VŠB – Technická univerzita Ostrava

Fakulta elektrotechniky a informatiky

Katedra informatiky

Absolvování individuální odborné praxe
Individual Professional Practice in the Company

2018 Jakub Konečný

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Jakub Konečný**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Absolvování individuální odborné praxe
Individual Professional Practice in the Company

Jazyk vypracování: čeština

Zásady pro vypracování:

1. Student vykoná individuální praxi ve firmě: ISSA CZECH s.r.o.
2. Struktura závěrečné zprávy:
   a) Popis odborného zaměření firmy, u které student vykonal odbornou praxi a popis pracovního zařazení studenta.
   b) Seznam úkolů zadaných studentovi v průběhu odborné praxe s vyjádřením jejich časové náročnosti.
   c) Zvolený postup řešení zadaných úkolů.
   d) Teoretické a praktické znalosti a dovednosti získané v průběhu studia uplatněné studentem v průběhu odborné praxe.
   e) Znalosti či dovednosti scházející studentovi v průběhu odborné praxe.
   f) Dosažené výsledky v průběhu odborné praxe a její celkové zhodnocení.

Seznam doporučené odborné literatury:

Podle pokynů konzultanta, který vede odbornou praxi studenta.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. David Ježek, Ph.D.**

Konzultant bakalářské práce: Ing. Michal Kolesár

Datum zadání: 01.09.2016

Datum odevzdání: 30.04.2018

doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 27. dubna 2018

Souhlasím se zveřejněním této bakalářské práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v bakalářských programech VŠB-TU Ostrava.

V Ostravě 27. dubna 2018

.....

ISSA CZECH s.r.o.
Hrušovská 3203/13a
702 00 Ostrava
DIČ: CZ25381920
www.issa.cz

ISSA

Rád bych poděkoval firmě ISSA CZECH s. r. o., která mi umožnila vykonávat odbornou praxi. Dále bych chtěl poděkovat především mému konzultantovi Ing. Michalu Kolesárovi, Ph.D., který mě provázel ve firemním prostředí. Také vedoucímu mé bakalářské práce Ing. Davidu Ježkovi, Ph.D., za odborné konzultace a rady v průběhu vykonávání praxe. V neposlední řadě bych chtěl poděkovat všem, kteří mi věnovali po dobu praxe svůj čas a energii.

Abstrakt

Bakalářská práce popisuje absolvování odborné praxe ve společnosti ISSA CZECH s. r. o. Má odborná praxe byla zaměřena na vývoj webových stránek a informačních systémů. V práci si dávám za cíl seznámit čtenáře s jednotlivými projekty, řešením a také s technologiemi, které jsem k projektům použil. Zmíním se také o mých zkušenostech a vědomostech, které jsem za celou praxi získal.

Klíčová slova: bakalářská práce, odborná praxe, ISSA CZECH s. r. o., PHP, MySQL, HTML, CSS, JavaScript, PLM, CMS, informační systém, vývoj webových stránek

Abstract

The Bachelor's thesis describes the completion of professional practice at the ISSA CZECH Company. My professional practice was focused on front-end web development and development of information systems. My main goal is to introduce reader with projects, its solution and technologies used in the projects. I will also mention my experience and knowledge, which I gained during whole practice.

Key Words: bachelor's thesis, professional practice, ISSA CZECH s. r. o., PHP, MySQL, HTML, CSS, JavaScript, PLM, CMS, information system, website development

Obsah

Seznam použitých zkratek a symbolů

API	**– Application Programming Interface**
CSS	**– Cascading Style Sheets**
CMS	**– Content Management Systém**
DDL	**– Data Definition Language**
HTML	**– HyperText Markup Language**
IS	**– Informační systém**
JS	**– JavaScript**
PHP	**– Hypertext Preprocessor**
PLM	**– Product Lifecycle Management**
SQL	**– Structured Query Language**
URL	**– Uniform Resource Locator**
WYSIWYG	**– What You See Is What You Get**

Seznam obrázků

Seznam výpisu zdrojového kódu

1 Úvod

Při výběru zadání bakalářské práce jsem zvolil možnost absolvovat a popsat odbornou praxi, protože tato možnost pro mě už od začátku představovala výhodu v podobě cenných zkušeností, které bych jinak pravděpodobně získal až po ukončení školy. Při výběru odborné praxe jsem si vybíral mezi několika firmami. Některé z nich měly pro mne velmi lákavé pozice, nicméně komunikace nebyla až tak dobrá. Proto jsem si vybral firmu, jejíž komunikace byla na velmi dobré úrovni, a to i za cenu jiného zaměření, než jsem původně vyhledával.

Praxe mi poskytla příležitost dostat se do firemního prostředí a poznat tak celý postup vývoje software i z jiného pohledu než jen ze školní lavice. To, že jsem na své pozici pracoval s jinými technologiemi už zase tak nevadilo, musel jsem se však toho naučit na začátku i v průběhu praxe podstatně více.

Firma, ve které jsem působil, se jmenuje ISSA CZECH s. r. o., sídlící v centru Ostravy. Má pozice byla zaměřena na vývoj „front-end“ a „back-end“ částí webových stránek a systémů.

V této práci popisuji realizaci a práci na celkem třech projektech. V prvním projektu jsem řešil vývoj jednoduché webové stránky. Jednalo se o informativní stránku pro jednu stavební společnost sídlící ve středních Čechách. Projekt sloužil hlavně k seznámení s firemním CMS. Jelikož jsem byl ve vývoji webových stránek v podstatě úplný začátečník, poskytl mi tento projekt možnost zopakovat, a hlavně prohloubit mé znalosti základních webových technologií, díky kterým jsem se následně mohl zaměřit na složitější projekty.

Webové stránky, které firma vytvářela, měly většinou stejný nebo velice podobný design, vyplývající z požadavků klientů. Další projekt, který jsem dostal, měl tento proces vývoje zautomatizovat a pokud možno co nejvíce omezit nutnost znát při tvorbě stránek nějaký programovací jazyk. Díky této automatizaci by si vzhled stránky mohl upravovat i klient, a to pouze za pomocí administračního prostředí.

Třetí projekt se zaměřoval na vývoj informačního systému na míru, který sloužil k řízení životního cyklu výrobků a správě úkolů. Na projektu pracovalo více lidí, moje práce se proto týkala specifické části.

V této práci se zmiňuji také o využití znalostí získaných v průběhu studia, chybějících znalostech, přínosu a hodnocení mé praxe.

2 Popis firmy a pracovní zařazení studenta

2.1 O firmě

Společnost ISSA CZECH s. r. o. sídlící v Ostravě, poskytuje služby zaměřující se na informační technologie už více než 20 let. Mezi tyto služby patří například vývoj a provoz webových stránek, informačních systémů a e-shopů. Kromě vývoje se zabývá také webdesignem, internetovým marketingem, optimalizaci pro vyhledávače (neboli SEO) a mnoha dalšími. Vedle toho poskytuje i v menší míře vývoj mobilních aplikací.

I když se nejedná o nějak velkou firmu, svým působením se bez pochyb dokáže vyrovnat i těm větším. To potvrzuje fakt, že firma poskytuje služby pro klienty jako jsou například Skylink – M7 Group S.A., Třinecké železárny, a. s., Barandov Studio a. s. nebo Nestlé Česko s. r. o., a to je pouze zlomek celého seznamu firem.

ISSA CZECH s. r. o. se v posledních letech začala také rozvíjet na poli komunikačních technologií, a to tím, že začala poskytovat vlastní internetové připojení pro oblast Ostrava-Heřmanice. Tato služba nese název ISSANET.

Firma je prodejním partnerem společnosti ESET, která se zabývá vývojem antivirových programů pro stolní i mobilní zařízení. V této oblasti poskytuje ISSA také poradenskou a konzultační činnost. [1]

2.2 Pracovní zařazení studenta

Jak jsem se již zmínil v úvodu, má pozice byla zaměřena na vývoj „front-end“ a „back-end“ částí webových stránek a systémů. Pod pojmem „front-end“ si lze představit vše, co může návštěvník na webové stránce vidět. Pojem „back-end“ pak představuje vnitřní chování webové aplikace, kterou návštěvník vidět nemůže. K realizaci zadaných úkolů jsem používal primárně firemní rámce.

Ve „front-end“ části jsem pracoval s nejznámějšími technologiemi, jako jsou HTML5, CSS3 a JavaScript (rámec jQuery). Při práci na „back-endu“ pak nejčastěji s jazyky PHP a SQL. Po celou dobu praxe jsem byl veden mým vedoucím bakalářské praxe, se kterým jsem konzultoval všechny návrhy a problémy. Vedle něj jsem měl po ruce kolegu, který mě provázel při práci s rámci.

2.3 Poskytované služby

Rád bych zde chtěl ještě popsat některé služby, které firma poskytuje.

2.3.1 Webové stránky

Firma poskytuje kompletní realizaci webových stránek od vývoje včetně originálních grafických návrhů a webdesignu až po jejich správu. K tvorbě se využívá jeden z firemních systémů, který je určen pro správu obsahu webových stránek. Tento redakční systém představuje užitečný nástroj, díky kterému může klient jednoduše editovat texty, obrázky, bannery, dokumenty nebo soubory.

Pouhým předáním hotových stránek klientovi spolupráce nutně nemusí končit, firma totiž nabízí i například monitoring a ladění. Vyhodnotí se chování a provoz návštěvníků, díky čemuž se mohou provést opatření zvyšující celkový výkon webové prezentace. [2]

2.3.2 Informační systémy

Jednou z nejžádanějších služeb této společnosti je realizace IS na míru. Vyvíjí různě velké projekty, které směruje v rámci nejnovějších trendů v oblasti. Využívá opět své vlastní řešení za pomocí vlastních rámců, přičemž je kladen důraz na práci s nejmodernějšími technologiemi. Mezi hlavní využívané internetové technologie patří například Java, PHP a .NET. [3]

2.3.3 E-shopy

Další velmi využívanou službou je realizace menších nebo robustních e-shopů. Stejně jako u většiny služeb firma využívá vlastní řešení, a tak dokáže vyhovět specifickým požadavkům klienta. Podle požadavků dokáže nasadit řešení, které se zaměří například na nároky na administraci položek, počet položek a frekvenci uživatelských přístupů nebo integraci se skladovým a ekonomickým systémem. K e-shopu je také nabízena implementace ověřených platebních metod jako jsou například GoPay, PayPal, GP webpay nebo platba pomocí SMS. [4]

2.3.4 Internetový marketing

Pokud chce dnes člověk něco propagovat, neměl by se zaměřit pouze na média jako je tisk, rádio nebo televize, ale měl by využít internetový marketing. Ten je v současnosti nedílnou součástí úspěšné marketingové reklamy, a pokud je realizován s dobrým poměrem „výkon/cena“, lze za poměrně nízké náklady zasáhnout velkou skupinu potenciálních zákazníků.

ISSA CZECH s. r. o. nabízí v této oblasti komplexní řešení, které zahrnuje webové stránky, webdesign, SEO, PPC kampaně, bannerovou reklamu a další typy reklam. Také se zaměřuje na největší sociální sítě, tvorbu PR článků, microsites (malé weby doplňující hlavní webovou prezentaci) a v neposlední řadě měřením účinnosti a optimalizací. [5]

2.3.5 Webdesign

Grafická řešení webových stránek jsou u této společnosti na profesionální úrovni. Při návrhu je kladen důraz na ergonomii, psychologii chování cílového uživatele, aktuální trendy v oblasti webdesignu, podporu všech prohlížečů a nasměrování návštěvníků ke konverzním cílům. [6]

2.3.6 SEO

Velmi důležitou součástí internetového marketingu je tzv. SEO optimalizace. Jedná se o úpravu webových stránek takovým způsobem, aby byly vhodné pro automatizované zpracování v internetových

vyhledávačích. Výsledkem je pak lepší pozice ve výsledcích vyhledávání, čímž lze na své stránky dostat větší počet návštěvníků.

Společnost se SEO optimalizací dlouhodobě zabývá, poskytuje krátkodobé řešení v podobě jednorázové SEO optimalizace nebo efektivnější rámcové řešení, kde se pravidelně udržují všechny SEO faktory podle aktuálního vývoje trendů. [7]

2.3.7 Datové centrum

Ve výčtu služeb se můžeme také dočíst o datovém centru. To může klient využít k provozu webových stránek, informačních systémů nebo třeba emailů. Datové centrum je postaveno na špičkových technologiích a je zajištěna bezpečnost na velmi vysoké úrovni. Můžeme se spolehnout na zabezpečení proti ztrátě a úniku dat a díky serverovým platformám od společnosti HP je zaručena vysoká dostupnost a minimalizovaný počet výpadků.

Za zmínku stojí také vlastnosti jako je klastrování, virtualizace, zálohovací systém, balancování výkonu podle aktuálního zatížení serverů a v neposlední řadě bezpečné umístění celého zařízení v klimatizované místnosti se záložním zdrojem. [8]

3 Technologie

3.1 CSS3

Kaskádové styly dnes představují jednoduchý způsob, jak naformátovat vzhled webových stránek. Od svého vzniku v roce 1997 prošel jazyk řadou modifikací a stal se velice oblíbeným. Pomocí stylů můžeme velmi přesně definovat vzhled každého elementu. Oproti staršímu způsobu formátování pomocí atributů v HTML umožňují kaskádové styly oddělit vzhled dokumentu od jeho obsahu, což výrazně zvyšuje přístupnost webu. Mezi další výhody patří například větší možnosti formátování, snazší správa větších prezentací a menší zatížení serverů. [9]

V nejnovější verzi můžeme najít například stíny u textu, transformace, zaoblení rohů prvku nebo třeba animace.

3.2 HTML5

HTML je značkovací jazyk, který slouží k sestavení struktury webové stránky. Struktura je tvořena značkami (elementy), které značí, jestli se jedná o nadpis, odkaz, text, obrázek, odstavec nebo například tabulku. K elementu mohou být přidány atributy, jejichž hodnoty definují adresu odkazu, cestu k obrázku, rozměr obrázku nebo třeba počet sloupců v tabulce. Element spolu s atributy je ohraničen špičatými závorkami, za které se následně může vložit obsah, tím může být text, další element atd. Pokud se jedná o párový element, je třeba za vložený obsah přidat koncový element, který je odlišen lomítkem před jeho názvem a je opět ohraničen závorkami. Tím se ukončí celý příkaz.

Nejnovější verze jazyka přináší například elementy pro vkládání multimediálního obsahu nebo plátno pro práci s vektorovou grafikou. Další zajímavou novinkou je možnost pracovat s lokálním uložištěm prohlížeče.

Spolu s kaskádovými styly tvoří tento jazyk základ pro tvorbu webových prezentací. Dalo by se říci, že HTML tvoří kostru a CSS kůži webové stránky. [10]

3.3 JavaScript

JavaScript pomáhá programátorovi řešit věci, na které HTML a CSS nestačí. Využití je poměrně rozsáhlé, od různých grafických úprav a efektů (rozbalovací menu nebo třeba efekt padajících sněhových vloček) přes validaci vstupních dat nebo formátování textu až po plnohodnotné aplikace.

Co se týče technických informací, JavaScript je jazyk interpretovaný, dynamicky typovaný, objektově orientovaný, využívající funkcionální paradigma (do běžné proměnné lze vložit funkci). Funguje na straně klienta, ale existují i tzv. server-side (zpracování požadavků na straně serveru) varianty jménem Node.js a Apache.

Také je nutno poznamenat, že mnoho lidí si plete nebo nějak spojuje JavaScript s jazykem Java. Pravdou je, že oba jazyky mají jen velmi málo společného. Kromě podobných názvů má JavaScript do jisté míry pouze podobnou syntaxi, ale jinak se sémanticky jedná o zcela různé jazyky. [11]

3.4 jQuery

Je nejrozšířenější javascriptový rámec licencovaný jako „Open-source“. Výrazně ulehčuje a urychluje tvorbu webových stránek a aplikací. Nejvyužívanějšími vlastnostmi je jednoduchá manipulace s CSS a HTML elementy pomocí selektorů, používání animací a efektů, práce s událostmi a AJAXem. jQuery se zaměřuje na psaní jednoho kódu, který funguje pro všechny prohlížeče. Při použití čistého JavaScriptu je totiž mnohdy nutné psát pro každý prohlížeč specifický kód. [12]

3.5 SQL

Jazyk SQL je standardním jazykem používaným pro manipulaci a získávání dat z relačních databází. Pomocí jazyka SQL může programátor nebo správce databáze upravovat strukturu databáze, měnit nastavení zabezpečení systému, přidávat uživatelská oprávnění k databázím či tabulkám, dotázat se databáze na nějakou informaci nebo aktualizovat obsah databáze. SQL neobsahuje řídicí příkazy a je neprocedurální, existují však rozšíření jako například T-SQL nebo PL-SQL, které tyto nedostatky doplňují.

Historie jazyka začala na konci sedmdesátých let dvacátého století v Kalifornii, kde byl v laboratořích společnosti IBM vyvinut.

Dnes se můžeme setkat s velkým počtem SQL databází (např.: Oracle, MS-SQL, MySQL, SQLite atd.), kde většina podporuje standard ANSI SQL. [13]

3.6 PHP

Je objektově orientovaný interpretovaný skriptovací jazyk fungující na straně serveru. Jedná se o nejrozšířenější webový skriptovací jazyk, který je neustále vyvíjen, nejnovější verze má pořadové číslo sedm. Je dynamicky typovaný se syntaxí podobnou jazyku C.

Pomocí PHP lze vytvořit informační systémy, e-shopy, diskuzní fóra nebo třeba sociální sítě. Jsou v něm napsané projekty jako například Facebook a Wikipedie.

Díky tomu, že PHP pracuje na straně serveru, je zdrojový kód skryt před uživateli. Těm se zobrazí v prohlížeči pouze HTML kód, který je poskládán na serveru. Oproti desktopovým aplikacím to přináší značnou bezpečnostní výhodu. Vedle toho odpadá i nutnost aplikace stahovat nebo instalovat.

Mezi příklady využití patří generování dynamicky se měnícího obsahu stránky, validace a zpracování formulářů, dotazování a manipulace dat v databázi nebo jejich šifrování. [14]

3.7 Bootstrap

Tento „front-end“ rámec byl původně navržen pro Twitter kvůli údržbě velkého množství nekonzistentního kódu. Později byl uvolněn jako otevřený software pro širokou veřejnost.

Bootstrap výrazně urychluje práci při tvorbě vzhledu webových stránek. Obsahuje hotové styly, které lze jednoduše nasadit na HTML kód. Doba učení je poměrně krátká, díky tomu se stal oblíbeným u méně zkušených vývojářů. Dá se říci, že odpadá nutnost spolupráce programátora s grafiky, jelikož programátor dokáže díky Bootstrapu vytvořit kompletní webovou prezentaci sám.

Rámec podporuje responzivní design, takže se rozvržení stránky přizpůsobí na základě použitého zařízení (telefony, počítače, tablety). Je podporován na všech nejpoužívanějších prohlížečích, a to i pro starší verze jako je například Internet Explorer 8. [15]

4 Seznam zadaných úkolů

4.1 Doteur.cz

4.1.1 Úvod k projektu

První věc, se kterou mě vedoucí praxe seznámil po mém nástupu, byl firemní systém pro správu obsahu, který se nazývá Content Management System (neboli CMS) Designer. Zkušený webový vývojář by měl ihned po zaslechnutí tohoto názvu určitě nějakou představu, o co se jedná a nebylo by to pro něj něco nového. To však nebyl můj případ, jelikož jsem byl v tomto odvětví prakticky úplný začátečník (pracoval jsem doposud pouze s HTML, CSS a jQuery při tvorbě primitivních statických stránek a v rámci předmětu TAMZ I), a tak byl CMS pro mě zcela nový pojem.

Vývoj stránek prostřednictvím redakčního systému byl jeden z nejlepších způsobů, jak proniknout do světa vývoje webových stránek. Za pomocí CMS Designeru jsem tedy tvořil můj první projekt. Webové stránky pro společnost DOTEUR Stavby s. r. o. bylo téma mé první práce ve firmě.

4.1.2 Zadání

Představa klienta byla celkem prostá, stránky měly být jednoduché a přehledné. Na stránkách se má nacházet menu spolu s logem a možností volby jazyka (klient požadoval jako druhý jazyk angličtinu). Pod logem má poutat pozornost „Posuvný banner“ (dále jen „posuvník“) poskytující informace o aktuálním dění ve firmě. O kus níže následuje sekce „O společnosti“. Pod těmito informacemi lze najít asi to nejdůležitější, a to jsou služby, které firma poskytuje. Ve spodní části se nachází sekce pro kontakt, a pokud by návštěvník chtěl vědět, kde firma sídlí, tak nalezne mapu na úplném konci stránky. Jedním s dalších požadavků je, aby byla webová prezentace responzivní.

K zadání jsem dostal obrázky a dokument s textovým obsahem stránek spolu s jeho anglickým překladem. Firemní grafik mi poslal návrh, podle kterého jsem měl stránku vytvořit.

4.2 Uniweb

4.2.1 Úvod k projektu

Poptávka po webech jako je Doteur.cz bylo mnoho a většinou se prvky nebo rozvržení stránky moc neměnily. U tohoto projektu jsem nepracoval pro žádného klienta, ale pro naši firmu. Měl jsem pro firmu vytvořit nástroj, který měl umožnit vytváření takových stránek bez nutnosti umět kódovat. Tyto stránky by mohl vytvářet kdokoliv, a také by to mnohem urychlilo samotnou tvorbu stránek. K realizaci jsem využil opět firemní CMS Designer.

Díky těmto vlastnostem se projekt dá přirovnat k nástrojům pro tvorbu webových stránek jako je například Wix.com, ve kterém lze sestavit celou stránku pouze pomocí operací „drag and drop“.

4.2.2 Zadání

Aby se dala stránka vytvořit pouze v CMS Designeru bez jakéhokoliv zásahu do kódu, musel projekt umožňovat definovat její vzhled, z jakých částí se bude stránka skládat (neboli strukturu stránky) a jak bude obsah na stránce stylován. Jak jsem se již zmínil, tak je tento projekt přizpůsoben pro vytváření podobných stránek jako je Doteur.cz. Z toho vyplývá, že nemáme při tvorbě vzhledu stránky takovou volnost jako například u přirovnávaného Wixu, kde si lze navolit šablony a používat „drag and drop“ operace. Takový projekt by určitě nebyl pro jednoho člověka v nějakém rozumném časovém úseku proveditelný. Celá stránka byla tedy rozdělena pouze do osmi nejpoužívanějších částí (které klienti na svých stránkách požadují), ze kterých se stránka dala poskládat, což pro tyto jednoduché stránky bohatě stačí. Těmito částmi můžeme rozumět například hlavičku, posuvník, text, patičku atd.

Co se týče vzhledu, tak měla mít každá část své vlastní nastavení. To zahrnovalo barvu textu, barvu odkazu, možnost pro podtržení odkazu, barvu pozadí nebo nastavení obrázku, který se opakováním vyplní přes celé pozadí (dále jen „dlaždice“).

Stylování obsahu mělo umožňovat definovat pro celou stránku styl písma, jeho velikost a barvu.

Toto byly hlavní požadavky. Vedle nich následovaly ještě další menší požadavky, které se týkaly jednotlivých částí, jako například pozice prvků v „Obrázkovém menu“ a možnost přepínání obrázků po jeho kontaktu s kurzorem. Dále možnost volby, jestli bude menu ve formě „hamburgerového menu“ nebo klasického a mnoho dalších.

4.3 HPLM (Hybrid Product Lifecycle Management)

4.3.1 Úvod k projektu

ISSA v době mého působení ve firmě pracovala na projektu s názvem HPLM (Hybrid Product Lifecycle Management). V tomto případě se tedy nejednalo pouze o můj projekt, jelikož na něm pracovalo více zaměstnanců. Jak název napovídá, jedná se o systém PLM, který slouží k řízení životního cyklu výrobku. Systém byl vyvíjen za pomocí dalšího firemního PHP rámce. Mým úkolem bylo tento systém o určitou část rozšířit.

4.3.2 Zadání

Pro lepší porozumění mého zadání, je nutné nejprve čtenáři představit část systému, se kterou jsem pracoval. V systému lze vytvořit položku, tou může být cokoliv, na co bude systém využíván, zde uvedu například sluchátka. Této položce lze pak přiřadit její typ, například že se jedná o sluchátka do uší. Na základě typu má pak položka specifické atributy, řekněme pro velikost konektoru Jack, barvu nebo třeba frekvenční rozsah. Je jasné, že každý typ může mít i jiný počet atributů (bezdrátová sluchátka nebudou mít žádnou informaci o délce kabelu). Vždy při změně typu položky se tedy změní všechny její atributy.

Vše, co mi k mému zadání bylo řečeno, je zprovoznit toto dynamické načítání atributů k položce na základě jejího typu (atributy se měly generovat v detailu položky). Ze začátku se to jevilo jako poměrně

malý úkol, který nezabere mnoho času, avšak abych toto mohl zprovoznit, bylo zapotřebí „rozchodit“ mnoho jiných věcí, které této funkčnosti předcházejí. Z tohoto jednoho požadavku tedy vzešlo mnoho dalších, které představím a popíši dále v kapitole 5.3.1, „Analýza a návrh“. Jak budu tyto věci řešit, už bylo ponecháno na mě.

5 Řešení zadaných úkolů

5.1 Doteur.cz

5.1.1 CMS Designer

Jak jsem se zmínil v úvodu k projektu, chtěl bych nejdřív popsat firemní CMS Designer. Tento systém byl úplně první, se kterým jsem v oblasti vývoje webových stránek setkal. První dávku informací jsem dostal ve formě prezentace na prvním sezení s mým vedoucím, kde jsme prošli uživatelské rozhraní systému. Při dalším setkání mi byl již poskytnut instalační balíček CMS Designeru a dozvěděl jsem se, jak jej nainstalovat. K provozu je třeba mít nějakou databázi a server. K tomu jsem použil balíček XAMPP, který obsahuje databázi MySQL a Apache server.

Před instalací je nutné vytvořit novou databázi, se kterou bude systém pracovat. Samotná instalace designeru je poměrně jednoduchá. Po rozbalení CMS souboru do pracovní složky stačí otevřít v prohlížeči instalační stránku, která požaduje vyplnění informací jako je název stránky, adresa, název databáze a její přihlašovací údaje, přihlašovací údaje do administračního rozhraní stránky atd. Dále je také možno specifikovat jazykové variace. V základu je nastaven jazyk čeština a angličtina, změna jazyka se dá vypnout. Díky poskytnutým informacím se systém napojí na databázi a automaticky vytvoří potřebné tabulky. Po dokončení instalace se zobrazí úvodní stránka, která má základní vzhled s firemním motivem.

Nyní je na řadě, abychom si začali stránku upravovat. Po otevření administračního prostředí nalezneme na první pozici v menu sekci s názvem „Struktura“. Ve struktuře lze vytvořit uzly, reprezentující například podstránky nebo odkazy. Podstránky mají dva typy, statická a dynamická. První slouží pro zobrazení statického obsahu, použijeme jej tedy například pro nějaký text (obsah editujeme pomocí WYSIWYG CKEditoru). Pokud potřebujeme dynamicky se měnící obsah, zvolíme typ druhý. Dynamickou podstránku je třeba propojit s modulem, ze kterého bude stránka čerpat data.

„Modul“ nalezneme v menu na třetí pozici. Pod tímto názvem si můžeme v tomto systému představit jednoduchou databázovou tabulku. Detail modulu se skládá ze čtyř záložek, v první definujeme základní vlastnosti jako název, popis nebo typ modulu. V druhé záložce lze tvořit komponenty (což jsou něco jako atributy v databázové tabulce), které se skládají z názvu a typu komponenty. Komponenty nám tedy definují, jaké záznamy bude modul obsahovat. Další záložka s názvem „Vzhled“ řeší, jak budeme záznamy modulu vypisovat na naši stránku. V záložce nalezneme editor spolu se seznam příkazů a proměnných, ve kterém můžeme způsob vypisování definovat. Seznam obsahuje například příkazy pro větvení a cykly, díky kterým procházíme záznamy modulu a vkládáme je do HTML kódu. Tento seznam lze přirovnat například k Twigu, což je zásuvný modul pro CMS WordPress. Podobně, jako v Twigu jsou příkazy v našem seznamu ohraničeny složenými závorkami. V tomto editoru je také nutné odlišit, jak budou vypadat záznamy modulu v administraci a jak na stránce. V poslední záložce nalezneme přehled všech záznamů s možností jejich přidání, editace a smazání.

Dalšími položkami v menu, které jsem ve svých projektech využil, jsou „Widgety“ a „Katalog textů“. Komponentu „Widget“ (dále jen „widget“) lze propojit s modulem a jeho obsah vypsat zavoláním metody

pro vypsání widgetu kdekoliv v „layoutu“ (v této práci budu používat pro pojem „layout“ výraz *rozvržení*), to v případě modulu propojeném s dynamickou stránkou nelze. Proto je v některých případech vhodnější pracovat s widgety. „Katalog textů“ pak obsahuje seznam proměnných, se kterými lze pracovat v kódu.

To by bylo asi tak vše k administrační části. Z pohledu programátora máme k dispozici v adresáři CMS Designeru složky s PHP soubory obsahující *rozvržení* administrace a samotné stránky, do kterých můžeme zasahovat. K tomu máme i příslušnou složku s CSS soubory, které jsou rozděleny rovněž pro administraci a stránku. Pokud chceme použít JavaScript, nic nám nebrání vytvořit si pro něj složku a jeho soubory následně připojit v *rozvržení*, kde je budeme používat.

Co se týče technických specifikací, systém je napsán v jazyce PHP a je podporován pro verzi 5.4, obsahuje rámec Bootstrap a knihovny pro javascriptový rámec jQuery, čímž odpadá nutnost je stahovat.

5.1.2 Analýza a návrh

Na začátku jsem si prošel grafický návrh a určil z jakých částí se stránka skládá. Než se k tomu ale dostanu, rád bych se nejdříve zmínil, s jakými nástroji jsem při manipulaci s grafikou pracoval.

Grafický návrh stránky jsem obdržel ve formátu spustitelném v programu Adobe Illustrator. Jedná se o vektorový grafický editor (veškeré grafické prvky si na rozdíl od rastrových editorů uchovají svou kvalitu i po aplikování různých transformací). Tento editor jsem využíval po celou dobu mé odborné praxe, všechny potřebné grafické prvky jsem mohl v případě potřeby jednoduše z návrhu vyřezat a použít na stránce. Velmi jednoduchá je také práce s barvami, jelikož po použití nástroje „kapátko“ na požadovanou barvu se vrátí její hexadecimální RGB formát, který je používán v CSS. Rastrové soubory jako jsou například obrázky a loga jsem dostal k návrhu samostatně bokem ve vysoké kvalitě.

V prvé řadě jsem si musel určit, pro které části bude nezbytné vytvořit moduly a jaké prvky bude obsahovat struktura. Z návrhu vyplývalo, že bude zapotřebí vytvořit dva moduly, pro „Posuvný banner“ (posuvník) a „Služby“, počet záznamu v těchto prvcích se bude totiž měnit. Dále bylo potřeba vytvořit ve struktuře dvě podstránky pro sekci „O společnosti“ a „Kontakt“. Tyto stránky budou statické, jelikož jejich obsah bude pouze prostý text. Zbývající části stránky jako je mapa společnosti a hlavička se bude řešit v *rozvržení*. Menu v hlavičce stránky obsahuje tři položky, ty odkazují na podstránky. Pro ně se vytvoří ve struktuře tři uzly typu „LINK“ tedy odkaz.

Jedním z požadavků pro menu je, aby po „rozkliknutí“ jeho obsah plynule sjel dolů (tzv. Dropdown efekt). S tímto jsem doposud neměl žádné zkušenosti, ale tušil jsem, že by bylo dobrou volbou pro tento efekt využít jQuery, což mi potvrdil i zkušený kolega.

Dalším úkolem bylo určit komponenty v modulech. Pro posuvník jsem navrhl na základě jeho vlastností komponenty pro nadpis, obrázek a pozici obrázku v posuvníku. Pro modul služeb stačily komponenty dvě, a to pro nadpis a popis dané služby.

Pro nadpisy v sekcích „O společnosti“, „Služby“ a „Kontakt“ jsem se rozhodl použít katalog textů. Nadpis služeb je totiž jediným statickým textem v této sekci, proto mi přišlo zbytečné vytvářet ve struktuře pro jeden nadpis další statickou podstránku. Nadpisy z katalogu textů lze vypsat kdekoliv na stránce, což

mi zrovna vyhovovalo, a jelikož mi přišlo dobré mít všechny nadpisy v administraci na jednom místě, přidal jsem do katalogu i zbylé nadpisy pro sekci „O společnosti“ a „Kontakt“.

5.1.3 Implementace

Po vytvoření databáze a dokončení instalace CMS Designeru, jsem v první řadě začal pracovat na *rozvržení* stránky. U tohoto projektu jsem pracoval s HTML5 a CSS3, složitější prvky jsem se rozhodl řešit pomocí Bootstrapu. V *rozvržení* jsem si vytvořil základní kostru pro všechny části stránky a všem elementům jsem přiřadil názvy tříd a identifikátorů pro práci ve stylech apod.

Hned v dalším kroku jsem se rozhodl přejít do administrace, kde jsem vytvořil a nastavil vše nezbytné. To hlavně z toho důvodu, abych věděl, jak se bude chovat obsah stránky, až ji budu stylovat. Vytvořil jsem tedy zmiňované moduly pro posuvník a služby a všechny navrhované podstránky ve struktuře. Dále položky typu „LINK“ pro menu a nadpisy v katalogu textů. Všechno jsem následně vyplnil přiloženým obsahem pro obě jazykové skupiny.

Nyní jsem se mohl vrhnout na stylování. Nebudu zde popisovat stylování nějak výrazně do hloubky, spíš bych se chtěl zmínit o zajímavostech a problémech, na které jsem při stylování narazil.

Práce na hlavičce byla celkem bezproblémová, zde jsem řešil zmiňovaný efekt „slidování“ menu při jeho „rozkliknutí“. Pro tento efekt je určena jQuery funkce `slideToggle()`. Vedle menu jsem ještě v hlavičce řešil přepínání loga, to mělo dvě různé verze pro oba jazyky. Jelikož se logo nebude nějak často měnit, tak bylo zbytečné pro něj vytvářet modul. Přepínání jsem tedy vyřešil jednoduše ternárním operátorem, který řešil, která cesta k obrázku loga se vloží do atributu příslušného elementu.

Při práci na posuvníku mi značně pomohl Bootstrap, který poskytuje pro posuvníky velmi pěkné řešení. Díky tomu jsem tedy řešil pouze to, jak na tento posuvník napojit vytvořený modul. Modul jsem propojil v administraci s nově vytvořeným widgetem, který lze následně vypsat pomocí metody `runWidget()`. Metodě je nutné předat parametry pro to, co chceme vypsat (v našem případě modul) a jeho ID. Nyní bylo zapotřebí ještě definovat v administraci vzhled modulu, ten nalezneme ve výpisu 1.

Kód popisuje výpis všech záznamů modulu pomocí příkazu `foreach`. Dále je z kódu patrné, že bylo potřeba rozlišit, jak bude záznam vypadat v administraci a jak na stránce. To za pomocí testování návratové hodnoty metody `isAdministrationMode()`. Uvnitř rozhodovacích bloků pak můžeme vidět HTML kód, který se z modulu vypíše na naší stránce.

V dalších krocích jsem řešil zobrazení statických podstránek „O společnosti“ a „Kontakt“. Zde nebylo skoro ani co řešit. Vytvořil jsem si widgety a obě podstránky jsem vypsal opět pomocí metody `runWidget()`, akorát s tím rozdílem, že jsem metodě předal parametr pro výpis stránky a ne modulu, to platilo rovněž i pro ID. Vzápětí jsem mohl vyřešit i interaktivní mapu se sídlem společnosti, která se má nacházet na konci stránky. K tomu jsem využil element `<iframe>`, který jsem vygeneroval pomocí The Google Maps Embed API.

Poslední zbývající sekce pro služby měla vypadat tak, aby se na jeden řádek vedle sebe vešly přesně tři služby, zbylé se následně vypíší na další řádek, a to vždy tak, aby byly co nejblíže ke středu. Pro tento

požadavek mi postačily pouze kaskádové styly. Další požadavek k těmto službám mě však donutil použít jQuery. Než se k tomu dostanu, je potřeba nejdříve popsat, jak služba vypadá.

```
{cond:if}(!isAdministrationMode()){cond:then}
  <div class="carousel-inner" role="listbox">
{cond:end}

{foreach:begin}

{cond:if}(isAdministrationMode()){cond:then}
<div>
    <a href="{itemDetailPath}">{resource:Show}</a>
    <img class="img-responsive" style="max-width: 200px; max-height:
      200px;" src="{img.Obrazek.value}" alt="screen">
</div>
{cond:else}
    <div class="item {cond:if}{itemNumber} == 1
      {cond:then}active{cond:end}">
      <img src="{img.Obrazek.value}" alt="{title.Titulek.value}">
    </div>
{cond:end}

{foreach:end}

{cond:if}(!isAdministrationMode()){cond:then}
  </div>
{cond:end}
```

Výpis 1: Definice vzhledu modulu pro posuvník

Každá služba má nadpis, pod kterým se nachází její popis, oba tyto texty pak rozděluje pruh. Celý tento obsah má bílé pozadí ve tvaru obdélníku. Požadavkem bylo, že pokud bude popis služby větší než pozadí, tak se výška pozadí automaticky přizpůsobí velikosti tohoto textu (zde jsem si při implementaci rovněž vystačil s kaskádovými styly). Zároveň se však výška pozadí všech služeb měla přizpůsobit výšce největší služby. Různě vysoké služby by jinak tvořily zuby, které by výrazně kazily celý vzhled. Pro tuto funkčnost jsem tedy využil jQuery, kde jsem pomocí funkce prošel výšku všech služeb a tu největší jsem aplikoval na ostatní. V opačném případě, kdy by byl text příliš krátký, jsem nastavil minimální výškovou hranici, pod kterou se pozadí nezmenší.

Poslední vlastnost, která stránkám scházela, je, aby byly responzivní. Mnoho zkušených vývojářů si říká, že toto jsem měl řešit ze všeho nejdřív, všechny moderní návrhy se dnes totiž řídí metodou „mobile-first“. V době, kdy jsem na tomto projektu pracoval, jsem to však nevěděl. Naštěstí se nejednalo o velký projekt, a tak mě moje metoda „desktop-first“ postupu nestála prakticky žádný čas navíc. Při řešení jsem

využil CSS Media Queries, kde jsem testoval šířku okna větší než 768px, 992px a 1200px. Tyto konkrétní hodnoty též nazývané „breakpointy“, má takto přednastavené například rámec Bootstrap. V podstatě jsem takto vytvořil čtyři intervaly, které obsahovaly své specifické styly. Stránka se tedy dokázala přizpůsobit většině zařízení.

Po dokončení projektu jsem ověřil, že vše funguje, jak má a ze svého lokálního uložiště jsem jej mohl předat kolegovi, který stránku vystavil na firemním datovém centru.

5.1.4 Poznatky k projektu

Vzhledem k tomu, že jsem doposud se systémy pro správu obsahu neměl žádné praktické zkušenosti, tak nemohu tento firemní systém srovnávat s jinými. Nicméně mohu říci, že se z mého pohledu CMS Designer jeví jako stabilní redakční systém, který umožňuje rychlý a pohodlný vývoj webových stránek.

Přínos projektu Tento projekt mi umožnil získat mnoho cenných zkušeností. S většinou věcí jsem se setkal poprvé a hodně jsem se toho musel učit „za pochodu“. Zopakoval a prohloubil jsem si znalosti kódování a stylování a přišel jsem na to, v jakých situacích je dobré využít JavaScript (resp. jQuery). Projekt mi přinesl první kontakt s jazykem PHP a měl jsem také možnost poznat, jaké je to pracovat na stránkách s redakčním systémem a stránkách vůbec. V budoucnu mi bude v případě práce s nějakým redakčním systémem doba potřebná k jeho ovládnutí mnohem menší. Taky mi už nebude tolik pojmů v této oblasti cizích, jako tomu bylo před nástupem do firmy.

5.2 Uniweb

5.2.1 Analýza a návrh

K projektu jsem dostal opět grafický návrh stránky, ten však sloužil spíše pro inspiraci. K návrhu byly přidány i odkazy na další stránky vytvořené touto firmou, u kterých jsem se mohl rovněž nechat inspirovat. Mezi těmito stránkami byla i mnou vytvořená stránka Doteur.cz. Vedle návrhu jsem dostal textový dokument, který shrnoval všechny požadavky.

Hlavní myšlenkou tohoto projektu je určit, z jakých částí se stránka bude skládat a jak budou tyto části vypadat, to vše bez nějakého zásahu do kódů. S vedoucím jsme se shodli, že by uživatel stránku skládal za pomocí „Katalogu textů“. Zde by jednoduše povolil, co na stránce bude. Hodnota proměnné by mohla být třeba „ANO“ nebo „NE“ (o zvoleném formátu vstupů se zmíním v následující kapitole „Implementace“). Katalog textů by posloužil ještě pro přepínání jazykové variace a definování velikosti a typu textu pro celou stránku. Dále si zde mohl uživatel vybrat mezi dvěma vzhledy menu nebo nastavit souřadnice interaktivní mapy.

Když přejdeme ke vzhledu jednotlivých částí, tak pro tento účel bylo nejvhodnější využít moduly, řešit to pomocí katalogů textů by bylo značně nevýhodné, jelikož texty nejde nijak organizovat (například do složek). Parametrů pro styly všech částí bude určitě mnoho, a tak by byl celý katalog velmi nepřehledný. Navíc bych v katalogu textů mohl pracovat pouze s řetězci. Nevýhodou však je, že bude nutné zjistit, jak získat data z modulů a použít je v *rozvržení*.

Jelikož každá část sdílí některé vlastnosti, tak by bylo dobré vytvořit jeden modul s názvem „Style“, který bude obsahovat sedm záznamů pro všechny části (sedm, jelikož u mapy není co stylovat). Každý z těchto záznamu bude mít na základě požadovaných vlastností komponenty pro barvu textu, barvu odkazu, barvu pozadí, možnost podtržení odkazu a komponentu pro nastavení „dlaždic“ místo pozadí (pro připomenutí dlaždicí rozumíme malý obrázek, který se opakováním vyplní přes celé pozadí).

Tímto se pokryly společné vlastnosti všech částí, zbývají ještě ty, které jsou pro každou část specifické. To se týká hlavičky, posuvníku a části nazvané „Image Menu“, která je podobná službám v projektu Doteur. Mezi jejich specifické vlastnosti patří například průhlednost pozadí, barva tlačítek nebo třeba barva konkrétní textové části, která měla být od ostatních odlišná.

Takto jsem pokryl většinu požadavků pro to, jak může uživatel stránku stylovat. Další moduly, které bylo nutné vytvořit, sloužily pro naplnění stránky obsahem. To zde není třeba rozebírat, je to v podstatě skoro to samé, co u předchozího projektu, přibyly akorát moduly pro logo a „favicony“ (ikony, které se zobrazují v záložce v prohlížeči), které logicky v tomto projektu nemohou být vytvořeny staticky. Všechny moduly jsem se rozhodl opět propojit s widgety.

Při návrhu struktury nebylo moc co řešit, pro všechny potřebné části se vytvoří podstránky podobně jako u předchozího projektu, akorát jejich počet byl o něco větší. Navíc zde vznikla například podstránka pro posuvník, který byl v tomto projektu složitější, ale o tom až později. Ve struktuře nebudou opět chybět ani uzly typu „LINK“, které se budou vypisovat v menu.

5.2.2 Implementace

Pro projekt jsem vytvořil novou databázi a klasickým způsobem nainstaloval CMS Designer. V prvním kroku jsem se zaměřil na administraci, kde jsem vytvořil všechny navržené moduly, komponenty, podstránky a prvky v katalogu textů.

Jakmile byly všechny základní prvky v administraci vytvořeny, mohl jsem začít pracovat na *rozvržení*. Oproti předchozímu projektu se *rozvržení* moc nelišilo, přibylo akorát pár bloků pro nové části, třeba hlavička měla dvě různé přepínatelné verze. Vytvořil jsem si tedy základní kostru pro všechny části stránky a hned poté jsem mohl rovnou naimplementovat skrývání těchto částí spolu s přepínáním menu prostřednictvím proměnných z katalogu textů. Jednoduše jsem u každé části v kostře testoval, zda hodnota z katalogu nabývá jedničky. Jednička (resp. cokoliv jiného než nula) pro zobrazení a nula pro skrytí této části. Takto jsem určil vstupní hodnoty, které uživatel v katalogů musí zadat. Vše jsem v katalogu popsal v popiscích. Prakticky jsem hned dodělal veškerou funkcionalitu pro celý katalog, jelikož se vesměs jednalo jen o testování vstupních hodnot a vkládání proměnných z katalogu do *rozvržení* nebo do stylů. Jediné, co jsem si nechal na později, bylo zadávání souřadnic do interaktivní mapy, jelikož způsobů, jak toto řešit, bylo více. Nakonec jsme se s vedoucím shodli, že bude pro tuto první verzi stačit, když si uživatel najde požadované souřadnice na mapě, nechá si vygenerovat URL pomocí Google Maps Embed API a to vloží do katalogu. V *rozvržení* jsem pak v elementu `<iframe>` měnil jednoduše pouze URL.

Dalším krokem bylo nastylovat celou stránku. Tvořit vzhled v dojmu již nějaké existující stránky mi přišlo zbytečně matoucí, a tak jsem pro celou stránku nastavil neutrální barvy. Obsah stránky jsem vyplnil pomocí generátoru pseudolatinského textu „Lorem Ipsum“. Rovněž jsem nenastavoval ani žádná již existující loga nebo obrázky, aby stránka vypadala jako čistý projekt, který si uživatel teprve začne upravovat.

Stylování bylo díky zkušenostem z předchozího projektu o něco jednodušší a rychlejší, nicméně ne vše bylo stejné. Nejvíce práce mi zabrala část „Posuvný banner“, „Image Menu“ a hlavička. Pro posuvník jsem využil opět Bootstrap. Posuvník navíc obsahoval ještě text, který ho měl překrývat a indikátory toho, na kterém obrázku v pořadí se zrovna nacházíme. Sekce „Image Menu“ byla podobna službám z Doteuru, její název napovídá, že se jednotlivé prvky neskládaly jen z textu, ale i z obrázku, který fungoval jako odkaz. „Image Menu“ se tedy dalo využít pro více účelů (mohla vzniknout třeba i jednoduchá galerie). Chování těchto prvků se oproti službám lehce lišilo, k řešení jsem opět využil jQuery.

Jak jsem se už zmínil, hlavička měla dvě různé verze, uživatel si v katalogu mohl nastavit, jestli chce mít logo uprostřed nebo na levém kraji. V prvním případě se menu skrylo pod tlačítko napravo, v druhém se menu vypsalo vedle loga po levé straně. Požadavkem na menu bylo, aby mělo více vnořených prvků. Problém byl v tom, že CMS není pro víceúrovňová menu optimalizován. Zeptal jsem se kolegy, jestli s tím nemá nějaké zkušenosti, naštěstí už se tímto problémem někdo ve firmě zabýval a vytvořil pro CMS třídu s názvem „MegaMenu“, která tento problém řeší. Netrvalo dlouho a třídu už jsem měl ve svém adresáři s projektem. Kolega mi ve stručnosti vysvětlil, jak se s třídou pracuje. V podstatě stačí zavolat metodu `printMegaMenu()`, které se předají parametry pro maximální počet úrovní (v našem případě dvě) a název CSS identifikátoru. Metoda sama sestaví HTML strukturu pro víceúrovňové menu, na mě tedy jen

zbývá, abych si menu pomocí předaných CSS identifikátorů nastyloval. Zobrazení vnořených podstránek mělo být doprovázeno několika „slidovacími“ efekty, toto jsem řešil opět pomocí jQuery.

To by bylo ke stylování asi vše, na závěr bych dodal, že optimalizace chování pro mobilní zařízení byla i u tohoto projektu samozřejmostí.

S hotovým vzhledem jsem mohl přejít k implementaci „dynamického stylování“. Bylo potřeba nějak získat data ze „stylovacích“ modulů a vložit je jako hodnoty do stylů v kódu. V CMS jsem našel pro tyto účely třídy `Module` a `ModuleData`. Druhá třída vlastnila metodu `load()`, která v poli vracela všechny záznamy z modulu kromě záznamů, které jsou souborového typu (v našem případě například komponenta „Image Background“, která obsahuje obrázky). Nejdříve jsem si tedy získal všechna „nesouborová“ data. K tomu bylo v prvé řadě potřeba vytvořit objekt třídy `Module`, kterým se inicializoval objekt třídy `ModuleData`. Objekt `module` jsem inicializoval za pomocí „tagu“ (značky), který lze přiřadit v administraci k jednotlivým modulům (pozor, nejedná se o HTML element). Následně jsem na objekt `module` zavolal metodu `load()` (obě třídy mají své vlastní metody `load()`), která na základě „tagu“ načetla do objektu modulu komponenty. Podobně jsem vytvořil objekt `moduleData`, kterému jsem v konstruktoru předal instanci objektu `module`. Vzápětí jsem na objekt `moduleData` zavolal metodu `load()`, jejíž výstup jsem si uložil do proměnné. Takto jsem za pomocí dalších „tagů“ načetl data ze všech modulů, které definují styl stránky. Zbývalo vyřešit, jak získat z modulů soubory. Z databáze jsem zjistil, že pro všechny soubory v modulu se vytváří samostatná tabulka, jejíž název se skládá z názvu modulu a přípony `_file`. Po podrobnějším prozkoumání několika metod jsem zjistil, že metoda `load()` třídy `ModuleData` uloží souborové záznamy z této tabulky v komponentě do proměnné `files` určené právě pro soubory. Komponenty jsou pak uloženy v objektu `module`. Z modulu jsem si získal komponentu s názvem „Image Background“ a zavoláním metody `getUrl()` na soubor v komponentě jsem získal potřebnou cestu k obrázku v adresáři. Toto jsem opět aplikoval pro všechny moduly, ve kterých se nacházel nějaký soubor. Získaná data mi kvůli přehlednosti přišlo dobré roztřídit do asociativních polí, které jsem pojmenoval podle modulů. Data z modulů jsem nyní mohl nasázet do všech potřebných stylů. Zde bych se chtěl pozastavit a zmínit se o tom, jak jsem data z modulů formátoval.

Většinou komponenty sloužily pro definování barvy nějakého prvku, jako vstupní formát barvy jsem určil šestimístné hexadecimální číslo. Na internetu lze dnes najít spousty „color pickerů“, kde uživatel prakticky okamžitě získá pro svou zvolenou barvu toto hexadecimální číslo. Hodnotu jsem do stylů tedy jednoduše vkládal z modulů jako text. CSS umí pracovat i s formáty RGB a HLS, ale hexadecimální formát je dle mého názoru rozšířenější a pracuje se s ním poměrně pohodlně, proto jsem jej zvolil. Nevýhodou však bylo, že jsem jej nemohl použít pro transparentní barvy, které jsem potřeboval například u pozadí rozbalovacího menu. Tuto barvu lze získat právě metodou pro RGB formát nazvanou `rgba()`, ve které se zadávají barvy pomocí decimálních hodnot. Poslední parametr pak slouží pro naši požadovanou transparentnost. Hodnota transparentnosti se zapisuje hodnotou v intervalu od 0 do 1 s přesností na dvě desetinná místa, což značí, na kolik procent má být barva průhledná. Aby i v tomto případě pracoval uživatel se stejným formátem, vytvořil jsem dvě komponenty, jednu pro barvu v hexadecimálním tvaru, a druhou pro průhlednost. V kódu jsem barvu rozdělil do tří částí pro `r`, `g` a `b` parametr a následně jsem jej převedl

do desítkové soustavy. Transparentnost uživatel bude pro jednoduchost zadávat v procentech v intervalu od nuly do sta, v kódu jsem pak tuto hodnotu jednoduše vydělil stem. Všechny tyto hodnoty jsem nakonec sloučil a vložil do funkce `rgba()`. Tento převod jsem pak ještě aplikoval na jednom místě pro transparentnost textu překrývajícího posuvník.

```
    $res = $cssSelector . ' {
              color: #' . $moduleData['colorText_TextField'] . ';
              background-color: #' . $moduleData['colorBackground_TextField'] . ';
              background-image: url(' . $moduleData['imageBackground_FileRef'] . ');
              background-repeat: repeat;
        } '.
        $cssSelector . ' a{
              color: #' . $moduleData['colorTextLink_TextField'] . ';
        } ';
        if ($moduleData['underlineTxtLinkOver_Number']) {
              $res .= $cssSelector . ' a:focus {
                  text-decoration: none;
              }' .
              $cssSelector . 'a:hover {
                  text-decoration: none;
              }' .
              $cssSelector . 'a:active {
                  text-decoration: none;
              }';
        } else {
              $res .= $cssSelector . ' a:focus {
                  text-decoration: underline;
              }' .
              $cssSelector . 'a:hover {
                  text-decoration: underline;
              }' .
              $cssSelector . 'a:active {
                  text-decoration: underline;
              }';
        }
        return $res;
```

Výpis 2: Tělo funkce `generateStyleModuleCSS()`

Abych nemusel vypisovat v *rozvržení* několikrát stejný styl pro všechny záznamy z modulu „Style“, vytvořil jsem si funkci `generateStyleModuleCSS()`, které se předá CSS selektor a data pro konkrétní část. Ve výpisu 2 lze vidět, že funkce nejdřív nastaví barvu textu a pozadí. Pokud se má pozadí vyplnit dlaždicemi, načte se URL adresou obrázek. Za pomocí vlastnosti `background-repeat` se tento obrázek v pozadí zopakuje po celé ploše. Dále funkce nastaví barvu pro všechny odkazy. Na konec se otestuje hodnota pro to, zda mají být odkazy podtrženy.

5.2.3 Poznatky k projektu

Tento projekt představoval spíše takový experiment toho, co se dá v tomto firemním CMS Designeru vytvořit. Bylo jasné, že minimálně s první verzí budou pracovat pouze zaměstnanci firmy, a tak se zde neřešily například validace vstupů v katalogu textů nebo v modulech. Jejich požadovaný formát jsem vypsal do kolonek pro popis, které se nacházejí v CMS snad u všeho. Pokud by se ukázalo, že je tvorba stránek pomocí tohoto projektu efektivní, přistoupilo by se k návrhu další verze, kde by se tyto problémy odladily. V budoucnu by se mohla k tomuto projektu vytvořit třeba uživatelská příručka a samotný vzhled by si mohli upravovat i koncoví zákazníci.

Přínos projektu Uniweb hodnotím kladně, práce na něm mě velmi bavila. Dozvěděl jsem se spoustu nových věcí, a zase jsem se dostal o něco dál v této oblasti. Projekt mě donutil projít firemní CMS Designer více do hloubky, a tak jsem do větší míry pochopil, jak je tento systém naprogramován a jak se s ním pracuje. Jediné, co mi při práci chybělo, byla právě nějaká podrobnější dokumentace. Od kolegy jsem dostal pouze dokument obsahující stručný návod, jak se řeší ty nejčastější problémy, čemuž se zrovna požadavky v mém projektu vyhýbaly. Většinou však byli po ruce zkušenější kolegové, kteří mi neváhali s čímkoliv pomoci.

5.3 HPLM (Hybrid Product Lifecycle Management)

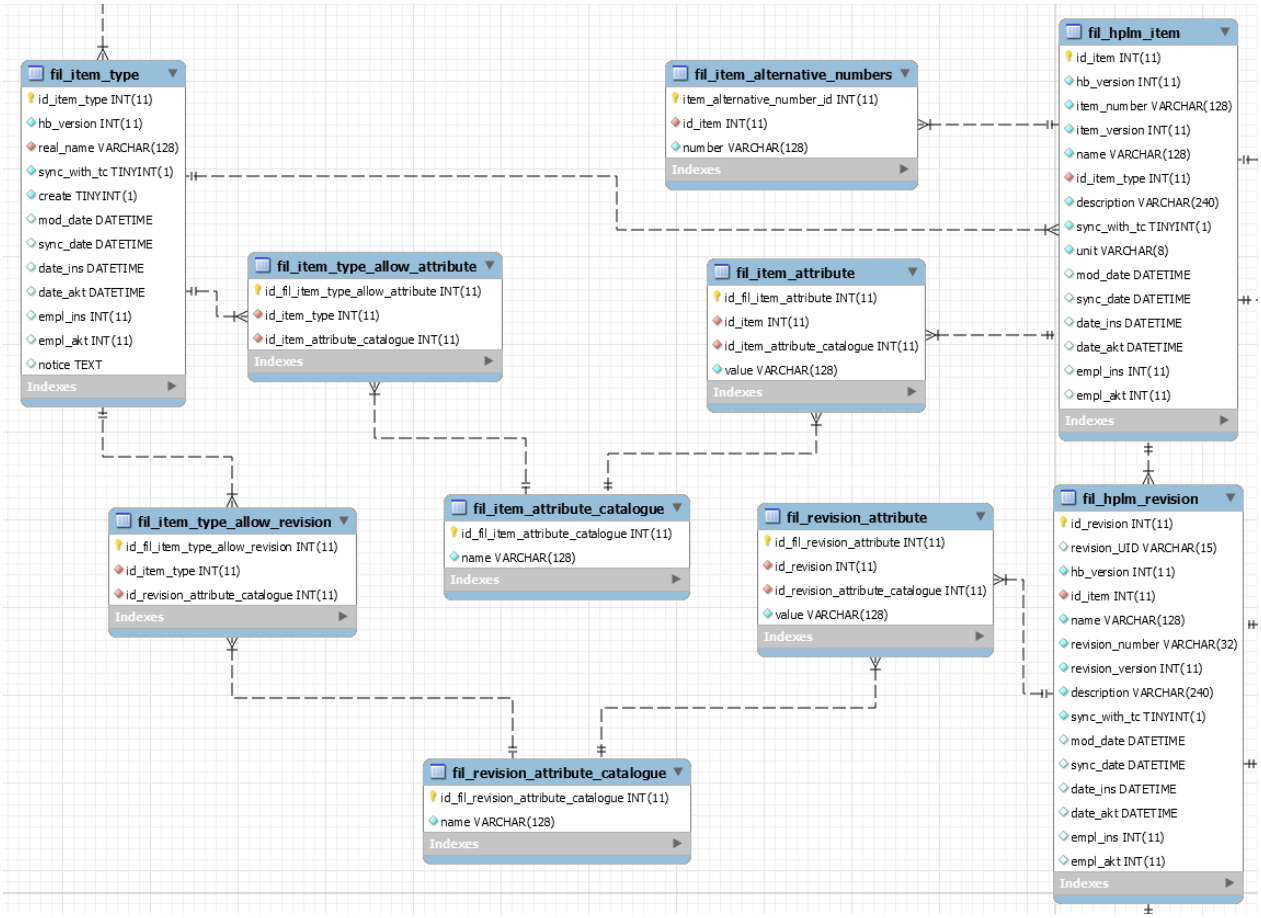
Na úvod jsem byl seznámen s dalším firemním rámcem. Kolega mi předal projekt s aktuální verzí systému a provedl mě instalací, k tomu jsem dostal ještě soubor, kde se nacházel návrh celé databáze. Byly mi stručně podány základní informace potřebné k práci na mém zadání a vysvětleny všechny rutinní povinnosti a pravidla, které je v rámci potřeba dodržovat. Tento firemní rámec byl nový a nebyla k němu vytvořena ještě žádná dokumentace. Z toho důvodu jsem byl odkázán na již funkční části systému, které tvořili kolegové, kde jsem mohl hledat inspiraci. Samozřejmě na cokoliv jsem se mohl doptávat kolegy, pokud byl ve firmě přítomen. Kolega mi rovněž doporučil, abych k práci s databází vyzkoušel program HeidiSQL, jelikož v tomto projektu budu pracovat s databází mnohem častěji a práce v prostředí phpMyAdmin je znatelně pomalejší. Tento program jsem si velmi oblíbil a určitě jej budu používat i k jiným projektům. Na rozdíl od předchozích projektů se zde nebudu věnovat „front-endu“, jelikož pro celý projekt jsou už předem nadefinované styly všech prvků a komponent, které budu v projektu vytvářet.

5.3.1 Analýza a návrh

Na začátku jsem si prošel celý projekt, zjistil jsem, v jakém stavu se nachází databáze a co vše je kolem položky v systému funkční. V databázi už byly potřebné tabulky vytvořeny, šlo však vidět, že od začátku vývoje systému s nimi nikdo nemanipuloval, a tak se v nich nacházely ještě atributy s neaktuální konvencí pojmenování a některé atributy v tabulkách chyběly, o tom ale až dále. Jak byly tyto tabulky s vazbami navrženy, můžeme vidět v diagramu na obrázku 1 (obrázek popisuje už mnou upravenou verzi). Tabulky jsem tedy nemusel navrhovat, stačilo je pouze upravit a zaktualizovat. V diagramu můžeme vidět, že pro atributy jsou vyhrazeny tři tabulky. Samotný atribut je rozdělen do dvou tabulek. Tabulka `fil_item_attribute_catalogue` drží informace o jméně atributu, hodnota tohoto atributu je pak uložena v tabulce `fil_item_attribute`, která zároveň slouží jako spojovací tabulka mezi katalogem a tabulkou pro položku. Třetí tabulka je spojovací tabulkou mezi katalogem a tabulkou pro typ položky, v té se tedy přiřazují atributy k danému typu položky. Typ položky a položka je rovněž propojena vazbou „1:N“, abychom mohli položce přiřadit typ. Co se týče tabulek, v jejichž názvu je „revision“, tak se principiálně jedná o totéž, jen pro jinou tabulku (resp. pro revizi položky). Na revizích a tabulce `fil_tem_alternative_numbers` jsem však pracoval pouze okrajově.

V dalším kroku jsem se přesunul do rámce, zde jsem zjistil, že pro atributy ani revize nebylo vytvořeno vůbec nic, nacházely se zde pouze rozpracované třídy pro typ položky a položku. Bylo tedy jasné, že abych mohl dynamické atributy implementovat v položce, musel jsem nejdřív zprovoznit možnost spravovat atributy. Nejprve mě napadlo, že by se atributy mohly vytvářet přímo v detailu typu položky, od této myšlenky jsem však rychle odstoupil, jelikož se jednalo o vazbu „M:N“. Mnohem lepší bylo pro atributy vytvořit samostatnou podstránku, kde by se nacházel seznam všech atributů a zde by se atributy spravovaly, V detailu typu položky by se tedy atributy nevytvářely, ale pouze pomocí komponenty typu „Selectbox“ (dále jen „selectbox“) přiřazovaly už existující. V tomto detailu by se nacházel seznam přiřazených atributů.

Podstránku pro atributy jsem pak zařadil jako položku v menu s názvem číselníky, kde se již nachází například podstránka pro typ položky.



Obrázek 1: HPLM – Část návrhu databáze

Zbývá zanalyzovat chování atributů v detailu položky. Podobně jako se přiřazuje atribut k typu položky, jsem se rozhodl přiřazovat i typ položky k položce, tedy za pomocí selectboxu, který bude obsahovat výpis všech vytvořených typů. Po vybrání typu položky se všechny „dynamické“ atributy vygenerují pod poslední statický atribut položky. Zde jsem nesměl zapomenout na to, že pokud se zvolí nový typ položky například při editaci položky a už existují u položky nějaké hodnoty ze starého typu položky, tak je nutné uživateli dát nějak najevo, že se při zvolení nového typu všechny staré hodnoty smažou. Podobná varování budou muset existovat i v případě, že odstraníme z číselníku nějaký atribut nebo typ položky.

5.3.2 Implementace

Zprvu jsem se vrhl na úpravu databázových tabulek, zde nebylo co řešit, většinou jsem prováděl úpravu názvů nebo datových typů. Veškeré provedené úpravy pak bylo pro kolegy potřeba nahrát do souboru v projektu, kde se nacházely všechny DDL příkazy, které byly nad databází provedeny.

S funkční databází jsem mohl vrhnout na implementaci chybějících tříd. Pro každou tabulku jsem tvořil vždy dvě třídy, jedna reprezentuje tabulku, druhá pak slouží pro logiku a reprezentaci podstránky. Všechny vytvořené třídy, je nutné vždy zapsat do databáze rámce, aby s nimi mohl pracovat. Další podobnou povinností, je vkládat do databáze i obě jazykové mutace všech prvků (nejčastěji text v komponentě typu „Label“ nebo text související s validací vstupů), které mají být překládány.

Než se dostanu dále, musím zde připomenout, že revizím a ostatním tabulkám, které nesouvisely s atributy, jsem se v rámci mého zadání věnoval jen okrajově. Má práce s nimi skončila u tohoto „namapování“.

Dalším krokem bylo vytvořit podstránku pro atributy. Podstránka se skládala ze seznamu vytvořených atributů, tlačítka pro vložení nového záznamu, který odkazuje na jednoduchý formulář a filtru. Přišlo mi užitečné atributy filtrovat na základě typu položky. Ve filtru jsem tedy vytvořil selectbox, ve kterém si uživatel vybral existující typy položek. Pro naplnění selectboxu bylo nutné sestavit vlastní dotaz, který do něj načítal záznamy z typu položky. Kliknutím na atribut v seznamu se uživatel dostane na jeho detail, kde jej může spravovat.

Při tvorbě této podstránky jsem nenarazil na nic těžkého, podobné věci se už v projektu řešily na mnoha místech, a tak stačilo prostudovat kódy od kolegů. Implementace poté už nezabrala mnoho času. Hotovou podstránku jsem vložil jako položku do menu v sekci „Číselníky“.

Po přesunutí do podstránky pro typ položky, jsem v detailu vytvořil pro atributy sekci, ve které se nacházely všechny přiřazené atributy, tyto záznamy se daly přepínáním tlačítek zobrazit buď v tabulce, nebo v mřížce. Tato volba se u přiřazených záznamů nacházela i v jiných, už hotových sekcích, proto jsem se to rozhodl respektovat a možnost zobrazit záznamy v mřížce jsem použil i v mém řešení. V sekci se nacházelo samozřejmě tlačítko pro přiřazení nového atributu, který odkazoval na formulář. V něm figuroval selectbox, který jsem popisoval v analýze. Na rozdíl od implementace podstránky pro atributy, jsem zde pracoval i s třídami, které pracovaly nad spojovací tabulkou. V těchto třídách jsem definoval například vzhled tabulky v sekci nebo zmíněný selectbox. Při řešení jsem opět nenarazil na žádný velký problém, omezovala mě pouze má neznalost rámce a doba jeho učení. Řešení bylo spíše pracné a kvůli absence dokumentace zdlouhavé. I pro toto řešení jsem mohl v projektu najít inspiraci v implementaci mých kolegů.

S hotovým základem pro atributy jsem mohl vyřešit generování atributů v položce. Každý atribut v položce pracoval s dvěma tabulkami. Název atributu se načítal z tabulky `fil_item_attribute_catalogue` a hodnota se pak načítala (případně ukládala) z tabulky `fil_item_attribute`. Nejprve jsem si získal z databáze všechny názvy atributů pro tuto položku a ty jsem poté vkládal do labelu pro název. V dalším kroku jsem si získal všechny hodnoty z tabulky `fil_item_attribute` (pokud nějaké existovaly) a vložil je do textového pole, ke svému příslušnému názvu atributu. Tímto bylo vyřešeno pouze generování atributů a načítání jejich hodnot, zbývalo

naimplementovat ukládání, editaci a mazání. Pro tyto účely jsem si vytvořil v položce metodu, `saveAttributes()`. V metodě jsem musel nejdříve zjistit, jestli byl v položce změněn typ položky a pokud ano, musel jsem smazat všechny staré záznamy hodnot atributů z tabulky `fil_item_attribute` a uložit zde hodnoty nové. V druhém případě (kdy se editovaly hodnoty atributů beze změny typu položky) se hodnoty pouze přepsaly. Tuto metodu jsem následně použil v metodě `afterSaveAction()`, která se volala po zmáčknutí tlačítka pro uložení.

To bylo vše ke generování, zbývalo ještě zařídit, aby se záznamy ze spojovacích tabulek `fil_item_attribute_catalogue` a `fil_item_attribute` mazaly vždy při smazání některého z nadřazených záznamů. K tomu stačilo upravit tyto tabulky v databázi pomocí kaskádového mazání. Na závěr bylo potřeba uživatele vždy při změně nebo smazání nějakého záznamu (který měl vliv na hodnoty atributu) nějak upozornit. Jelikož o tom, jak bude upozornění vypadat, neměl nikdo z vedení zatím představu, tak jsem použil dočasné řešení v podobě jednoduchých dialogových oken (tzv. Message boxů).

Po dokončení implementace jsem projekt předal kolegovi, který řešení zkontroloval a nahrál za pomocí systému pro správu verzí CVS, který firma využívá pro většinu svých projektů.

5.3.3 Poznatky k projektu

V tomto projektu jsem se seznámil s dalším firemním rámcem, pomocí kterého firma vyvíjí rozsáhlejší systémy. V tomto případě jsem se zaměřil mnohem více na práci s jazykem PHP a vzhledové části jsem se téměř vůbec nevěnoval. Nad řešením jsem strávil více času, než jsem původně odhadoval, jelikož rámci chyběla dokumentace. Což pro někoho, kdo s podobnými rámci nemá žádné zkušenosti, není zrovna ideální. Zde jsem musel procházet kvanta kódu, abych spoustu věcí pochopil, ale to k práci programátora patří.

Přínos projektu Tento projekt mi umožnil pracovat na něčem, na čem nepracuji jen já, ale celý tým. Zde jsem získal nejvíce zkušeností s jazykem PHP. Naučil jsem se pracovat s novým rámcem a měl jsem možnost prozkoumat rozsáhly projekt.

6 Závěr

6.1 Uplatnění teoretických a praktických znalostí a dovedností získaných během studia

Za celou dobu studia jsem měl možnost načerpat mnoho znalostí a dovedností, bez kterých bych si při vykonávání odborné praxe nedokázal poradit. Jelikož jsem se v minulosti webovým technologiím nevěnoval, musel jsem se opírat právě o předměty, které se v rámci mého studijního programu vyučují.

Nejvíce jsem zužitkoval zkušenosti z předmětů, které úzce souvisejí se zaměřením mé odborné praxe. Mezi tyto předměty patří *Vývoj informačních systémů*, *Databázové a informační systémy* a *Úvod do databázových systémů*, které mě dobře připravily na práci s databázemi a vývojem informačních systému v celém jeho spektru.

Dále bych chtěl zmínit přínos předmětu *Tvorba aplikací pro mobilní zařízení I*, který se sice zabývá vývojem mobilních aplikací, ale představuje stejné technologie, které se využívají při tvorbě webových stránek a systémů. *Programování I* a *II* mi představily základní programovací paradigmata a poskytly základní znalosti v oblasti objektově orientovaného programování. Předměty *Programovací jazyky I* a *II* pak tyto znalosti ještě více prohloubily. Nesmím zapomenout ani na předměty *Algoritmy I* a *II*, které mají v této oblasti rovněž velký přínos. Díky těmto předmětům, bylo seznámení s pro mě novým jazykem PHP mnohem rychlejší a jednodušší. Vyzdvihnout musím také předmět *Uživatelská rozhraní*, jenž studenty seznamuje s principy a správnými postupy při návrhu uživatelského rozhraní.

Na konec je třeba dodat, že ruku k dílu přidaly i předměty, které se zaměřením mé praxe přímo nesouvisejí.

6.2 Znalosti a dovednosti scházející v průběhu vykonávání odborné praxe

Chybějících znalostí a dovedností, které by výrazně ulehčily průběh mé praxe, je poměrně mnoho. Hlavním důvodem jejich absence je to, že vývoj webových prezentací a aplikací nepatří mezi mé hlavní zájmy. Volbu předmětů v mém studiu jsem směroval spíše k vývoji mobilních aplikací a počítačové grafice. V tomto směru jsem však pro vykonání praxe nenalezl vhodnou firmu, a tak jsem zvolil firmu se zaměřením na webový vývoj. V této oblasti jsem neměl téměř žádné zkušenosti a přišlo mi dobré tuto mezeru zaplnit právě prostřednictvím praxe.

V průběhu praxe mi nejvíce chyběly zkušenosti s jazykem PHP a hlubší znalosti HTML, CSS, JavaScriptu a Bootstrapu. Rovněž by bylo užitečné, kdybych měl alespoň všeobecný přehled o redakčních systémech. S firemními nástroji jsem se sice seznámit před nástupem do firmy nemohl, avšak kdybych měl zkušenosti s některými známými PHP rámci, jako je třeba Laravel, Nette apod., tak by byla práce s těmi firemními mnohem jednodušší.

6.3 Přínos a celkové zhodnocení odborné praxe

Během mé odborné praxe ve firmě ISSA CZECH s. r. o. jsem získal mnoho cenných zkušeností a nových znalostí. Praxe mi umožnila prozkoumat další oblast informačních technologií, ve které jsem neměl téměř žádné zkušenosti. Prohloubil jsem své znalosti získané během studia a osvojil jsem si práci s novými jazyky a technologiemi. Zjistil jsem, jak probíhá vývoj velkých projektů a poznal jsem mnoho nových nástrojů, které budu používat i nadále.

Dokážu si představit, že získané znalosti v budoucnu využiji třeba pro mé osobní projekty (např. pro webovou prezentaci k nějaké aplikaci nebo portfolio), a tak nebudu muset plýtvat prostředky, s čímž jsem před absolvováním praxe nepočítal.

Další velkou výhodou je, že mohu nabyté zkušenosti uplatnit také při hledání mého budoucího zaměstnání. Praxi tudíž z výše uvedených důvodu nemůžu hodnotit jinak než kladně.

Na závěr bych chtěl nesmírně poděkovat panu Ing. Michalu Kolesárovi, Ph.D. a panu Ing. Davidu Ježkovi, Ph.D., že mi věnovali svůj čas a energii a měli se mnou trpělivost i ve špatných chvílích. Také všem, kteří mě při studiu podporovali.

7 Literatura

[1] *Internet Software & Security Advice.* [Online] ISSA, c2013. [Citace: 10. 4 2018.] https://www.issa.cz/index.html?lang=cz.

[2] *Tvorba webových stránek.* [Online] ISSA, c2013. [Citace: 10. 4 2018.] https://www.issa.cz/tvorba-webovych-stranek-637.html.

[3] *Informační systémy.* [Online] ISSA, c2013. [Citace: 10. 4 2018.] https://www.issa.cz/informacni-systemy-626.html.

[4] *E-shopy.* [Online] ISSA, c2013. [Citace: 10. 4 2018.] https://www.issa.cz/e-shopy-624.html.

[5] *Internetový marketing.* [Online] ISSA, c2013. [Citace: 10. 4 2018.] https://www.issa.cz/internetovy-marketing-622.html.

[6] *Webdesign.* [Online] ISSA, c2013. [Citace: 10. 4 2018.] https://www.issa.cz/webdesign-11.html.

[7] *SEO optimalizace pro vyhledávače.* [Online] ISSA, c2013. [Citace: 10. 4 2018.] https://www.issa.cz/seo-optimalizace-pro-vyhledavace-6.html.

[8] *Datové centrum.* [Online] ISSA, c2013. [Citace: 10. 4 2018.] https://www.issa.cz/datove-centrum-625.html.

[9] *Kaskádové styly.* [Online] Adaptic.cz, c2005-2018. [Citace: 10. 4 2018.] http://www.adaptic.cz/znalosti/slovnicek/kaskadove-styly/.

[10] *HTML.* [Online] ComputerHope.com, 2017. [Citace: 10. 4 2018.] https://www.computerhope.com/jargon/h/html.htm.

[11] *1. díl - Úvod do JavaScriptu.* [Online] ITnewtwork.cz, c2018. [Citace: 10. 4 2018.] https://www.itnetwork.cz/javascript/zaklady/javascript-tutorial-uvod-do-javascriptu-nepochopeny-jazyk.

[12] *What Is jQuery?* [Online] Elated.com, c1996-2018. [Citace: 10. 4 2018.] https://www.elated.com/articles/what-is-jquery/.

[13] STEPHENS, Ryan K., Ronald R. PLEW a Arie JONES. *Naučte se SQL za 28 dní: [stačí hodina denně].* Brno : Computer Press, 2010. ISBN 978-80-251-2700-1.

[14] *PHP /základy/.* [Online] Tvorba-webu.cz, c2003-2008. [Citace: 10. 4 2018.] https://www.tvorba-webu.cz/php/.

[15] Michálek, Martin. *K čemu je dobrý Bootstrap a frontend frameworky?* [Online] Zdroják.cz, 2013. [Citace: 10. 4 2018.] https://www.zdrojak.cz/clanky/k-cemu-je-dobry-bootstrap-frontend-frameworky/.