ZÁPADOČESKÁ UNIVERZITA V PLZNI

FAKULTA EKONOMICKÁ

Bakalářská práce

Návrh a implementace webového informačního systému

Design and implementation of a web-based information system

Snizhana Marchuk

Plzeň 2023

Čestné prohlášení

Prohlašuji, že jsem bakalářskou práci na téma

„Návrh a implementace webového informačního systému“

vypracovala samostatně pod odborným dohledem vedoucího bakalářské práce za použití pramenů uvedených v přiložené bibliografii.

Plzeň dne 23.4.2023 v. r. Snizhana Marchuk

Zásady pro vypracování práce

1. Uveďte zásady tvorby webového IS a specifikujte požadavky jazykové školy na webový IS.
2. Navrhněte funkcionality a obsah webového IS.
3. Navrhněte datové objekty, jejich strukturu, relace mezi objekty, integritní omezení a implementujte databázovou vrstvu.
4. Navrhněte a implementujte aplikační a prezentační vrstvu.
5. Otestujte použitelnost WIS a diskutujte jeho omezení a možností rozšíření.

Poděkování

Ráda bych poděkovala vedoucímu bakalářské práce doc. RNDr. Mikuláši Gangurovi, Ph.D. za cenné rady a připomínky.

Obsah

Úvod

V dnešní informační době se neustále hovoří o datech a o jejich zpracování. S rostoucí závislostí všech oblastí života na počítačových a internetových technologiích roste i rozšířenost použití informačních systémů. Na informační systém můžeme nahlížet jako na systém technických prostředků, dat a lidí, jehož cílem je shromažďovat data, přeměňovat je na informace, a ty pak transformovat na využitelné znalosti.

Jako samostatný typ informačních systémů lze vyčlenit webové informační systémy, vytvořené s využitím internetových a webových technologií. Jedná se o informační systémy, které umožňují diferenciaci přístupových práv a poskytnutí přístupu velkému počtu uživatelů. V současnosti jsou webové stránky často prvním kontaktem zákazníků s podnikem (i když se o něm dozvědí z jiného zdroje, vždy se podívají na webové stránky, kdykoli je to možné). Webové informační systémy se tak staly významnou součástí prezentace informací, produktů a služeb.

Hlavním cílem této bakalářské práce je vytvořit webový informační systém pro jazykovou školu pomocí běžně dostupných technologií. Systém bude sloužit k prezentaci školy na internetu, bude zprostředkovávat první kontakt zákazníků se školou a uschovávat záznamy jak o zájemcích o výuku, tak o stávajících studentech, lektorech, nabízených kurzech a vyučovacích hodinách. Naplnění tohoto cíle je rozděleno do následujících částí:

- analýza požadavků jazykové školy na webový informační systém;
- návrh funkcionalit a obsahu systému, návrh datových objektů, jejich struktury, relací mezi objekty, integritních omezení a následná implementace databázové vrstvy;
- návrh a implementace aplikační a prezentační vrstev;
- Testování a návrhy na zlepšení systémů.

Práce je členěna do tří hlavních kapitol. V první kapitole je provedena literární rešerše, kdy jsou definovány základní pojmy a jsou popsány technologie, které budou pro návrh a implementaci využity. Druhá kapitola práce se věnuje metodickým přístupům k tvorbě informačních systémů a uvádí metody použité v praktické části. Následující kapitola popisuje samotný proces tvorby webového informačního systému. Nejprve je definován účel systému, následně jsou představeny požadavky na systém. Dále je pozornost

věnována již samotnému návrhu a implementaci systému, ale i jeho následnému testování a možným vylepšením.

1 Literární rešerše

V této kapitole jsou uvedena teoretická východiska práce. Nejdřív se kapitola zaměřuje na přiblížení podstaty informačních systémů a vysvětlení principu fungování webových informačních systémů. Dál jsou popsány implementační nástroje a jazyky, ve kterých bude webový informační systém implementován.

1.1 Informační systém

Informační systémy se obvykle vytvářejí pro potřeby reálných organizací, což jsou sociální systémy skládající se z mnoha jednotlivých prvků, včetně velkého počtu lidí, kteří spolupracují na dosažení společného cíle. Pro tvorbu informačních systémů je zásadní systémový pohled na byznys. Byznys systém je tedy celistvým pohledem na celou organizaci, která má definované byznys cíle a záměry. Součástí tohoto systému jsou lidé (pracovníci a manažeři), kteří vykonávají činnosti za účelem dosažení cílů organizace, a zdroje, které při tom používají (Bruckner, a další, 2012).

Komponenty informačního systému obvykle odpovídají komponentám byznysových systémů. Nicméně informace o daném komponentu (jako například o osobě, stroji, materiálu atd.) jsou často důležitější než samotný byznysový komponent. Z tohoto hlediska může být informační systém vnímán jako součást byznysového systému. I když se oba systémy mohou shodovat v použitých komponentách, liší se v účelu. Účelem informačního systému je poskytovat správná data a informace správné osobě (uživateli IS) ve správný čas. Kritériem pro správnost dat a informací je vhodnost podpory byznysových systémů při plnění jejich účelu (Bruckner, a další, 2012).

Obr. 1: Složky informačního systému

Zdroj: Vlastní zpracování (2022)

Z výše uvedeného lze vymezit pět samostatných složek, které tvoří jednotný informační systém organizace. Mezi těchto pět složek patří: hardware, software, data, lidé a procesy. Každá z těchto komponent je nezbytnou pro funkčnost a efektivitu informačního systému.

1.2 Webový informační systém

Pokud pojmeme informační systém jako počítačovou reprezentaci nějakého systému z reálného světa, která slouží k podpoře procesů probíhajících v tomto systému, pak každý datově náročný informační systém, ke kterému mohou uživatelé přistupovat prostřednictvím webových prohlížečů, lze označovat jako webový informační systém (WIS) (Schewe & Thalheim, 2019). To vypovídá o tom, že webové informační systémy jsou v podstatě jen speciální podtřídou informačních systémů.

Při tvorbě informačního systému je důležité odpovědět na následující základní otázky:

1. Kdo bude systém používat?
2. Co je náplní systému?
3. Proč by se měl systém používat? (Gangur, 2014, str. 5)

Dřív se předpokládalo, že se informační systémy slouží pouze k podpoře obchodních a provozních procesů uvnitř podniku. Za tohoto předpokladu se lehce hledá odpověď na otázku, kdo bude systém používat. Uživatelé systému jsou zaměstnanci, kteří jej budou používat jako nástroj pro svou pracovní činnost. Z odpovědi na tuto otázku obvykle vznikají modely pracovních postupů a uživatelských rozhraní, které odrážejí různé pohledy na systém definovaný uživateli, managementem a návrhářem.

Otázka, co je náplní informačního systému, se týká obsahu dat. Obvykle souvisí s modelováním a návrhem databáze, v níž budou tato data uložena. Kritéria, která určují podobu modelu, jsou ovlivněna skutečností, že data musí být trvale uložena a současně zpřístupněna více uživatelům, aniž by byla porušena jejich konzistence nebo integrita (Schewe & Thalheim, 2019). Kromě toho se to týká také externích pohledů, tj. dat, která jsou zpřístupněna uživateli pro konkrétní úlohu. To znamená, že se tato otázka týká především statických aspektů systému.

U intranetových informačních systémů otázka, proč bude systém používán, vede k definování operací, které podporují podnikové procesy. Tyto operace jsou obvykle vyvolány vznikem určitých obchodních případů. Lze tedy říci, že se tato otázka týká především dynamických aspektů systému a jeho využití.

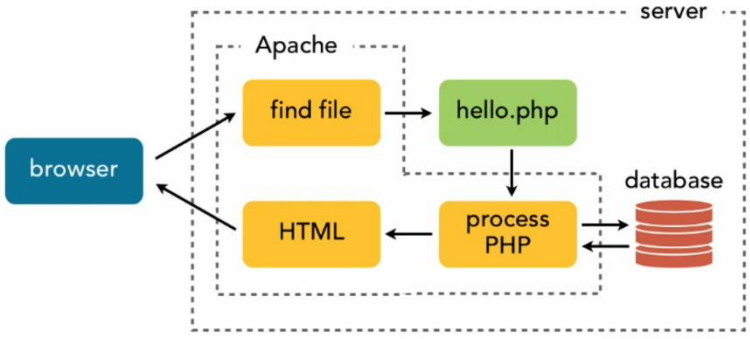
Když uvažujeme webový informační systém, najít odpovědi na výše uvedené otázky již není tak snadné. Stále platí, že otázka, co je náplní systému, se týká obsahu dat, a proto stále souvisí s modelováním a návrhem databáze. Podobně otázka, proč bude systém využíván, vede k definici operací, které podporují určité procesy.

Odpověď na otázku, kdo bude systém používat, se však výrazně liší od odpovědí, které jsme uvedli pro tradiční podnikové informační systémy. Uživatelé webového informačního systému nejsou předem známi. Jinými slovy, webové informační systémy jsou otevřené téměř každému, kdo by je chtěl používat. Z toho vyplývá, že při modelování webových informačních systémů je třeba určit cílové uživatele. Stejně tak není jasné, jak bude potenciální uživatel systém používat. Omezujeme-li pozornost na cílové uživatele, musíme předvídat jeho chování. To se týká záměrů uživatele, navigace v systému a také požadované podpory (Schewe & Thalheim, 2019). Uživatelé nejsou povinni systém používat. Naopak systém musí být navržen tak, aby se pro cílové uživatele stal atraktivním.

1.3 Architektura klient-server

Síťová architektura klient-server je jedním z nejdůležitějších pojmů pro pochopení principu fungovaní webových aplikací. Tato architektura odděluje klienta (prohlížeč nebo jinou aplikaci s grafickým uživatelským rozhraním) a server, kteří spolu komunikují přes počítačovou síť. V reálném provozu jsou webový a databázový server také oddělené z důvodu bezpečnosti a z důvodu zvýšení výkonnosti a rozložení zátěže (Gangur, 2014). Na obrázku je zobrazen proces zpracovávání požadavku klienta a zasílaní odpovědí.

Obr. 2: Architektura klient-server



Zdroj: Almendral (2018)

Požadavek webového klienta je zpracován v následujících krocích:

1. Prohlížeč zadá požadavek v podobě žádosti o provedení PHP scriptu webovému serveru.
2. Webový server převezme od klienta požadavek a vyvolá PHP interpret, kterému předá požadovaný PHP skript ke zpracování.
3. Interpret PHP skript provede.
4. Část PHP skriptu obsahuje příkazy SQL. PHP interpret se spojí s databázovým serverem MySQL a požádá server o zaslání dat z databáze zasláním SQL příkazu.
5. Databázový server zpracuje SQL dotaz a vrátí jeho výsledek PHP interpretu.
6. PHP interpret připraví data k prezentaci dle instrukcí v PHP skriptu a vrátí výsledky na webový server.
7. Webový server odešle data klientovi, který je zobrazí (Gangur, 2014).

Uživatelé jako klienti tedy komunikují pouze s webovým serverem, nikoli se serverem databázovým. S tím komunikuje pouze webový server.

1.4 Použité technologie

Webový informační systém se vytváří propojením frontendových a backendových technologií. Backend je serverová stránka aplikace. Její hlavní funkcí je zajištění správného chodu celé aplikace. Pro vývoj backendu budou použity PHP a MySQL, které jsou nejpoužívanějšími nástroji pro vývoj malých informačních systémů. Frontend spravuje vše, co uvidíte jako první v aplikaci nebo prohlížeči. Pro vývoj frontendu budou použity HTML, CSS a Javascript.

1.4.1 MySQL

MySQL je systém pro správu relačních databází. Relační databáze umožnuje efektivně ukládat, načítat, řadit a vyhledávat data (Welling & Thomson, 2017). Přístup k datům v databázi řídí MySQL server. Internetové stránky MySQL popisují MySQL jako „nejoblíbenější open source databázi na světě“ (MySQL, 2022). Její popularita je nepochybně podpořena skutečností, že pro nekomerční použití je možné si stáhnout kopii MySQL zdarma, jelikož MySQL je k dispozici pod volné dostupnou licencí s otevřeným zdrojovým kódem General Public Licence (GPL) (MySQL, 2022).

Pro přístup k systémům pro správu relačních databází se používá jazyk SQL (Structured Query Language). SQL umožňuje, jak definovat databáze pomocí příkazů pro definici dat (CREATE, ALTER, DROP), tak se jich dotazovat pomocí příkazů pro manipulaci s daty (SELECT, INSERT, UPDATE, JOIN, SHOW, DELETE) (Gangur, 2014). Pomocí jazyka PHP můžeme všechny tyto příkazy směrovat přímo do systému MySQL, aniž bychom museli spouštět systém samotný, nebo používat jeho příkazový řádek. Výsledky lze ukládat do polí pro další zpracování a provádět více vyhledávání, závislých na výsledcích vrácených z předchozích vyhledávání, abychom se dostali až k položce dat, kterou potřebujeme.

Hlavními konkurenty MySQL v oblasti relačních databází jsou Oracle, Microsoft SQL server, PostgreSQL a MariaDB, jež vznikla jako alternativní větev MySQL. Databázový systém MySQL má však mnoho předností. MySQL je k dispozici zdarma pod licencí otevřeného softwaru, pokud aplikace nebude komerčně distribuovaná (MySQL, 2022), což platí pro naši aplikaci. Další výhodou MySQL je snadnost použití. Lze ho totiž nastavit snadněji než jiné podobné systémy. Další výhodou je, že MySQL má otevřený zdrojový kód, ale zároveň dostupnou podporu poskytovanou rodičovskou společností Oracle (Welling & Thomson, 2017).

1.4.2 PHP

PHP je serverový skriptovací jazyk, který byl navržen speciálně pro webové aplikace. Byl vytvořen v roce 1994 a dnes je nejpoužívanějším jazykem pro tvorbu webových stránek. Jeho syntax je inspirována jazyky C a Perl. Původně bylo PHP zkratkou pro Personal Home Page, nicméně význam zkratky se později změnil a nyní znamená PHP: Hypertext Preprocessor (Welling & Thomson, 2017).

Kód jazyka PHP můžeme vkládat přímo do stránky HTML, na které se kód provede při každé návštěvě této stránky. PHP script je interpretován webovým serverem, přičemž generuje dokument HTML nebo jiný výstup, který uživatel uvidí (Welling & Thomson, 2017).

PHP je open source projekt, což znamená, že je k dispozici zdarma a máme přístup k jeho zdrojovému kódu. Aktuální hlavní verzí jazyka PHP je verze 8.2. Hlavními konkurenty jazyka PHP jsou jazyky Java, Ruby, Python, Perl, prostředí Node.js a Microsoft.NET. V porovnání s těmito produkty má jazyk PHP řadu výhod. Obsahuje velké množství vestavených funkcí, vykonávajících mnoho užitečných webových úloh, nativně

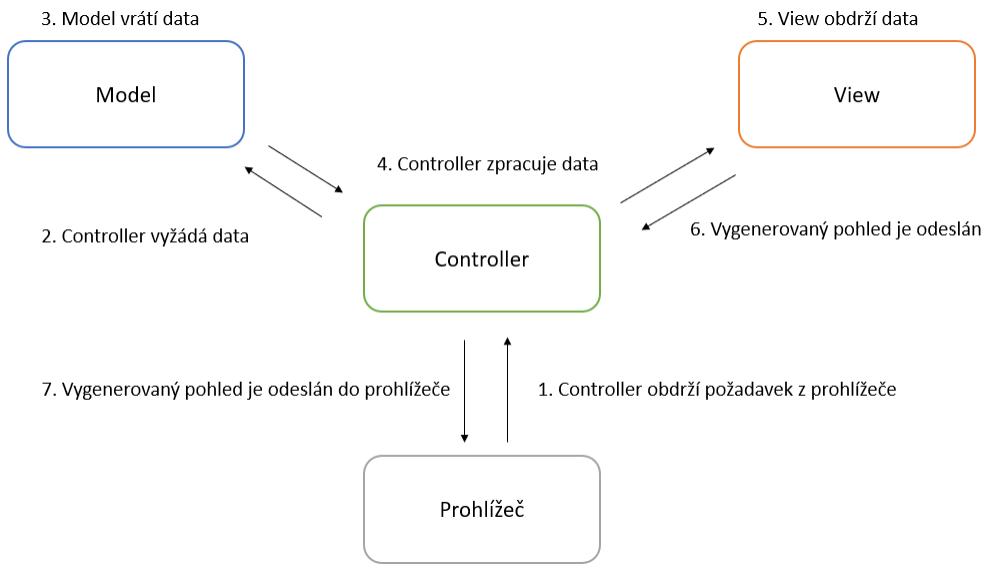
podporuje většinu databázových systémů a dokáže zpracovávat miliony návštěv denně pomocí jediného levného serveru (Welling & Thomson, 2017).

Hlavní předností jazyka PHP je rozšířenost jeho používání – PHP používá totiž 78,1 % všech webových stránek (W3Techs, nedatováno). PHP tak má skvělou online podporu, vyspělou komunitu a bohaté zdroje velkého množství informací. Další výhodou je flexibilita vývojového přístupu. V jazyce PHP můžeme vyřešit jednoduché úkoly jednoduše, stejně tak jednoduše můžeme vytvářet obrovské aplikace s pomocí vybraného frameworku založeného na architektuře MVC (Welling & Thomson, 2017).

1.4.3 MVC

MVC je architektonický vzor, který je založen na principu oddělení řídicí logiky (Controller), datové části (Model) a prezentační části (View) webových informačních systémů a aplikací. Model se stará o data a obchodní logiku aplikace. Jeho zodpovědností je také správa stavu aplikace a poskytovaní přístupu k tomuto stavu. View neboli pohled představuje uživatelské rozhraní aplikace, které slouží k zobrazení dat z modelu v uživatelsky přívětivém formátu. Controller neboli kontrolér propojuje Model a View a slouží k řízení toku dat. Controller poskytuje řídicí logiku aplikace, spravuje příchozí požadavky a přijímá uživatelské vstupy a vykresluje konečný výstup.

Obr. 3: Diagram MVC



Zdroj: Vlastní zpracování (2022)

Interakce mezi těmito třemi části je znázorněna na Obr. 3. Uživatel z prohlížeče požádá o přístup k systému, případně zadá vstupní parametry. Kontrolér obdrží požadavek, zpracuje ho a zavolá model s požadavkem na potřebná data. Model vyhledá data v datovém uložišti a vrátí data kontroléru. Kontrolér uloží data do proměnných, zpracuje je a předá pohledu proměnné s příslušnými daty. Pohled přijme data od kontroléru a vloží je do připravené šablony. Tento připravený pohled kontrolér odešle do prohlížeče, kde se on zobrazí uživateli (Čápka, 2012).

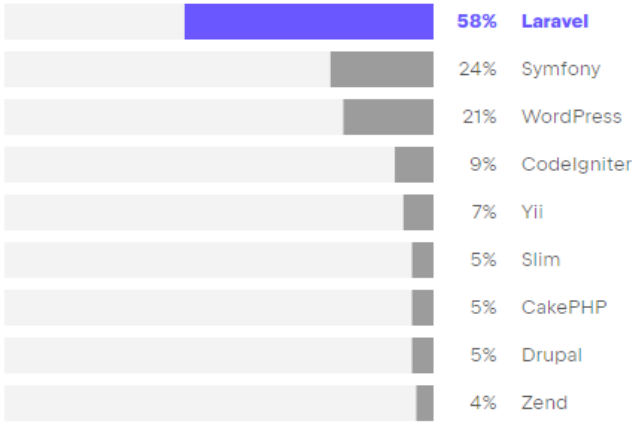
Využití MVC při návrhu webové aplikace má řadu výhod. MVC odděluje jednotlivé části aplikace, čímž zajišťuje přehlednost kódu, jeho snadnější údržbu a modifikaci. Další výhodou je lepší testovatelnost, protože existuje možnost testovat jednotlivé části a funkce. Jako poslední výhodu lze zmínit snadný přechod na jiný databázový systém. Architektonický vzor MVC používá většina moderních frameworků, jako je například Nette, Symfony, Laravel, Ruby On Rail, ASP .NET. a další.

1.4.4 PHP frameworky pro vývoj webových aplikací

Frameworky neboli aplikační rámce jsou klíčovou součástí moderního vývoje webových aplikací, protože vývojářům umožňují vytvářet webové aplikace efektivnějším a organizovanějším způsobem. Framework je kolekce předem připravených knihoven, nástrojů a šablon, které vývojářům umožňují zaměřit se na implementaci obchodní logiky namísto nízkoúrovňových implementačních detailů (Codecademy, 2021). Frameworky poskytují také sadu osvědčených postupů, standardů a konvencí, které zajišťují organizovanou strukturu, konzistentní kvalitu a udržitelnost kódu.

S rostoucí poptávkou po webových aplikacích se rozšířil počet frameworků pro vývoj webových aplikací. Mezi nejoblíbenější frameworky pro vývoj webových aplikací patří: CakePHP, Yii, Laravel, Symfony, Zend, CodeIgniter. Prvních pět zmíněných frameworků je postaveno na vzoru MVC, v CodeIgniteru je použití kontroléru nezbytnou součástí vývoje, ale použití modelu a pohledu je volitelné. Podle průzkumu umístěného na portálu Statista byly nejpoužívanějšími PHP frameworky mezi vývojáři v roce 2022 Laravel a Symfony (Vailshery, 2022). V průzkumu společnosti JetBrains se tyto dva PHP frameworky umisťují na stejných místech (JetBrains, 2022).

Obr. 4: Nejpoužívanější PHP frameworky a platformy



Zdroj: JetBrains (2022)

Laravel je full-stack framework, protože generuje HTML, obsluhuje web a spravuje databáze. Laravel má vynikající nástroj příkazového řádku s názvem Artisan, který obsahuje průkazy pro činnosti opakující se při tvorbě webových aplikací. Artisan umí Artisan umí generovat třídy a šablonový kód, spravovat databázi a databázové migrace, vypsat seznam rout atd. Všechny požadavky HTTP odesílané do prohlížeče a z něj zpracovává třída Controller. Mezi prohlížečem a kontrolérem je umístěna třída Middleware, která kontroluje HTTP požadavky vstupující do webového informačního systému. Ve třídě Viewer se využívá odlehčený šablonovací nástroj s názvem Blade, který slouží pro reprezentaci informací a bezproblémově spolupracuje s HTML. S využitím nástroje pro objektově-relační mapování (ORM) s názvem Eloquent se v aplikaci Laravel implementují aktivní záznamy, které usnadňují práci s databází. Každá tabulka v databázi MySQL má odpovídající model, který slouží k interakci s danou tabulkou. Aplikace Laravel umožňuje snadné začlenění balíčků třetích stran prostřednictvím několika příkazů Artisan, což výrazně usnadňuje vývoj a šetří čas. Většina těchto balíčků je dobře otestovaná, stabilní a zdarma k dispozici. Laravel má mnoho schopností, které činí možným rychlý vývoj aplikací, elegantní syntax usnadňující vykonávaní úloh, které je třeba dělat často, jako jsou autentizace, směrování, relace (sessions), ukládání do mezipaměti.

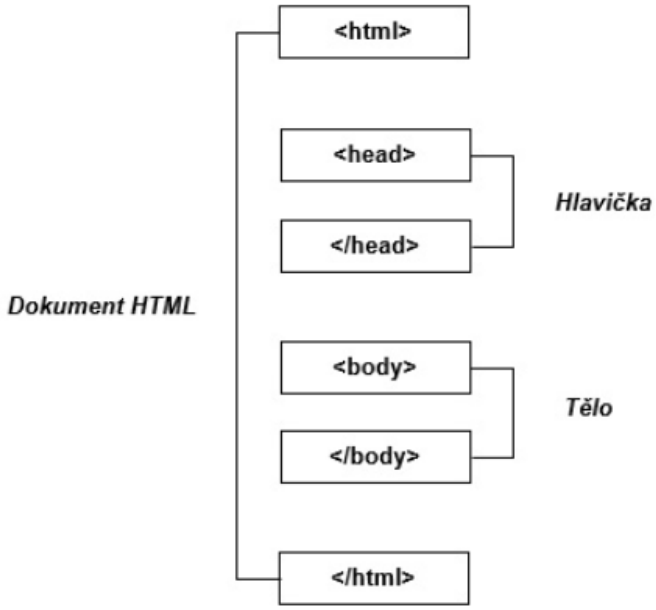
1.4.5 HTML

HTML (Hypertext Markup Language) je značkovací jazyk sloužící k usnadnění procesu tvorby webových stránek. Při tvorbě webové stránky si je třeba uvědomit, že všechny texty, které do webové stránky zapíšeme, slouží pouze jako informace pro webový

prohlížeč. Způsob, jak prohlížeč v okně webové stránky zobrazí, můžeme ovlivnit použitím řidičích příkazů jazyka HTML, označovaných jako tagy. HTML tagy jsou pokynem pro prohlížeč, jak text zobrazit na obrazovce, ale nemusí se uživateli nutně zobrazit. V jazyce HTML jsou tyto tagy pevné nebo předdefinované, což znamená, že názvy, které lze ve značkách použít, jsou omezeny na to, co jsou prohlížeče schopny rozpoznat.

HTML poskytuje strukturu, která definuje prvky, jako jsou tabulky, formuláře, seznamy a nadpisy, a určuje, kde začínají a končí různé části obsahu. Lze jej použít mimo jiné ke vkládání dalších formátů souborů, jako jsou videa, zvukové soubory, dokumenty typu PDF. Jazyk HTML je nejpoužívanějším jazykem při tvorbě webových stránek (Nixon, 2014). Jazyk HTML je nyní v aktuální verzi HTML5. Normy a doporučení pro jazyk HTML schvaluje nezávislé mezinárodní konsorcium W3C.

Obr. 5: Struktura webové stránky



Zdroj: Laurenčík (2019, s. 44)

Webová stránka, zapsaná v kódu HTML, se skládá ze dvou částí: textů, které budou na webové stránce zobrazeny, a HTML elementů. Obsah webové stránky je uzavřen mezi tagy <html> a </html>. Ostatní elementy musí být do toho elementu vnořené (Laurenčík, 2019). Hlavička obsahuje metadata, která se vztahují k celému dokumentu. Definují např. název dokumentu, jazyk, kódování, klíčová slova. Tělo dokumentu obsahuje veškerý zobrazovaný obsah stránky.

1.4.6 CSS

Druhou ze dvou základních technologií pro tvorbu webových stránek je CSS (Cascading Style Sheets). CSS je zkratka pro kaskádové styly a používá se k nastavení barev, písma a grafického rozvržení stránek. Určuje také, kdy mají být tato pravidla použita, a to na základě informací, jako je například typ zařízení, které se ke stránce připojuje. CSS lze použít nejen v jazyce HTML, ale v jakémkoli jazyce založeném na XML. Nejlepším přístupem je používat jazyk HTML k definování struktury (a pouze struktury) webových stránek, kdykoli je to možné, čímž položíme základ pro CSS, aby vědělo, kde má použít skutečný styl (Nixon, 2014). Tím, že co nejvíce oddělíme styl stránky od její HTML struktury, vlastně oddělíme obsah od vzhledu. To umožňuje rychle vytvořit několik verzí vzhledu naší stránky, aniž bychom museli v každé verzi znovu vytvářet obsah.

1.4.7 JavaScript

JavaScript je skriptovací jazyk na straně klienta, který slouží k přidání interaktivity na webové stránce. V kombinaci s CSS stojí JavaScript za dynamickými webovými stránkami. Skripty na straně klienta se běžně používají pro ověření dat před odesláním na webový server a upravení rozhraní v návaznosti na zpětnou vazbu uživatele.

JavaScript se poprvé objevil v prohlížeči Netscape Navigator v roce 1995 současně s přidáním podpory technologie Java do prohlížeče (Mendez, 2014). Kvůli původnímu mylnému dojmu, že JavaScript je odvozený od Javy, docházelo dlouhodobě k nejasnostem ohledně jejich vztahu. Pojmenování však bylo pouze marketingovým krokem, který měl novému skriptovacímu jazyku pomoci využít popularity programovacího jazyka Java. JavaScript získal na významu, když prvky webové stránky v jazyce HTML dostaly formálnější, strukturovanější definici v takzvaném objektovém modelu dokumentu (Document Object Model, DOM) (Nixon, 2014).

Nejlepším postupem pro použití JavaScriptu na webu je vytvořit celý web bez něj, a poté jej přidat tam, kde může zlepšit uživatelský zážitek (proces se nazývá „postupné vylepšení“) (Mendez, 2014). Tento postup pomůže zajistit funkčnost webových stránek i bez JavaScriptu.

1.5 Tvorba webový stránek pro jazykovou školu

V předchozích kapitolách jsme popsali technologie pro vytvoření webových stránek. V této kapitole se zaměříme na jejich obsah.

1.5.1 Obsah webových stránek

Před vytvořením obsahu je vhodné definovat obsahovou strategii. Nejdřív musíme formulovat cíle webových stránek a promyslet jejich strukturu. Struktura webových stránek by měla být přehledná, jednoduchá a každá stránka by měla plnit určitý účel. Správnou struktura webových stránek umožní vyhledávačům snadno najít webové stránky a zajistí vyšší pozici ve výsledcích vyhledávání. Nejdůležitější informace ("O nás", "Nabídka", "Ceník", "Kontakty") by měly být dostupné ihned po kliknutí, a proto je lepší vyhnout se rozbalovacím nabídkám, které se mohou v různých prohlížečích chovat nepředvídatelně (LangLion Blog, 2014). Při tvorbě webových stránek bychom měli mít neustále na paměti, že potenciální zákazník, který naše stránky navštíví, chce získat konkrétní informace – seznámit se s naší nabídkou a cenami. Tyto záložky by měly být viditelné ihned po vstupu na stránku. Skrývání tohoto typu informací může potenciální zákazníky účinně odradit. Pokud nebudou moci snadno najít to, co je zajímá, stránku zavřou a jako zákazníci budou ztraceni.

Webová stránka, kromě funkce informační, musí plnit i funkci prodejní. Z tohoto důvodu by měl být povinnou součástí webové stránky formulář umožňující přihlášení na kurzy. Může se jednat o jednoduchý formulář, kde zákazníci mohou zanechat své jméno, telefon, e-mailovou adresu a výběr kurzů/tříd (LangLion Blog, 2014). Na svých webových stránkách by škola měla mít snadno přístupnou záložku Kontakt, kde potenciální zákazník najde všechny adresní údaje (název školy, adresu, poštovní směrovací číslo, město, telefon, e-mailové adresy a přístupovou mapu).

1.5.2 Cílové stránky

Cílová nebo vstupní stránka (z anglického termínu landing page) je webová stránka, která je vytvořena pro marketingové a reklamní účely (Mioweb, nedatováno). Cílové stránky jsou v online marketingu klíčové, protože nabízejí návštěvníkům vysoce cílené a zaměřené vzkazy. Poskytují jasnou a stručnou nabídku, která přímo hovoří o potřebách a přáních návštěvníka, jehož má povzbudit k provedení požadované akce, například k vyplnění formuláře, nákupu nebo přihlášení k odběru newsletteru. Cílové stránky

obvykle obsahují několik klíčových prvků, které společně vytvářejí účinné přesvědčivé sdělení a jsou doplněny titulky a podtitulky, které jsou jasné, stručné, poutají pozornost a obsahují výraznou výzvu k akci, která je také nazývána CTA (z anglického termínu call-to-action). Ta návštěvníky nasměruje k provedení požadované akce. Vstupní stránky jsou tak cenným nástrojem pro podporu konverzí a generování potenciálních zákazníků. Konverze je situace, kdy návštěvník webu nebo příjemce marketingového sdělení provede požadovanou akci (MarketingPPC, 2022).

Design, rozložení a obsah vstupní stránky mohou ovlivnit její efektivitu při přeměně návštěvníků na zákazníky. Dále budou popsány zásady a osvědčení postupy pro vytváření efektivních vstupních stránek. Vstupní stránka by měla být přehledná a cílená. Obsah vstupní stánky by neměl obnášet nadbytečné informace. Informace by měly být maximálně srozumitelné a jednoduché, neměl by se využívat odborný jazyk nebo zkratky. Výhody nabízených služeb nebo spoluprací by měly vynikat hned (Koďousková, 2021).

Podle statistik si pouze 2 z 10 lidí přečtou celou vstupní stránku, ale 8 z 10 lidí si přečte titulek (Chowles, 2023). Po vstupu na neznámou stránku uživatel věnuje pouze pár vteřin rychlému průzkumu a následně vyhodnotí, zda má smysl studovat další obsah. Proto je doporučeno umisťovat podstatné informace již v nadpisu, případně v několika větách pod ním. Statistika také tvrdí, že umístění více nabídek na vstupní stránce může snížit konverze až o 266 % (Chowles, 2023). Cílová stránka by měla obsahovat jednu jasně určenou nabídku a neměla by nutit uživatele k více akcím.

Vizuální podoba stránky hraje stejně významnou roli jako obsahová náplň stránky. Uživatelé si z 80 % spíše přečtou obsah, který obsahuje barevné vizuální prvky (Chowles, 2023). Zařazení obrázků nebo videí na vstupní stránku může pomoci zprostředkovat výhody nabídky a učinit stránku poutavější, ale grafické prvky by měly být použity strategicky a neměly by odvádět pozornost od výzvy k akci. Navíc bez ohledu na délku cílové stránky je třeba členit text pomocí podtitulků, odrážek či vizuálně oddělených bloků, a tak pomáhat návštěvníkovi s jeho snadnějším čtením (Pilka, 2018).

Nejdůležitějším prvkem cílové stránky je výzva k akci. Kvalitní výzva měla by obsahovat akčně zaměřené sloveso v rozkazovacím způsobu: získejte, přihlaste se, objednejte, objevte atp. Ideálně by měla výzva k akci přesně popisovat, co může návštěvník webu po provedení akce očekávat. Pro výzvu k akci se nejčastěji využívají tlačítka, která můžou mít rozlišné funkce. Patří mezi ně například přesměrování uživatele k objednávce,

přechod na stránku s registrací nebo odeslání kontaktního formuláře. Tlačítko by mělo být dostatečně velké a mít kontrastní barvu vůči pozadí. V případě, že je cílová stránka delší, se doporučuje umístit na ni konverzní tlačítko vícekrát (Pilka, 2018).

1.5.3 Optimalizace pro vyhledávače

Optimalizace pro vyhledávače (SEO z anglického termínu Search Engine Optimization) je optimalizace webových stránek za účelem zvýšení jejich viditelnosti a umístění na stránkách s výsledky vyhledávání. Zahrnuje řadu technik, včetně výzkumu klíčových slov, optimalizace na stránce (například optimalizace titulních tagů, meta popisů a obsahu), budování odkazů a technické optimalizace (například zlepšení rychlosti webu a jeho přizpůsobení mobilním zařízením) (Search Engine Land, 2023).

Optimalizace klíčových slov je důležitou součástí SEO a zahrnuje identifikaci a používání klíčových slov, která jsou relevantní pro obsah webových stránek. Podle Erica Engeho (Enge, Spencer, & Stricchiola, 2015) existuje několik zásad a technik, které mohou pomoci optimalizovat klíčová slova pro účely SEO. Jednou z nejdůležitějších zásad optimalizace klíčových slov je provedení pečlivého průzkumu klíčových slov. Ten zahrnuje identifikaci klíčových slov, která jsou relevantní pro obsah webových stránek a která uživatelé pravděpodobně použijí při vyhledávání tohoto obsahu. Výzkum klíčových slov lze provádět pomocí různých nástrojů, jako je například Google Keyword Planner a SEMrush.

Po identifikaci relevantních klíčových slov je důležité začlenit je do obsahu webových stránek přirozeným a organickým způsobem. To zahrnuje použití klíčových slov v nadpisech, meta popisech a v hlavním textu obsahu. Je však důležité vyvarovat se přeplňování klíčových slov, které může být vyhledávači penalizováno a může zhoršit uživatelský zážitek. Kromě začlenění klíčových slov do obsahu webových stránek je také důležité optimalizovat další prvky na stránce, jako jsou obrázky, videa a adresy URL (Search Engine Land, 2023). To může zahrnovat začlenění relevantních klíčových slov do alt tagů a názvů souborů u obrázků a videí, ale také začlenění klíčových slov do struktury URL webových stránek.

V neposlední řadě je důležité pravidelně sledovat a upravovat úsilí o optimalizaci klíčových slov na základě změn v algoritmech vyhledávačů a chování uživatelů. To zahrnuje pravidelnou analýzu analytických dat webových stránek a úpravu používání klíčových slov a dalších technik SEO podle potřeby.

1.6 Požadavky na webový informační systém

Pro návrh kvalitního informačního systému je nezbytné pochopit a zdokumentovat požadavky na systém. Existuje několik typů požadavků, které je třeba při návrhu informačního systému zohlednit. Mezi to patří funkční požadavky, nefunkční požadavky, požadavky uživatelů, obchodní požadavky a právní a regulační požadavky (Bruckner, a další, 2012).

Funkční požadavky popisují funkce, které musí systém plnit, aby vyhověl potřebám uživatelů. Mohou zahrnovat zadávání dat, vyhledávání dat, analýzu dat a jejich zobrazování. Podle Brucknera (2012) funkční požadavky definují, co má systém dělat, jaké má být chování systému a jeho interakce s uživateli a jinými systémy. Jinými slovy, funkční požadavky popisují činnosti, které musí systém vykonávat, aby dosáhl svého zamýšleného účelu. Funkční požadavky lze dále rozdělit do dvou kategorií: obchodní požadavky a uživatelské požadavky. Obchodní požadavky popisují cíle a záměry organizace, které bude systém podporovat, zatímco uživatelské požadavky popisují potřeby a očekávání uživatelů systému.

Nefunkční požadavky popisují kvalitativní charakteristiky systému, jako je výkonnost, spolehlivost, udržovatelnost, škálovatelnost, bezpečnost a použitelnost (Riordan, 2005). Brucknera et al. (2012) uvádí, ze nefunkční požadavky definují, jak se má systém chovat, jeho výkon, kvalitu a další atributy, které nesouvisejí s funkčností systému. Požadavky na výkonnost popisují schopnost systému zpracovávat data a reagovat na požadavky uživatelů ve stanoveném časovém rámci. Požadavky na spolehlivost popisují schopnost systému fungovat správně a bez chyb v průběhu času. Požadavky na dostupnost popisují schopnost systému být k dispozici pro použití v případě potřeby. Požadavky na udržovatelnost popisují schopnost systému snadno se udržovat a aktualizovat. Požadavky na škálovatelnost popisují schopnost systému zvládat zvýšenou pracovní zátěž při růstu systému. Požadavky na bezpečnost popisují schopnost systému chránit data před neoprávněným přístupem. Požadavky na použitelnost popisují schopnost systému být uživatelsky přívětivý a snadno použitelný (Riordan, 2005).

Riordan (2005) zdůrazňuje důležitost shromažďování požadavků při návrhu databázového systému. Proces sběru požadavků obvykle zahrnuje několik kroků, mezi něž patří identifikace zúčastněných stran, sběr dat, analýza dat, definice požadavků a ověřování požadavků. Identifikace zúčastněných stran zahrnuje určení osob a skupin,

které budou systémem ovlivněny a které mohou poskytnout vstupní informace o jeho návrhu. Sběr dat zahrnuje shromažďování informací o uživatelích systému, obchodních procesech a dalších relevantních faktorech. Analýza údajů zahrnuje analýzu shromážděných údajů s cílem identifikovat vzory, trendy a další poznatky, které mohou být podkladem pro návrh systému. Definování požadavků zahrnuje zdokumentování požadavků na systém, včetně funkčních a nefunkčních požadavků, požadavků uživatelů, obchodních požadavků (Davis & Yen, 1998). Ověřování požadavků zahrnuje jejich projednání se zúčastněnými stranami, aby se zajistilo, že jsou úplné, přesné a proveditelné.

Požadavky mohou pocházet z různých zdrojů, včetně zainteresovaných stran, existující dokumentace, právních a regulačních omezení a analýzy konkurence. Zúčastněné strany mohou poskytnout cenné podněty k požadavkům na systém na základě svých potřeb a očekávání. Existující dokumentace může zahrnovat dokumentaci současných obchodních procesů, zásad a postupů, právní a regulační omezení definují omezení, která musí systém splňovat. Analýza konkurence zahrnuje analýzu systémů používaných konkurencí, která poskytuje přehled o funkcích a možnostech, které jsou pro uživatele důležité (Bruckner, a další, 2012).

1.7 Testování použitelnosti systému

Základní podmínkou úspěchu webových stránek není jen kvalitní obsah, ale i dobrá použitelnost. Dobře navržené stránky jsou přehledné a srozumitelné, návštěvník se na nich snadno orientuje a intuitivně je ovládá. Pokud stránky neplní svůj účel, problém často spočívá v překážkách v jejich použitelnosti. Proto je potřeba provádět testování použitelnosti. Testy použitelnosti spočívají v pozorování reprezentativních koncových uživatelů, kteří používají produkt k provádění realistických úkolů a k odhalení věci, které je matou nebo zklamávají (Rubin & Chisnell, 2008).

Existuje široká škála testů, které lze provádět, od klasických experimentů se složitými testovacími plány, až po velmi neformální kvalitativní studie s jediným účastníkem. Každý přístup k testování má jiné cíle a jiné požadavky na čas a zdroje (Davis & Yen, 1998). V této práci je kladen důraz na neformálnější, méně složité testy určené pro rychlé získání výsledků při vývoji malého webového informačního systému.

Jeffrey Rubin (Rubin & Chisnell, 2008) uvádí tři typy testů použitelnosti: explorační, hodnotící, validační, a to podle přibližného okamžiku v životním cyklu vývoje produktu, ve kterém se každý z nich provádí.

Explorační testování se provádí na začátku vývojového cyklu, kdy je produkt ještě v předběžné fázi definování a navrhování (proto se někdy nazývá formativní). V této fázi vývojového cyklu by měl být definován profil uživatele a měla by být provedena analýza úkolů produktu. Hlavním cílem exploračního testování je prověřit vhodnost předběžných návrhů (Rubin & Chisnell, 2008). Význam tohoto typu testování nelze podceňovat, neboť právě v tomto okamžiku se přijímají kritická rozhodnutí o návrhu, která určují směr celého projektu. Pokud projekt začíná s chybnými předpoklady o uživateli, je téměř zaručeno, že produkt bude mít později problémy s použitelností.

V průběhu vývoje se provádí hodnoticí test. Účelem hodnoticího testu je rozšířit zjištění exploračního testu o hodnocení použitelnosti prvků a funkcionalit produktu na nižší úrovni. Cílem tohoto testu je prozkoumat a vyhodnotit, nakolik efektivně byl návrh implementován. Na konci vývojového cyklu se obvykle provádí validační test, jehož cílem je vyhodnotit, jak všechny součásti produktu fungují společně, protože komponenty jsou často vyvíjeny relativně izolovaně od sebe (Rubin & Chisnell, 2008).

Před provedením formálního testu použitelnosti se nejdřív musí vypracovat plán testu a zvolit testovací prostředí. Pro testování je nutné vybrat vhodné participanty. Nejvhodnějšími jsou participanti, kteří patří do cílové skupiny, ale není to závazná podmínka. Dál je nutné připravit materiály pro test. Nejdřív se sepíše seznam úkolů, které se participanti mají pokusit splnit. Pak úkoly rozšířené do scénářů, které přidávají podrobnosti o kontextu, který potřebují znát, aby mohli úkol splnit. Následuje vlastně uživatelské testování použitelnosti pod dohledem moderátora (Rubin & Chisnell, 2008). V průběhu nebo po skončení testu účastníci vyplní dotazník, dotazník, který sbírá informace o tom, jak participanti „vnímají“ jednoduchost používání.

2 Metodika

Vývoj webového informačního systému zahrnuje řadu etap, které jsou nezbytné pro zajištění dodání funkčního výsledného systému. Těmito základními etapami jsou analýza, návrh, implementace a testování.

V etapě analýzy bude probíhat příprava na vytvoření webového informačního systému, bude určen jeho účel a budou definovány cíle, které by měly být dosaženy pomocí informačního systému. Tato etapa bude také zahrnovat shromáždění požadavků na informační systém, zejména funkčních požadavků, které budou popisovat funkcionality systému, a nefunkčních požadavků, které budou definovat, jak se má systém chovat. Po shromáždění požadavků proběhne jejich analýza, na jejímž základě bude vytvořen diagram případu užití systému a proběhne předběžné rozdělení systému na funkční části.

Na základě analýzy funkcionalit systému budou navrženy a popsány objekty, aplikační pravidla a integritní omezení, následně bude vytvořen konceptuální datový model. Konceptuální model bude následně převeden na fyzický datový model, na jehož základě bude vytvořena databáze, která bude sloužit pro uschovávaní dat. V této etapě bude také proveden návrh uživatelského rozhraní, bude určen vizuální vzhled stránek, jejich základní struktura a rozložení jednotlivých prvků na stránkách webového informačního systému.

Po dokončení návrhu informačního systému bude možné postoupit k jeho vlastní implementaci. Předpokladem pro zahájení implementace bude instalace implementačních nástrojů. Následně na základě hotových návrhů informačního systému vytvořených v předchozí fázi bude vytvořen software pomocí technologií popsaných v teoretické části této práce, zejména frameworku Laravel, programovacího jazyka PHP a dalších.

Implementovaný informační systém bude otestován, aby se zajistilo, že funguje podle očekávání a zda splňuje požadavky specifikované v etapě analýzy. Testování použitelnosti systému proběhne na základě seznamu předdefinovaných úkolů a uživatelských scénářů. Na základě nedostatků odhalených testováním budou představeny návrhy na jeho zlepšení.

3 Praktická část

Tato kapitola se bude zabývat procesem vytvoření webového informačního systému pro jazykovou školu. Vytvoření webového informačního systému proběhne ve čtyřech etapách, které jsou zmíněné ve druhé kapitole této práce. Pro analýzu, návrh, implementaci a testování použijeme teoretické znalosti a technologie uvedené v první kapitole této práce.

3.1 Analýza požadavků na informační systém jazykové školy

Definice a důkladné pochopení požadavků, které musí systém splňovat, jsou základními prvky úspěšného vývoje jakéhokoliv webového informačního systému. V této kapitole se budeme zabývat funkčními a nefunkčními požadavky na webový informační systém jazykové školy. Specifikace těchto požadavků zajistí splnění potřeb budoucích uživatelů systému a jeho správné fungování.

3.1.1 Cíl a rozsah informačního systému

Webový informační systém bude vyvinut pro jazykovou školu SpeakSmart, což je start-up nápad autora této práce. Autor práce ze své mnoholeté zkušenosti s výukou cizích jazyků v rolích studenta a vyučujícího dospěl k závěru, že osobnost lektora hraje ve výuce cizích jazyků významnou roli. Osobní vlastnosti lektorů určují způsob, jakým budou látku předávat. Díky tomu stejná lekce z jazykové učebnice může probíhat velmi odlišně, a to jen díky tomu, že je vyučovaná dvěma různými lektory. Podle postřehu autora této práce lze rozlišit dva aspekty osobnosti lektora, které mají největší vliv na efektivitu výuky cizích jazyků. Těmito aspekty jsou vyučující styl a přísnost lektora. Studenti se liší ve způsobu, jakým vnímají a zpracovávají informace, což je dáno tím, že mají různé styly učení. Pokud vyučující styl lektora není kompatibilní s určitým stylem učení studenta, k učení příliš nedochází. Nelze jednoznačné určit, který lektor je lepší – shovívavý nebo striktní. Lze však říct, že pro studenty s různými cíli a s různými úrovněmi schopnosti motivovat sám sebe jsou vhodní lektoři s různými úrovněmi přísnosti. Student, který se připravuje na jazykovou zkoušku povinnou pro studium v zahraničí a nedokáže se přinutit k systematickému učení, prospěje pouze tehdy, když bude lektor velmi striktní a bude pečlivě opravovat každou jeho chybu. Na druhu stranu studentovi, který se chce naučit jazyk jen pro komunikaci na dovolené a kvůli porozumění memům, výuka s velmi

striktním lektorem vyhovovat nebude. V zájmu zvýšení efektivity výuky je třeba věnovat pozornost kompatibilitě vyučovacích stylů lektoru a stylů učení studentu. Dalším aspektem, kterému se musí věnovat pozornost je přísnost lektora. Informační systém by měl usnadňovat výběr nejvhodnějšího lektora pro každého studenta. Dalším úkolem informačního systému by mělo být představení školy na internetu a získání kontaktů na studenty, kteří by měli zájem o výuku ve škole SpeakSmart.

3.1.2 Funkční požadavky

Funkční požadavky na informační systém jazykové školy SpeakSmart jsou uvedeny níže.

Anonymním návštěvníkům by měl systém umožnit:

- zobrazit si údaje o jazykové škole, jako je adresa, metodika a materiály využívané ve škole, proces zahájení výuky v jazykové škole;
- zobrazit si čtyři náhodně zvolené kurzy a jejich krátký popis;
- zobrazit si tři náhodně zvolené lektory s informacemi o jejich přísnosti a vyučovacím stylu;
- zobrazit si balíčky, ve kterých je možné zakoupit lekci a jejich ceny;
- vyplnit a odeslat kontaktní formulář. Pro odeslaní by bylo nutné uvést své jméno, e-mailovou adresu a zvolit jazyk, o který má zájem, zaměření kurzu, typ a způsob výuky, vyučovací styl a přísnost lektora, o které má návštěvník zájem;
- přejít na sociální síť školy pomocí odkazu na webové stránce;
- přihlášení k aktivnímu účtu. Pro přihlášení by bylo nutné zadat unikátní e-mailovou adresu a heslo. Mělo by proběhnout ověření, že hodnoty zadané uživatelem odpovídají hodnotám v databázi.

Přihlášenému uživateli by měl systém umožnit:

- odhlášení ze svého účtu;
- zobrazit si svůj profil;
- měnit údaje ve svém profilu;
- změnit své heslo. Pro změnu hesla by musel uživatel zadat stávající heslo, nové heslo a potvrdit nové heslo.

Přihlášenému uživateli s rolí Administrátor by systém měl umožnit:

- přidávat nové uživatele, odstraňovat uživatele a měnit údaje jiných uživatelů;
- zobrazeni a správu studentů školy;
- zobrazení a správu lektorů pracujících ve škole;
- zobrazení a správu kurzů nabízených ve škole;
- zobrazení a správu vyučujících hodin ve škole;
- zapisovat studenty na vyučující hodiny.

Přihlášenému uživateli s rolí Editor systém by měl umožnit:

- zobrazení cenových plánů;
- zobrazení údajů získaných z kontaktních formulářů.

3.1.3 Nefunkční požadavky

Informační systém by měl:

- běžet na serveru a být přístupný z webového prohlížeče;
- být kompatibilní s prohlížeči Google Chrome, Microsoft Edge, Opera Mini, Firefox a Safari;
- být spolehlivý, fungovat správně, bez chyb a výpadků;
- být zabezpečen přístupovými údaji a chránit citlivé informace před neoprávněným přístupem. Heslo by se mělo ukládat do databáze v zašifrované formě;
- být dále rozšiřitelný. Návrh informačního systému by měl umožňovat přidávání modulů a bloků;
- mít responzivní design. Obsah informačního systému by se měl korektně zobrazovat na různých zařízeních;
- být snadno použitelný. Grafické uživatelské rozhraní by mělo umožňovat intuitivní ovládání. Navigační menu by mělo být pro uživatele přehledné a poskytovat rychlou navigaci do všech částí informačního systému.

3.1.4 Vymezení modulů informačního systému

Po shromáždění funkčních požadavků na informační systém je možné určit a vymezit jeho moduly, které by splňovaly tyto požadavky. Na základě analýzy a kategorizace požadavků byl systém rozdělen na 4 menší části, které by měly zajistit snadnější spravování a údržbu informačního systému. Dále bylo určeno, jak by jednotlivé moduly měly být propojené, aby bylo možné požadovaných funkcionalit dosáhnout.

Uživatelé

Tento informační systém bude vyvinut pro jazykovou školu a měl by sloužit dvěma základním skupinám uživatelů – administraci jazykové školy a potencionálním studentům této jazykové školy. Uživateli informačního systému by byla přiřazena jedna ze 4 rolí: Administrátor, Editor, Student, Lektor.

Tento modul by především měl sloužit pro uschovávaní přístupových práv pro uživatele informačního systému a informací o nich. Uživatel s rolí Administrátor by měl mít přístup ke všem funkcionalitám systému a moct spravovat celý systém. Administrátor by měl mít možnost uschovávat podrobné informace o jazykové škole, vkládat a editovat tyto informace, dále by měl mít možnost přidávat další uživatele do systému.

Uživatel s rolí Editor by měl mít přístup jenom ke funkcím, které zajišťují správu webu, ale i ke kontaktům obdrženým od návštěvníků webové stránky. Potencionální studenti neboli anonymní uživatelé by si měli moct zobrazovat informace o jazykové škole na veřejné webové stránce.

Uživatelům s rolí Lektor a Student by nebyl přístup k informačnímu systému poskytnut. V modulu Uživatelé by se pouze uschovávaly informace o lektorech školy a studentech. Jedná se zejména o osobní informace a kontaktní údaje, jako je jméno, příjmení, datum narození, telefonní číslo, pohlaví, e-mail, fotografie. U lektorů by rovněž měl byl uveden jazyk, který vyučují, a to pouze jeden. Před začátkem práce se studenty by lektor měl projít tréninkem a absolvovat osobnostní testy, které by měly určit jeho vyučovací styl a úroveň přísnosti. Tyto údaje musí být také uschované v informačním systému. Za tímto účelem musí existovat katalog vyučovacích stylů a úrovní přísnosti s jejich krátkou specifikací. Tento katalog by měl být spravován Administrátorem. U studentů by se měl uschovávat jazyk, který se učí. V seznamech studentů a lektorů by mělo být umožněno vyhledávání podle všech jednotlivých údajů.

Kurzy

Tento modul by měl sloužit k evidenci kurzů nabízených v jazykové škole. U všech kurzů by měl být uveden jazyk, ve kterém se kurz vyučuje, úroveň znalosti jazyka, která by měla být dosažena po jeho absolvování, specializace nebo zaměření kurzu, věková kategorie, pro kterou je kurz vhodný a čas, který bude muset strávit student týdně nad vykonáním domácích úkolů. Tyto údaje by se měly volit z existujících katalogů, které by měl spravovat Administrátor.

Výuka

Tento modul úzce spojuje předchozí moduly. V tomto modulu by se ke kurzu měl přiřadit lektor, který bude kurz vyučovat. Dále by se mělo specifikovat datum zahájení výuky kurzů, předpokládané datum ukončení výuky kurzu, dny v týdnu, ve kterých bude výuka probíhat a délka jednotlivých vyučovacích hodin. U výuky by mělo jít uvést, jestli bude probíhat osobně nebo online, či zda bude skupinová nebo individuální. Dále musí být uvedena cena vyučovaného kurzu. Systém by měl umožňovat zapsat studenty na výuku celého kurzu. Výuka kurzu by měla mít jednoznačný identifikátor.

Web

Tento modul by měl sloužit ke spravování obsahu webu a také k zobrazení kontaktů a zpráv obdržených od návštěvníků webu. Uživatel s rolí Editor by mohl upravovat obsah webu, spravovat informace o cenových nabídkách. Jak již bylo uvedeno, uživatelé s rolí Student a Lektor by přístup k systému neměli, jelikož systém pro ně nebude poskytovat žádné funkcionality. Přidání funkcionalit pro studenty a lektory je jednou z možností rozšíření systému.

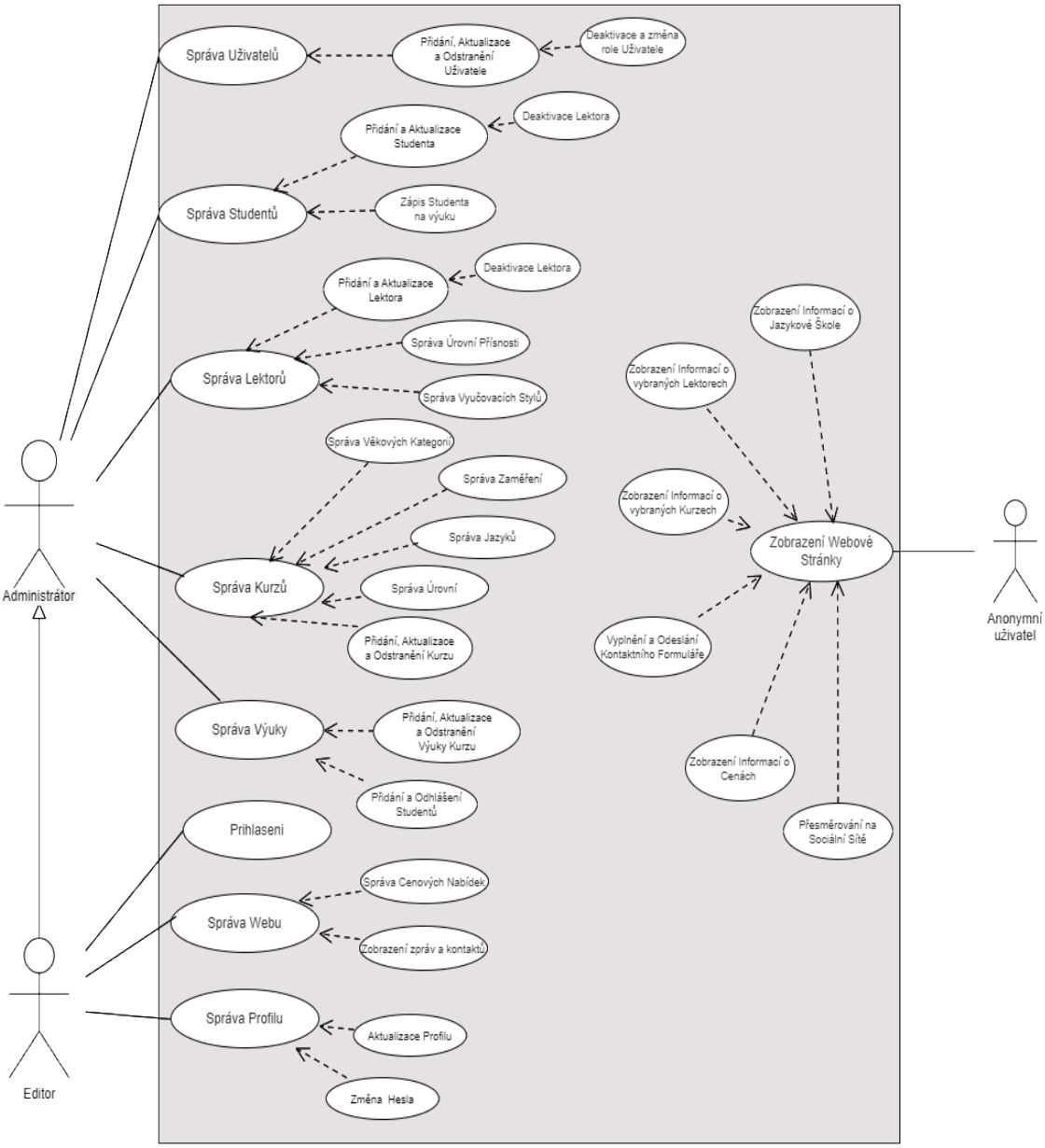
Pro lepší znázornění a vizualizaci funkčních požadavků a interakcí systému z pohledu jeho uživatelů byl vytvořen diagram případů užití. Tento diagram zobrazuje různé případy užití, které popisují způsoby interakce uživatelů se systémem za účelem dosažení jejich cílů. Diagram na Obr. 6 by měl pomoct zajistit, aby byly během procesu vývoje zohledněny všechny potřeby uživatelů.

3.2 Návrh datové vrstvy

V předchozí kapitole jsme sepsali požadavky kladené na informační systém jazykové školy. V této kapitole budeme pokračovat konceptuálním neboli myšlenkovým návrhem databáze, při kterém budeme postupovat technikami ERA-modelovaní. Na základě

konceptuálního datového modelu dále vytvoříme v systému řízení báze dat MySQL fyzický datový model.

Obr. 6: Diagram případů užití



Zdroj: Vlastní zpracování (2022)

3.2.1 Konceptuální datový model

První fází konceptuálního modelování je identifikace entit v systému. Entity můžeme blíže popsat určitými charakteristickými vlastnostmi. Současně s identifikací konkrétních entit v systému budeme také popisovat jejich jednotlivé atributy. Základ jazykové školy tvoří dvě nejdůležitější datové entity. První z nich je uživatel, zejména uživatel s rolí

Student nebo Lektor. U uživatele s rolí Student by bylo nutné uschovávat jeho jméno, příjmení, telefonní číslo, e-mail, datum, narození, pohlaví a jazyk, který se učí. Mezi atributy uživatele s rolí Lektor, stejně jako u Studenta, patří jméno s příjmením, telefonní číslo, e-mail, datum, narození pohlaví, a navíc datum nástupu. Uživatel s rolí Lektor je také charakterizován úrovní jeho přísnosti, má uvedené informace o jeho vyučujícím stylu a vždy jednom jazyku, ve kterém vyučuje. V entitě Uživatel budou také definovány údaje o uživatelském účtu pro informační systém. Uživatel má přidělené právo, které definuje jeho roli a rozsah přístupu k systému. Může se nacházet v určitém stavu, má uvedený e-mail a heslo, jejichž kombinace slouží k přístupu do systému. Dalším atributem je nepovinné foto.

Druhou základní entitou je Kurz, který má název, krátký popis, informace, do kolika lekcí je rozdělen a kolik hodin bude muset strávit na domácích úkolech týdně. U kurzu je vždy uvedené jeho zaměření, jazyk, ve kterém se vyučuje, úroveň znalosti jazyka, která by měla být dosažena po jeho absolvování a věková kategorie, pro kterou je kurz vhodný.

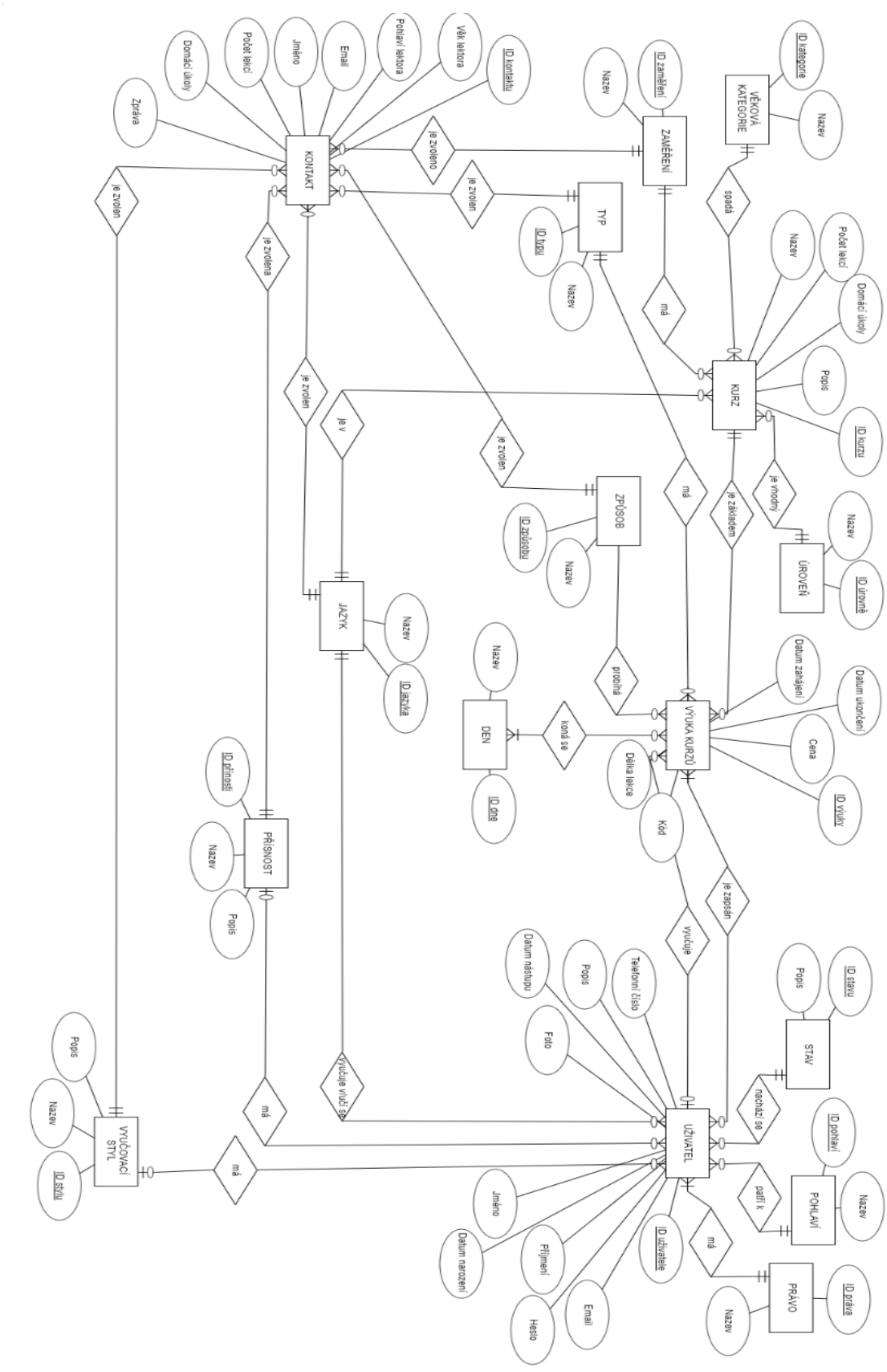
V okamžiku, kdy si alespoň jeden student zakoupí kurz, musíme zajistit jeho výuku. Tímto vzniká další entita výuka kurzu, která tvoří určitou relaci mezi studenty, lektory a kurzy. Na výuku kurzu je vždy zapsán jeden nebo více studentu a kurz vyučuje vždy jeden lektor. Výuka probíhá určitým způsobem v určité dny v týdnu a má určený typ. Dalšími atributy výuky jsou datum zahájení, datum ukončení, cena výuky pro studenta, délka jednotlivé lekce a kód výuky.

Při popisu entit už jsme se dotkli vazeb mezi entitami, ale explicitně jsme je nepopisovali. Nyní se zaměříme na podrobnější popis relací mezi entitami a určení jejich kardinality.

- Relace mezi entitami *Úroveň* a *Kurz* – pod každou úroveň může spadat více kurzů, anebo žádný, ale každý kurz vždy musí spadat pod jednu určitou úroveň. Relace 1: N.
- Relace mezi entitami *Jazyk* a *Kurz* – v každém jazyce může být vyučováno více kurzů, anebo žádný, ale každý kurz vždy musí být vyučován právě v jednom jazyce. Relace 1: N.
- Relace mezi entitami *Věková Kategorie* a *Kurz* – pro každou věkovou kategorii může být nabízeno více vhodných kurzů, anebo žádný, ale kurz vždy musí být vhodný pro právě jednu věkovou kategorii. Relace 1: N.

- Relace mezi entitami *Zaměření* a *Kurz* – do každého zaměření může patřit více kurzů, anebo žádný, ale kurz musí mít vždy mít právě jedno zaměření. Relace 1:N.
- Relace mezi entitami *Kurz* a *Výuka Kurzu* – každý kurz může být základem pro výuku jednou, anebo vícekrát, ale jednotlivá výuka se musí vždy zakládat na jednom kurzu. Relace 1: N.
- Relace mezi entitami *Typ* a *Výuka Kurzu* – každým typem může být vyučováno více kurzů, anebo žádný, ale výuka musí vždy mít jeden typ. Relace 1: N.
- Relace mezi entitami *Způsob* a *Výuka Kurzu* – stejným způsobem může být vyučováno více kurzů, anebo žádný, ale jednotlivá výuka musí vždy probíhat jedním způsobem. Relace 1: N.
- Relace mezi entitami *Výuka Kurzu* a *Den* – v každém dne týdne se může konat více výuk anebo žádná, výuka se musí konat alespoň v jenom dne, ale může se konat i ve více dnech. Relace M: N.
- Relace mezi entitami *Výuka Kurzu a Uživatel s rolí Student* – každý student se může zapsat do žádného, jednoho nebo více vyučovaných kurzů, na výuce kurzu může být zapsán jeden student, více studentů, anebo žádní studenti. Relace M: N.
- Relace mezi entitami *Uživatel s rolí Lektor* a *Výuka Kurzu* – každý lektor může vyučovat jeden nebo více kurzů, ale nemusí vyučovat žádný, výuka kurzu je vždy zajištěna jedním lektorem. Lektor musí vyučovat ve stejném jazyce, ve kterém se vyučuje kurz. Relace 1: N.
- Relace mezi entitami *Uživatel s rolí Lektor/Student* a *Jazyk* – v každém jazyce může vyučovat více lektorů, anebo žádný, ale lektor vyučuje vždy v jednom jazyce. Pro studenta platí, že každý jazyk se může učit více studentů, ale student se učí vždy jeden jazyk. Relace N:1.
- Relace mezi entitami *Uživatel s rolí Lektor* a *Vyučovací Styl* – stejný vyučovací styl může mít více lektorů, ale lektor musí vždy mít jeden vyučovací styl. Relace N:1.
- Relace mezi entitami *Uživatel s rolí Lektor* a *Přísnost* – stejnou úroveň přísnosti může mít jeden lektor, více lektorů nebo žádný, ale každý lektor vždy musí mít jednu úroveň přísnosti. Relace N:1.

Obr. 7: Konceptuální datový model



Zdroj: Vlastní zpracování (2023)

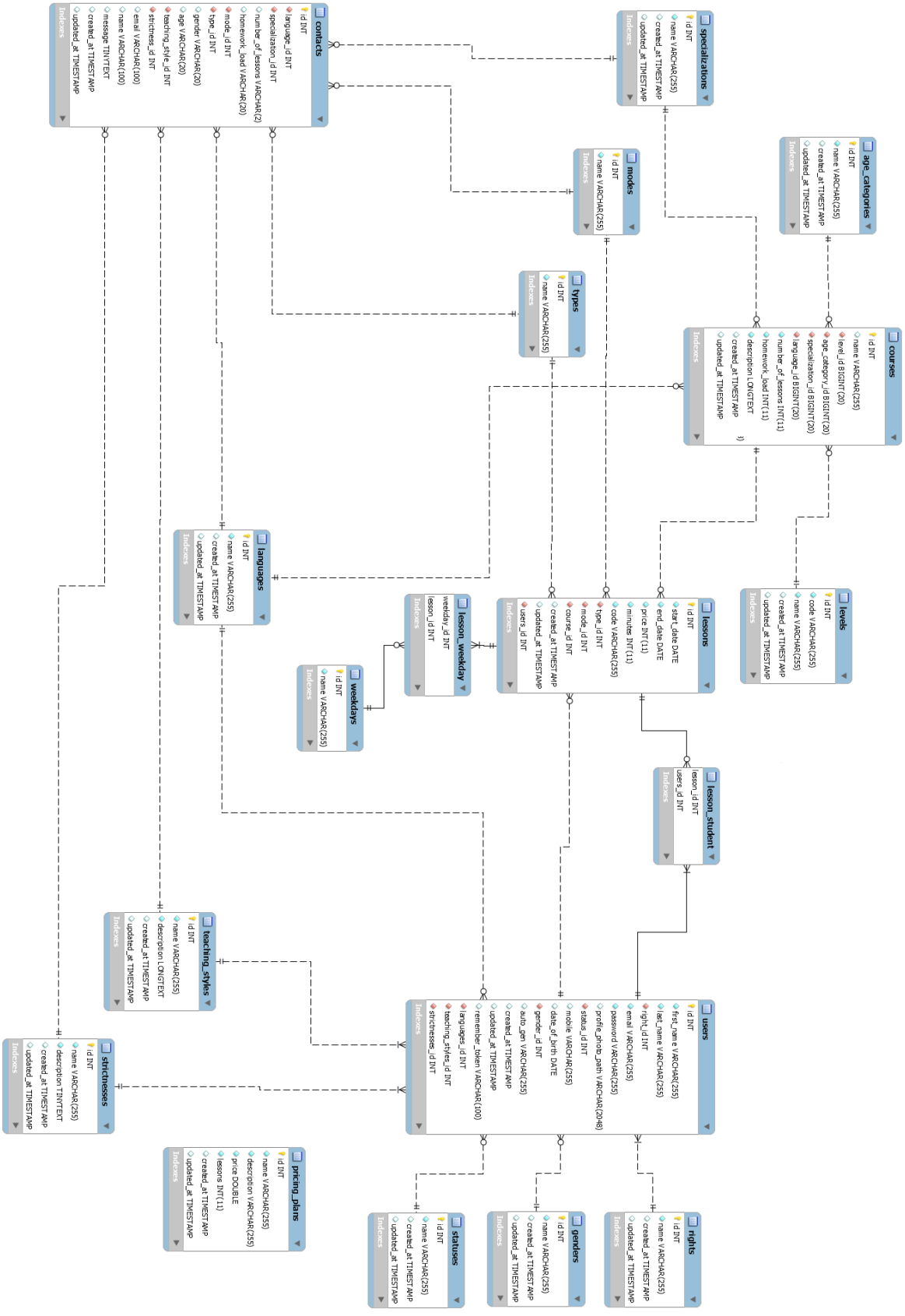
- Relace mezi entitami *Uživatel* a *Stav* – ve stejném stavu se může nacházet více uživatelů, ale každý uživatel se vždy musí nacházet v právě jednom stavu. Relace N:1.
- Relace mezi entitami *Uživatel* a *Právo* – více uživatelů může mít stejná práva, ale každý uživatel vždy musí mít právě jedno právo. Relace N:1.
- Relace mezi entitami *Uživatel* a *Pohlaví* – stejné pohlaví může mít více uživatelů, anebo žádný, každý uživatel musí patřit k právě jednomu pohlaví. Relace N:1.
- Relace mezi entitou *Kontakt* a entitami *Jazyk*, *Zaměření*, *Přísnost*, *Vyučovací Styl*, *Způsob*, *Typ* – ve formuláři pro každou entitu je vždy možné vybrat právě jednu hodnotu z nabízených. Relace N:1.

Při definování funkčních požadavků jsme uvedli, že uživatelé systémy s přihlašovacími právy budou administrátor školy a správce webu. Zároveň jsme uvedli, že návrh informačního systému by měl zajišťovat další rozšiřitelnost systému v budoucnu. V budoucnu by mohl přijít požadavek o přidání dalších funkcionalit, zejména to, že k systému by mohli mít přístup studenti a lektoři, pro které teď tento přístup poskytován nebude.

3.2.2 Relační datový model

Na Obr. 7 můžeme vidět výsledný konceptuální datový model. Dalším krokem při návrhu datové vrstvy našeho informačního systému je převedení konceptuálního modelu datového modelu na relační model, na kterém bude založena naše relační databáze. V relačním datovém modelu jsou data reprezentována jako sada tabulek, z nichž každá představuje entitu nebo objekt. Každá tabulka se skládá ze sloupců a z řádků. Každý řádek představuje jedinečnou instanci nebo záznam této entity. Sloupce tabulky představují atributy entity a jsou definovány svými datovými typy a omezeními. Vztahy mezi tabulkami se vytvářejí pomocí klíčů. Primární klíč je sloupec nebo sada sloupců, které jednoznačně identifikují každý řádek v tabulce, zatímco cizí klíč je sloupec nebo sada sloupců, které odkazují na primární klíč jiné tabulky. Pomocí klíčů reprezentujeme relace mezi tabulkami.

Obr. 8: Relační datový model



Zdroj: Vlastní zpracování (2023)

Uvádíme seznám vytvořených tabulek:

- *users* – tabulka pro ukládání informací o uživatelích systému a jejich přihlašovacích údajů;
- *lessons* – tabulka pro ukládání informací o výuce kurzu;
- *lesson_student* – tabulka, která vzniklá rozkladem relace M: N. Tabulka pro uchování seznamu studentů zapsaných na jednotlivou výuku kurzů;
- *weekdays* – číselník pro uschování názvu dnu v týdnu. Hodnoty číselníku: pondělí, úterý, středa, čtvrtek, pátek, sobota, neděle;
- *lesson_weekday* – tabulka uschovává informaci, ve kterých dnech probíhá jednotlivá výuka kurzu;
- *courses* – tabulka pro ukládání informací o kurzech;
- *contacts* – tabulka pro ukládání informací z kontaktních formulářů;
- *specialization* – číselník pro uschování zaměření kurzu;
- *age_categories* – číselník pro uschování názvů věkových kategorií, do kterých jsou kurzy zařazené;
- *levels* – číselník pro uschování úrovni znalosti jazyka. Hodnoty číselníku: A1, A2, B1, B2, C1, C2;
- *modes* – číselník pro uschování názvů způsobů výuky. Hodnoty číselníku: osobně, online, hybridně;
- *types* – číselník pro uschování názvů typu výuky, nabízených v jazykové škole Hodnoty číselníku: individuální, skupinová;
- *genders* – číselník pro uschování názvů pohlaví. Hodnoty číselníku: muž, žena, neuvedeno;
- *rights* – číselník pro uschování názvů uživatelských rolí. Hodnoty číselníku: administrátor, editor, student, lektor;
- *statuses*– číselník pro uschování názůu stavů, ve kterých se uživatel může nacházet. Hodnoty číselníku: aktivní, neaktivní,
- *languages* – číselník pro uschování názvů jazyků, které jsou ve škole vyučované,

- *strictnesses* – číselník pro uschování názvů a popisů úrovní přísností, které múzou mít lektoři,
- *teaching_styles* – číselník pro uschování názvů a popisů stylů výuky, které můžou mít lektoři,
- *pricing_plans* – tabulka pro ukládání informací o cenových nabídkách jazykové školy.

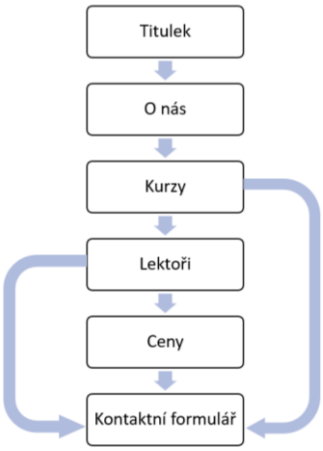
3.3 Návrh uživatelského rozhraní

V této kapitole popíšeme proces návrhu uživatelského rozhraní pro náš informační systém. Návrh jsme se rozhodli začít u veřejně přístupné webové stránky, která má prezentovat školu na webu, a tak se na její design a vzhled musí klast velký důraz. Webová stránka by neměla být dlouhá a neměla by být přeplněna textem, ale zároveň by měla návštěvníkovi poskytovat všechny potřebné informace a přesvědčit návštěvníka k rozhodnutí pro naši jazykovou školu.

3.3.1 Struktura webové stránky

Webovou stránku jsme se rozhodli rozdělit do šesti částí, přičemž každá část bude mít svůj jasně vymezený cíl a bude sdělovat stručné informace o důležitých aspektech školy. Tyto části, zejména "Titulek" "O nás", "Kurzy", "Lektoři" "Ceny" a "Kontakt", budou tvořit sekce stránky a budou vizuálně a tematické oddělené. Výsledná struktura stánky a její hierarchie je znázorněna na Obr. 9 a je popsána níže.

Obr. 7: Struktura webové stránky



Zdroj: Vlastní zpracování (2022)

Titulek je to první, co návštěvník uvidí, když přejde na webovou stránku z vyhledávače, a proto měl by hned upoutávat pozornost. Klíčovým elementem tedy bude hlavní nadpis *"Dokonalé jazykové kurzy pro nedokonalé lidi"*, který jasně sděluje podstatné informace o tom, co nabízíme. Poté bude následovat podtitulek, který je informativnější a sděluje přístup jazykové školy ke svým studentům a naznačuje klady naší školy: *„Nepočítáme s ideálními studenty, protože víme, že ti neexistují. Každý jsme jiný, máme odlišné cíle a učíme se trochu jinak. Uděláme Ti kurz na míru, aby to byl pro Tebe 100% užitek a 100% požitek"*. Na této stránce bude umístěno tlačítko "Objev více", které návštěvníka přesměruje na část "O nás".

Sekce "O nás" bude minimalistická a bude dále rozvádět výhody výuky v naší škole, což jsou individuální přístup a komunikativní metodika výuky. Za ní bude následovat sekce „Kurzy". Tato sekce bude obsahovat čtyři bloky, z nichž v každém bude informace o jednom kurzu z nabídky školy. Rozčlení informací do bloku zajistí jejich zobrazení přehlednějším způsobem. Pod nabídkou kurzu bude umístěno tlačítko s nadpisem "*Pomozte mi s výběrem kurzu*", které návštěvníka přesměruje na vyplnění kontaktního formuláře.

Po sekci „Kurzy" bude umístěna sekce Lektoři. Tato sekce bude stejně jako sekce „Kurzy" rozdělena do bloků, stylizovaných jako vizitky lektorů. Na každé ze třech vizitek bude foto lektora, jeho jméno, jazyk, který vyučuje, úroveň jeho přísnosti a vyučovací styl. Pod nabídkou kurzu bude umístěno tlačítko s nadpisem "*Najděte mi vysněného lektora*", které návštěvníka přesměruje na vyplnění kontaktního formuláře.

Předposledním blokem bude blok "Ceny". Ceny budou uvedeny v cenových balíčcích. U balíčku bude uveden název, stručný popis, počet lekcí v balíčku a vlastní cenu balíčku. Dále bude u každého balíčku tlačítko s nadpisem "*Zakoupit*", které návštěvníka přesměruje na vyplnění kontaktního formuláře.

Poslední sekcí bude "Kontakt", ve které se nachází kontaktní formulář, k němuž směruje každé tlačítko na webové stránce. Při návrhu kontaktního formuláře se musíme rozhodnout, jaké informace v něm budeme vyžadovat a kolik polí bude obsahovat. Na jednu stranu, u krátkého formuláře, který vyžaduje pouze dva nebo tři údaje, je vetší pravděpodobnost, že ho návštěvník vyplní a odešle. Na druhu stranu, od návštěvníka chceme získat obsáhlejší informace o jeho preferencích a požadavcích, abychom jeho poptávku mohli zpracovat a zpětně ho kontaktovat s cílenou nabídkou. Tento problém

jsem se rozhodli vyřešit vytvořením delšího formuláře, ve kterém budou určitá pole vyžadována. Formulář se bude skládat z dvanácti poli, z nichž osm bude vyžadováno. Bude požadováno vyplnění zaměření, jazyk, typ výuky, způsob výuky, přísnost lektora a vyučující styl lektora, což bude následně implementováno jako rozbalovací nabídka. Návštěvník tak nebude muset nic vyplňovat a bude si moct jednoduše vybrat z položek nabízených ve škole. Dalšími vyžadovanými poli budou "Jméno" a "E-mailová adresa". Nepovinnými poli budou počet lekcí týdne, časová náročnost domácích úkolů, preferované pohlaví učitele a preferovaný věk učitele. Formulář bude doplněn nepovinným polem, do kterého návštěvník bude moct napsat svůj vzkaz.

V hlavičce stánky bude umístěna navigační lišta s logem školy a položkami "O nás" "Kurzy", "Lektoři", "Ceny", "Kontakt". Navigační lišta zůstane vidět i při procházení stránky a umožní uživatelům rychle přecházet mezi sekcemi stránky. Při kliknutí na logo stránky bude uživatel přesměrován na začátek stránky. V patičce budou umístěny ikonky s odkazy na sociální sítě školy, zejména Facebook, Twitter, LinkedIn a Instagram.

3.3.2 Vizuální návrh webové stránky

Po vytvoření struktury webu a navigace následuje další fáze, a to vytvoření vizuálního návrhu webu. To zahrnuje volbu barevného schématu a typografického provedení webové stránky. Pomocí barev a typografie usilujeme o vytvoření vizuální hierarchie a zaměření pozornosti návštěvníků na důležité informace na stránce.

Obr. 8: Barevné schéma

Zdroj: Vlastní zpracování (2023)

Základní barvou byl zvolen odstín fialové barvy #904FD0, což je poměrně sytá barva s mírným jasem. Doplňující barvou pro tlačítka a drobné akcenty byl zvolen zářivý odstín žluté barvy #FFD708, který se často používá v brandingových a marketingových materiálech k vyvolání pocitu energie, optimismu a vzrušení. Třetí základní barvou byla zvolena bílá barva #F6F6F6.

Stejně důležitým aspektem webového designu je typografie, jelikož ovlivňuje čitelnost a použitelnost webových stránek. Písmo použité na webových stránkách může mít významný vliv na to, jak uživatelé stránky vnímají a jak snadno se na nich čte a orientuje. Pro webovou stránku jazykové školy jsme zvolili písmo Raleway, které si v posledních letech získalo oblibu zejména v moderním a minimalistickém designu webových stránek. Jedná se o bezpatkové písmo, které bylo vydáno v roce 2010. Raleway má čistý a elegantní vzhled a působí moderně.

Jednou z výhod písma Raleway je jeho všestrannost. Dodává se v různých variantách, od nejtenčí až po tučnou, což usnadňuje jeho použití pro nadpisy, podnadpisy a hlavní text. Má také širokou škálu znaků včetně diakritiky, interpunkčních znamének a symbolů, tudíž je správnou volbou pro naši stránku v českém jazyce.

Z hlediska použitelnosti je písmo Raleway snadno čitelné jak na desktopu, tak na mobilních zařízeních. Díky čistým liniím a otevřeným mezerám je dobrou volbou pro hlavní text, zatímco jeho tučná tloušťka se dobře hodí pro nadpisy a další prvky designu.

Zvolené barvy a písmo jsme použili i pro vytvoření loga jazykové školy, protože je důležité, aby zapadalo do designu webové stránky. Logo firmy je tvořeno jejím názvem a dvěma písmeny „S", které v kombinaci vytváří jedno stylizované písmeno „S".

Obr. 9: Logo

Zdroj: Vlastní zpracování (2023)

3.3.3 Návrh administrátorských stránek informačního systému

Barevné a designové zpracování stránek informačního systému určených pro správu školy, ke kterým budou mít přístup jen přihlášení uživatelé, bude přizpůsobeno barevnému zpracovaní a designu webové stránky školy. Struktura a rozložení těchto stránek bude shodná se strukturou a s rozložením stránek mnoha informačních systémů, což by mělo zajistit snadnou a intuitivní orientaci v systému. V levé části bude umístěn boční pruh s navigačním menu, v jehož nabídce bude šest sekcí s podsekcemi, které budou obsahovat odkazy na stránky informačního systému. Nad navigačním menu vlevo bude

umístěno logo jazykové školy. Po kliknutí na logo bude uživatel přesměrován na úvodní stránku informačního systému, na které se bude nacházet dashboard se základními statistikami školy. V horní části stráňky bude umístěna hlavička, v jejíž pravé části bude možné nalézt obrázek přihlášeného uživatele. Po kliknutí na obrázek se otevře menu, v jehož nabídce budou dvě položky: „Můj Profil“ a „Odhlásit se“.

Obr. 10: Návrh administrátorské stránky

Zdroj: Vlastní zpracování (2022)

V dolní části stránky se bude nacházet patička s informacemi o autorských právech. Mezi hlavičkou a patičkou se bude nacházet hlavní obsah stránky, který bude umístěn do kontejneru. Nad kontejnerem s hlavním obsahem se bude nacházet navigační lišta, jejímž úkolem bude zajistit přehlednost a jednoduchost navigace mezi stránkami.

3.4 Implementace

Ve fázi, kdy máme návrh informačního systému hotový, můžeme přistoupit k jeho implementaci. V této kapitole uvedeme nástroje pro vývoj informačního systému a popíšeme vybrané třídy a metody naimplementované v systému se zaměřením na jeho MVC architekturu.

3.4.1 Implementační nástroje

Prvním krokem implementace systému je výběr a instalace implementačních nástrojů a prostředí. Pro vývoj budeme používat open-source softwarový balík XAMPP, který

obsahuje webový server Apache (A), databázi MySQL (M), programovací jazyk PHP (P) a programovací jazyk Perl (P). XAMPP umožňuje vytvořit na lokálním počítači prostředí lokálního webového serveru, které použijeme k lokálnímu vývoji a j testování webové aplikace před jejím nasazením na ostrý webový server. Budeme využívat XAMPP verzi 8.0.15 pro Windows, staženou z oficiální webové stránky.

Dále bude provedena instalace PHP frameworku Laravel, ve kterém bude aplikace vyvíjena. Prerekvizitou pro instalaci Laravel je instalace programu Composer, což je nástroj pro správu balíčků a jejich závislostí v PHP. Po stažení souboru Composer-Setup.exe. z oficiálních stránek a jeho následným spuštěním se provede instalace, po které už je možné spustit program Composer z libovolného adresáře v příkazovém řádku pomocí příkazu *composer*. Teď můžeme provést vytvoření Laravel projektu pomocí příkazu *composer create-project laravel/laravel speaksmart-management-system*.

Jako editor kódů bude použito Visual Studio Code (VS Code) od společnosti Microsoft. VS Code je navržen tak, aby byl vysoce přizpůsobitelný, a je k dispozici široká škála rozšíření, která přidávají funkce a podporu pro různé programovací jazyky a frameworky, zejména PHP a Laravel.

3.4.2 MVC architektura

Laravel projekt budeme spouštět pomocí vestavěného webového serveru v PHP voláním příkazu *php artisan serve* ve složce projektu. Informační systém bude vyvíjen a odladěn na lokálním hostiteli na URL adrese http://127.0.0.1:8000/.

V první kapitole bylo zmíněno, ze Laravel framework je postaven na vzoru MVC, který rozděluje zdrojový kód do tří komponent. Z tohoto rozdělení vyplývá struktura projektu, která bude popsána dále, a to včetně modelů, pohledů a kontrolérů.

3.4.2.1 Modely

Laravel poskytuje nástroj pro objektově-relační mapování Eloquent ORM, který usnadňuje práci s databázovými záznamy. Hlavní výhodou ORM je synchronizace mezi používanými objekty v aplikaci s tabulkami v databázi. Pro každou tabulku v databázi je vytvořen odpovídající model. Název modelu bude stejný jako název tabulky, ale v jednotném čísle, protože zastupuje právě jeden záznam databázové tabulky. K tabulce *courses* tak vytvoříme přes příkazový řádek ve složce s naším projektem korespondující model Course pomocí artisan příkazu *php artisan make:model Course*.

Tento příkaz vytvoří do složky *speak-smart-management-system/app/models* soubor Course.php se třídou Course. Seznam všech vytvořených modelů s korespondujícími tabulkami je uveden níže:

- Model User zastupuje databázovou tabulku *users*.
- Model Lesson zastupuje databázovou tabulku *lessons*.
- Model Weekday zastupuje databázovou tabulku *weekdays*.
- Model Course zastupuje databázovou tabulku *courses*.
- Model Contact zastupuje databázovou tabulku *contacts*.
- Model Specialization zastupuje databázovou tabulku *specializations*.
- Model AgeCategory zastupuje databázovou tabulku *age_categories*.
- Model Level zastupuje databázovou tabulku *levels*.
- Model Mode zastupuje databázovou tabulku *modes*.
- Model Type zastupuje databázovou tabulku *types*.
- Model Gender zastupuje databázovou tabulku *genders*.
- Model Right zastupuje databázovou tabulku *rights*.
- Model Status zastupuje databázovou tabulku *statuses*.
- Model Language zastupuje databázovou tabulku *languages*.
- Model Strictness zastupuje databázovou tabulku *strictnesses*.
- Model TeachingStyle zastupuje databázovou tabulku *teaching_styles*.
- Model PricingPlan zastupuje databázovou tabulku *pricing_plans*.

Každá vytvořená třída dědi ze třídy Model, která obsahuje proměnné pro definování atributů a hodnot a metody, které budeme potřebovat pro práci s databází (vyhledávání, vytváření, upravování). Instance třídy Course také obsahuje atributy z databáze. Tabulky v databázi jsou mezi sebou propojené vazbami. Aby bylo s těmito vazbami možné pracovat, musíme je definovat v modelu. Každá vazba bude definována metodou v modelu, jejíž název je odvozen od názvu modelu, na který se odkazujeme. Například metoda *type()* v modelu Lesson definuje vazbu mezi tabulkami *types* a *lessons*. Metoda

students() definuje vazbu M:N mezi tabulkami *lessons* a *users*. Tato vzniklá metoda definující vztah s jiným modelem pak může být následně zavolána jako třídní metoda.

Obr. 11: Ukázka vybraných metod třídy Lesson

```
namespace App\Models;

use Illuminate\Database\Eloquent\Model;

class Lesson extends Model
{
    public function type()
    {
        return $this->belongsTo(Type::class, 'type_id', 'id');
    }
    public function students()
    {
        return $this->belongsToMany(User::class, 'lesson_student', 'lesson_id', 'student_id')->where('right_id', '3');
    }
}
```

Zdroj: Vlastní zpracování (2023)

Tím dostaneme instanci třídy daného vztahu. V databázi máme vytvořenou tabulku *lesson_student.* Pro tuto tabulku nebyl vytvořen vlastní model, jelikož spojující databázové tabulky zpravidla neuchovávají ani svůj jedinečný identifikátor. K hodnotám v této tabulce budeme přistupovat buď z modelu Lesson, nebo z modelu User.

3.4.2.2 Pohledy

HTML stránky webového informačního systému jsou umístěné ve složce *speak-smart-management-system/resources/views.* V této složce byly vytvořeny dvě podsložky: *admin* a *landing_page.* Ve složce *admin* jsou uloženy pohledy pro administrátorské stránky systému, ve složce *landing page* je uložena veřejně přístupná webová stránka. Název každého souboru s pohledem obsahuje příponu *.blade .php*. Tato přípona slouží k označení, že pohled používá šablonovací systém Blade. Šablony Blade obsahují HTML a také direktivy Blade, které umožňují snadné vypisovaní hodnot, vytváření příkazů "if", iteraci dat a další.

Do projektu s webovým informačním systémem byl naimportován framework Bootstrap. Při tvorbě administrátorských stánek byla využita šablona Bootstrap 5, kterou jsme upravili podle požadavků uvedených v kapitole “Návrh administrátorských stránek informačního systému”. Tato šablona obsahuje celkový vzhled webového informačního systému. Aby byl kód přehlednější, byly jeho sémantické části vyčleněny do jednotlivých souborů, které byly uloženy do složky *speak-smart-management-*

system/resources/views/admin/layout. Soubor *header.blade.php* obsahuje hlavičku stránky, soubor *footer.blade.php* obsahuje patičku stránky a soubor *sidebar.blade,php* obsahuje boční pruh s navigačním menu. Dále vytvoříme soubor *master_layout.blade.php*, který bude sloužit jako hlavní pohled pro zajištění celkového vzhledu informačního systému. Do souboru *master_layout.blade.php* budou vloženy soubory *header.blade.php*, *footer.blade.php*, *sidebar.blade,php*, také v tomto souboru připojíme externí styly a knihovny pomocí Blade direktivy *@include*.

Obr. 12: Soubory zajišťující vzhled informačního systému

```
@include('admin.layout.header')
@include('admin.layout.sidebar')
@yield('content')
@include('admin.layout.footer')
```

Zdroj: Vlastní zpracování (2023)

Blade direktiva *@yield()* převezme sekci kódu z pohledu, který dědí pohled *master _layout.blade.php.* Hodnota této sekce se definuje Blade direktivami *@section* a *@endsection*. Rovněž v pohledu *add_lesson.blade.php*, který obsahuje formulář pro přidávání nové vyučovací hodiny, dědíme pohled *master_layout.blade.php*, který se nachází ve složce *resources/views/admin/layout*. Zároveň v tomto pohledu definujeme sekci, kde se nachází formulář.

Obr. 13: Vymezení sekce v souboru add_lesson.blade.php

```
@extends('admin.layout.master_layout')
@section('content')
 …
@endsection
```

Zdroj: Vlastní zpracování (2023)

Administrátorské stránky obsahují formuláře pro přidávání nových lektorů, studentů, jazyků, kurzů a tak dále. Proto tady bude ukázán fragment kódu pro vytvoření formuláře, který slouží pro přidání nového lektora. V tomto formuláři pro definici způsobu předávání dat používáme metodu POST a odkazujeme na routu *'insert.new.teacher'* (routování bude podrobněji popsáno v další kapitole). Formulářem odesíláme obrázek lektora, a proto byl nastaven způsob zakódování dat *enctype="multipart/form-data"*. Pomocí Blade direktivy *@csrf* bude ve formuláři vygenerováno skryté pole s CSRF tokenem, pomocí kterého se

ověřuje, zda byl požadavek odeslán přes naši stránku. Tato direktiva tak poskytuje ochranu proti CSRF útoku.

Obr. 14: Ukázka formuláře pro přidání nového lektora

```
<form method="post" action="{{ route('insert.new.teacher') }} enctype="multipart/form-data">
 @csrf
 <div class="form-group">
  <h5>Jméno<span class="text-danger">*</span></h5>
  <input type="text" name="first_name" class="form-control" required value="{{ old('first_name') }}">
  @error('first_name')
   <span class="text-danger">{{ $message }}</span>
  @enderror
 </div>
…
 <input type="submit" class="btn btn-rounded btn-info mb-5" value="Přidat">
 </div>
</form>
```

Zdroj: Vlastní zpracování (2023)

Helper funkce *old(),* která má přístup k datumu z formuláře uloženého do sesions, zajistí, že hodnoty vyplněné uživatelem zůstanou v poli, pokud dojde k obnovení stránky, například v případě, kdy jméno neprojde přes validační pravidla. V tomto případě výpis chyby, ke které došlo při validaci hodnot zadaných uživatelem do pole *<input>,* zajistí Blade direktiva *@error*. Nyní už stačí jen vypsat chyby v našem pohledu, abychom uživatele informovali o omylech, kterých se dopustil.

3.4.2.3 Kontroléry

Kontrolér na základě požadavku od uživatele vytvoří model a vygeneruje pohled s daty z modelu. Kontroléry jsou umístěné ve složce *speak-smart-management-system/app/controllers/*. V této složce byly vytvořeny dvě podsložky *backend* a *web*. V podsložce *backend* jsou v dalších podsložkách uloženy kontroléry spravující požadavky nadcházející z administrátorských stránek webového informačního systému, ve složce web jsou uložené kontroléry spravující požadavky z veřejné přístupné webové stránky. Kontroléry byly vytvořeny tak, aby logika zpracování souvisejících požadavků

byla umístěna do jediné třídy. Například třída LessonController zpracovává všechny příchozí požadavky týkající se vyučovacích hodin, včetně zobrazování, přidání, aktualizace a mazání vyučovacích hodin. Kontrolér pro práci s vyučujícími hodinami byl vytvořen následujícím Artisan příkazem v příkazové řádce z kořenové složky projektu: *php artisan make:controller backend/lesson/LesssonController*. Tento příkaz vytvoří ve složce *speak-smart-management-systém/app/controllers/backend/lesson* soubor *LessonController.php*, který obsahuje pouze definování třídy s názvem LessonController, dědící z třídy Controller.

V kontroléru byly definovány metody pro spravování vyučovacích hodin v jazykové škole. Metoda *showEnrolledStudentList($id)* slouží pro výpis seznamu studentů zapsaných na vyučovací hodinu. Tato metoda přebírá parametr *$id*, což je id vyučovací hodiny, pro kterou má být vypsán seznam studentů. Pomocí modelu Lesson vyhledáme v databázi vyučující hodinu, jejíž id odpovídá hodnotě v proměnně $id a načteme pomocí metody *students()* všechny studenty zapsané na tuto vyučovací hodinu. Metoda *students()* je metoda třídy Lesson a byla popsána v kapitole Modely.

Obr. 15: Ukázka metody pro výpis seznamu studentů zapsaných na vyučovací hodinu

```
public function showEnrolledStudentList($id)
 {
  $data['lessons'] = Lesson::where('id', $id)->with('students')->get();
  return view('admin.lesson.enrolled_student_list', $data);
 }
```

Zdroj: Vlastní zpracování (2023)

Dále volíme helper funkci *view()* s dvěma parametry. Jako první parametr této funkci předáváme pohled s názvem *enrolled_student_list.blade.php*. Druhým parametrem je pole s proměnnými, které se předají pohledu.

3.4.3 Routování

Routování (česky směrovaní) je proces přijímaní HTTP požadavků z prohlížeče a jejich přiřazení odpovídajícím kontrolérům. Definování rout zajišťuje to, že podle URL adresy se pozná, jakou akci uživatel chce provést a zavolá se příslušný kontrolér obsahující metodu, která tuto akci provede. Všechny routy jsou definované v souboru *routes/web.php.*

Routy mohou být definované několika způsoby. Součástí definice routy musí být voláni statické metody třídy Route odpovídající HTTP požadavku. Mezi parametry metody patří adresa, ze které budou přicházet požadavky, název metody v kontroléru zpracovávající daný požadavek. Jednotlivé routy lze pojmenovat pro přehlednost a snadnější odkazování na ni pomocí helper funkce *route()*. Například u formulářů pro přidání nové vyučující hodiny použijeme *route('insert.lesson')*. Routy lze seskupit pomocí metody *group()*, pro vytvořenou skupinu můžeme definovat společnou předponu pomocí metody *prefix()* a přiradit všechny routy ve skupině jednomu kontroléru pomocí metody *controller()*. Níže je přidána ukázka skupiny rout definovaná pro webový informační systém jazykové školy.

Obr. 16: Ukázka skupiny rout s definovaným prefixem

```
use Illuminate\Support\Facades\Route;
use App\Http\Controllers\backend\lesson\LessonController;
Route::prefix('lessons')->controller(LessonController::class)->group(function () {
    Route::get('/', 'showLesson')->name('lesson');
    Route::get('/add', 'addLesson')->name('add.lesson');
    Route::post('/insert', 'insertLesson')->name('insert.lesson');
    Route::get('/edit/{id}', 'editLesson')->name('edit.lesson');
    Route::post('/update/{id}', 'updateLesson')->name('update.lesson');
    Route::get('/details/{course_id}', 'showLessonDetails')->name('lesson.details');
    Route::get('/delete/{id}', 'deleteLesson')->name('delete.lesson');
    Route::get('/student/unenroll/{student_id}/{lesson_id}', 'unenrollStudent')->name('unenroll.student');
    Route::get('/enrolled/student/list/{id}', 'showEnrolledStudentList')->name('enrolled.student.list');
    Route::get('/student/enroll/{lesson_id}', 'enrollStudents')->name('lesson.enroll.student');
    Route::post('/student/insert', 'insertStudentsIntoLesson')->name('insert.students.into.lesson');
  });
```

Zdroj: Vlastní zpracování (2023)

Každá routa v této skupině zpracovává požadavek tykající se vyučující hodiny a volá metody třídy LessonController a předává jim příslušné parametry z URL adresy uvedené ve složených závorkách {}.

3.5 Testování použitelnosti webového informačního systému

Po dokončení implementace webového informačního systému je třeba ho otestovat, aby bylo zajištěno, že systém funguje podle očekávaní a splňuje stanovené požadavky. Testování webového informačního systém proběhlo podle uživatelských testovacích scénářů, které simulovaly reálné situace a případy použití webového informačního systému.

Testování veřejně přístupné webové stránky bylo uskutečněno za účasti třech osob, jejichž profil odpovídá profilu cílových uživatelů vytvořené webové stránky – často si vyhledávají služby na internetu a mají zájem o výuku cizích jazyků. Při testování veřejné přístupné webové stránky byly otestovány všechny odkazy a tlačítka. Bylo otestováno, zda CTA tlačítka směrují ke kontaktnímu formuláři a odkazy v navigační listě směrují ke správné části stránky.

Dále byl otestován kontaktní formulář. Proběhla kontrola, zda se číselníkové hodnoty z databáze načítají bez chyb, nebo jestli formulář nelze odeslat, když nejsou vyplněna požadovaná pole. Následně byl formulář vyplněn a odeslán. Po odeslaní formuláře bylo zkontrolováno, zda se odeslaná zprava uložila do databáze a zda byla návštěvníkovi webu vypsána zprava, že formulář byl úspěšně odeslán.

Následujícím krokem bylo testování přihlašovací stránky. Bylo vyzkoušeno, že se nepřihlášený uživatel nedostane do administrátorských stránek a při pokusu takto učinit bude nepřihlášený uživatel přesměrován na stránku s přihlašovacím formulářem. Do přihlašovacího formuláře byla zadána kombinace e-mailové adresy a hesla, která v databázi neexistuje. Bylo ověřeno, že přístup k administrátorským stránkám nebude povolen a bude vypsána chybová hláska. Do registračního formuláře byl zadán e-mail a heslo uložené v databázi u uživatele s rolí administrátor a bylo ověřeno, že přístup k systému a ke všem funkcionalitám je povolen. Dále proběhlo ověření, že přihlášený uživatel s rolí admin může otevřít stránku svého profilu, aktualizovat údaje ve svém profilu a změnit si heslo.

V další fázi proběhlo testování jednotlivých funkcionalit systému přístupných pouze pro přihlášené uživatele.

Spravování uživatelů: přidání nového uživatele, aktualizace údajů existujícího uživatele.

Spravování studentů: přidávání nového studenta, aktualizace údajů studenta, vyhledávání studenta v seznamu.

Spravování lektorů: přidání lektora, aktualizace údajů lektora, vyhledávání lektora a vyfiltrování lektora podle vyučovacího stylu a přísnosti. Dále bylo otestováno přidávání, aktualizace a odstranění vyučovacích stylů a přísnosti. Bylo ověřeno, že pokud v systému existuje lektor s daným vyučujícím stylem nebo přísností, nelze tento vyučující styl a přísnost odstranit.

Spravování kurzů: přidání nového jazyka, úrovně, věkové kategorie a zaměření, aktualizace těchto údajů. Dále byl vytvořen nový kurz a po vytvoření byl aktualizován. Bylo ověřeno, že pokud v systému existuje alespoň jeden kurz v daném jazyce, pro danou úroveň, věkovou kategorii nebo zaměření, nelze tento jazyk, úroveň, věkovou kategorii a zaměření odstranit.

Spravování výuky: byla přidána vyučovací hodina, proběhlo vypsání vyučovacích hodin kurzu "Angličtina pro teenagery", údaje o vyučovací hodině byly aktualizovány, vyučovací hodina byla odstraněna, byl vypsán seznam studentů zapsaných na vyučovací hodinu "Angličtina pro teenagery" a byl zapsán další student.

Spravování cenových plánů: přidání nového cenového plánu, aktualizace cenového plánu, odstranění cenového plánu, vyhledávání v seznamu cenových plánů.

Po otestování všech funkcionalit systému lze říci, že všechny požadované funkcionality fungují tak, jak bylo zamýšleno.

Informační systém byl otestován v prohlížečích Google Chrome, Microsoft Edge, Opera Mini, Firefox a Safari na zařízeních iPhone 11, Lenovo IdeaPad S540 a iPad air. Testování ověřilo, že informační systém je kompatibilní s různými prohlížeči a obsah informačního systému se korektně zobrazuje na různých zařízeních.

3.6 Omezení a možnosti rozšíření webového informačního systému

Tento informační systém byl navržen především pro prezentaci školy na internetu a přidání na webovou stránku školy aktuálních údajů z databáze školy. Administrátor může tento systém používat pro zaznamenání údajů o lektorech, kurzech, vyučovacích hodinách a dalších záležitostech školy. Vzhledem k tomu, že škola je jen start-up a její administrativa není rozsáhlá, jsou tyto funkcionality dostačující. V budoucnosti by mohlo

dojít k rozšíření administrativních činností školy a k nárustu počtu lektorů a studentů. V tomto případě by na informační systém mohly být kladeny větší nároky.

Při návrhu informačního systému byly zohledněny potřeby dalšího rozšíření, a proto by bylo možné zavést nové moduly pro studenty a lektory. Studenti by měli přístup k informačnímu systému, ke svému profilu, k rozvrhu, k učebním plánům a podrobným informacím o každé vyučující hodině, a to včetně studijních materiálu k ní. Lektoři by měli přístup ke svým rozvrhům, k profilům svých studentů, ke každé vyučující hodině by lektor mohl přidávat studijní materiály.

Další užitečnou funkcionalitou by byl chat mezi uživateli webového informačního systému. Administrace by mohla komunikovat s lektory i se studenty. Lektoři a studenti by rovněž mohli komunikovat mezi sebou. Uvítanou funkcionalitou by bylo i automatické rozesílaní SMS zpráv s oznámením v případě, že by vyučovací hodina byla zrušena.

Dalším modulem, který by mohl být přidán, je modul Accounting. V současném systému nejsou řešeny studentské platby a mzdy lektorů. V budoucnosti by bylo vhodné přidat platební bránu pro platby na internetu a umožnit generovaní faktur pro studenty a výplatní pásky pro lektory.

Na veřejnou webovou stránku by bylo vhodné přidat blog, na kterém by mohly být publikovány články se zajímavostmi o cizích jazycích, trendy ve výuce cizích jazyků a tipy na to, jak se učit cizí jazyky.

Závěr

Cílem této bakalářské práce bylo navrhnout a implementovat webový informační systém pro jazykovou školu, který bude prezentovat školu na internetu, zprostředkovávat první kontakt zákazníků se školou a uchovávat záznamy jak o zájemcích o výuku, tak o stávajících studentech, lektorech, nabízených kurzech a vyučovacích hodinách. Před zahájením tvorby webového informačního systému byla provedena literární rešerše, což umožnilo přiblížení podstaty informačních systémů a vysvětlení principu jejich fungování. Byl diskutován architektonický vzor MVC a framework Laravel, které se staly základem pro vytvoření informačního systému, a další implementační nástroje a jazyky, ve kterých byl webový informační systém naprogramován.

Po nastudovaní teorie byl stanoven postup pro vytvoření webového informačního systému pro jazykovou školu. Vytvoření webového informačního systému proběhlo ve čtyřech etapách. Nejprve byly definovány funkční a nefunkční požadavky na systém. Důkladná analýza těchto požadavků odehrála klíčovou roli v tom, že vytvořený informační systém splnil potřeby uživatelů systému.

V okamžiku, kdy byly požadavky specifikované, jsme pokračovali návrhem datových objektů, jejich struktury, relací mezi objekty a integritních omezení. Nakonec jsme navrhli konceptuální datový model a implementovali jsme databázovou vrstvu. V této fázi proběhl též návrh uživatelského prostředí a vizuálního vzhledu systému.

Hotový návrh informačního systému umožnil přistoupit k vlastní implementaci systému. Systém byl vytvořen s použitím frameworku Laravel a při jeho implementaci byly respektovány principy architektonického vzoru MVC. Jednotlivé vrstvy sytému byly implementovány v různých programovacích jazycích, zejména HTML, CSS, SQL, PHP a JavaScript. Pro prezentační vrstvu byl využit šablonovací nástroj Blade a pro práci s databází nástroj Eloquent ORM.

Vytvořený webový informační systém byl otestován, aby bylo zajištěno, že systém funguje podle očekávaní a splňuje stanovené požadavky. Následně byla diskutována omezení systému, rovněž byly řešeny návrhy na jeho rozšíření a vylepšení. Po otestování byl systém nasazen na veřejný webový server. Webová stránka jazykové školy SpeakSmart je tak dostupná na adrese www.speaksmart.space.

Seznam použitých zdrojů

Almendral, K. (24. 7. 2018). *PHP Request-Response Cycle. Kennyalmendral.github.io.* Získáno 21. leden 2022, z Kenny Almendral. Web/Mobile App Developer: https://kennyalmendral.github.io/php-request-response-cycle

Bruckner, T., Voříšek, J., Buchalcevová, A., Stanovská, I., Chlapek, D., & Řepa, V. (2012). *Tvorba informačních systémů. Principy, metodiky, architektury.* Praha: Grada Publishing.

Čápka, D. (20. 12 2012). *Lekce 1 - Popis MVC architektury*. Získáno 13. 12 2023, z itnetwork.cz: https://www.itnetwork.cz/php/mvc/objektovy-mvc-redakcni-system-v-php-popis-architektury

Chowles, L. (4. 4 2023). *100+ landing page statistics (2023 edition)*. Načteno z MarketSplash: https://marketsplash.com/landing-page-statistics

Codecademy. (23. 9 2021). *What Is a Framework?* Načteno z Codecademy: https://www.codecademy.com/resources/blog/what-is-a-framework

Davis, W. S., & Yen, D. C. (1998). *The Information System Consultant's Handbook-Systems Analysis and Design.* Boca Raton, Florida, USA: CRC Press.

Enge, E., Spencer, S., & Stricchiola, J. (2015). *Art of SEO* (3. vyd.). O'Reilly Media,.

Gangur, M. (2014). *Základy implementace webového informačního systému. Praktický průvodce.* Plzeň: Západočeská univerzita v Plzni.

JetBrains. (2022). *PHP Programming – The State of Developer Ecosystem in 2022*. Načteno z JetBrains: https://www.jetbrains.com/lp/devecosystem-2022/php/.

Koďousková, B. (21. 7 2021). *CO JE LANDING PAGE A JAK BY MĚLA VYPADAT*. Načteno z Rascasone: https://www.rascasone.com/cs/blog/co-je-landing-page

LangLion Blog. (16. 9 2014). Načteno z How to create a GOOD website for a language school?: https://blog.langlion.com/en/how-to-create-a-good-website-for-a-language-school/

Laurenčík, M. (2019). *Tvorba www stránek v HTML a CSS.* Praha: Grada Publishing. Načteno z https://www.bookport.cz/tvorba-www-stranek-v-html-a-css-6057

MarketingPPC. (19. 3 2022). *Co Je to Konverze*. Načteno z MarketingPPC: https://www.marketingppc.cz/ppc/co-je-to-konverze/

Mendez, M. (2014). *The Missing Link. An Introduction to Web Development and Programming.* Geneseo: Open SUNY Textbooks.

Mioweb. (nedatováno). *Co je landing page*. Načteno z Mioweb slovníček webových pojmů: https://www.mioweb.cz/slovnicek/landing-page/

MySQL. (6. 1. 2022). Načteno z Mysql.com: https://www.mysql.com

Nixon, R. (2014). *Learning PHP, MySQL, JavaScript, CSS & HTML5. Step-by-Step Guide to Creating Dynamic Websites* (3. vyd.). Sebastopol: O'Reilly Media.

Pilka, L. (28. 1 2018). *10 pravidel pro účinnou landing page*. Načteno z Blueghost.cz: https://www.blueghost.cz/clanek/10-pravidel-pro-ucinnou-landing-page

Riordan, R. M. (2005). *Designing effective database systems.* Upper Saddle River: Addison-Wesley.

Rubin, J., & Chisnell, D. (2008). *Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests* (2. vyd.). Indianapolis: Wiley Publishing.

Schewe, K.-D., & Thalheim, B. (2019). *Design and Development of Web Information Systems.* Berlin: Springer.

Search Engine Land. (8. 2 2023). *What Is SEO – Search Engine Optimization?* Načteno z Search Engine Land: https://searchengineland.com/guide/what-is-seo

Vailshery, L. S. (9. 8 2022). *Most used web frameworks among developers worldwide, as of 2022*. Načteno z Satista: https://www.statista.com/statistics/1124699/worldwide-developer-survey-most-used-frameworks-web/

W3Techs. (nedatováno). *Usage statistics of PHP for websites*. Načteno z W3Techs: https://w3techs.com/technologies/details/pl-php

Welling, L., & Thomson, L. (2017). *Mistrovství PHP a MySQL. Kompletní průvodce vývojáře.* (O. Baše, Překl.) Brno: Computer Press.

Seznam obrázků

Seznam příloh

Příloha A: CD-R se zdrojovým kódem webového informačního systému

Abstrakt

Marchuk, S. (2023). *Návrh a implementace webového informačního systému* [Bakalářská práce, Západočeská univerzita v Plzni].

Klíčová slova: webový informační systém, analýza a návrh, webové stránky, jazyková škola, MVC

Bakalářská práce popisuje proces návrhu a implementace malého webového informačního systému pro jazykovou školu sloužícího k prezentaci školy na internetu a pro administraci školy. Práce je rozdělena na tři části: teoretickou, metodickou a praktickou. V teoretické části práce je přiblížena podstata informačních systémů a jsou vysvětleny principy jejich fungování, rovněž zde jsou diskutovány technologie pro vývoj informačních systémů. Dále je pozornost věnována vytvoření obsahu webových stránek, typům požadavků na webový informační systém a postupům při testování systému. Další kapitola se zabývá stanovením metodického postupu, podle kterého se řídil proces vytvoření informačního systému. Praktická část je věnována analýze požadavků na informační systém jazykové školy, návrhu datové a prezentační vrstvy, implementaci, nakonec pak testování použitelnosti systému. Na závěr jsou diskutována omezení a možnosti rozšíření systému.

Abstract

Marchuk, S. (2023). *Design and implementation of a web-based information system* [Bachelor Thesis, University of West Bohemia].

Key words: information system, analysis and design, website, language school, MVC

The bachelor thesis outlines the design and implementation process of a simple web-based information system for a language school. The main purpose of the system is to showcase the school on the internet while allowing basic school administration. The thesis is divided into three parts: a literature review, a description of the methodology, and a practical chapter. The literature review focuses on the principles of information systems and the technologies for information system development. It also gives an overview of website content creation, types of requirements for a web-based information system, and procedures for testing the system. The next chapter defines the methodological procedure that guided the process of creating the information system. The practical part is devoted to the requirements analysis for the language school information system, the design of the data layer and presentation layer, the implementation of the system, and finally the usability testing of the system. Lastly, the limitations and possibilities of extending the system are discussed.