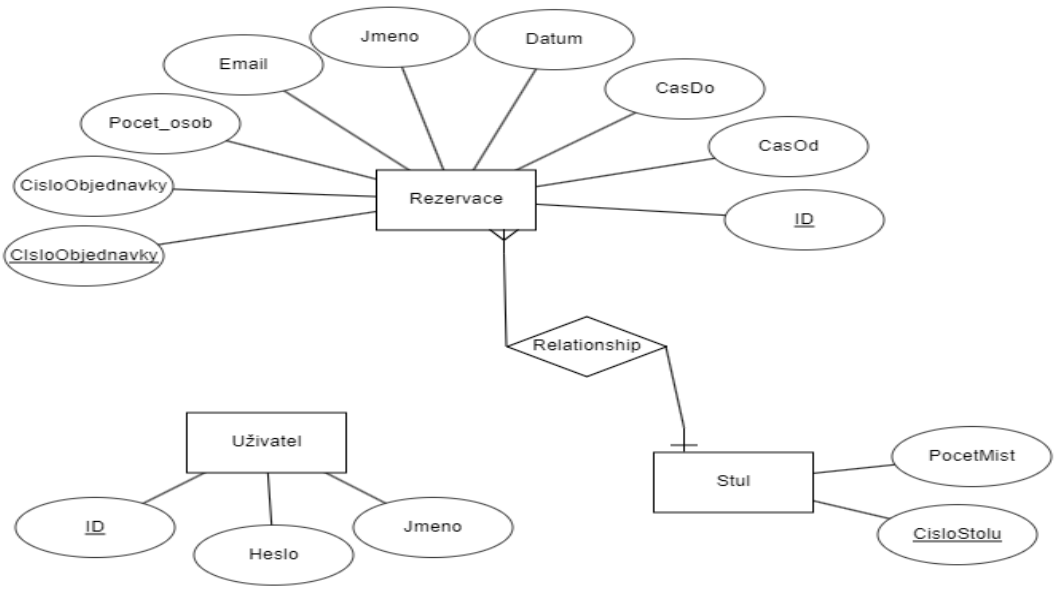
Diagram tříd se skládá ze dvou aktérů, jenž reprezentují tabulky Uživatel a Zákazník. Tito aktéři pracují s třídou rezervace, jenž ještě pracuje s třídou stůl.



Obrázek 14 Diagram tříd, vlastní tvorba

5.7 ER Diagram

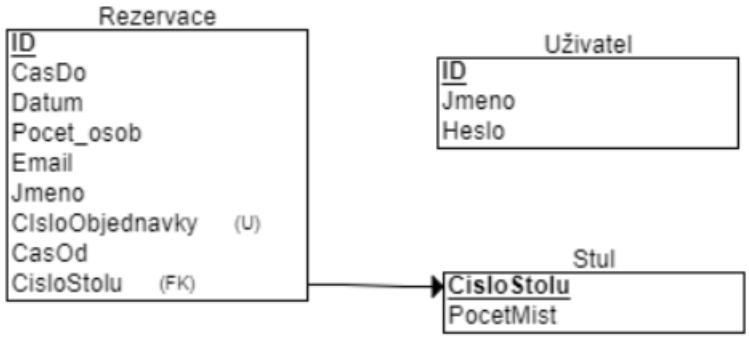
ER Diagram znázorňuje, jakým způsobem je realizovaná databáze, se kterou pracuje webová aplikace. Tento návrh zobrazuje, jaké atributy bude mít jaká tabulka a jaké vazby budou mezi entitami. „Relationship“ na obrázku reprezentuje vazbu mezi Rezervací a stolem. Tento vztah je vazba u „Rezervace“ je to jedna a u „Stul“ je to many.



Obrázek 15 ER Diagram, vlastní tvorba

5.8 Schéma databáze

Cílem tohoto schématu je přiblížit co nejvíce vzhled databáze pro implementaci. V návrhu se nachází tabulka „Rezervace“ s „ID“ s datovým typem „INT“, „CasDo“ a „CasOd“ mají datový typ „TIME“, „Datum“ má datový typ „DATE“, „Jmeno“ a „Email“ mají datový typ „VARCHAR(100)“ a „CisloObjednavky“, „CisloStolu“ mají také hodnotu „INT“. Zároveň u objednávky je „(U)“, což znamená, že je to unikátní hodnota a nemůže se opakovat. A u „CisloStolu“ je „(FK)“, což znamená, že tato hodnota je cizí klíč z tabulky „Stul“, kde má také datový typ „INT“. Dále se v této tabulce nachází „PocetMist“ se stejným datovým typem, jako „CisloStolu“, což je zároveň i primárním klíčem pro tabulku. Jako poslední lez vidět tabulku „Uživatel“. Tato tabulka má „ID“, což je primární klíč pro danou tabulku. Dále pak má „Jmeno“ a „Heslo“. Oba tyto atributy mají datový typ „VARCHAR“, „Jméno“ má hodnotu 50 a „Heslo“ 100.



Obrázek 16 Databáze, vlastní tvorba

6 IMPLEMENTACE

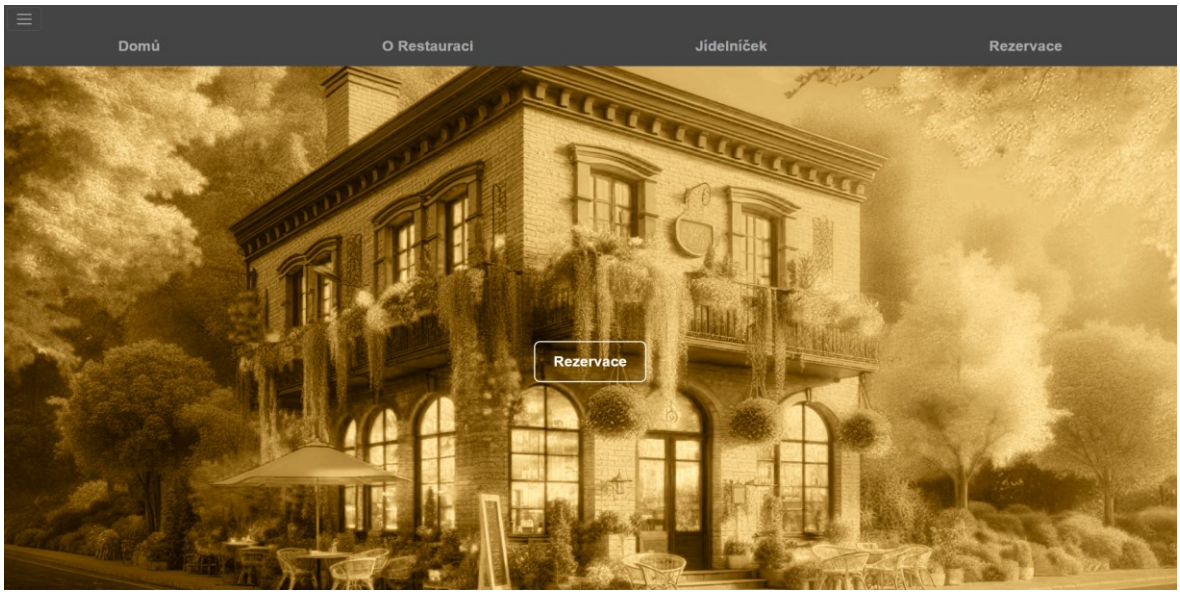
Webová aplikace byla vytvořena ve Visual Studio Code pomocí HTML, CSS, JavaScriptu, Bootstrapu a PHP. Provoz webové aplikace je zajištěn prostřednictvím XAMPP, jenž funguje jako místní vývojové prostředí pro webové aplikace. XAMPP je vybaven Apache pro provoz webového serveru, MySQL pro správu databází a PHP pro zpracování skriptovacích jazyků. To umožňuje vývoj a testování webové aplikace na vlastním zařízení, které simuluje serverové prostředí. Používám také knihovnu jQuery k odesílání a ukládání dat na server pomocí AJAXu. Při tvorbě webové aplikace jsem využíval ChatGPT. [31]

6.1 Vzhled

Vzhledem k tomu, že tato stránka byla vytvořena pro fiktivní restauraci tak, aby její vzhled co nejvíce vyhovoval mně osobně. Vzhled této webové aplikace byl navržen v minimalistickém stylu. Tento styl byl vybrán, protože je mi velmi blízký a domnívám se, že minimalistické stránky tolik neodrazují zákazníky nadbytkem aktivit, které se na nich odehrávají.

6.1.1 domu.php

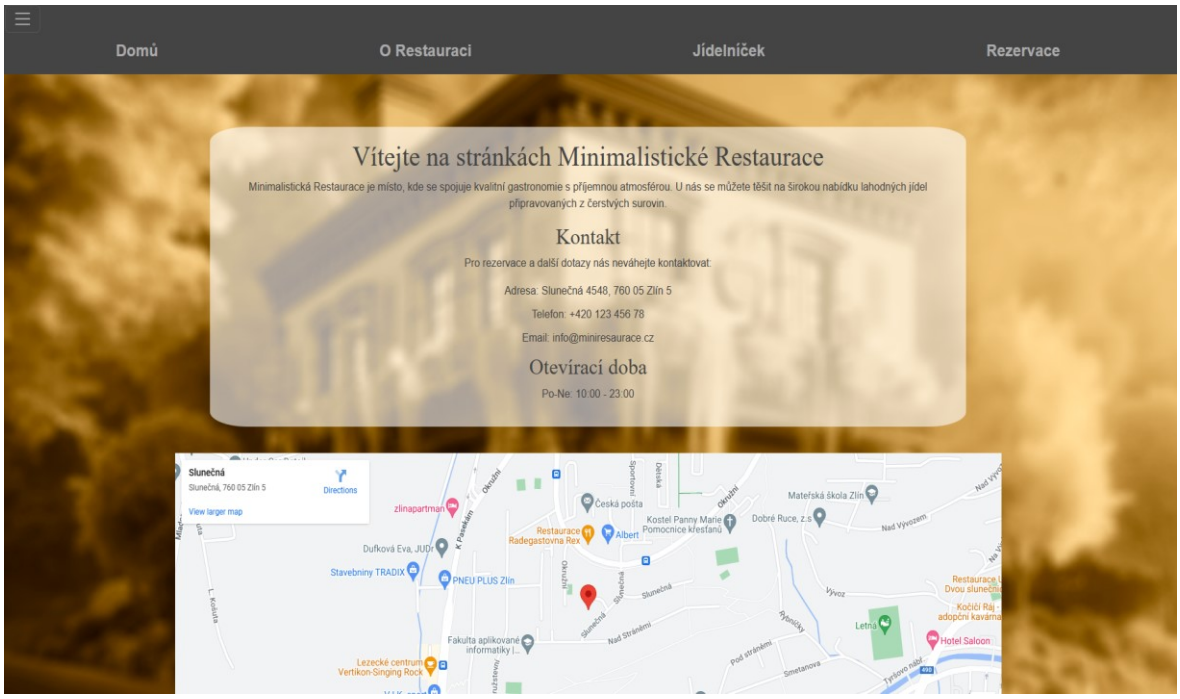
Tato stránka je úvodní stránkou pro webovou aplikaci Minimalistické restaurace. V horní části se nachází navigace s možnosti zobrazení jiných stránek, než je tato. Veprostřed stránky se nachází tlačítko „Rezervace“, které při stisknutí přesměruje zákazníka na stránku „rezervace.php“. Obrázek na pozadí byl generován pomocí umělé inteligence. (Copilot, M. Abstraktní umělecké dílo. 2024) [32]



Obrázek 17 Úvodní stránka, vlastní tvorba

6.1.2 restaurace.php

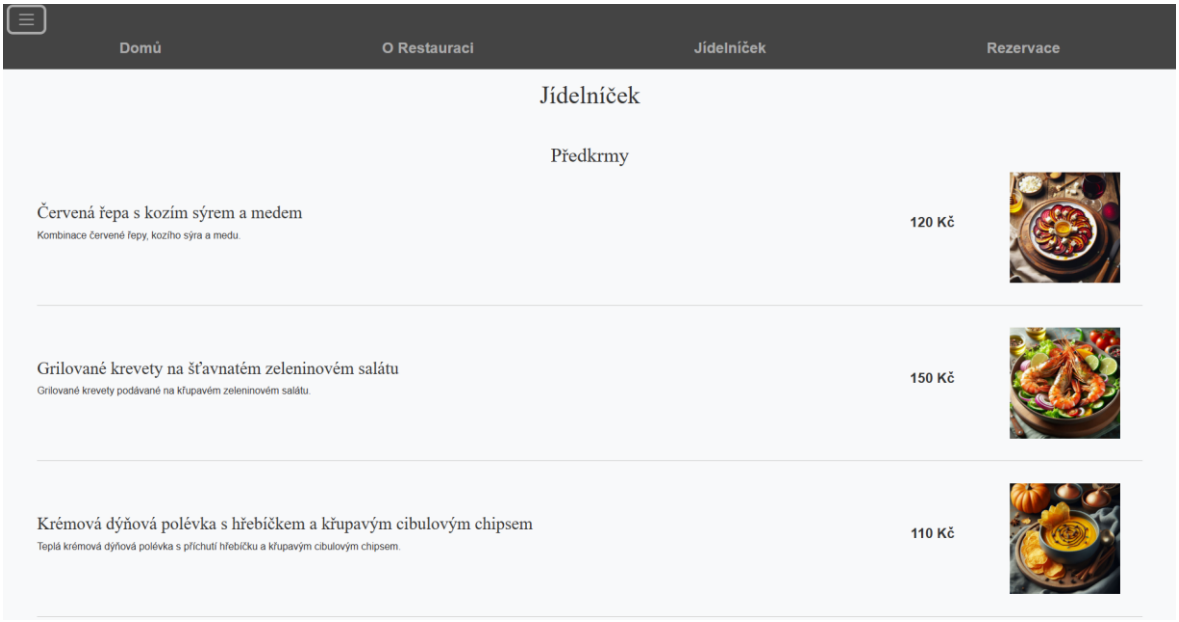
Stránka „restaurace.php“, na webové stránce označená, jako „O Restauraci“, a je určena k poskytnutí potřebných informací o restauraci zákazníkům. Zákazník zde může najít kontakt na restauraci, který lze použít k rezervaci stolu pomocí mobilního telefonu. K dispozici je také e-mailová adresa. Stránka dále obsahuje polohu restaurace zobrazenou na mapě, kterou lze přibližovat nebo oddalovat podle potřeby. Tato mapa může také pomoci zákazníkovi naplánovat cestu do restaurace. Nakonec je zde uvedena i otevírací doba restaurace. Obrázek na pozadí této stránky byl generován pomocí umělé inteligence. (Copilot, M. Abstraktní umělecké dílo. 2024)



Obrázek 18 O Restauraci, vlastní tvorba

6.1.3 jidelnicek.php

Stránka Jídelníček má za cíl informovat zákazníky o nabídce pokrmů, které jsou v nabídce Minimalistické restaurace. Na této stránce se nacházejí Předkrmy, Hlavní jídla a Dezerty. Vše je doplněno ilustračními obrázky. Vedle obrázků je také uvedena cena, kterou zákazník zaplatí, pokud si bude chtít objednat dané jídlo. Obrázky na této stránce byly vygenerovány pomocí umělé inteligence. (Copilot, M. Abstraktní umělecké dílo. 2024)

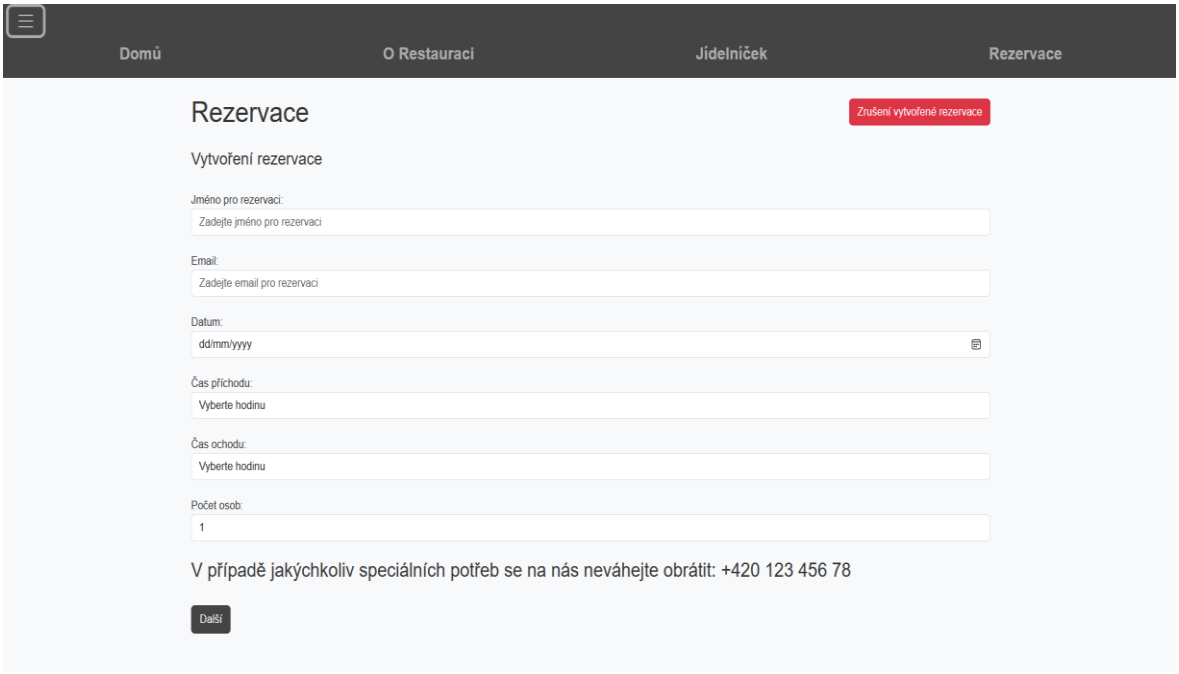


Obrázek 19 Jídelníček, vlastní tvorba

6.1.4 rezervace.php

Stránka rezervace je považována za klíčový prvek ovlivňující celkovou funkčnost webové aplikace. Na této stránce je zákazníkovi umožněno vytvořit rezervaci. Po něm jsou vyžadovány informace, jako je jméno pro identifikaci zákazníka při příchodu do restaurace, dále e-mail, datum, čas příchodu, čas odchodu a počet osob. Všechna tato pole jsou považována za nezbytná pro správné vytvoření rezervace.

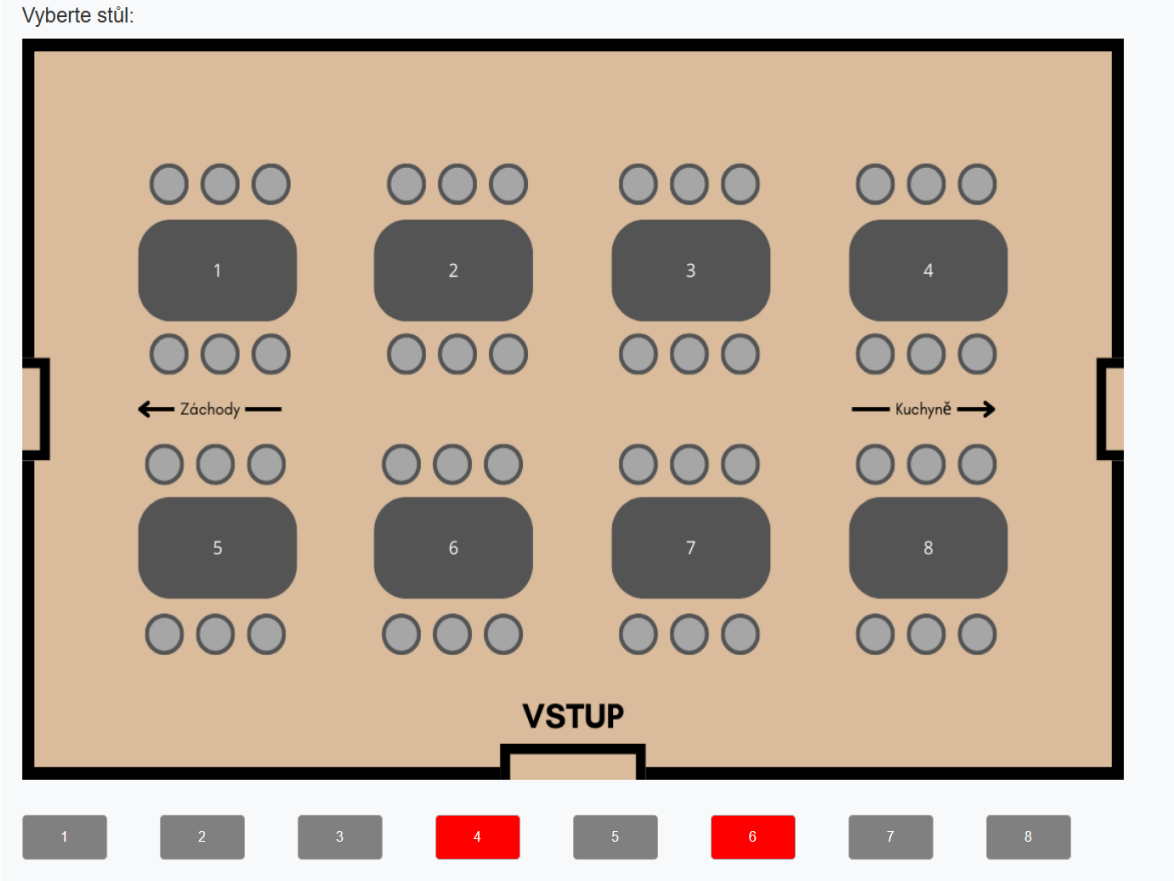
Na stránce se vedle navigace také v pravém horním rohu nachází tlačítko „Zrušení vytvořené rezervace“, které zákazníka přenese na stránku umožňující zrušení již vytvořené rezervace. Pod formulářem je uveden text, který vybízí zákazníky k tomu, aby v případě specifických požadavků, jako je rezervace pro více osob, než je povoleno, kontaktovali restauraci telefonicky. Pod tímto textem se nachází tlačítko „Další“, které zobrazuje zákazníkovi další krok v rámci jedné stránky. Stránka také sama provádí kontrolu, aby nebylo zákazníkem zadáno datum nebo čas v minulosti, pokud k tomu dojde, zákazník je na to upozorněn a není mu umožněn přechod do dalšího kroku.



Obrázek 20 Rezervace 1. část, vlastní tvorba

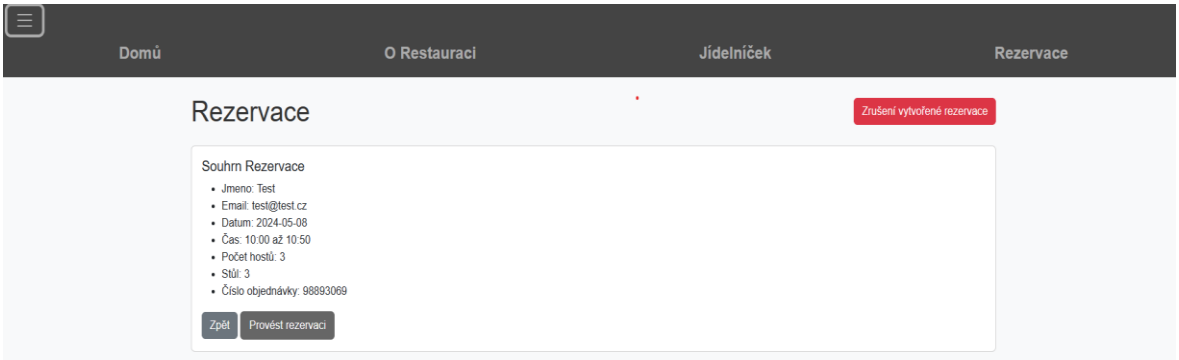
Další krok, který má být proveden zákazníkem, je výběr stolu. Stůl je vybírán zákazníkem hned pod půdorysem restaurace, který může být nápomocný při výběru vhodného místa pro konkrétního zákazníka. Čísla jedna až osm představují jednotlivé stoly, které mohou být upraveny pro potřeby rezervace podle požadavků zákazníka. Pokud je stůl zvýrazněn červeně, znamená to, že je již obsazen pro dané datum a hodinu. Pokud se zákazník pokusí vybrat tento stůl a přejít do dalšího kroku, bude upozorněn systémem, že tato akce není možná, protože je stůl obsazen, a zákazník zůstane na této stránce. Může si tedy vybrat neobsazený stůl. Poté bude systémem povolen přechod do dalšího kroku.

Tlačítko „Další“ je umístěno hned pod nabídkou stolů, ze kterých si může zákazník vybrat. Nachází se zde také tlačítko „Zpět“ hned vedle tlačítka „Další“ a toto tlačítko přemístí zákazníka zpět na první krok, pokud chce upravit nějaká data. Při návratu zpět na stránku s formulářem jsou jeho předvyplněná data již ve formuláři, takže zákazník může provést pouze úpravy tam, kde je to potřeba, a není nutné vyplňovat celý formulář znovu.



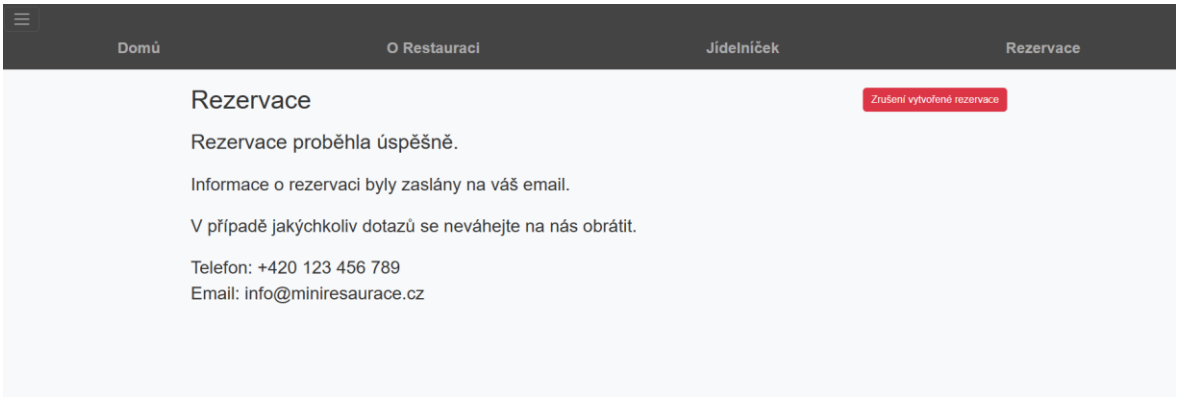
Obrázek 21 Rezervace 2. část, vlastní tvorba

Posledním krokem rezervace pro zákazníka je samotné potvrzení, které je prováděno na této stránce. Zde je zákazníkovi zobrazen souhrn rezervace k ověření. Pokud zákazník není spokojen s rezervací, může se pomocí tlačítka „Zpět“ vrátit do některého z předchozích kroků a případně provést úpravy. Pokud však zákazník nenajde žádnou chybu v rezervaci, může kliknout na tlačítko „Provést rezervaci“. Toto tlačítko uloží objednávku do databáze a zároveň zašle zákazníkovi e-mail s informacemi o objednávce a zobrazí mu potvrzení.



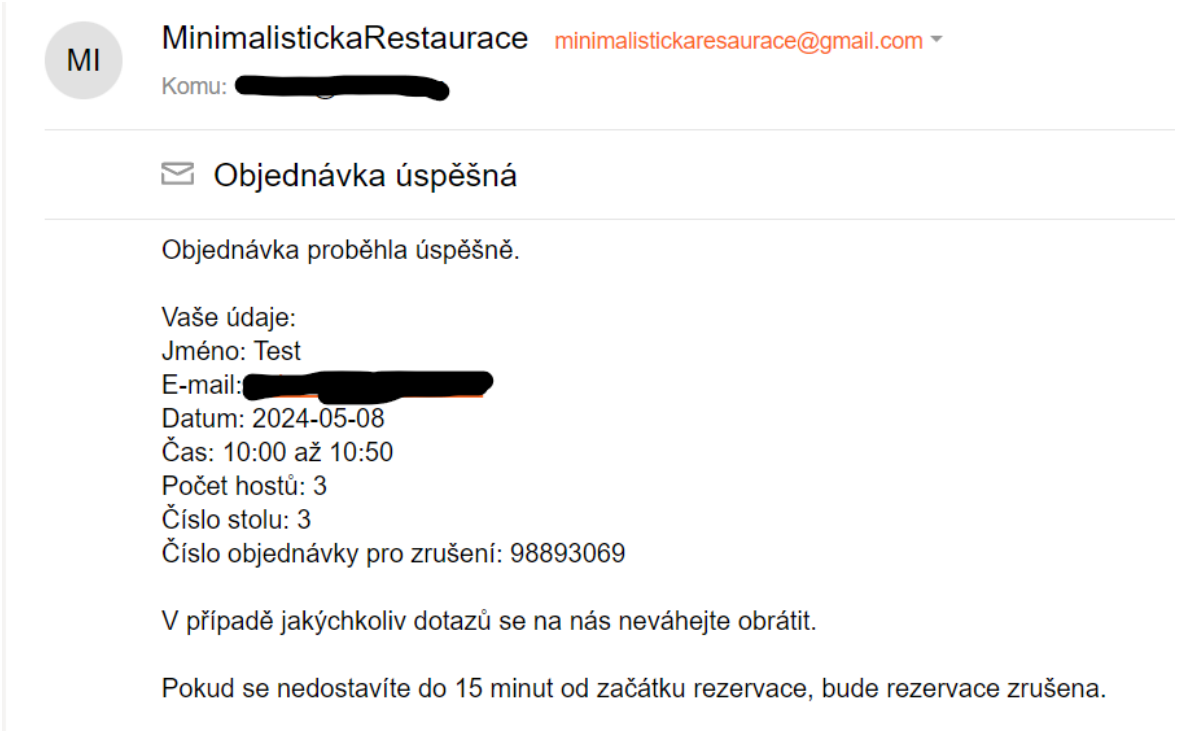
Obrázek 22 Rezervace 3. část, vlastní tvorba

Toto potvrzení je zákazníkovi zobrazeno po úspěšném dokončení rezervace.



Obrázek 23 Rezervace 4. část, vlastní tvorba

Tělo e-mailu, který je zákazníkovi odeslán po zkompletování rezervace, vypadá takto.

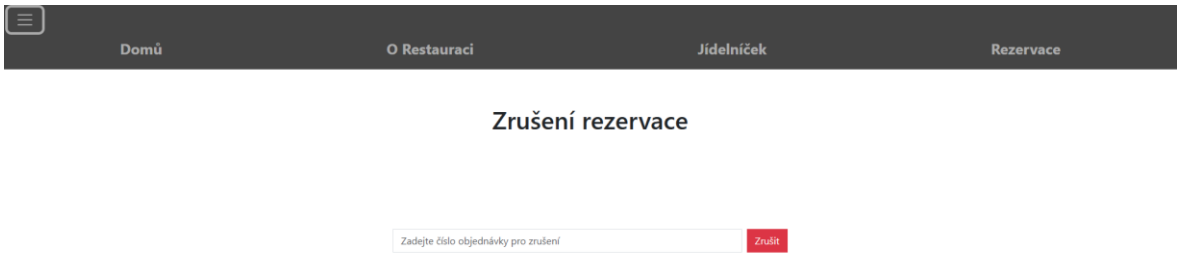


Obrázek 24 Ukázka emailu, vlastní tvorba

6.1.5 zruseni.php

Na této stránce může být objednávka zákazníka zrušena bez nutnosti telefonického kontaktu s restaurací. Zde je zákazníkem zadáváno číslo objednávky, které mu bylo vygenerováno při vytváření objednávky, a pomocí tohoto čísla může být zrušení provedeno. Toto číslo je kontrolováno, aby nedošlo k tomu, že dvě rezervace mají stejná čísla. Objednávka je vymazána z databáze, pokud je zákazníkem zadáno jeho číslo objednávky do formuláře a je kliknuto

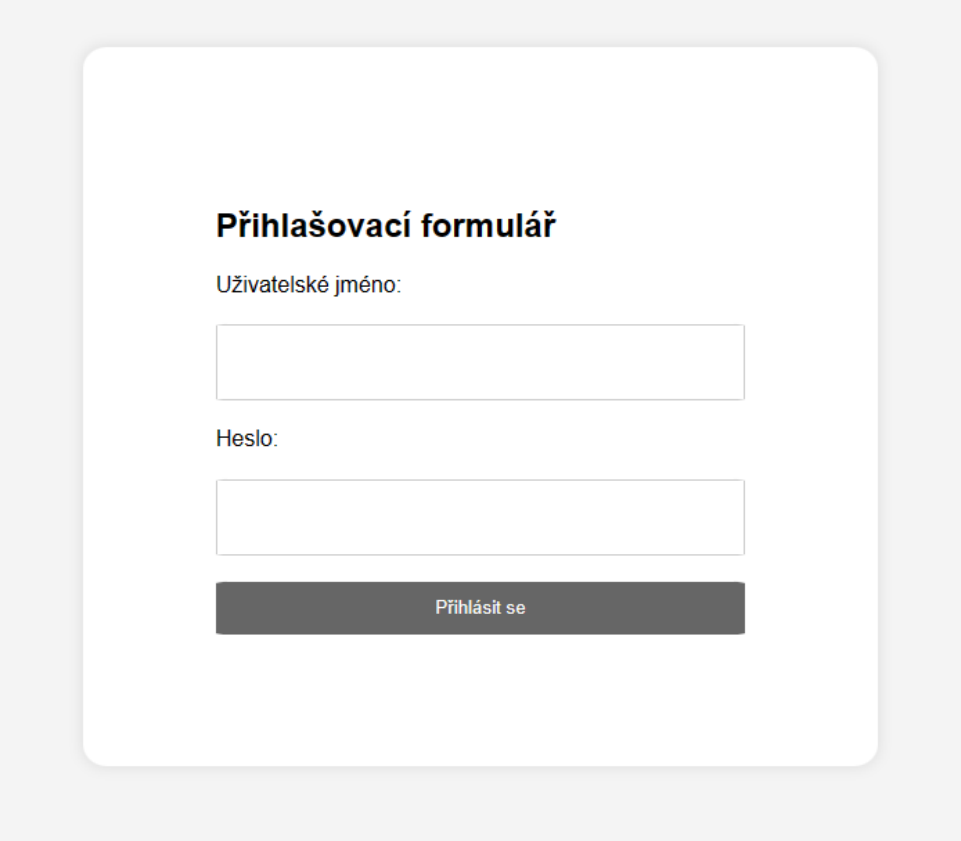
na tlačítko „Zrušit“. Pokud není zákazníkem zadáno číslo objednávky a je pouze kliknuto na tlačítko „Zrušit“, je zákazník systémem vyzván k vyplnění čísla objednávky.



Obrázek 25 Zrušení, vlastní tvorba

6.1.6 admin.php

Tato stránka obsahuje přihlašovací formulář, pomocí kterého se může zaměstnanec přihlásit do systému a tak získat přístup k objednávkám. Pokud zaměstnanec zadá nesprávná data, která jsou uložena v databázi jako jméno a heslo v podobě hashe, je na to stránkou upozorněn. Pokud uživatel nevyplní nějaké pole, je systémem znovu vyzván k jeho vyplnění. Pokud uživatel zadá správné jméno a heslo pro přihlášení, je přihlášen a přesměrován na stránku „databáze.php“.



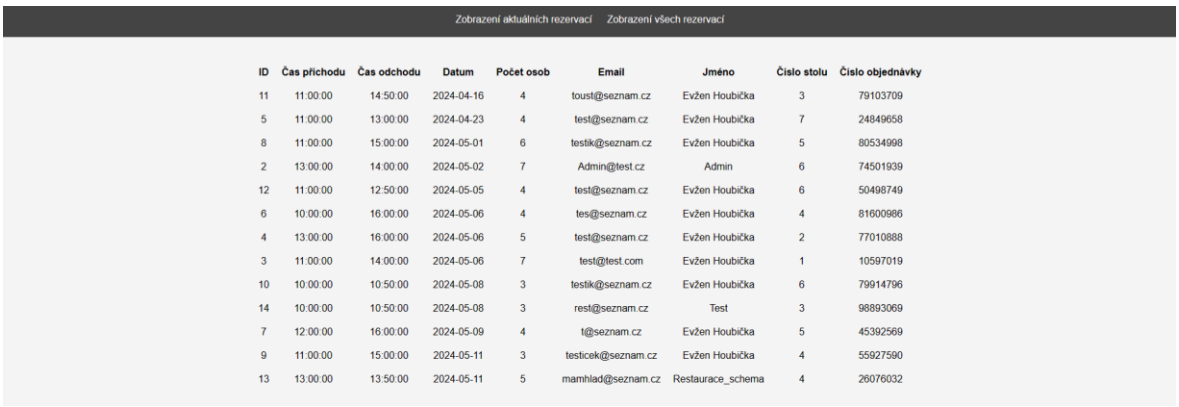
Obrázek 26 Přihlašovací formulář, vlastní tvorba

6.1.7 databaze.php

Stránka databáze umožňuje obsluze nebo administrátorovi přístup k objednávkám, které mohou být rozděleny na aktuální nebo všechny objednávky pomocí navigace v horní části obrazovky. První zobrazení na této stránce ukazuje uživateli aktuální objednávky, které jsou platné. Pro zobrazení všech objednávek je nutné v navigaci kliknout na „Zobrazení všech rezervací“. Zůstanou-li však uživatelé na této stránce, mohou rezervace upravit pomocí tlačítka „Upravit“, které se nachází hned vedle řádku s daty. Po kliknutí na tlačítko „Upravit“ jsou uživatelé přesměrováni na stránku „upraveni.php“.

Obrázek 27 Aktuální rezervace, vlastní tvorba

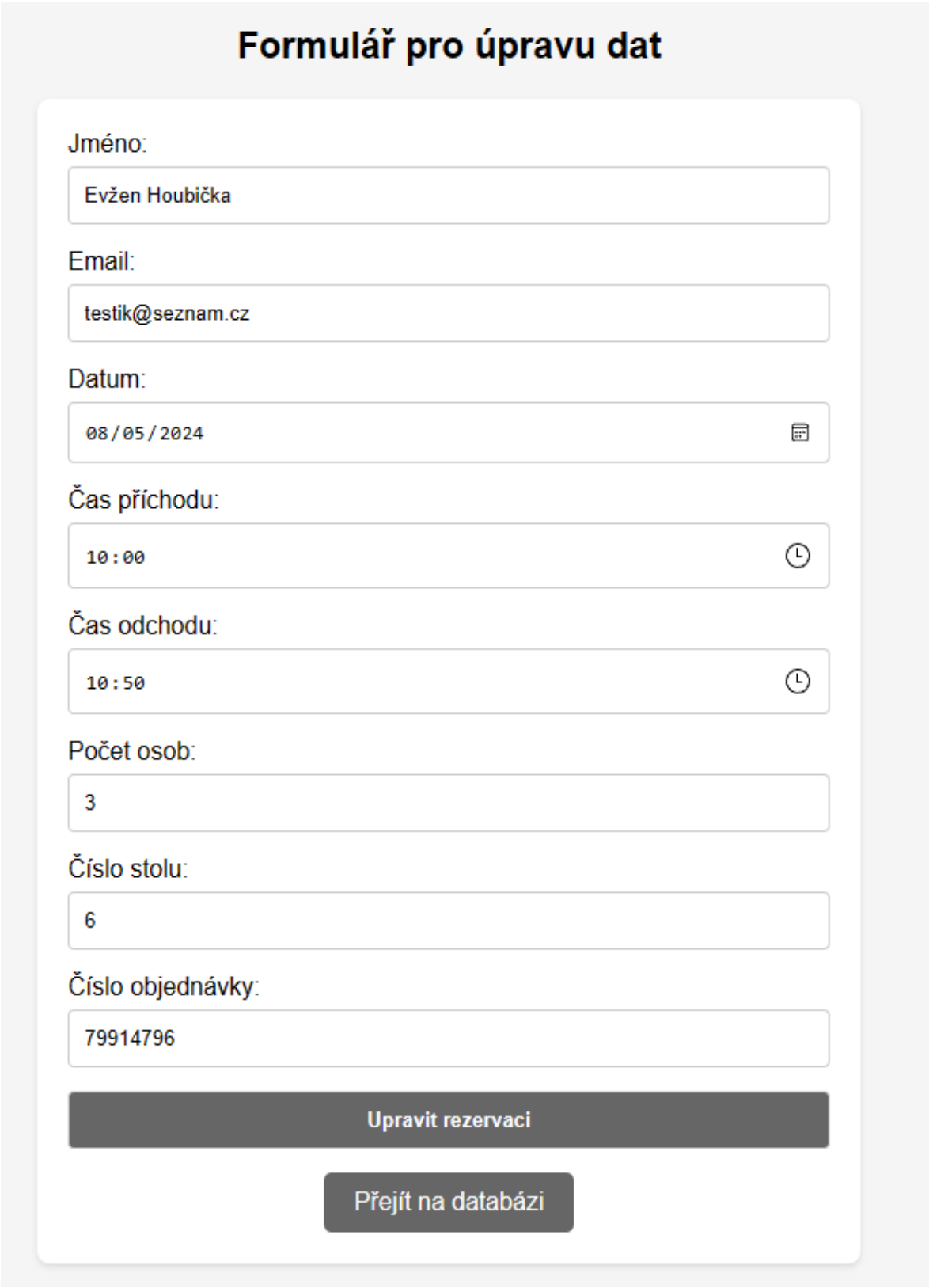
Pokud je kliknuto na „Zobrazení všech rezervací“, zobrazí se uživateli tato stránka obsahující všechny provedené rezervace.



Obrázek 28 Všechny rezervace, vlastní tvorba

6.1.8 upraveni.php

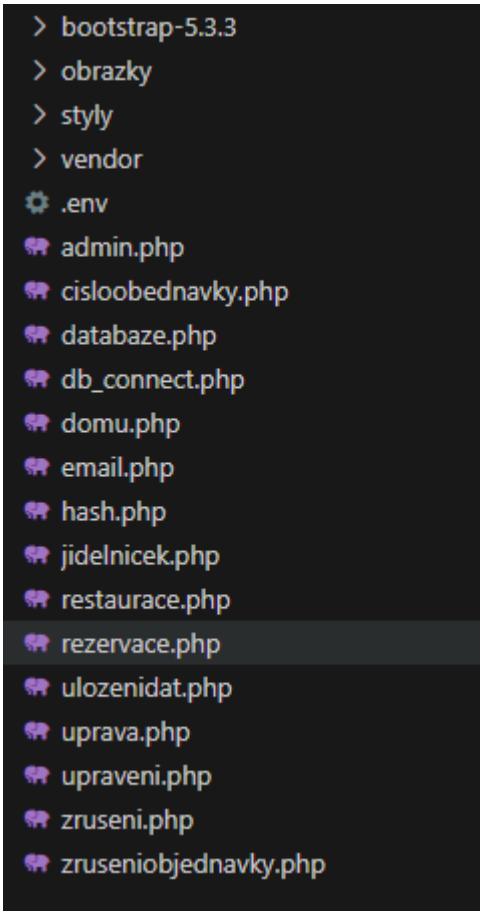
Na této stránce jsou data z konkrétního řádku, na kterém uživatel klikl na tlačítko „Upraveni“, načítána do formuláře pro úpravu dat. Tato data jsou načítána z databáze a jejich úprava má být co nejjednodušší. Po provedení úpravy je uživatelem kliknuto na tlačítko „Upravit rezervaci“ a tím jsou upravená data uložena do databáze. Pokud však uživatel nechce upravovat data a chce se pouze vrátit zpět, pak tlačítko „Přejít na databázi“ jen přesměrovává na stránku s databází bez nutnosti měnit data.



Obrázek 29 Formulář pro upravení, vlastní tvorba

6.2 Struktura aplikace

Na následujícím obrázku bude vidět struktura webové aplikace.



Obrázek 30 Struktura aplikace, vlastní tvorba

Ve složce "bootstrap-5.3.3" je uložen Bootstrap, který je následně využíván pro tvorbu navigace. Dále ve složce "obrázky" jsou uloženy obrázky používané na webu. Tyto obrázky jsou využívány k vylepšení vzhledu a uživatelského zážitku, jsou zobrazovány na různých stránkách a mohou obsahovat ilustrační, informativní či dekorativní prvky. Většina těchto obrázků byla vygenerována pomocí nástroje Microsoft Copilot, s výjimkou půdorysu restaurace, který byl vytvořen pomocí online programu Canva. Složka „styly“ obsahuje Následně je složka „vendor“ zahrnuta v projektu a je v ní uložen Composer pro PHP. Knihovna SendGrid pro PHP je využívána k odesílání e-mailů v projektu. Tato knihovna je instalována a spravována pomocí nástroje Composer a nachází se ve složce `vendor`, která obsahuje všechny závislosti projektu. Třída `SendGrid\Mail\Mail` je využívána k sestavování e-mailů, včetně nastavení odesílatele, příjemců, předmětu, těla e-mailu a případně i příloh. Následně je e-mail odesílán pomocí API klíče pro SendGrid.

6.3 Popis stránek

V této kapitole se zaměříme na ukázky kódu jednotlivých stránek webu. Pro každý webový modul vysvětlíme jeho roli a význam v rámci celkového projektu. Kapitola bude doplněna příklady kódu, které ilustrují nejdůležitější části práce.

6.3.1 .env

Soubor „.env“ obsahuje API klíč pro rozhraní API služby sendgrid.com, jenž je klíčový pro odesílání e-mailů v rámci projektu. Tento klíč je využíván k autentizaci a autorizaci požadavků na SendGrid API a zajišťuje, že e-maily jsou odesílány bezpečně a v souladu s požadavky platformy. Udržování API klíče v souboru „.env“ poskytuje bezpečnost, protože klíč není sdílen ve zdrojovém kódu. Použití tohoto klíče je nezbytné pro integraci s e-mailovou službou SendGrid a pro efektivní správu e-mailové komunikace v projektu.

6.3.2 db_connect.php

V souboru „db_connect.php“ je obsažen kód, který se stará o navázání připojení k databázi. Tento soubor je považován za klíčový prvek projektu, protože zajišťuje komunikaci s databází, což umožňuje práci s daty uloženými v databázi. Správné navázání připojení je zásadní pro bezproblémový a bezpečný provoz celého systému. Dále jsou v něm obsaženy důležité konfigurační údaje, jako jsou údaje o hostiteli, uživateli, heslu a názvu databáze, které jsou nezbytné pro úspěšné připojení.

6.3.3 admin.php

Na této stránce se nachází přihlašovací formulář, který je využíván k ověřování administrátorů nebo obsluhy. Prostřednictvím tohoto formuláře mohou uživatelé zadat své přihlašovací údaje, jako jsou uživatelské jméno a heslo. Pokud jsou tyto údaje správně zadány a přihlášení je úspěšné, uživatel je automaticky přesměrován na stránku „databaze.php“, kde jsou zobrazována data z databáze.

```php
    <?php
require_once('db_connect.php');
session_start(); // Zahájení relace

// Připojení k databázi
$conn = DBConnect::connect();
if (!$conn) {
    die('Chyba připojení k databázi.');
}

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Získejte data z formuláře
    $username = trim($_POST['username']);
    $password = trim($_POST['password']);

    // Připravte a proveďte dotaz pro tabulku administrator
    $stmt = $conn->prepare('SELECT Heslo FROM administrator WHERE Jmeno = ?');
    $stmt->bind_param('s', $username);
    $stmt->execute();
    $stmt->bind_result($hashedPassword);
    $stmt->fetch();
    $stmt->close();
    $Pass = password_hash($password, PASSWORD_DEFAULT);
    // Porovnání hesla s Hash v databázi
    $adminValid = password_verify($password, $hashedPassword);

    // Připravte a proveďte dotaz pro tabulku obsluha, pokud předchozí ověření bylo neúspěšné
    if (!$adminValid) {
        $stmt = $conn->prepare('SELECT Heslo FROM obsluha WHERE Jmeno = ?');
        $stmt->bind_param('s', $username);
        $stmt->execute();
        $stmt->bind_result($hashedPassword);
        $stmt->fetch();
        $stmt->close();
        $obsluhaValid = password_verify($password, $hashedPassword);
    }

    // Ověření hesla a přihlášení
    if ($adminValid || $obsluhaValid) {
        $_SESSION['username'] = $username;
        header('Location: databaze.php');
    }
    //Alert při nesprávném zadání jména nebo hesla
    else {
        echo '<script type="text/javascript">';
        echo 'alert("Neplatné uživatelské jméno nebo heslo.");';
        echo '</script>';
    }
    // Zavření spojení s databází
    $conn->close();
}
?>
```

Obrázek 31 Ukázka kódu přihlašovací formulář, vlastní tvorba

Na tomto obrázku je ukázka PHP kódu pro ověřování, zda jsou zadané údaje do formuláře shodné s heslem v podobě hashe v databázi. Nejprve je provedeno samotné připojení k databázi. Poté se ověřuje, zda je uživatel přihlašován jako administrátor. Pokud ne, ověřuje se, zda je uživatel přihlašován jako obsluha. Pokud ani to neplatí, je vyhozena zpráva `Neplatné uživatelské jméno nebo heslo. Pokud jsou však přihlašovací údaje správné, je uživatel přesměrován na stránku „databaze.php“.

6.3.4 cisloobjednavky.php

Tento soubor vrací číslo objednávky z databáze pro následnou kontrolu, aby se nestalo, že budou mít dvě objednávky stejné číslo.

```php
<?php

require_once('db_connect.php');

// Připojení k databázi
$conn = DBConnect::connect();

// Funkce pro načtení CisloObjednavky z tabulky rezervace
1 reference
function fetchCisloObjednavky($conn) {
    // SQL dotaz pro načtení CisloObjednavky
    $query = "SELECT CisloObjednavky FROM rezervace";

    // Proveďte dotaz
    $result = $conn->query($query);

    if ($result === false) {
        echo json_encode(['success' => false, 'message' => 'Query failed']);
        $conn->close();
        exit;
    }

    // Načtěte data z výsledku dotazu
    $data = [];
    while ($row = $result->fetch_assoc()) {
        $data[] = $row['CisloObjednavky'];
    }

    // Vraťte data jako JSON
    echo json_encode(['success' => true, 'data' => $data]);
}

// Zavolejte funkci pro načtení CisloObjednavky
fetchCisloObjednavky($conn);
// Uzavření připojení k databázi
$conn->close();
?>
```

Obrázek 32 Ukázka kódu funkce fetchCisloObjednavky, vlastní tvorba

Na začátku je provedeno připojení k databázi. Poté je definována funkce „fetchCisloObjednavky“, která je určena k získávání čísla objednávky „CisloObjednavky“ z tabulky „rezervace“ v databázi. V rámci této funkce je proveden SQL dotaz, jsou načtena data a jsou vrácena zpět jako JSON. Po dokončení funkce je připojení k databázi uzavřeno.

6.3.5 databaze.php

Na této stránce jsou zobrazeny všechny rezervace, které byly uživatelem uskutečněny, včetně těch probíhajících i minulých. Tato stránka je vybavena navigací, která umožňuje přepínání mezi aktuálními a všemi objednávkami. Kromě toho umožňuje úpravu stávajících rezervací, a to jak těch, které jsou aktuálně aktivní, tak i těch, které se mají teprve uskutečnit v budoucnu. To se děje za pomocí tlačítka upravit, které se nachází vedle informací o rezervaci.

```php
$sql = "SELECT * FROM rezervace WHERE CONCAT(Datum, ' ', CasOd) > DATE_SUB(NOW(), INTERVAL 15 MINUTE) ORDER BY Datum;";

// Spuštění dotazu
$query = $conn->query($sql);

// Zkontrolujte, zda dotaz byl úspěšný
if ($query === false) {
    die("Chyba při provádění dotazu.");
}

echo "<table>";
echo "<tr><th>ID</th><th>Čas příchodu</th><th>Čas odchodu</th><th>Datum</th><th>Počet osob</th><th>Email</th><th>Jméno</t

while ($row = $query->fetch_assoc()) {
    echo "<tr>";
    echo "<td>" . htmlspecialchars($row['ID']) . "</td>";
    echo "<td>" . htmlspecialchars($row['CasOd']) . "</td>";
    echo "<td>" . htmlspecialchars($row['CasDo']) . "</td>";
    echo "<td>" . htmlspecialchars($row['Datum']) . "</td>";
    echo "<td>" . htmlspecialchars($row['Pocet_osob']) . "</td>";
    echo "<td>" . htmlspecialchars($row['Email']) . "</td>";
    echo "<td>" . htmlspecialchars($row['Jmeno']) . "</td>";
    echo "<td>" . htmlspecialchars($row['Cislo_stolu']) . "</td>";
    echo "<td>" . htmlspecialchars($row['CisloObjednavky']) . "</td>";
```

Obrázek 33 Ukázka kódu pro výpis objednávek, vlastní tvorba

Na obrázku je ukázka kódu, pomocí kterého jsou data načítána z databáze na základě podmínky, že kombinované hodnoty sloupců Datum a CasOd jsou více než 15 minut před aktuálním časem nebo po aktuálním čase. Data jsou následně načítána pomocí SQL příkazu SELECT * FROM rezervace WHERE CONCAT(Datum, ' ', CasOd) > DATE_SUB(NOW(), INTERVAL 15 MINUTE) ORDER BY Datum;. SQL dotaz $query = $conn->query($sql); je poté proveden. Pokud je dotaz úspěšný, výsledky jsou iterovány pomocí smyčky while, v níž jsou hodnoty jednotlivých sloupců rezervace zobrazovány v tabulce pomocí PHP. Funkce htmlspecialchars je použita pro bezpečné zobrazení dat z databáze.

6.3.6 domu.php

```
<!-- Navigace -->
<nav class="navbar navbar-md navbar-dark" style="background-color: #444;">
    <div class="container-fluid">
        <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-
            <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarNav">
            <ul class="navbar-nav" style="margin-left: auto; flex-direction: row;">
                <li class="nav-item">
                    <a class="nav-link " href="domu.php">Domů</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="restaurace.php">O Restauraci</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="jidelnicek.php">Jídelníček</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link" href="rezervace.php">Rezervace</a>
                </li>
            </ul>
        </div>
    </div>
</nav>

    <!-- Obsah stránky -->
    <div class="center">
    <a href="rezervace.php" class="btn rezervace-btn">Rezervace</a>
```

Obrázek 34 Ukázka kódu navigace, vlastní tvorba

Na tomto obrázku lze vidět, jak je navigace řešena na všech stránkách, které jsou uživateli zobrazeny. Jednotlivé odkazy v navigaci jsou směrovány na různé stránky. Navigace je vytvořena pomocí Bootstrapu. Na dolní části obrázku je vidět tlačítko, kterým je uživatel přesměrován na stránku „rezervace.php“.

6.3.7 email.php

Soubor je zaměřen na posílání e-mailů uživatelům, a je volán ze stránky „rezervace.php“. Uživatelům je odesíláno potvrzení o provedené rezervaci, které obsahuje informace, jako je například jméno uvedené při rezervaci, čas rezervace a případně i číslo objednávky pro možné zrušení rezervace. Na tuto stránku jsou převáděna data ze stránky „rezervace.php“ a poté jsou zadávána v těle e-mailu.

```php
use SendGrid\Mail\Mail;
// Ověřte, zda byl odeslán POST požadavek
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Načtení dat z JSONu odeslaného z klienta
    $data = json_decode(file_get_contents('php://input'), true);

    // Ověření, že data byla dekódována správně
    if ($data === null) {
        echo json_encode(['success' => false, 'message' => 'Chyba při dekódování JSON dat']);
        exit;
    }

    // Načtení hodnot z pole $data
    $name = $data['name'] ?? 'Není k dispozici';
    $emailInput = $data['email'] ?? 'Není k dispozici';
    $date = $data['date'] ?? 'Není k dispozici';
    $timeod = $data['timeod'] ?? 'Není k dispozici';
    $timedo = $data['timedo'] ?? 'Není k dispozici';
    $guests = $data['guests'] ?? 'Není k dispozici';
    $cisloObjednavky = $data['cisloObjednavky'] ?? 'Není k dispozici';
    $selectedTableId = $data['selectedTableId'] ?? 'Není k dispozici';
```

Obrázek 35 Ukázka kódu odesílání emailů, vlastní tvorba

Na tomto obrázku je v horní části zobrazeno použití SendGrid pro odesílání e-mailů. Kód je následně kontrolován, zda byl obdržen POST požadavek. Pokud ano, data z JSONu poslaného klientem jsou načtena a zpracována. Poté je ověřováno, zda byla data správně dekódována. Pokud se dekódování nepodaří, je vrácena chyba. Ve spodní části obrázku jsou data z JSONu ukládána do jednotlivých proměnných, které obsahují informace jako jméno, e-mail, datum, časové intervaly, počet hostů, číslo objednávky a vybraný stůl.

```php
// Vytvoření instance třídy Mail
$email = new Mail();
$email->setFrom('minimalistickaresaurace@gmail.com', 'MinimalistickaRestaurace');
$email->setSubject('Objednávka úspěšná');
$email->addTo($emailInput, $name);

// Vytvoření zprávy obsahující informace o rezervaci
$message = <<<EOT
Objednávka proběhla úspěšně.

Vaše údaje:
    Jméno: $name
    E-mail: $emailInput
    Datum: $date
    Čas: $timeod až $timedo
    Počet hostů: $guests
    Číslo stolu: $selectedTableId
    Číslo objednávky pro zrušení: $cisloObjednavky

V případě jakýchkoliv dotazů se na nás neváhejte obrátit.

Pokud se nedostavíte do 15 minut od začátku rezervace, bude rezervace zrušena.
EOT;


// Přidejte obsah e-mailu
$email->addContent("text/plain", $message);

// Získání SendGrid API klíče z prostředí
$apiKey = $_ENV['API_KEY'];

// Vytvoření instance SendGridu a odeslání e-mailu
$sendgrid = new \SendGrid($apiKey);
```

Obrázek 36 Ukázka těla eamilu, vlastní tvorba

Na tomto obrázku je v horní části zobrazeno, od koho je e-mail odesílán a komu je určen. Ve střední části je zobrazen obsah samotného e-mailu, včetně informací o rezervaci, které byly zadány uživatelem na stránce „rezervace.php“. Ve spodní části je zobrazen API klíč, který je použit v rámci SendGrid funkce pro odeslání e-mailu.

6.3.8 hash.php

Tento soubor je pouze pomocný a slouží k vytvoření hashe z hesel pro případné ukládání do databáze. Pokud je stránka zobrazena, hash je vypisován a následně se může použít jako heslo v databázi.

```php
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <?php
  $password = "AdminHeslo"; // Předpokládáme, že heslo je zadáno jako vstup uživatelem.
  $hashedPassword = password_hash($password, PASSWORD_DEFAULT);
  echo $hashedPassword;

    ?>
</body>
</html>
```

Obrázek 37 Ukázka kódu pro hashování hesla, vlastní tvorba

6.3.9 jidlenicek.php

Tato stránka určena k zobrazování a pokrmů, jejich popisu včetně ceny. Součástí jsou i obrázky, jak pokrm může vypadat. Obrázky jsou pouze ilustrativní a vygenerovány umělou inteligencí. Jídla se dělí do kategorií jako jsou předkrmy, hlavní jídla a dezerty.

```
        <h3>Červená řepa s kozím sýrem a medem</h3>
        <p>Kombinace červené řepy, kozího sýra a medu.</p>
    </div>
    <div class="menu-item-price">
        <p>120 Kč</p>
    </div>
    <img src="obrazky/repa.png" alt="Červená řepa s kozím sýrem a medem">
/li>
li class="menu-item">
    <div>
        <h3>Grilované krevety na šťavnatém zeleninovém salátu</h3>
        <p>Grilované krevety podávané na křupavém zeleninovém salátu.</p>
    </div>
    <div class="menu-item-price">
        <p>150 Kč</p>
    </div>
    <img src="obrazky/kreveta.png" alt="Grilované krevety na šťavnatém zeleninovém salátu">
/li>

li class="menu-item">
    <div>
        <h3>Krémová dýňová polévka s hřebíčkem a křupavým cibulovým chipsem</h3>
        <p>Teplá krémová dýňová polévka s příchutí hřebíčku a křupavým cibulovým chipsem.</p>
    </div>
    <div class="menu-item-price">
        <p>110 Kč</p>
    </div>
    <img src="obrazky/dyne.png" alt="Krémová dýňová polévka s hřebíčkem a křupavým cibulovým chipsem">
/li>
```

Obrázek 38 Ukázka kódu Jídelníčku, vlastní tvorba)

V ukázce lze vidět příklady kódu, který tvoří strukturu stánky.

```css
body {
    font-family: Arial, sans-serif;
    background-color: #f8f9fa;
    color: #333;
    margin: 0;
    padding: 0;
}
h1, h2 {
    color: #333;
    text-align: center;
}
.menu-section ul {
    list-style-type: none;
    padding: 0;
}
.menu-section li {
    margin-bottom: 2%;
    border-bottom: 1px solid #ccc;
    padding-bottom: 2%;
    display: flex;
    align-items: center;
    justify-content: space-between;
}
.menu-section li img {
    max-width: 10%;
    margin-right: 2%;
}
```

Obrázek 39 Ukázka kódu stylování Jídelníčeku, vlastní tvorba

Zde je možno vidět ukázku stylování některých prvků na stránce. Konkrétněji se jedná o body, kterému je přidělováno písmo, barva pozadí, barva písma, a vnitřní a vnější ohraničení je nastaveno na 0. Poté

6.3.10 restaurace.php

```
    <h2>Otevírací doba</h2>
    <p>Po-Ne: 10:00 - 23:00<br>
</div>
<div class="map">
    <iframe src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d2727.8960717
            width="100%"
            height="450">
    </iframe>
</div>
```

Obrázek 40 Ukázka kódu pro zobrazování mapy, vlastní tvorba

V horní části této ukázky je umístěna otevírací doba a ve spodní části je umístěna mapa s místem, kde je restaurace umístěna.

6.3.11 rezervace.php

Rezervace.php je nejobsáhlejší stránka ze všech. Na této stránce se provádí rezervace, která je pomocí souboru „ulozenidat.php“ následně ukládána do databáze. Zároveň je využíván soubor „email.php“ pro odesílání informací o rezervaci zákazníkovi na email. Stránka je rozdělena do sekcí, které mizí nebo se objevují podle kroků uživatele, ale stále je to jedna stránka.

```
function filtraceRezervace(casod, casdo, date){
    casdo = date + "T" + casdo;
    casod = date + "T" + casod;
    const CasPrichod = new Date(casod);
    const CasOdchodu = new Date(casdo);

    reservationData.forEach((row) => {
        const CasOdDatabaze = new Date(row.Datum + "T" + row.CasOd);
        const CasDoDatabaze = new Date(row.Datum + "T" + row.CasDo);

        if (row.Datum === date) {

            if (CasOdDatabaze < CasOdchodu && CasOdDatabaze >= CasPrichod) {
                // Změna barvy pozadí
                var tableElement = document.getElementById(row.Cislo_stolu);
                tableElement.style.backgroundColor = "red";
            }
        }
    });
}
```

Obrázek 41 Ukázka kódu pro kontrolu času, vlastní tvorba

Zde je ukázka scriptu, kterým je zákazníkovi zobrazován již obsazený stůl.

```
// Ošetření odeslání formuláře pro krok dva
$('#go-to-step-three').on('click', function() {
    // Získání vybraného stolu
    var selectedTable = document.getElementById(selectedTableId);

    $('#summary-cislo').text(cisloObjednavky);
    // Zkontrolujte, zda byl vybrán stůl
    if (!selectedTableId) {
        alert("Vyberte prosím stůl před pokračováním do dalšího kroku.");
        return;
    }

    // Zkontrolujte, zda má vybraný stůl pozadí červené barvy
    var bgColor = window.getComputedStyle(selectedTable).backgroundColor;
    if (bgColor === 'rgb(255, 0, 0)' || bgColor === '#ff0000') {
        alert("Vybraný stůl nelze rezervovat, protože je obsazený.");
        return;
    }

$('#summary-table').text(selectedTableId);

    // Skrytí kroku dva a zobrazení kroku tři
    $('#step-two').hide();
    $('#step-three').show();
});

// Ošetření navigace zpět ke kroku jeden
$('#back-to-step-two').on('click', function() {
    // Skrytí kroku dva a zobrazení kroku jeden
    $('#step-three').hide();
    $('#step-two').show();
});
```

Obrázek 42 Ukázka kódu kontrola stolu, vlastní tvorba

Na tomto obrázku lze vidět, že číslo vybraného stolu je ukládáno do proměnné „selectedTable“. Tato proměnná je poté kontrolována, zda není „null“ nebo není nastavena. Po této podmínce je přecházeno do kroku, ve kterém je kontrolována barva pozadí vybraného objektu. Pokud je vybraná barva stolu červená, zákazník je upozorněn na to, že stůl je v daný čas již obsazený a že jej není možné rezervovat. Pokud je však stůl volný, číslo stolu je uloženo do „summary-table“. Tato informace je poté využita při zobrazení kontroly objednávky. V posledním kroku je skryta část dva a zobrazena část tři.

```
function minulost() {
    // Získání aktuální hodiny
    var currentHour = new Date().getHours();

    // Získání hodnot z formuláře a převod na objekty Date
    var timeodValue = $('#timeod').val(); // Hodnota z formuláře pro čas příchodu
    var timedoValue = $('#timedo').val(); // Hodnota z formuláře pro čas odchodu


    // Kontrola, zda je alespoň jedna z hodnot null
    if (timedoValue === null || timeodValue === null) {
        alert("Chybný časový údaj.");
        return false;
    }

    // Kontrola, zda jsou obě hodiny v minulosti
    if (timeodValue <= currentHour && timedoValue <= currentHour) {
        alert("Časové intervaly musí být nastaveny do budoucnosti.");
        return false;
    }

    // Kontrola, zda `timeod` není po `timedo`
    if (timeodValue >= timedoValue) {
        alert("Čas příchodu musí být před časem odchodu.");
        return false;
    }

    // Pokud všechny podmínky prošly, vracíme true
    return true;
}
```

Obrázek 43 Ukázka kódu funkce minulost, vlastní tvorba

Zde je možné vidět, že se kontroluje, zda zadaný čas v daný den není v minulosti, a také se kontroluje, aby čas příchodu nebyl po čase odchodu. Pokud se tak stane, zákazník je upozorněn na nesprávně nastavený časový údaj.

6.3.12 ulozenidat.php

Do tohoto souboru jsou posílána data ze stránky „rezervace.php“. A v tomto souboru jsou data ukládána do databáze.

```php
// Získání jednotlivých hodnot z pole
$name = $data['name'];
$email = $data['email'];
$date = $data['date'];
$timeod = $data['timeod'];
$timedo = $data['timedo'];
$guests = $data['guests'];
$CisloObjednavky = $data['cisloObjednavky'];
$selectedTableId = $data['selectedTableId'];

// Připojení k databázi
$conn = DBConnect::connect();
if (!$conn) {
    // Chyba při připojení k databázi, ukončení skriptu
    exit;
}

// SQL dotaz pro vložení rezervace
$sql = "INSERT INTO rezervace (Jmeno, Email, Datum, CasOd, CasDo, Pocet_osob, CisloObjednavky, Cislo_stolu)
        VALUES (?, ?, ?, ?, ?, ?, ?, ?)";
```

Obrázek 44 Ukázka kódu ukládání do databáze, vlastní zdroj

Příchozí data jsou ve formě pole, takže je nejprve nutné získat konkrétní data a správně je uložit do proměnných. Tato data jsou poté ukládána do databáze pomocí příkazu „INSERT“.

6.3.13 uprava.php

Tento soubor je používán k aktualizaci dat v databázi na nové hodnoty. Soubor je volán ze stránky „upraveni.php“.

```php
    // SQL dotaz pro aktualizaci údajů
    $sql = "UPDATE rezervace SET Jmeno = ?, Email = ?, Datum = ?, CasOd = ?, CasDo = ?, Pocet_osob = ?, CisloObjednavky = ?,
Cislo_stolu = ? WHERE ID = ?";

    // Příprava a vykonání dotazu
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("ssssiiii",
        $input['name'],
        $input['email'],
        $input['date'],
        $input['timeod'],
        $input['timedo'],
        $input['guests'],
        $input['cisloObjednavky'],
        $input['selectedTableId'],
        $input['id']
    );
```

Obrázek 45 Ukázka kódu pro aktualizaci dat, vlastní tvorba

Data v databázi jsou aktualizována pomocí SQL příkazu „UPDATE“ podle parametru „ID“. Následně je metoda „bind_param“ použita k první specifikaci typů dat („s“ pro řetězce a „i“ pro celá čísla) a poté k zadání proměnných, které obsahují hodnoty, jež se vkládají do databáze.

6.3.14 upraveni.php

Na stránce „upraveni.php“ je zobrazován formulář pro úpravu dat z předešlé provedené rezervace, která je uložena v databázi. Tato stránka je přístupná ze stránky „databaze.php“ a umožňuje upravovat konkrétní rezervaci.

```
function saveChanges() {
    // Získejte prvky z formuláře
    const form = document.getElementById('editForm');
    const formData = new FormData(form);

    // Vytvořte objekt s daty z formuláře
    var data = {
        id: id,
        name: formData.get('name'),
        email: formData.get('email'),
        date: formData.get('date'),
        timeod: formData.get('timeod'),
        timedo: formData.get('timedo'),
        guests: formData.get('guests'),
        cisloObjednavky: formData.get('cisloObjednavky'),
        selectedTableId: formData.get('selectedTableId')
    };

    // Vytvoření XMLHttpRequest objektu
    var xhr = new XMLHttpRequest();

    // Otevření POST požadavku na URL "uprava.php"
    xhr.open("POST", "uprava.php", true);
    xhr.setRequestHeader("Content-Type", "application/json");

    // Odeslání dat ve formátu JSON
    xhr.onreadystatechange = function () {
        if (xhr.readyState == XMLHttpRequest.DONE) {

        }
    };
    console.log(data);
    // Odeslání objektu `data` jako JSON řetězce
    xhr.send(JSON.stringify(data));
}
```

Obrázek 46 Ukázka kódu funkce saveChanges, vlastní tvorba

V této ukázce je ukázáno, jakým způsobem jsou upravená data předávána do souboru „uprava.php“, čímž jsou tato data uložena do databáze.

6.3.15 zruseni.php

Tato stránka je určena k tomu, aby se pomocí čísla objednávky smazala rezervace. Zákazník nebo obsluha mohou zadat číslo objednávky a kliknout na tlačítko „Zrušit“, čímž je ID

objednávky odesláno do souboru „zruseniobjednavky.php“ a rezervace je tak smazána z databáze.

```
function handleCancel() {
    // Zachycení hodnot z formuláře
    var order_id = document.getElementById('order_id').value;

    // Kontrola, zda je zadané číslo objednávky
    if (order_id.trim() === "") {
        alert("Prosím, zadejte číslo objednávky.");
        return;
    }

    // Vytvoření objektu s daty
    var data = {
        order_id: order_id
    };

    // Vytvoření XMLHttpRequest objektu
    var xhr = new XMLHttpRequest();
    xhr.open("POST", "zruseniobjednavky.php", true);
    xhr.setRequestHeader("Content-Type", "application/json");

    xhr.onreadystatechange = function () {
        if (xhr.readyState === XMLHttpRequest.DONE) {
            if (xhr.status === 200) {
                alert("Objednávka byla úspěšně zrušena.");
            } else {
                alert("Nastala chyba při zrušení objednávky: " + xhr.responseText);
            }
        }
    };

    // Odeslání dat jako JSON
    xhr.send(JSON.stringify(data));
}
```

Obrázek 47 Ukázka kódu funkce handleCancel, vlastní tvorba

Kód nejprve získává hodnotu „order_id“ z inputu HTML s ID rovno „order_id“. Zda je hodnota „order_id“ prázdná zobrazí se uživateli „Prosím, zadejte číslo objednávky.“. Pokud však je tato hodnota zadaná, tak se uloží do objektu data. Kód následně vytvoří „XMLHttpRequest“ jenž využívá pro odeslání HTTP požadavku na server. Následně odešle za pomocí „POST“ data na stránku „zruseniobjednavky.php“. Následná funkce kontroluje stav požadavku, pokud je status 200, tak se zobrazí, že rezervace byla úspěšně zrušena, pokud ne zobrazí se chybová hláška. Na konci stránky se data odesílají za pomocí JSON.

6.3.16 zruseniobjednavky.php

V tomto souboru je daný řádek z tabulky rezervace smazán podle čísla objednávky. Tento soubor je volán ze stránky „zruseni.php“.

```php
$order_id = $data['order_id'];

// Připojení k databázi
$conn = DBConnect::connect();
if (!$conn) {
    echo json_encode(['success' => false, 'message' => 'Chyba při připojení k databázi.']);
    exit;
}

// Připravte SQL dotaz pro odstranění záznamu podle order_id
$stmt = $conn->prepare("DELETE FROM rezervace WHERE CisloObjednavky = ?");
if ($stmt === false) {
    echo json_encode(['success' => false, 'message' => 'Failed to prepare statement']);
    $conn->close();
    exit;
}
```

Obrázek 48 Ukázka kódu pro smazání z databáze, vlastní tvorba

Na začátku tohoto kódu se získává hodnota „order_id“ z pole „$data“. Poté se pokouší o připojení k databázi za pomocí „DBConnect“. Pokud se připojení k databázi nezdaří, je vrácena odpověď ve formátu JSON s informací o neúspěchu a chybovou zprávou.

Následně se kód snaží připravit SQL dotaz pro odstranění záznamu z tabulky `rezervace` na základě hodnoty `CisloObjednavky` (zřejmě odpovídající `order_id`). Pokud příprava dotazu selže (`$stmt` bude `false`), vrací odpověď s informací o neúspěchu a chybovou zprávou, uzavírá připojení k databázi a ukončuje skript.

ZÁVĚR

Cílem této bakalářské práce s názvem Restaurační Rezervační Systém bylo navrhnout a vyvinout plně funkční rezervační systém, který umožňuje zákazníkům pohodlně rezervovat stoly v restauraci podle jejího půdorysu bez nutnosti telefonických hovorů. Práce je rozdělena na dvě části: teoretickou a praktickou.

V teoretické části prezentuji koncepty a základy rezervačních systémů. Také v této část popisuji použité technologie k vypracování návrhu a následné implementaci webové aplikace.

Praktická část se zaměřuje na implementaci samotné aplikace, včetně jejího návrhu, jejích funkcionalit a procesů. Tento krok zahrnuje vývoj a integraci klíčových prvků systému, které umožní uživatelům pohodlně a efektivně využívat rezervační systém pro rezervaci stolů v restauraci.

Výsledkem mé práce je moderní a uživatelsky přívětivý rezervační systém, který přináší mnoho výhod restauracím i zákazníkům. Zákazníci mohou snadno a rychle rezervovat stoly podle svých preferencí a restaurace mohou optimalizovat své provozní procesy. Práce ukázala, že digitální rezervační systém může výrazně zlepšit efektivitu v restauračním průmyslu a zlepšit zážitky zákazníků.

I když program plní svůj účel, stále existují oblasti pro zlepšení. Například možnost zavedení flexibilních velikostí stolů a dalších parametrů podle půdorysu restaurace by umožnila vyšší personalizaci.

Celkově tato práce může ukázat restauracím, že celý proces rezervace může probíhat bez nutnosti telefonických hovorů.

SEZNAM POUŽITÉ LITERATURY

[1] *The Advantages and Disadvantages of Online Booking Systems*. Online. 2022. Dostupné z: https://www.bookinglive.com/wp-content/uploads/2022/05/bookinglive_ebook_final.pdf. [cit. 2024-02-08].

[2] *Flying revue*. Online. 2018. Dostupné z: https://www.flying-revue.cz/svet-letecke-dopravy-2. [cit. 2024-03-08].

[3] *Emerging Trends in Online Reservation Systems: Navigating the Future*. Online. C2024. Dostupné z: https://fhahoreca.com/blog/trends-in-online-reservation-systems/. [cit. 2024-02-08].

[4] *Security in an Online Reservation System*. Online. 2020. Dostupné z: https://www.otrams.com/security-in-an-online-reservation-system/. [cit. 2024-02-08].

[5] *U Ševce*. Online. C2018. Dostupné z: https://www.restauraceusevce.cz/cz/rezervace. [cit. 2024-05-08].

[6] *Restaurace Kozlovna Celnice Zlín*. Online. C2023. Dostupné z: https://kozlovnazlin.cz/. [cit. 2024-03-08].

[7] *RESTAURACE U DVOU SLUNEČNIC*. Online. C. Dostupné z: https://www.udvouslunecnic.cz/kontakt.php. [cit. 2024-05-08].

[8] *What are Functional and Non-Functional Requirements and How to Document These*. Online. C2024. Dostupné z: https://enkonix.com/blog/functional-requirements-vs-non-functional/. [cit. 2024-03-08].

[9] *What is UML Collaboration Diagram?* Online. C2024. Dostupné z: https://www.visual-paradigm.com/guide/uml-unified-modeling-lan-guage/what-is-uml/. [cit. 2024-03-08].

[10] *Unified Modeling Language (UML) Diagrams*. Online. 2024. Dostupné z: https://www.geeksforgeeks.org/unified-modeling-language-uml-introduction/. [cit. 2024-03-08].

[11] *Use Case Diagrams | Unified Modeling Language (UML)*. Online. 2024. Dostupné z: https://www.geeksforgeeks.org/use-case-diagram/. [cit. 2024-03-08].

[12] *UML Use Case Diagram Tutorial*. Online. C2024. Dostupné z: https://www.lucidchart.com/pages/uml-use-case-diagram. [cit. 2024-03-08].

[13] *Class Diagram | Unified Modeling Language (UML)*. Online. 2024. Dostupné z: https://www.geeksforgeeks.org/unified-modeling-language-uml-class-diagrams/. [cit. 2024-03-08].

[14] *What is Class Diagram?* Online. C2024. Dostupné z: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/. [cit. 2024-03-08].

[15] *Introduction of ER Model*. Online. 2024. Dostupné z: https://www.geeksforgeeks.org/introduction-of-er-model/. [cit. 2024-03-08].

[16] *All about ER model cardinality with examples*. Online. C2024. Dostupné z: https://www.gleek.io/blog/er-model-cardinality. [cit. 2024-03-08].

[17] *Requirements Traceability Matrix – RTM*. Online. 2023. Dostupné z: https://www.geeksforgeeks.org/requirement-traceability-matrix/. [cit. 2024-03-08].

[18] *HTML Introduction*. Online. 2024. Dostupné z: https://www.geeksforgeeks.org/html-introduction/. [cit. 2024-03-08].

[19] *What is HTML – Definition and Meaning of Hypertext Markup Language*. Online. 2021. Dostupné z: https://www.freecodecamp.org/news/what-is-html-definition-and-meaning/. [cit. 2024-03-08].

[20] *Learn to style HTML using CSS*. Online. C1998-2024. Dostupné z: https://developer.mozilla.org/en-US/docs/Learn/CSS. [cit. 2024-03-08].

[21] *What is CSS ?* Online. 2023. Dostupné z: https://www.geeksforgeeks.org/what-is-css/. [cit. 2024-03-08].

[22] *What is JavaScript?* Online. C1998-2024. Dostupné z: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript. [cit. 2024-03-08].

[23] *What is JavaScript ?* Online. 2024. Dostupné z: https://www.geeksforgeeks.org/what-is-javascript/. [cit. 2024-03-08].

[24] *What is PHP*. Online. C2023. Dostupné z: https://www.phptutorial.net/php-tutorial/what-is-php/#:~:text=PHP%20is%20a%20server-side%20and%20general-purpose%20scripting%20language,PHP%20was%20created%20by%20Rasmus%20Lerdorf%20in%201994. [cit. 2024-03-08].

[25] *PHP Introduction*. Online. 2024. Dostupné z: https://www.geeksforgeeks.org/php-introduction/. [cit. 2024-03-08].

[26] *What is Bootstrap ?* Online. 2024. Dostupné z: https://www.geeksforgeeks.org/what-is-bootstrap/. [cit. 2024-03-08].

[27] *A Beginner's Guide to Bootstrap*. Online. C2024. Dostupné z: https://www.codecademy.com/resources/blog/what-is-bootstrap/. [cit. 2024-03-08].

[28] *CO JE MYSQL?* Online. C2024. Dostupné z: https://www.websiterating.com/cs/web-hosting/glossary/what-is-mysql/. [cit. 2024-03-08].

[29] *MySQL | Common MySQL Queries*. Online. 2021. Dostupné z: https://www.geeksforgeeks.org/mysql-common-mysql-queries/. [cit. 2024-03-08].

[30] *What is a LAMP stack?* Online. C2024. Dostupné z: https://aws.amazon.com/what-is/lamp-stack/#:~:text=A%20LAMP%20stack%20is%20a,and%20the%20programming%20language%2C%20PHP. [cit. 2024-03-08].

[31] OPENAI. ChatGPT. San Francisco, CA: OpenAI, 2024. [online]. [cit. 2024-04-07]. Dostupné z: https://openai.com/

[32] COPILOT, M. Abstraktní umělecké dílo [online]. 2024 [cit. 2024-04-07]. Dostupné z: https://www.bing.com/chat

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

CSS	Cascading Style Sheets
ER	Entity-Relationship.
HTML	Object Modeling Technique
HTTP	HyperText Transfer Protocol
ID	Identifikátor
ISO	International Organization for Standardization
OMT	HyperText Markup Language
PHP	Hypertext Preprocessor
RTM	Requirement Traceability Matrix
SQL	Structured Query Language
UML	Unified Modeling Language

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I: CD s elektronickou verzí práce a zdrojovým kódem aplikace

SEZNAM PŘÍLOH

Příloha P I: fulltext.pdf

Příloha P II: Složka prilohy.zip se zdrojovým kódem aplikace a databází.

PŘÍLOHA P I: STRUKTURA CD

Text bakalářské práce v digitální podobě – soubor .pdf

Složka prilohy.zip – Obsahující zdrojový kód webové aplikace ve složce restaurace_stranky a databázi restaurace_databaze