Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Bakalářská práce
Porovnání PHP frameworků Nette a Laravel

Lukáš Oplt

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Lukáš Oplt

Informatika

Název práce

Porovnání PHP frameworků Nette a Laravel

Název anglicky

Comparison of PHP frameworks Nette and Laravel

Cíle práce

Hlavním cílem práce je porovnání PHP frameworků Nette a Laravel.
Dílčím cíle jsou:
- charakterizovat technologie a nástroje pro tvorbu webových aplikací
- charakterizovat framework Laravel
- charakterizovat framework Nette

Metodika

Metodika práce je založena na studiu a analýze odborných informačních zdrojů. Pro podporu porovnání budou vytvořeny dvě stejné webové aplikace demonstrující získané znalosti, jedna s použitím frameworku Nette, druhá s použitím frameworku Laravel. Poté bude provedeno porovnání těchto aplikací pomocí vhodných metod a parametrů. Na základě syntézy teoretických poznatků a výsledků praktické části budou formulovány závěry bakalářské práce.

Doporučený rozsah práce

45 stran

Klíčová slova

webová aplikace, PHP, HTML, CSS, SQL

Doporučené zdroje informací

HTML 5 Okamžitě, Tiffany B. Brown, Kerry Butters, Sandeep Panda, Computer Press, ISBN 978-80-251-4296-7
Mistrovství – PHP a MySQL Luke Welling, Laura Thomson, Computer Press 2017, ISBN 978-80-251-4892-1
MySQL Okamžitě, Timothy Boronczyk, Computer Press, ISBN 978-80-251-4737-5
PHP okamžitě, Callum Hopkins,Computer Press 2014 ISBN 978-80-251-4196-0

Předběžný termín obhajoby

2018/19 LS – PEF

Vedoucí práce

Ing. Jan Masner, Ph.D.

Garantující pracoviště

Katedra informačních technologií

Elektronicky schváleno dne 18. 9. 2018

Ing. Jiří Vaněk, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 19. 10. 2018

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 11. 03. 2019

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Porovnání PHP frameworků Nette a Laravel" jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 14.3.2019 _____

Poděkování

Rád bych touto cestou poděkoval vedoucímu této práce, Ing. Janu Masnerovi, Ph.D. za cenné a užitečné rady při psaní této práce. Dále bych rád poděkoval rodině, přítelkyni a přátelům, kteří při mně stáli po celou dobu studia.

Porovnání PHP frameworků Nette a Laravel

Abstrakt

Bakalářská práce „Porovnání PHP frameworků Nette a Laravel“ se zabývá porovnáním a popsáním rozdílů těchto dvou frameworků. Framework Nette byl vybrán jakožto framework pocházející z České republiky a také v České republice nejpoužívanější framework, zatímco framework Laravel jakožto nejpoužívanější framework pro PHP celosvětově.

V teoretické části se tato bakalářské práce zabývá popsáním technologií pro tvorbu webových aplikací, a to z pohledu Client side (na straně uživatele) a z pohledu Server side (z pohledu serveru). Mezi popsanými Client side technologiemi jsou HTML a CSS, popsanými Server side technologiemi jsou PHP a databázové dotazovací jazyky NoSQL a SQL.

V praktické části mé práce porovnávám dva frameworky, Nette a Laravel. Porovnávám je v oblastech rychlosti, bezpečnosti, podpory Dependency injection, počtu řádků aplikace, popularitě frameworku, což jsem měřil dle výskytu ve fórech a také dle uplatnění na trhu práce. Dále jsem frameworky porovnal i dle zdrojů, odkud je možné se daný framework naučit a dle rychlosti implementace nových technologií.

Klíčová slova: webová aplikace, PHP, HTML, CSS, SQL, Nette, Laravel

Comparison of PHP frameworks Nette and Laravel

Abstract

The bachelor thesis “Comparison of PHP frameworks Nette and Laravel” deals with comparison and description of the differences between these two frameworks. The Nette framework was chosen as a framework originating from Czech Republic and the most widely used framework in Czech Republic, while the Laravel framework is the most widely used framework for PHP worldwide.

In the theoretical part, this bachelor thesis deals with describing technologies for creating web applications, from the side of the Client side and from the Server side. Among the Client side technologies are described the HTML and CSS, among the Server side technologies are described PHP and the database query languages NoSQL and SQL.

In the practical part of my work, I compare two frameworks, Nette and Laravel. I compare them in the areas of speed, security, Dependency injection support, application line counts, popularity of the framework, which I measured according to the occurrence in the forums and according to the application on the labor market. In addition, I compared the frameworks by sources, where the framework can be learned and the speed of implementation of new technologies.

Keywords: web application, PHP, HTML, CSS, SQL, Nette, Laravel

Obsah

1. Úvod

V dnešní době, kdy jde vývoj softwaru stále dopředu, jsou stále více využívány webové aplikace. Nejvíce webových aplikací využívá programovací jazyk PHP. Tento programovací jazyk nabízí několik frameworků neboli softwarových základů, které pomáhají s rychlejším vývojem aplikací. Frameworky také pomáhají vytvářet aplikace bezpečnější a usnadňují porozumění kódu od různých vývojářů.

Frameworků PHP existuje nepřeberné množství, nicméně z hlediska českého prostředí vyniká především jeden framework. Je jím Nette framework, který je nejpoužívanějším PHP frameworkem na území České republiky a taktéž z České republiky pochází jeho autor David Grudl. Pozici nejpoužívanějšího PHP frameworku celosvětově oproti tomu má framework Laravel. Ačkoli oba tyto frameworky mají jistá specifika otázkou zůstává, v kterých parametrech a jak výrazně se od sebe tyto frameworky odlišují a zdali je některý framework výhodnější za určité situace.

Srovnání frameworků je jednou z aktuálně diskutovaných problematik v rámci programování v jazyce PHP. Intenzivně je srovnávána nejen rychlost či bezpečnost daných frameworků, ale i podpora Dependency injection, počet řádků kódu nebo počet výskytů na fórech. Neméně jsou pak porovnávány možnosti uplatnění na trhu práce, možnosti zdrojů učení a rychlost implementace nových technologií do frameworků. Na základě výsledků porovnání jednotlivých frameworků v těchto parametrech lze tedy vybrat vhodný PHP framework pro dané prostředí, podmínky či situace bez pozdějších problémů s přepisováním kódu do frameworku vhodnějšího pro aktuální potřebu.

2. Cíl práce a metodika

2.1 Cíl práce

Hlavním cílem této bakalářské práce je porovnání frameworků Nette a Laravel. Porovnány budou na základě testování rychlosti, bezpečnosti, podpoře Dependency injection, počtu řádků aplikace, výskytu ve fórech, uplatnění na trhu práce, dostupných zdrojů učení a rychlosti implementace nových technologií.

Dílčími cíli této práce jsou charakterizování frameworků Nette a Laravel. Dále charakterizování technologií pro tvorbu webových aplikací, a to z pohledu Client side technologií a Server side technologií.

2.2 Metodika

Metodika práce je založena na studiu a analýze odborných informačních zdrojů. Na základě syntézy teoretických poznatků, získaných při vypracování teoretické části, bude vypracovaná praktická část.

Pro podporu porovnání byly vytvořeny dvě stejné webové aplikace demonstrující získané znalosti, jedna s použitím frameworku Nette, druhá s použitím frameworku Laravel. Obě aplikace měly vlastní databázi a tyto databáze byly shodné svou struktury i data.

Pro testování rychlosti PHP aplikací bude použit volně dostupný software Apache JMeter, pro testování bezpečnosti bude využito ruční vkládání kódu do aplikace. Porovnání na základě Dependency injection bude provedeno na základě syntézy teoretických poznatků, stejně tak jako u porovnání zdrojů učení a rychlosti implementace nových technologií. Porovnání počtu řádků aplikace bude prováděno ručně, po sečtení řádků ze zdrojových kódů. Pro porovnání výskytu na fórech a uplatnění na trhu práce byly použity webové portály sloužící k dané problematice.

Následně byly hodnoceny frameworky vůči sobě, kdy nejlepší hodnocení z dané oblasti bude 100 % a hůře umístěný framework dostane odpovídající množství procent dle svých výsledků, pokud není u testu uvedeno jinak. Dle těchto výsledků praktické části byly formulovány závěry bakalářské práce.

3. Teoretická část

Teoretická část bakalářské práce je věnována teorii tvorby webových aplikací a definici jednotlivých frameworků. V této části jsou popsány technologie pro tvorbu webové aplikace, jmenovitě HTML a CSS, SQL, PHP. Dále jsou popsány oba frameworky, Nette i Laravel.

3.1 Client side technologie

V informačních technologiích dělíme technologie na ty, které se vykonávají na straně klienta a na ty, které se vykonávají na straně serveru. Technologie, které se vykonávají na straně klienta nazýváme Client side technologie. V oblasti webového vývoje se dají technologie Client side definovat jako technologie, které zajišťují zobrazení na monitoru, ať se jedná o text, obrázky či jakékoli další části uživatelského rozhraní. (1)

3.1.1 HTML

HTML (celým názvem HyperText Markup Language) je značkovací jazyk využívaný pro tvorbu webových stránek. HTML je hlavním z jazyků pro vytváření stránek v systému World Wide Web, který umožňuje publikaci dokumentů na internetu. Jazyk HTML definuje parsovací algoritmus pro generování DOM (objektový model dokumentu). HTML využívá zápis pomocí elementů a tyto elementy jsou reprezentovány tagy. Tento zápis je převzat z SGML (Standard Generalized Markup Language) a postupem času byl upravován v závislosti na aktuálních trendech. Názvy jednotlivých tagů a jejich vlastností jsou uzavřeny uvnitř hranatých závorek „<“ „>“. Dále dělíme tagy na párové a nepárové. Většina tagů je párová, tzn. Má počáteční a koncovou značku, uvnitř je uložen nějaký obsah. (2) Například:

* Nějaký text *

Opak párového tagu je tag nepárový. Jedná se o tagy, které nemají obsah a tudíž nepotřebují koncovou značku. Příkladem může být odřádkování:

K tagům lze přidávat i jejich vlastnosti. Vlastnosti jsou vepsány do počáteční značky a uvádějí učitou informaci k danému tagu. Například tag *<a>*, který slouží pro odkaz, obsahuje vlastnost *href*, která nám udává, kam daný odkaz odkazuje. Příkladem může být odkaz, který bude odkazovat na webové stránky www.seznam.cz (2):

* Text odkazu*

V rámci HTML dokumentu lze též využívat komentáře, které jsou viditelné pouze ve zdrojovém kódu, tudíž nejsou zobrazeny v obsahu dané stránky. Komentáře slouží lidem pracující na dané stránce pro lepší orientaci či dopsání si nějakých poznámek. Komentáře jsou zapsány tímto způsobem:

<!-- text komentáře -->

HTML má též pevnou strukturu, tato struktura se skládá z kořenového elementu (tag *html*), hlavičky dokumentu (tag *head*) a těla dokumentu (tag *body*). V kořenovém elementu je uloženo vše, co se týká dané stránky. V hlavičce dokumentu jsou uloženy metadata pro daný dokument a tělu dokumentu je uveden obsah dokumentu. Velmi jednoduchá struktura poté může vypadat takto:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Titulní název stránky</title>

</head>

<body>

* Toto je ukázková struktura html *

</body>

<html>

Značky můžeme též dle významu dělit na strukturální (rozvrhují strukturu dokumentu), popisné, též sématické, značky (popisují povahu obsahu daného prvku) a stylistické značky (určují vzhled prvku při zobrazení, avšak od těchto značek se postupně upouští a vzhled se určuje pomocí CSS stylů). (3)

3.1.2 CSS

Kaskádové styly (anglicky Cascading Style Sheet, zkratkou CSS) jsou informace pro způsob zobrazení dokumentů v značkovacích jazycích jako jsou HTML, XML či XHTML. (3)

Kaskádové styly jsou jednoduchým mechanismem pro přidání stylů (např. barev, fontů) do webových dokumentů. (4) Definice kaskádových stylů je složena ze selektoru a bloku deklarací. Selektor ukládá, na jaký element se vztahuje daná úprava zobrazení. V bloku deklarací jsou poté nastaveny hodnoty pro jednotlivé chtěné úpravy. Jednoduchým příkladem syntaxe může být nastavení pozadí dokumentu na modrou barvu:

```
body{
    background-color: blue;
}
```

3.1.2.1 Selektory

CSS má mnoho různých selektorů, které je navíc možné i kombinovat, pro případné vnoření. Deklarovat CSS úpravu můžeme pro element (například *p*), což poté provede danou úpravu u všech výskytů daného elementu. Dále můžeme provést deklaraci pro vnořený element, kde rozlišujeme dva typy. Prvním typem je označení elementu vnořeného na jakékoli úrovni (př. *body p*), druhým typem je označení elementu, který je přímým potomkem (př. *body>p*). Pro upřesnění mějme tento příklad v HTML dokumentu:

```
<body>
    <div>
        <p> Tento text chci vizuálně upravit</p>
    </div>
</body>
```

Pokud bychom měli tento případ, tak selektor „*body>p“* daný text nijak neovlivní, neb element <*p*> je vnořen ještě do elementu <*div*>, zatímco selektor *„body p“* by daný text již ovlivnil.

Za selektor nemusíme vždy vybírat všechny html elementy, můžeme vybrat i elementy s určitým atributem class (selektor .) či s nastaveným identifikátor id (selektor #). Příklad:

.class{...} či *#id{...}*

Posledním mnou zmíněným způsobem selektoru jsou selektory reagující na určité akce (např. navštívené odkazy, přejetí myší, …), jedná se o selektory *:visited*, *:hover.* (4)

3.1.2.2 Připojení CSS do HTML dokumentu

Existuje několik způsobů, jak k HTML dokumentu přiřadit CSS, avšak nejčastěji se setkáme s použitím odkazu na externí soubor. Připojení CSS souboru k HTML dokumentu pomocí elementu <*link*> poté v HTML dokumentu vypadá následovně:

```
<head>
    <link rel=´stylesheet´ href=´styly.css´ type=´text/css´>
</head>
```

Toto však není jediný možný způsob propojení HTML dokumentu s CSS. K propojení můžeme též využít elementu <*style*>, který se vkládá přímo do HTML dokumentu, a tudíž nepotřebuje samostatný CSS soubor.

Dalším možným způsobem propojení je tzv. inline zápis pomocí atributu *style*. Dané změny se poté aplikují pouze na daný element, u kterého se nachází tento atribut. (9) Kód poté může vypadat následovně:

```
<p style="color: blue"> Tento odstavec bude modrý </p>
```

3.2 Server side technologie

Technologie Server side se na rozdíl od Client side technologií provádí na serveru. V minulosti se všechny logické úkoly zpracovávaly na serveru. Server vypočte data, která až následně posílá na zařízení, kde jsou data pomocí Client side technologií zobrazena. Mezi Server side technologie patří vytváření dynamických stránek, komunikace s databází, autorizace uživatelů a notifikace uživatelům. (1)

3.2.1 Databáze

Databáze jsou systémem pro ukládání a zpracování dat. Databáze obsahují data a jejich vztahy. V databázi můžeme data třídit a řadit dle různých požadavků. Přirovnáním k databázi může být kartotéka u lékaře, kde jsou záznamy pacientů. (5)

Databáze jsou dnes používány pro uložení informací a následnou práci s nimi. Příkladem může být e-shop, kde produkty jsou uloženy v databázi jako jednotlivé záznamy.

Výhodou databází oproti uložení do souborů je jejich rychlost zpracování a optimalizace pro přístup více uživatelů. (5)

Dnes využíváme tzv. SQL a NoSQL databáze. SQL databází rozumíme databáze, nad kterými je možné využít SQL dotazování. Tyto databáze využívají tabulek, ve kterých jsou uložena data. Jednotlivé tabulky jsou mezi sebou navzájem propojeny tzv. vztahy. Zatímco NoSQL databáze nevyužívají ukládání do tabulek a nepodporují SQL dotazování. Doménou NoSQL databází je vysoká optimalizace pro vyhledávání, avšak nabízí jen nízkou funkcionalitu, což často bývá jen prosté ukládání dat. (6)

3.2.1.1 NoSQL

NoSQL (Non SQL) databáze nejdou cestou tabulkového rozdělení. Namísto toho jdou spíše cestou stromových či grafických struktur. NoSQL se hodí k ukládání velkých objemů dat, kde není zapotřebí uchovávat vzájemné vztahy. NoSQL nenabízí u transakcí plnou podporu ACID, což umožňuje lepší škálovatelnost. NoSQL nabízí distribuovanou architekturu odolnou vůči chybám. Je to výsledkem uložení dat na několika rozdílných serverech, což nabízí přípustnost výpadku serveru. (6)

Výhody NoSQL:

- Velmi rychlé čtení/zápis z/do databáze
- Snadné přidávání nových atributů
- Dobrá škálovatelnost databáze (7)

Nevýhody NoSQL:

- Integrita dat se musí hlídat pomocí aplikace, která s danou databází komunikuje
- Velká redundance dat
- Pro úpravu dat se prochází všechny záznamy (7)

3.2.1.2 SQL

SQL (Structured Query Language) je dotazovací jazyk, který se používá pro práci s daty v relačních databázích. Jedná se o standardní jazyk pro přístup k systémům pro správu relačních databází. Pomocí SQL můžeme ukládat i načítat data z databáze. (8)

Jedná se o jeden z nejpoužívanějších jazyků pro práci s daty, čemuž vděčí i relativně jednoduché syntaxi. Jednoduchý příklad pro vypsání všech dat z databáze zákazníků:

*SELECT * FROM Zakaznici;*

SQL příkazy můžeme dělit na čtyři skupiny:

1. Pro manipulaci s daty (SELECT, UPDATE, …)
2. Pro definici dat (CREATE, ALTER TABLE, …)
3. Pro řízení přístupu (GRANT, REVOKE)
4. Pro řízení transakcí (COMMIT, ROLLBACK)

Výhody SQL:

- Nemá problémy ani s velkými databázemi ve velmi velkých firmách
- SQL splňuje standardy společnosti ANSI&ISO.
- Jednoduchá migrace (9)

Nevýhody SQL:

- Přestože SQL databáze odpovídají normám ANSO&ISO standardů, tak některé databáze jsou chráněny autorským právem proti rozšíření standardního SQL
- Rozhraní databáze jsou složitá (9)

3.2.1.3 MySQL

MySQL je databázový systém postavený na dotazovacím jazyce SQL. MySQL je volně šiřitelný software, je publikován pod licencí GNU (General Public License). Vzhledem k tomu, že se jedná o volně šiřitelný software, tak je spousta podobných verzí jako například MariaDB či Percona Server, běžný uživatel si však rozdílů mezi distribucemi nevšimne. (10)

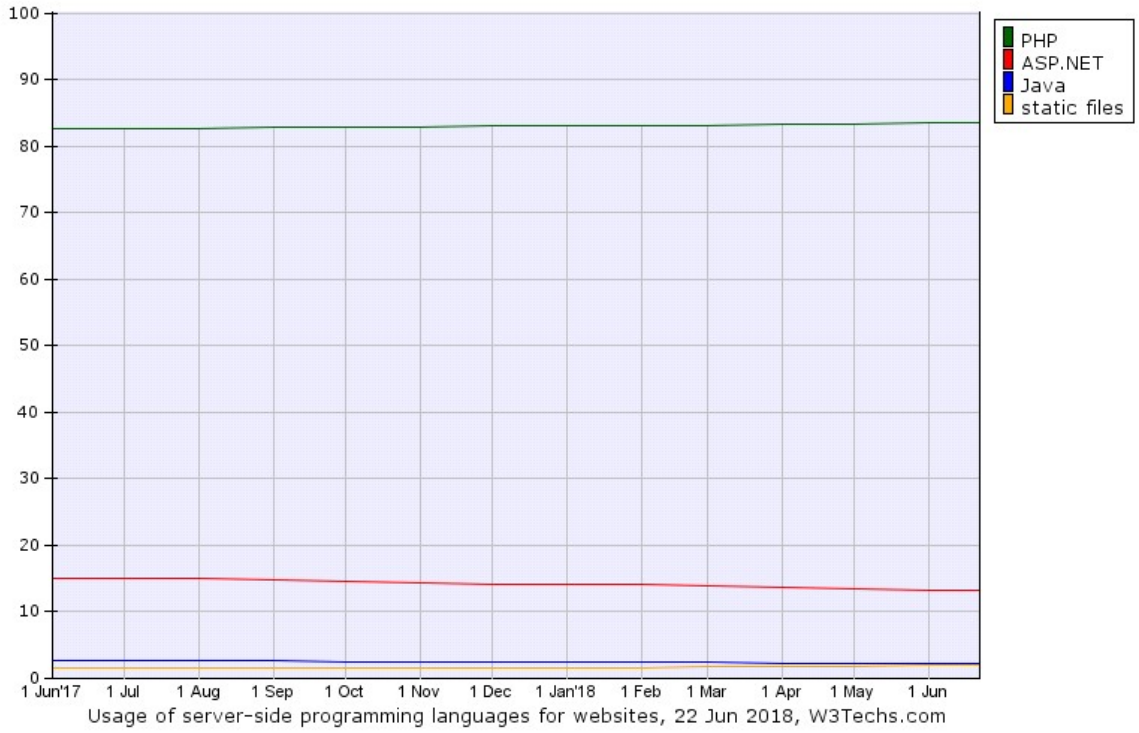
MySQL je k dispozici ke stažení na stránkách www.mysql.com/dowloads. Zde je možné stáhnout installer, který uživatele již krok po kroku navede při instalaci.

3.2.2 PHP

PHP (původně Personal Home Page) je skriptovací jazyk. Je určen pro tvorbu dynamických webových stránek a aplikací. PHP využívá toho, že skript je prováděn na straně serveru, tudíž k uživateli je přenášek až výsledek skriptu. Z tohoto důvodu není jazyk PHP závislý na platformě. Těží též z podpory různých knihoven, například pro grafiku, práci se soubory, přístup k různým databázím atd. (11)

Výhodou dynamického přístupu k webovým stránkám je, že pokud se nám opakuje nějaký kód, např. hlavička s odkazy, patička či menu, tak můžeme z této stránky vytvořit šablony. Do šablony je poté dle dané stránky vložen příslušný kód, který je již pro jednotlivé stránky rozdílný. (12)

PHP je nerozšířenějším skriptovacím jazykem pro web, k červnu 2018 s podílem 83,5 %, viz. obrázek č. 1. Nejvíce získal právě z důvodu své jednoduchosti.



Obrázek 1 - Porovnání programovacích jazyků pro tvorbu webových aplikací (13)

Velmi častou kombinací technologií je Linux, Apache (webový server), MySQL (databáze) a PHP. Pro toto uspořádání vznikla zkratka LAMP, popřípadě MAMP pro MacOS či WAMP pro Windows. (14)

3.2.2.1 Požadavky PHP

Aby bylo možné provozovat PHP projekt, musí být projekt uložen na serveru, který bude splňovat určité požadavky. Tyto programu musejí být nakonfigurovány, což provádí zpravidla správce serveru. Mezi tyto požadavky patří: (15)

- HTTP server
- Podpora PHP

- Databázový program (15)

Pro PHP aplikaci můžeme využít internetový server nebo server lokální. Internetový server je placenou službou, avšak v dané ceně máme hosting (uložení aplikace na serveru), podporu a zpravidla i doménu, přes kterou je poté naše aplikace přístupná z internetu. Oproti tomu lokální server znamená, že daná aplikace je uložena na našem zařízení. Často se využívají lokální servery při psaní a ladění aplikace, kterou následně pošleme na internetový server. (15)

3.2.2.2 Základní vlastnosti PHP

V PHP se používají tzv. otevírací a zavírací značky, mezi které poté píšeme náš kód. Otevírací značky říká serveru, že následují kód má zpracovávat jako kód jazyka PHP a uzavírací značka zas, že má přestat zpracovávat následující kód jako kód jazyka PHP. (8) Kód poté vypadá takto:

<?php ... náš kód ... ?>

Pro výpis do stránky slouží v PHP funkce *echo*. Ta do HTML stránky vloží hodnotu, kterou uvedeme v jejích vstupních parametrech. Vstupní parametry pro funkci *echo* uvádíme do závorek, které píšeme za název této funkce. (8) Kód vypadá následovně:

echo(„Dobrý den!“)

Jazyk PHP je dynamicky typovaný, tzn. Neuvádíme exaktně datový typ. Proměnné jsou v kódu označovány pomocí symbolu *$* a je nutné jej vždy použít při práci s danou proměnou. Na rozdíl od HTML u jakzyka PHP po příkazu píšeme znak *;*. (11) Zápis poté může vypadat následovně:

$pozdrav = „Ahoj“;

echo ($pozdrav);

Při práci s textovými řetězci je občas potřeba propojit textový řetězec a promměnou, ve které máme uloženou nějakou hodnotu. K takovému účelu používáme operátor . [tečky], který propojí řetězce. (16) Následký kód může poté vypadat takto:

$jmeno = „Lukáš Oplt“;

echo („Vaše jméno je “ . $jmeno);

Pro uložení více hodnot stejného typu nabízí PHP pole. Pole jsou, jak je zvykem v IT, indexovány od 0 a je možné nad nimi provádět různé operace. Pro definici pole se v PHP používá klíčové slovo *array()*, od PHP 5.4 mimo něj lze pole definovat také znaky *[]*. (16) Kód pro definici pole vypadá následnovně:

$pole = array(); nebo $pole = [];

Pro uložení hodnot do pole můžeme použít dva přístupy. Prvním je uložení hodnoty na určitou pozici, druhým je při deklaraci pole uvést již hodnoty najednou. Při použití druhého přístupu jsou data ukládaná na pozice jdoucí po sobě od 0, těmto hodnotám říkáme index. (16) Příklad přiřazení hodnot do pole vypadá následovně: (první příklad uvádí první přístup, druhý uvádí druhý přístup):

$pole[0] = 1;
$pole = array(1, 2);

Dalším typem strukuty, do které lze uložit hodnoty je asociativní pole. Funguje na podobném principu jako klasické (indexové) pole, avšak místo indexování od 0 nabízí místo indexů klíče, což jsou textové hodnoty. Definice pole má podobný způsob jako indexové pole, ale mimo hodnot zadáváme také klíče. Klíče k hodnotám poté přiřazujeme operátorem dvojité šipky (=>). (17) Příkladem takového pole může vypadat následovně:

$zkratky = array(
‘czu‘ => ‘Česká Zemědělská univerzita‘,
‘uk‘ => ‘Univerzita Karlova‘,
);

Pomocí PHP lze odesílat z webové aplikace i emaily. K tomuto účelu slouží SMTP server. V PHP nastavíme adresu SMTP serveru v souboru php.ini, což je soubor, kde je uloženo nastavení našeho PHP. Nastevení adresy má pak následující tvar: (15)

smtp = adresa serveru

3.2.2.3 Formuláře v PHP

Pomocí formulářů můžeme do PHP skriptu získat hodnoty od uživatele. Formulář musí mít definovaný atribut *action* (adresa skriptu, do kterého jsou hodnoty odeslány) a atribut *method*. Atribut *method* pracuje s dvěmi základníma metodami HTTP, konkrétně se jedná o metodu GET a metodu POST. Hodnoty metody GET jsou uváděny v adresním řádku (URL), zatímco hodnoty metody POST jsou uváděny jako http objekt, a tudíž je

nevidíme v adresním řádku a nejsou tak snadno editovatelná jako u metody GET. Pokud použijeme metodu GET, k datům přistupujeme pomocí $_GET[„název proměnné“]. V případě použití metody POST k datům přistupujeme pomocí $_POST[„název proměnné“]. (18) Kód formuláře vypadá takto:

```
<form action=“poslat.php“ method=“GET“>
<input type=“text“ name=“promenna“ />
</form>
```

3.2.2.4 Podmínky v PHP

V PHP, stejně jako například v céčkových jazycích, můžeme větvit běh programu. Jedná se rozhodování na základě hodnoty vstupu. V PHP se používá operátor *if*, ze kterým je v závorce uveden logický výraz. V případě, že je logický výraz (podmínka) pravdivý, tak se vykoná následující příkaz. V případě, že logický výraz pravdivý není, další příkaz je přeskočen a program pokračuje v běhu dál. K operátoru *if* lze také přidat operátor *else*. Příkazy v operátoru *else* jsou vykonány za předpokladu, že logický výraz, který je u operátoru *if*, není pravdivý. V případě, že v podmínce chceme vykonat více něž jeden příkaz, lze tyto příkazy uzavřít do složených závorek ({}), avšak tyto závorky fungují i jen s jedním příkazem. (19) Kód podmínky vypadá takto:

```
if(a > b){
        echo(‘A je větší než B‘);
}
else{
        echo(‘A není větší než B‘);
}
```

V podmínce můžeme využít operátory nebo funkce. V případě použití operátorů porovnáváme hodnoty vůči jiným, možnosti porovnání jsou uvedeny v tabulce č. 1, a výsledkem je buď pravda nebo nepravda. V případě použití funkce voláme takovou funkci, která nám vrátí hodnotu true nebo false. V podmínce můžeme také využít více podmínek zároveň, a to zkombinování výrazů pomocí || (operátor „nebo“) a pomocí && (operátor „a zároveň“). (19)

Operátor	Zápis
Je rovno	==
Je větší	>
Je menší	<
Je větší nebo rovno	>=
Je menší nebo rovno	<=
Není rovno	!=
Negace	!

Tabulka 1 - Operátory (19)

3.2.2.5 Cykly v PHP

Pro opakování kódu se v IT využívají tzv. cykly. Jedná se o smyčku, která se dokola opakuje dle daných pravidel. Cykly se dají dělit na while cykly, ty mají podmínku na začátku a opakují se, dokud je podmínka splněna, a for cykly, které mají předem nastavený počet opakování. (20)

V PHP známe tři typy cyklů. While cyklus, for cyklus a foreach cyklus. While cyklus je nejjednodušším typem opakování v PHP. V kódu je while cyklus označen klíčovým slovem *while*, za kterým následuje podmínka. Dokud je podmínka pravdivá, cyklus se opakuje. (21) Kód poté může vypadat následovně:

```
$cislo = 1;
while ($cislo <=5){
        echo $i;
        $i++;
}
```

Oproti while cyklu, který nemusí mít přesný počet opakování a může čekat na nějaký stav, for cyklus začíná s předem určeným počtem opakování. Kód for cyklus se skládá z klíčového slova *for*, za kterým následují v závorce tři parametry. Prvním parametrem je proměnná, dle které se počítá počet opakování. Tato proměnná musí být číslo a lze jí

definovat i v rámci for cyklu. Druhým parametrem je podmínka, dle které se vyhodnocuje, zda dojde k dalšímu opakování. Třetím parametrem je, co se po daném opakování stane s proměnnou, která se vyhodnocuje. (22) Příklad for cyklu pro vypsání čísel 1 až 10:

```
for($cislo = 1; $cislo <= 10; $cislo++){
        echo $cislo;
}
```

Posledním cyklem používaným v PHP je foreach cyklus. Foreach cyklus slouží pro procházení polí. Deklarace foreach cyklu probíhá pomocí klíčového slova *foreach*, za kterým následují vstupní parametry, konkrétně jaké pole se má procházet a do jaké pomocné proměnné mají být ukládány jednotlivé záznamy z pole v rámci daného cyklu. Tyto parametry jsou odděleny slovem *as*. (23) Příklad foreach cyklu pro výpis jmen ze seznamu:

```
foreach($jmena as $jmeno){
        echo $jmeno;
}
```

3.2.2.6 Historie PHP

V této části práce popisuji vývoj hlavních verzích jazyka PHP. Mimo tyto verze vyšlo již několik dalších, které vždy nesou označení ‘číslo verze**.**číslo‘. První zmínkou o PHP byl PHP Tools (Personal Home Page Tools). Jednalo se o sadu nástrojů, kterou vytvořil Rasmus Lerdorf v roce 1994. Původním účelem daného nástroje bylo sledování návštěvy webu. Postupem času Rasmus Lerdorf chtěl na svůj web více funkcí, a tak PHP Tools přepsal. Tato nová sada nabídla navíc interakce s databází a rámec, dle kterého bylo možné vytvářet jednoduché webové aplikace. V červnu 1995 Rasmus Lerdorf uvolnil zdrojový kód na internet, což dovolilo vývojářům jej využívat.

PHP/FI

Dalšího rozšíření se PHP dočkalo v září 1995, kdy PHP odkazoval na nástroj FI (Forms Interpreter). PHP obsahovalo proměnné, nabízelo automatickou interpretaci formulářových proměnných a syntaxe byla obsažená v HTML. Pro přidání PHP syntaxí do HTML bylo využíváno HTML komentářů. V říjnu 1995 Rasmus Lerdorf vydal kompletní přepis kód. PHP bylo záměrně strukturováno jako jazyk C. Bylo tomu z důvodu, aby se vývojáři, kteří programovali v C, Perl či podobně stavěných jazycích, lépe adaptovali. Do této doby byl PHP omezen jen pro unixové a posix-kompatibilní systému, avšak zvyšoval se potenciál i pro podporu Windows NT.

V dubnu 1996 představil Rasmus Lerdorf PHP/FI, byl vydán i další přepis a PHP se ze sady nástrojů začalo vyvíjet jako samostatný programovací jazyk. To však znamenalo nutnost přidání podpory pro DBM, mSQL, Postgres95 databáze, cookies a další. V červnu 1996 byl PHP/FI označen za PHP 2.0. Zásadní informací je, že v danou dobu byla jen jedna samostatně plnohodnotná verze PHP 2.0 V listopadu 1997 byl přepsán celý parsovací nástroj.

V letech 1997 a 1998 mělo PHP/FI tisíce příznivců po celím světě. Dle průzkumu Netcraft z května 1998 používalo PHP/FI na 60 tisíc domén. To, v dané době, znamenalo přibližně 1 % všech domén na internetu. (24)

PHP 3

V roce 1997 se začali zabývat Andi Gutmans a Zeev Suraski přepisem PHP/FI 2.0. Důvodem bylo, že PHP/FI 2.0 neobsahovalo různé funkce, například funkce pro elektronické obchodování. Později byl tento jazyk přejmenován z PHP/FI 2.0 na PHP (PHP: Hypertext Preprocessor).

Tento jazyk nabízel rozhraní pro práci s více databázemi, protokoly a API, což vzbudilo zájem mnoha vývojářů. V červnu 1998, za pomoci mnoha vývojářů z celého světa, byl vývojářským týmem PHP představen PHP 3 jakožto nástupce PHP/FI 2.0. Jednou z klíčových funkcí PHP 3 byla podpora objektově orientovaného programování. Po devíti měsících veřejného testování bylo po celém světě PHP 3 nainstalováno na více než 70 tisících doménách a PHP již nebyl omezen jen na posix-kompatibilní systémy. To znamenalo, že PHP 3 již běželo pod systémy Windows 95, 98, NT i Macintosh. V době své největší obliby bylo PHP 3 využíváno přibližně na 10 % webových serverů. (24)

PHP 4

V zimě 1998 začali Andi Gutmans a Zeev Suraski pracovat na přepsání PHP jádra. Cílem bylo zvýšit výkon a efektivitu u komplexních aplikací. Dále chtěli zvýšit modularitu PHP jádra. Hlavním problémem bylo, že PHP 3 sice podporovalo řady databází a API třetích stran, ale nebylo navrženo tak, aby efektivně zpracovávaly velké a složité operace.

Nové jádro, které bylo cílem PHP 4, bylo nazváno „Zend Engine“ (slovní hříčka z jmen autorů, Zeev a Andi). Požadavky dokázali splnit a v polovině roku 1999 bylo poprvé toto jádro představeno. PHP 4, které se zakládalo na tomto jádře, přineslo mnoho nových

funkcí ať už pro mnoho webových serverů, buffering výstupů a mnoho dalších. Oficiálně byla verze PHP 4 představena v květnu 2000. (24)

PHP 5

Po delší době vývoje a několika předběžných verzí byla v červenci 2004 představena verze PHP 5. Ta se zakládala na jádře Zend Engine 2.0, která přinesla vylepšený objektový přístup a mnoho nových funkcí. Od verze PHP 5.4 je součástí nativní podpora Unicode. (24)

PHP 6

V roce 2005 se začalo pracovat na verzi PHP 6. Hlavním cílem PHP 6 měla být plná podpora Unicode UTF-16. Bohužel v dané době nebyla celosvětově podporována ani verze UTF-8, což nakonec vedlo k negativním výsledkům ohledně výkonu. Na přelomu let 2009/2010 bylo v plánu verzi PHP 6 vydat, avšak tomu se nestalo z důvodů právě zmíněných problémů s poklesem výkonu. (25)

PHP 7

V prosinci roku 2015 byla vydána nejnovější verze PHP 7. Ta přinesla opět přestavěné jádro a nové jádro nese název PHP#NG (Next Generation). PHP 7 přineslo mnoho novinek, ale zásadní bylo až dvojnásobné zrychlení oproti poslední oficiálně vydané verzi, tedy PHP 5.6.

S dobou však šel i vývoj operačních systémů, a tak tomu verze PHP 7 šla naproti a přinesla podporu 64bitových systémů. Pozdější verze PHP 7, tedy verze 7.1 a 7.2, dále přinesly návratový typ void, možnost nastavení viditelnosti konstant a podporu algoritmu Argon2 sloužícího pro šifrování hesel. (26)

3.2.2.7 Výhody PHP

- Není závislý na operačním systému
- Kód nevyžaduje kompilaci, a tudíž zůstává v čitelné podobě
- Umí přímo komunikovat s SQL (27)

3.2.2.8 Nevýhody PHP

- Větší nároky na hosting

- Nejsou povinné standardy pro pojmenování, tudíž pokud je kód následně upravován někým jiným, například z důvodu změny programátora, tak je pro danou osobu složitější se v daném kódu vyznat (28)

3.3 PHP frameworky

Frameworky jsou dnes velmi rozšířené pro svou možnost pomoci programátorovi. Nabízí efektivnější řešení a zkrátí kód, kód se poté nemusí stále znovu psát. Framework je definován jako softwarová struktura, která může obsahovat podpůrné programy, různé API či návrhové vzory. (29)

V této práci jsou přiblíženy dva PHP frameworky. Prvním PHP frameworkem bude Nette, což je PHP framework vyvíjený českým open source vývojářem Davidem Grudlem. Jako druhý byl zvolen Laravel, což je celosvětově nejpoužívanější PHP framework. (30)

Velmi rozšířeným důvodem pro používání PHP frameworků není jen poskytnutá pomoc při programování, ale fakt, že frameworky mohou pomoci i s bezpečnostními dírami ve webových aplikacích. Spousta PHP frameworků nabízí různé restrikce pro různá pole, příkladem tak může být restrikce k zapsání jmen do formuláře. Pro příklad můžeme brát seznam lidí, kteří se někam zapisují a je tak požadováno jejich jméno a přímení. Pokud však do tohoto pole, které by neobsahovalo restrikce, napsali toto: *<script>alert(„Pozdrav“)</script>* tak při výpisu jmen nám vyskočí upozornění, ve kterém bude napsáno „Pozdrav“. Toto představuje velkou bezpečnostní hrozbu, od které může pomoci použití frameworku. (31)

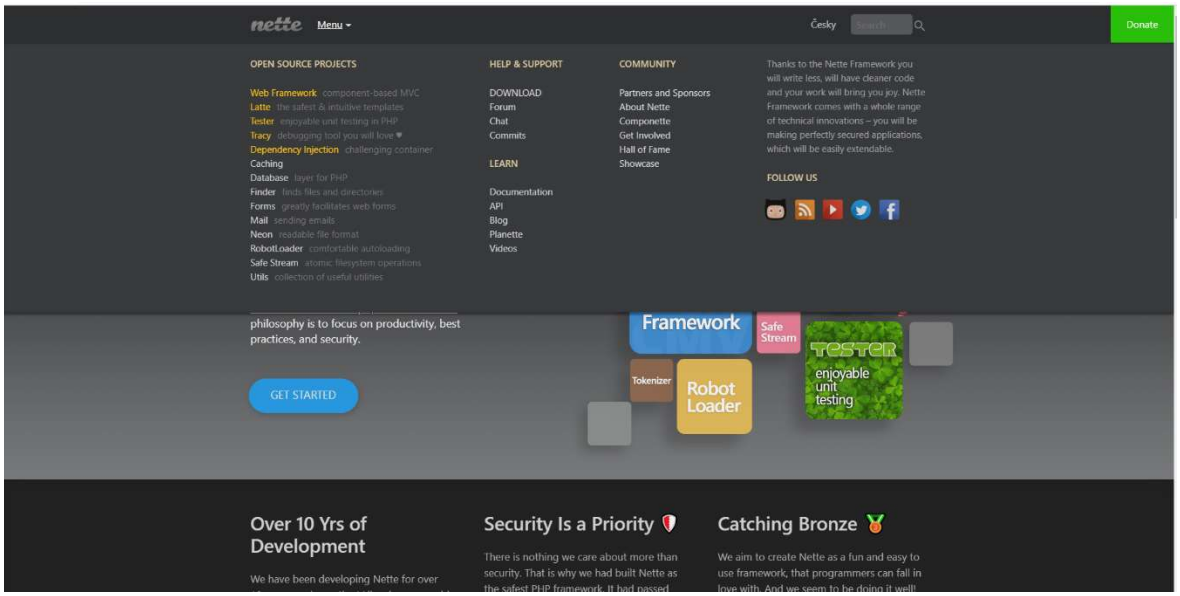
Výhody použití PHP frameworku

- Rychlejší vývoj aplikací
- Vyšší zabezpečení aplikací
- Podpora komunity (32)

Nevýhody použití PHP frameworku

- Frameworky jsou poměrně obsáhlé, a tak nemusí být moc čitelné na první pohled
- Frameworky přinášení vnější rizika
- Frameworky přistupují k aplikacím obecně (32)

3.3.1 Nette



Obrázek 2 - Domovská stránka frameworku Nette (zdroj: autor)

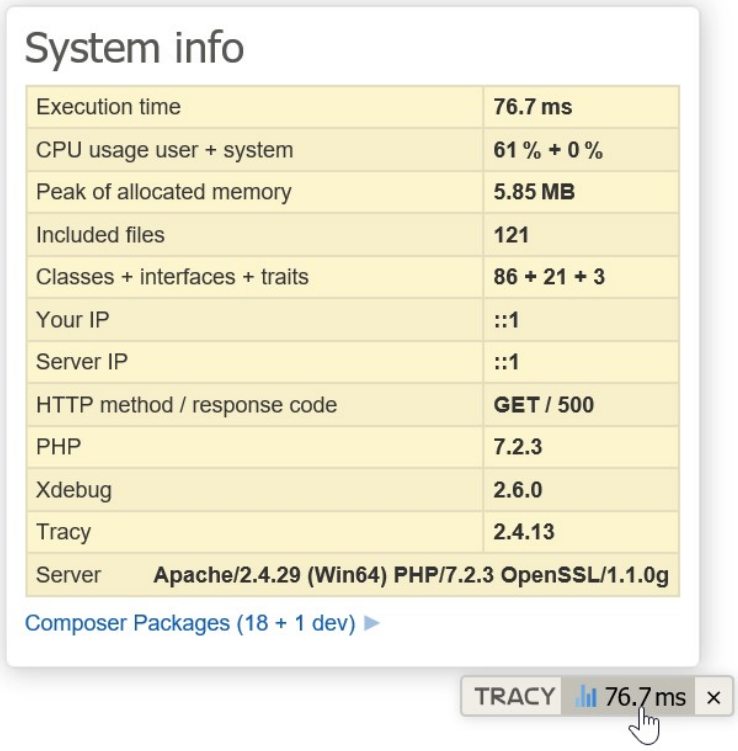
Nette je open-source framework pro tvorbu webových aplikací s využitím programovacího jazyka PHP. Nette je navrženo tak, aby případné rozšiřování kódu bylo snadné, rychlé a bezpečné. Autorem Nette je český open source programátor David Grudl. Nette se zaměřuje na eliminaci bezpečnostních chyb a na znovu použitelnost kódu. (33)

Oficiální webové stránky frameworku jsou www.nette.org, dokumentaci frameworku nalezneme na stránkách www.doc.nette.org. Framework Nette dále nabízí oficiální fórum, kam uživatelé mohou psát, pokud narazí na nějaký problém. Fórum nalezneme na stránkách www.forum.nette.org/cs/. Na všechny tyto stránky se dostaneme z domovské stránky frameworku www.nette.org, viz. obrázek č. 2.

V roce 2015 se dle ankety na Sitepoint.com jednalo o třetí nejpoužívanější PHP framework. Dle vyhodnocených dat byl Nette nejpoužívanějším PHP frameworkem na území České a Slovenské republiky. Dle věkového rozdělení se jednalo o druhý nejpoužívanější framework ve věkové skupině 18-25 let. (34)

Vždy poslední sobotu v měsíci se koná setkání fanoušků Nette, které je i samotným Nette oficiálně zastřešeno. Jedná se o možnost probrat možnosti řešení nějakého problému s jinými programátory. V rámci setkání se konají i přednášky na různá témata vývoje webových aplikací. Setkání se střídavě pořádají v Praze a Brně, výjimečně v jiných městech. (35)

Nette nabízí dvě možnosti spuštění na serveru. První možností je vývojářský režim, ten zobrazuje rychlost načtení stránky, viz obrázek č. 3, dobu komunikace s databází, či stav, zda je někdo přihlášen uživatelským účtem v aplikaci a informace o routě, přes kterou byla daná stránka volána. Další možností spuštění na serveru je produkční režim, který nezobrazuje časy komunikace s databází, dobu vykreslení a při chybě nezobrazí chybu se zdrojovým kódem, což slouží k ladění, ale místo toho odkáže uživatele na chybovou stránku. (36)



Obrázek 3 - Tracy Debugger (36)

3.3.1.1 Stažení

Pro stažení základní struktury Nette můžeme využít těchto možností:

- Instalace pomocí Composeru

  V Composeru najdeme cestu k projektu. Poté zadáme příkaz *composer create-project nette/web-project nette-blog*. Tímto příkazem se nám web project stáhne do složky nette-blog. (36)

- Stažení balíčku Nette

  Nette je také možní stáhnout jako .zip archiv. Tento archiv nalezneme na oficiálních stránkách www.nette.org/cs/download. (37)

3.3.1.2 Požadavky na server

Pro použití nejnovější verze Nette frameworku, v tuto chvíli verze 2.4, je nutné, aby server podporoval PHP 5.6.0 nebo novější. Mezi další požadavky patří .htaccess file protection, .htaccess mod_rewrite, Function ini_set(), Function error_reporting(), Function flock(), Register_globals, Zend.ze1_compatibility_mode, Session auto-start, Reflection extension, SPL extension, PCRE extension, ICONV extesion, PHP extesion, PDO extesion, Multibyte String extesion, Multibyte String function overloading, Memcache extension, GD extension, Bundled GD extension, Fileinfo extension nebo mime_content_type(). (38)

3.3.1.3 Struktura Nette

Nette využívá MVP (Model-View-Presenter) strukturu, ta rozděluje aplikaci do tří vrstev. Díky tomuto rozdělení je kód přehlednější, dá se lépe upravovat a poskytuje možnost každou část zvlášť testovat. Nette dále využívá tzv. Router. Router má za úkol z příslušné URL adresy zavolat příslušný presenter. Struktura Nette se skládá z:

- Model

  Datový a funkční základ aplikace. Obsahuje aplikační logiku, kterou můžeme volat a model nám vrací dle svých funkcí odpověď na daná volání. Akcí modelu je chápána jakákoli akce uživatele.
- View

  Tato vrstva má na starosti zobrazení výsledku požadavku, běžně k zobrazení využívá šablonovací systém
- Presenter

  Řadič, který na základě vstupů od uživatele volá příslušnou aplikační logiku. (39)

3.3.1.4 Dokumentace

Framework Nette nabízí dokumentaci na stránkách www.doc.nette.org. Dokumentace je dostupná jak v českém, tak v anglickém jazyce. Dokumentace se dělí na několik částí: Framework (popis práce s frameworkem), MVC aplikace (zde najdeme presentery, routování a mnoho dalšího), Latte (šablonovací systém), Dependency Injection (odebírání práv třídám), Formuláře (popis práce s formuláři), Databáze (popis práce s databází), Tracy (popis práce s Tracy – debugovací nástroj), HTTP (popis HTTP requestů a responsů), Užitečné třídy a Různé (vytažení často hledaných věcí).

3.3.1.5 Práce s databází

Framework Nette nabízí více než jeden způsob, jak webovou aplikaci propojit s databází. K propojení databáze můžeme využít Nette Database nebo knihovnu dibi. Zatímco dibi je externí knihovna, tak Nette Database je knihovna, která je součástí Nette. Použít lze i jiný nástroj jakým je například Doctrine2. (40)

Zadání příkazu do databáze je možné jak z presenteru, tak i z modelu. Pro přehlednost kódu je v Nette doporučeno využívat modelů. (40)

3.3.1.6 Debugger

Framework Nette nabízí ladící nástroj Tracy. Tracy slouží k usnadnění ladění PHP kódu. Nabízí mimo jiné i vizualizaci a logování chyb. V případě vyvolání chyby Tracy nabízí možnost danou chybu vyhledat pomocí internetového vyhledávače www.google.com. V tomto ladícím nástroji je také možné danou chybu potlačit a pokračovat v aplikaci. (41)

Debugger je možné i zakázat. Toto se využívá v případě, kdy aplikaci publikujeme. Dojde-li pak následně k nějaké chybě na produkci, místo zobrazení kódu je uživatel přesměrován na chybovou stránku. (36)

3.3.1.7 Struktura adresáře

```
www/                    ← kořenový adresář webu
└── nette-blog/
    ├── app/            ← adresář webové aplikace
    │   ├── config/     ← konfigurační soubory
    │   ├── presenters/ ← třídy presenterů
    │   │   └── templates/← šablony
    │   ├── router/     ← konfigurace URL adres
    │   └── bootstrap.php ← spouštěcí soubor
    │
    ├── log/            ← zde naleznete chybové logy
    ├── temp/           ← místo pro dočasné soubory (cache, sessiony, atd.)
    │
    ├── vendor/         ← knihovny pro vaši aplikaci
    │   └── nette/      ← jako například váš oblíbený framework :-)
    │
    └── www/            ← místní kořenový adresář webu - pouze tento adresář je přístupný z internetu
```

Obrázek 4 - Struktura Web Projekt (36)

Struktura adresáře projektu je popsána na Obrázku 4. Z internetu je však dostupný pouze adresář www, který je určen pro uložení obrázků, JavaScriptových souborů a CSS stylů. Z důvodu, že jen tento adresář je dostupný z internetu, tak by měl kořenový adresář směřovat právě sem. (36)

3.3.1.8 Historie Nette

Autorem PHP frameworku Nette je David Grudl. Ten začínal jako programátor rodinného e-shopu a časem začal hledat framework, který by mi práci usnadnil. Tehdy však nenašel žádný framework, který by splňoval jeho požadavky, a tak si funkci po funkci začal budovat vlastní. Na daném frameworku pracoval již od roku 2004 a v roce 2007 jej prezentoval na konferenci PHP frameworky. Tímto představením byla uvedena první verze Nette. (42)

Nette 2.0

V roce 2010 byla na WebExpu představena verze 2.0, ta však ze začátku nepřinesla nic nového. Jednalo se tedy pouze o povýšení verze 1.0 na verzi 2.0. Postupem času přibývaly ve verzi nové funkce jako například Debug Bar, API pro rozšíření Latte (šablonovací systém) či novou strukturu jmenných prostorů a tříd. (43)

Nette 2.1

V roce 2013 byla představena verze 2.1. Ta přinesla podporu pro Bootstrap, velké změny pro práci s databázemi, rozšíření pro Latte a v neposlední řadě přepracovaná vrstva Presenter, která nyní zabraňuje vkládání polí do persistentního parametru. (44)

Nette 2.2

V roce 2014 byla představena verze 2.2. Ta přinesla novou třídu Nette\Security\Passwords, která řeší hashování hesel. Hlavní změnou, kterou přinesla verze 2.2 bylo rozdělení původního repositáře Nette do celkem 19-ti samostatných komponent, například Application, Bootstrap, Nette Database, Forms, atd. (45)

Nette 2.3

Ještě v roce 2014 byla představena verze 2.3. Velkou změnou byl přechod na case sensitive, Nette se tak stalo závislé na velikosti písmen u tříd, metod i proměnných. (46)

Mezi další novinky, které verze 2.3 přinesla patří rozšíření funkcí Bootstrapu, práce s databázemi či funkce „Finder“. Vylepšení se též dočkali formuláře, které získali přímé napojení na Nette DI. (47)

Nette 2.4

V roce 2016 byla představena verze 2.4. Ta přinesla možnost nastavení průhlednosti k img souborům, podporu PHP7 typehintů. Dále přibyl funkce pro šablonovací systém Latte. V neposlední řade přibyla varování pro nekompatibilní funkce/kontext. (48)

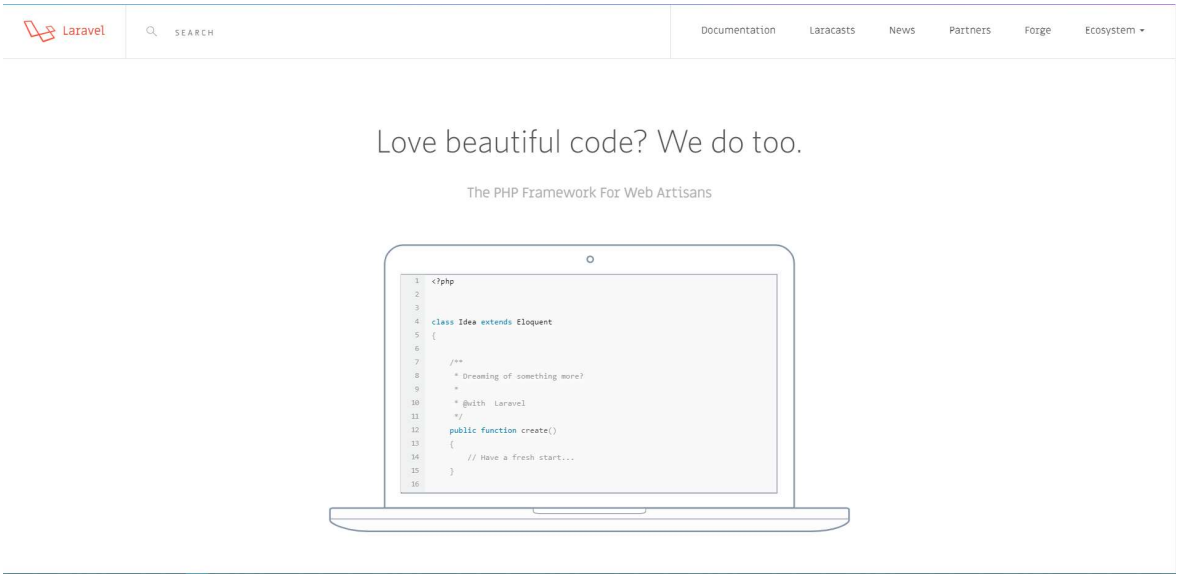
3.3.1.9 Výhody frameworku Nette

- Automaticky generuje validační JavaScript
- Podpora automatického překladu
- Obsahuje silný validační jazyk (49)

3.3.1.10 Nevýhody frameworku Nette

- Nutnost promazávat cache paměť
- Slabá dokumentace (49)

3.3.2 Laravel



Obrázek 5 - Domovská stránka frameworku Laravel (zdroj: autor)

Framework Laravel je dle průzkumů různých portálů nejpoužívanějším PHP frameworkem na světě. Mezi tyto portály patří sitepoint.com, coderseyes.com či medium.com.

Laravel je PHP framework vytvořený Taylorem Otwellem za účelem tvorby webových aplikací. Mezi výhody, které Laravel nabízí, patří jednoduchá integrace s různými knihovnami, různé možnosti připojení k relačním databázím a v neposlední řadě velká a aktivní komunita. (50)

Oficiální webové stránky, které jsou zachyceny i na Obrázku 5, frameworku jsou www.laravel.com, dokumentaci frameworku nalezneme na stránkách www.laravel.com/docs. Oficiální Laravel fórum nalezneme na adrese www.laracast.com/discuss. Všechny odkazy jsou dostupné hlavní stránky.

Laravel nabízí celkem velký ekosystém. Nabízí hostingy s automatickým deploymentem, video tutoriály, implementaci HttpKernelu a spoustu dalších rozšíření. Poskytuje širokou podporu implementovaných komponent, například podporu platebních bran, autorizaci uživatelů či šifrování. (51)

3.3.2.1 Stažení

Pro využití frameworku Laravel máme několik možností stažení frameworku. Tyto možnosti jsou:

1. Přes Laravel Installer

   Nejprve je nutné stáhnout Laravel Installer využívající Composer. Zde zadáme příkaz *composer global require "laravel/installer"*. Poté stačí pomocí příkazu *laravel new* instalaci dokončit. Tento způsob instalace je mnohem rychlejší než pomocí Composer Create-Project. (52)

2. Pomocí Composer Create-Project

   Při instalaci pomocí Composeru stačí do Composeru zadat příkaz *composer create-project --prefer-dist laravel/laravel blog*. (52)

3.3.2.2 Požadavky na server

Aby běžela nejnovější verze Laravelu, v tuto chvíli verze 5.7, potřebuje Laravel, aby server měl nainstalováno PHP ve verzi 7.1.3 nebo novější, OpenSSL PHP Extension, PDO PHP Extension, Mbstring PHP Extension, Tokenizer PHP Extension, XML PHP Extension, Ctype PHP Extension, JSON PHP Extension a BCMath PHP Extension. (53)

3.3.2.3 Struktura Laravelu

Laravel využívá MVC (Model-View-Controller) strukturu, ta rozděluje aplikaci do tří vrstev. Díky tomuto rozdělení je kód přehlednější, dá se lépe upravovat a poskytuje možnost každou část zvlášť testovat.

- Model

  Vrstva obsahující logiku programu.

- View

  Tato vrstva má na starosti zobrazení, běžně k zobrazení využívá šablonovací systém

- Controller

  Řadič, který na základě vstupů od uživatele volá příslušnou aplikační logiku. (39)

3.3.2.4 Dokumentace

Framework Laravel nabízí oficiální dokumentaci na webové stránce www.laravel.com/docs. Dokumentace je dostupná pouze v anglickém jazyce, za to je nabízená i formou videí na portále laracasts.com, na který je v dokumentaci odkaz pro danou

verzi frameworku. V dokumentaci každé verze Laravelu je také odkaz na příslušný laracast, což je video seznámení s danou verzí frameworku.

3.3.2.5 Práce s databází

Framework Laravel umožňuje připojit databáze několika typů. Konkrétně se jedná o databáze MySQL, PostrageSQL, SQLite a SQL Server. Pro práci s databází slouží databázová vrstva Eloquent, avšak lze použít i Query Builder. (54)

Standardně se databázové příkazy zadávají do Rout. Pro větší přehlednost však lze využít i Controller. (55)

3.3.2.6 Debugger

Framework Laravel nabízí ladící nástroj Whoops. Whoops nabízí jednoduché API pro řešení chyb v kódu, zobrazení vstupních zdrojů (např. JSON, XML, SOAP), zobrazení místa chyby v kódu. Whoops dále nabízí vyhledání dané chyby na internetovém prohlížeči www.google.com. (56)

3.3.2.7 Struktura adresáře

Struktura frameworku Laravel je rozdělena do adresářů app, bootstrap, config, database, public, resources, routes, storage, tests a vendor. Hlavním adresářem je adresář app, kde je uložena logika dané aplikace. (57)

Adresář bootstrap slouží napojení na CSS framework Bootstrap, adresář config obsahuje konfigurační soubory. Adresář database obsahuje informace o migracích databáze a její struktuře. V adresáři public nalezneme soubory upravující CSS styly či JavaScript soubory. Adresář resources obsahuje pohledy, adresář routes zas obsahuje definici všech rout. V adresáři storage nalezeme soubory s informacemi jako různé HTML stránky, cashe a soubory generované frameworkem k dřívějším spuštěním. Adresář tests obsahuje automatizované testy a adresář vendor obsahuje propojení na Composer. (57)

3.3.2.8 Artisan console

Framework Laravel nabízí vestavěný příkazový řádek, který se nazývá Artisan. Artisan nabízí 72 příkazů a ke každému příkazu funguje i parametr -help, který vypíše nápovědu pro daný příkaz. Pro použití nějakého příkazu se do příkazové řádky zadá příkaz ve tvaru: (58)

php artisan [příkaz]

Artisan příkazy pomáhají urychlit vývoj, pomocí Artisan-u je možné například provést migraci schématu databáze, vytvořit login systém v aplikaci. Výpis všech příkazů lze použít příkaz v příkazové řádce: (58)

php artisan list

3.3.2.9 Historie Laravelu

V říjnu roku 2009 přišla verze PHP 5.3, tato verze přinesla do PHP namespace a anonymní funkce (closure). Avšak mnoho frameworků tyto funkce neimplementovala a zaměřily se spíše na podporu kompatibility s předchozí verzí PHP. V tehdejší době byl oblíbený PHP framework Codeigniter, avšak nepodporoval různé funkcionality, například přihlašování uživatelů. To byl dle Taylora Otewlla důvod, proč vydal Laravel – aby zaplnil mezeru v nabídce frameworků. (59)

Laravel 1

Taylor Otwell implementoval autentizaci uživatelů, správu sessions, databázovou vrstvu Eloquent a mnoho dalších funkcí. První beta verze Laravel 1 byla publikována 9. června 2011. Byť se ještě nejednalo o plnohodnotný framework – chyběly controllery, tak i přes to se stal Laravel tématem mezi vývojáři. Během stejného roku byly přidány nové funkce, například podpora formulářových validací či instalace z příkazové řádky. (59)

Laravel 2

V listopadu 2011 byla vydána verze 2, která rozšířila Laravel o podporu controllerů, šablonovací systém Blade a IoC kontejner. Nyní se již dá mluvit o plnohodnotném frameworku. (59)

Laravel 3

V únoru 2012 vyšla verze 3, ta se soustředila na podporu unit testování. Dále k funkcionalitám přibyl Artisan, příkazová řádka, ze které je možné Laravel ovládat. Mezi další novinky se řadila podpora databázových migrací. Shodou okolností byl ve verzi 3 na chvíli odstraněn Eloquent (databázová vrstva), avšak při ohlasu vývojářů byl Eloquent zas vrácen do Laravelu. (59)

Laravel se stal nejrychleji se šířícím frameworkem. Po pár měsících po vydání verze 3 se Taylor Otwell rozhodl přepsat Laravel, aby byl balíčkovatelný prostřednictvím Composeru. (59)

Laravel 4 „Illuminate“

V květnu 2013 byla představena verze 4, která nesla kódové označení „Illuminate“. Tato verze se zaměřila na trendy v programování, což obecně bylo cílem Laravelu – držet krok s novými technologiemi. Ve verzi 4 tak přišla podpora front, dále přibyla možnost přes různé API odesílat emaily skrze různé providery. (59)

Byť Laravel vydával verze poměrně rychle a implementoval tak nejmodernější technologie, tak vývojáři volali po zpomalení a zvýšení stability, aby byli schopni upgradovat vlastní aplikace. Toto přání bylo vyslyšeno, a tak na Laraconu (konference pro Laravel) byly v roce 2013 oznámeny klasické LTS a STS verze. (59)

Laravel 5

V únoru 2015 byl představen Laravel 5, který opět přinesl nové technologie. Nově Laravel obsahoval Elixir – nástroj pro buildění Javascriptu, CSS a obrázků. Dále se změnila adresářová struktura aplikace. Verze 5 přinesla navíc knihovnu DotEnv, pomáhá s detekcí prostředí. (59)

Laravel 5.1

V červnu 2015 byla vydána verze 5.1, ta byla první LTS verzí (verze s dlouhodobou podporou). Dále verze přinesla podporu PSR-2, což umožnilo využíval tabulátory namísto mezer či otevírací závorky na stejném řádku jako funkce. (59)

Laravel 5.2

V prosinci 2015 byla představena verze 5.2. Ta přinesla implementaci rozšířené autentizace, podporu Moddleware skupin a validace polí. Dále přibyla metoda pro snadnější implementaci Eloqeunt. (60)

Laravel 5.3

V srpnu 2016 byla představena verze 5.3. Ta byla zaměřena na zrychlení vývoje webových aplikací. Mezi nové funkcionality patří podpora notifikací, fulltextové vyhledávání v Eloquent models, ukládání uploadovaných souborů či přidání funkce *loop.*

Dále Laravel přidal autentizaci pomocí API, přes Laravel Passport, která poskytuje plnou implementaci serveru OAuth2 v řádech minut. (61)

Laravel 5.4

V lednu 2017 byla vydána verze 5.4. Ta přinesla Laravel Dusk, který podporuje end-to-end testování. Mezi další funkce patří možnost využití značkovacího jazyka Markdown pro tvorbu emailových šablon. Tato verze také přinesla jednodušší zápis cesty a zkratky pro procházení kolekcemi. (62)

Laravel 5.5 LTS

V srpnu 2017 vyšla verze 5.5 LTS, jedná se tedy o další verzi s dlouhodobou podporou. Novou funkcí, která přibyla, byl Laravel Horizon, což je dashboard s informacemi o dané webové aplikaci. Nabízí možnost konfigurace souborů a nabízí vylepšení pro práci ve více lidech. Další funkcí, která přibyla, je „Console Command Auto-Registration“, díky ní není potřeba ručně přidávat vytvořený nový příkaz pro konzoli do konzolového listu příkazů. (63)

Laravel 5.6

V únoru 2018 byla představena verze 5.6. Tato verze přinesla funkci „Task Scheduling“, díky které je možné, pro vývojáře, kteří najednou používají více serverů, spouštění skriptů pouze na jednom serveru. Dále přibyl nový systém logování a defaultní podpora Bootstrap 4. (64)

Laravel 5.7

V srpnu 2018 byla představena doposud poslední verze, jedná se tedy o aktuální verzi, Laravel 5.7. Tato verze přináší funkci Laravel Nova, což je přepracovaný dashboard nástroj pro správu nejen aplikace, ale nyní je možné přes tento dashboard spravovat též Eloqeunt databázovou vrstvu. Dále přibyla funkce pro verifikaci emailu, funkce „Guest User Gates“, která nabízí pro nepřihlášené uživatele univerzálního uživatele Guest. Další novou funkcí je „Notification Localization“, díky které je možné nastavit lokalizaci pro dané notifikace či funkce „Filesystem Read / Write Streams“, která přináší implementaci *readStream* a *writeStream* metod. (52)

3.3.2.10 Výhody frameworku Laravel

- Využívá nejnovějších funkcí PHP
- Vestavěný nástroj pro příkazový řádek (Artisan)
- Skvělá dokumentace (65)

3.3.2.11 Nevýhody frameworku Laravel

- Verze mezi sebou nejsou kompatibilní
- U upgradů se stává, že upgrady jsou s chybami (65)

4. Praktická část

V praktické části budou porovnány frameworky Nette a Laravel. Porovnané budou rychlosti jednotlivých frameworků ve stejných úlohách, jejich bezpečnost, podpora Dependency injection, počet řádků při stejné aplikaci, výskyt na fórech a uplatnění na trhu práce. Dále porovnání spočívá v porovnání zdrojů pro naučení se frameworku a rychlosti implementace nových technologií.

4.1 Rychlost frameworků

Rychlost jednotlivých frameworků byla testována ve stejných úlohách, kdy k testování byl použit open-source program Apache JMeter 5.0. Tyto testy jsou uvedeny v příloze č. 2 – Nette.jmx a příloze č. 3 – Laravel.jmx. Každý z těchto testů testuje výpis všech informací z databáze, kde bylo uloženo 10000 záznamů, které obsahovali jednu číselnou a dvě textové hodnoty. Dále byla testována rychlost zobrazení jednoho záznamu z databáze a update záznamu v databázi.

Pro zamezení náhody bylo testování provedeno 1000krát pro každou operaci. Z těchto výsledků je následně vypočítána minimální hodnota, maximální hodnota, průměr a medián. Pro každý framework byla vytvořena samostatná databáze, která měla stejné parametry a stejné hodnoty.

Parametry zařízení, na kterém bylo prováděno testování:

- Procesor: Intel® Core™ i5-8250 CPU @ 1.60GHz
- RAM: 8,00 GB
- Typ disku: M.2 SSD

4.2 Bezpečnost

Dobrá aplikace není jen rychlá, ale také bezpečná. Vývojáři používají frameworky také z důvodu, aby jim daný framework pomohl s bezpečností aplikace a nemuseli tak všechny hrozby odstraňovat sami.

V rámci tohoto porovnání jsem použil metody SQL injection a Cross Site scripting. První uvedená metoda spočívá ve vložení SQL kódu do POST metody, která je následně odeslána na server ke zpracování. Když framework odolá tomuto útoku, projeví se to tak, že daný příkaz nevykoná. Tudíž v případě, že pomocí SQL injection odešleme na server kód

pro smazání databáze, databáze, v případě odolání útoku, nebude smazána. Cross Site scripting funguje na principu vkládání javascriptového kódu do polí pro text. V případě neodolání frameworku se následně script provede. Při odolání frameworku script nefunguje a má jen funkci textu, často daný kód framework ještě poupraví, např. odstraní závorky. (66)

Pro testování SQL injection byl použit vkládaný kód „‘; DROP TABLE [název tabulky]; --“. Dále byl použit kód „‘; SELECT * FROM [název tabulky]“. Při porovnání Cross Site scrippting byl nejdříve použit skript „<script type=“text/javascript“> alert(‘ahoj‘); </script>“. Dalším skriptem pro Cross Site scrippting úkol byl skript „<script type=“text/javascript“> <body onload=alert('test1')> </script>“, který navíc oproti prvnímu útoku upravuje atribut stránky aplikace.

4.3 Dependency injection

Dependency injection (česky „vkládání závislostí“) slouží pro předání závislosti nadřazené třídě. Příkladem může být metoda, která ukládá do databáze. Bez Dependency injection musí mít tato metoda propojení na databázi. Pokud použijeme Dependency injection, nutnost zajistit propojení s databází přechází z metody na třídu. Volně by se dalo Dependency injection tedy přeložit jako „Ať se postará někdo jiný“. Tento princip umožňuje rychlejší vývoj, ale navíc je možné implementovat do kódu i metody jiných vývojářů jako jsou například platební brány. (67)

Využívat Dependency injection nelze do nekonečna. Nakonec narazíme na nějaký bod v stromové struktuře, který, např. dané databázové spojení, musí zajistit. Tomuto bodu se říká DI kontejner. DI kontejner obstarává tyto závislosti. Z důvodu, že celý DI kontejner je dosti komplikovaný, spousta frameworků je rovnou zahrnuje v sobě, a tak vývojář může Dependency injection rovnou používat namísto psaní zdlouhavého a složitého kódu. (67)

4.4 Počet řádků kódu

Počet řádků aplikace reprezentuje číslo, kolik řádků je nutné napsat, aby aplikace fungovala. Čím vyšší je toto číslo, tím vyšší je pracnost na dané aplikaci Z tohoto důvodu bude porovnaný počet řádků aplikace, který bylo nutné napsat pro vytvoření stejné aplikace při použité jednotlivých frameworků.

4.5 Výskyt ve fórech

V případě, že programátor má nějaký problém s kódem, většinou použije internet pro nalezení řešení. I samy frameworky nabízejí, v případě chyby, danou chyby vyhledat pomocí internetového vyhledávače www.google.com.

Výskyt ve fórech jsem porovnával na největších IT fórech na internetu. Jedná se o www.stackoverflow.com, www.quora.com a www.github.com. Oba frameworky dále nabízejí vlastní fóra, kde je možné vyhledávat, z důvodu, že se jedná o fóra určená pouze k danému frameworku a tudíž druhý framework se na nich nebude vyskytovat, tyto fóra nejsou zahrnuta do srovnání.

4.6 Uplatnění na trhu práce

Programování je velmi oblíbenou a dobře placenou prací. Pro další způsob porovnání jsem si tak vybral porovnání uplatnění na trhu práce. Pro toto porovnání jsem vyhledal, kolik nabídek práce se nachází na českých (Jobs.cz, Startupjobs.cz, Prace.cz, Jobnes.cz) a amerických (Us.jooble.org, Monster.com, Indeed.com) pracovních portálech. Pracovní nabídky byly vyhledány k datu 10.2.2019.

4.7 Zdroje učení

Abychom ovládali nějaký framework, musíme se jej nejdříve naučit. Pro učení mohou posloužit různé materiály jakými mohou být například knihy, online články, tutoriály, videa či kurzy. Mezi oblíbené v dnešní době patří video tutoriály, které můžeme najít na spoustu věcí na jednom z největších portálů pro online videa, YouTube.

V rámci mého porovnání porovnám, jaké zdroje nabízí jednotlivé frameworky. Přes tištěné materiály k online materiálům. Zde budu hodnotit jaké nabízí daný framework možnosti, protože porovnání kvality materiálů je do určité míry dost subjektivní hodnocení.

4.8 Rychlost implementace nových technologií

V dnešním světě IT se stále častěji setkáváme s tím, že software rychle zastarává. Důkazem toho mohou být i stále častější updaty verzí různých webových technologií. Z tohoto důvodu jsem se rozhodl do porovnání přidat i to, jak rychle frameworky reagují na implementaci nových technologií.

5. Výsledky

V této části mé práce uvádím výsledky pro jednotlivá porovnání frameworků Nette a Laravel. Všechny výsledky z testování jsou dostupné v příloze č. 1 – data.xlsx.

5.1 Rychlost frameworku

První testovanou oblastí v rámci rychlosti jednotlivých frameworků je porovnání času, za který framework vypíše 10000 záznamů, každý o třech hodnotách. Aby bylo zamezeno náhodnému efektu, bylo měření provedeno 1000krát a z hodnot byl vypočítán průměr a medián, kdy menší hodnota znamená kratší část vypsání a tím i lepší výsledek.

	Nette	Laravel
Průměr [ms]	5223	6520
Medián [ms]	5668,5	6895
Nejlepší medián / Medián [%]	100	82,21

Tabulka 2 - Zobrazení všech záznamů

Z tabulky č. 2 je vidět, že medián doby, za kterou danou činnost vykonal framework Nette je o téměř 20% nižší než medián doby frameworku Laravel. Z tohoto srovnání tak jasně vyhrává framework Nette.

V dalším testování rychlosti frameworků byla testována rychlost vypsání jednoho záznamu. Toto měření bylo opět opakováno 1000krát a z výsledných hodnot byla vypočítána průměrná doba a medián doby.

	Nette	Laravel
Průměr [ms]	4087	5328
Medián [ms]	4494,5	5765
Nejlepší medián / Medián [%]	100	77,96

Tabulka 3 - Zobrazení jednoho záznamu

I z tohoto měření opět vyhrává framework Nette, viz. tabulka č. 3, a to ještě s mírně lepším výsledkem něž při testování zobrazení všech dat z databáze, který byl prováděn jako první test.

Posledním testem rychlosti bylo porovnání rychlosti v zápisu do databáze, kdy byl z aplikace odeslán požadavek na zapsání dat do databáze. Tento úkon se opět opakoval 1000krát pro zamezení efektu náhody. Z těchto hodnot byla vypočítána průměrná doba a medián doby.

	Nette	Laravel
Průměr [ms]	9221	16657
Medián [ms]	10420,5	18916
Nejlepší medián / Medián [%]	100	55,09

Tabulka 4 - Update záznamu

V tabulce č. 4 můžeme vidět, že v tomto měření je rozdíl v rychlosti frameworků největší. Při tomto úkolu potřeboval framework Laravel o více jak 80 % více času než framework Nette.

	Nette	Laravel
Zobrazení všech záznamů	100	82,21
Zobrazení jednoho záznamu	100	77,96
Update záznamu	100	55,09
∑Výsledků / 3	100	71,75

Tabulka 5 - Výsledky testováni rychlosti

Pokud bychom tyto výsledky srovnáme, zjistíme, že framework Nette byl rychlejší ve všech testech rychlosti než framework Laravel, viz. tabulka č. 5. Z tohoto důvodu Nette získává z této části porovnání celých 100 %, zatímco Laravel získává jen 71,75 %.

5.2 Bezpečnost

Dalším porovnáním byla bezpečnost. V rámci bezpečnosti byly simulovány útoky SQL injection a Cross Site scripting. V tabulce č. 6 je uvedeno, zda si frameworky poradili s daným útokem či nikoli.

	Nette	Laravel
První SQL injection	ano	ano
Druhý SQL injection	ano	ano
První Cross Site scripting	ano	ano
Druhý Cross Site scripting	ano	ano

Tabulka 6 - Zvládnutí útoků na framework

Jak je vidět z tabulky č. 6, tak oba frameworky odolali daným útokům. Tudíž je možné považovat oba frameworky za bezpečné. Z důvodu, že oba frameworky obstáli v testu bezpečnosti na výbornou dostávají z části bezpečnosti hodnocení 100 %.

5.3 Dependency injection

Podpora Dependency injection je důležitá pro znovupoužití kódu, tudíž i na tuto oblast jsem se v porovnání zaměřil. Dependency injection potřebuje tzv. DI kontejner, který spravuje závislosti. Od frameworku se očekává, že DI kontejner nabízí, vzhledem k tomu, že se jedná o rozsáhlejší kód. (67)

Nette framework podporuje Dependency injection. Znamená to, že v rámci tohoto frameworku je již připraven DI kontejner pro správu závislostí. Nette samotné na Dependency injection odkazuje a věnuje mu značnou část své dokumentace. (68)

Podporu Dependency injection nabízí také framework Laravel, avšak pro tuto funkcionalitu používá název Service Container. Byť se jedná o rozdílné názvy, tak v použití této funkcionality to ničemu nebrání a v dokumentaci pro Service Container od Laravelu nalezneme též označení „Dependency injection“, tudíž není problém dohledat tuto funkcionality i v případě, že neznáme tento název. (69)

Z tohoto porovnání tedy oba frameworky získávají 100 % z důvodu, že Dependency injection podporují. Jediným rozdílem mezi frameworky je název, zatímco Nette framework používá označení „Dependency injection“, tak Laravel danou funkcionalitu označuje jako „Service Container“.

5.4 Počet řádků kódu

Pro vytvoření aplikace v obou frameworcích bylo nutné upravit určité množství souborů. Proto porovnávám, kolik souborů bylo nutné upravit a kolik řádků kódu znamenalo vytvoření daných aplikací.

	Nette	Laravel
Počet upravených souborů	7	5
Počet řádků kódu	155	138
Nejnižší počet řádků / Počet řádků [%]	89,03	100

Tabulka 7 - Počet upravovaných souborů

Počet upravovaných souborů je, viz. tabulka č. 7, vyšší u Nette. Bylo nutné upravit přesně o 2 soubory více, což v tomto malém množství dělá o 40 % více souborů než u Laravelu. V řádcích tento rozdíl činní 17 řádků, což dělá už jen 12% rozdíl. Laravel tak za tuto oblast získává plných 100 %, zatímco Nette získává jen 89,03 %.

Důvodem tohoto rozdílu je struktura frameworků. Zatímco Laravel využívá MVC architekturu, Nette využívá MVP architekturu. Rozdíl v tomto použití je takový, že aplikační logiku pro rozdílné stránky můžeme u MVC architektury psát do jednoho souboru, Controlleru, u MVP architektury musíme pro jednotlivé stránky zakládat vlastní soubor, Presenter. Toto porovnání je tedy dosti subjektivní, protože byť více souborů znamená i více řádků, protože pro každý soubor definujeme minimálně namespace, může někomu připadat přehlednější rozdělení aplikační logiky do více souborů.

5.5 Výskyt ve fórech

Jak již bylo zmíněno, ve fórech často vývojáři hledají odpovědi na problémy, na které narazili. Proto je také porovnání výskytu ve fórech důležitou součástí celkového porovnání frameworků. Mimo to je dobré zmínit, že oba frameworky nabízí také vlastní fóra, která jsou dostupná z hlavních webových stránek frameworků.

	Nette	Laravel
Stackoverflow	418	108216
Quora	1	~3400
Github	92	~7000
∑Výskytu / 3	153,25	29654
Počet výskytů / Nejvyšší počet výskytů [%]	0,52	100

Tabulka 8 - Výskyt ve fórech k 10.2.2019

V tabulce č. 8 můžeme vidět výskyt pro jednotlivé frameworky na největších vývojářských fórech. Dle těchto výsledků má jasně navrch framework Laravel, který je mnohem diskutovanější a tudíž je, v případě potřeby, vyšší šance najít odpověď na otázku, kterou hledáme.

5.6 Uplatnění na trhu práce

Každý vývojář hledá uplatnění, proto jsem do porovnání zahrnul také uplatnění na trhu práce. Toto kritérium porovnávám na základě nabídek práce na internetových portálech v České republice a USA, jakožto zástupce zahraničí.

	Nette	Laravel
Jobs.cz	42	8
Startupjobs.cz	31	0
Prace.cz	41	7
Jobdnes.cz	33	6
∑Nabídek / 4	36,75	5,25
Počet nabídek / Nejvyšší počet nabídek [%]	100	14,29

Tabulka 9 - Pracovní nabídky v České republice k 10.2.2019

Jak je vidět v tabulce č. 9, v České republice má je větší poptávka po lidech, kteří umí pracovat s frameworkem Nette než po lidech, kteří umí pracovat s frameworkem Laravel. Rozdíl v poptávce po těchto lidech je dokonce několikanásobný.

Důvodem pro tento výsledek bude s největší pravděpodobností důvod, že v České republice je nejčastěji používaný PHP framework Nette, a tudíž je tu i největší zázemí projektů v Nette. Dalším důvodem může být česká podpora Nette, kterou Laravel nenabídne. (70)

	Nette	Laravel
Us.jooble.org	0[1]	430
Monster.com	0	300
Indeed.com	0	661
∑Nabídek / 3	0	347,75
Počet nabídek / Nejvyšší počet nabídek [%]	0	100

Tabulka 10 - Pracovní nabídky v USA k 10.2.2019

V tabulce č. 10 jsou uvedeny výsledky hledání pracovních nabídek v USA. Jak je vidět, Nette zde nemá žádné uplatnění. Nette framework pochází od českého vývojáře a ještě se tolik nerozšířil, aby našel uplatnění i v USA.

	Nette	Laravel
Uplatnění v ČR [%]	100	14,29
Uplatnění v USA [%]	0	100
Průměr [%]	50	57,145

Tabulka 11 - Uplatnění v ČR a USA

V tabulce č. 11 můžeme vidět průměr uplatnění frameworků pro Českou republiku a USA. Laravel v tomto srovnání tak získává lepší hodnocení, čemuž odpovídá i fakt, že je možné se s tímto frameworkem uplatit i v zahraničí. Pokud bychom hledali uplatnění pouze

[1] Výsledky byly vyhledány, ale nebyly relevantní s Nette framework

v České republice, tak bychom našli lepší uplatnění s Nette framework, avšak mimo Českou republiku je uplatnění velmi nízké, a tak z tohoto srovnání získává pouze 50 %.

5.7 Zdroje učení

Pro porovnání zdrojů učení jsem vybral hned několik způsobů porovnání. Prvním způsobem porovnání je porovnání množství dostupné literatury bez ohledu na jazyk literatury. Další porovnání bude na základě dostupných online videí, které se nacházejí na portále www.youtube.com.

Při porovnání literatury vyhrává Laravel, který k 10.2.2019 na webovém portálu www.amazon.com nabízí 75 knih, oproti tomu Nette nenabízí na tomto portálu, ani žádném jiném, jedinou knihu. V rámci literatury tak nelze k Nette sehnat žádné podpůrné materiály.

Nette framework nabízí vlastní YouTube kanál, který k 10.2.2019 má 249 nahraných videí. Sem se nahrávají jednotlivé přednášky o Nette, srazy fanoušků Nette či záznamy z různých akcí. Mimo tento kanál je na YouTube nahráno jen několik málo videí ohledně tohoto frameworku. Oproti tomu framework Laravel nabízí na internetovém serveru YouTube tisíce videí a nabízí navíc svůj vlastní portál laracast.com, kam se nahrávají videa ohledně verzí a novinek v Laravel. Z tohoto pohledu tak Laravel nabízí mnohem více videí, které může člověk vidět a z kterých se může učit. Za zmínku také stojí, že zatímco videa ohledně Nette jsou primárně v českém jazyce, videa ohledně Laravelu jsou primárně v anglickém jazyce.

Z důvodu nemožnosti přesně určit podíl jednotlivých frameworků a tím tak přidělit adekvátní počet procent, rozhodl jsem se obou frameworkům udělit 100 %. Oba frameworky nabízejí možnosti, jak získat znalosti ohledně těchto frameworků a bylo by pouze subjektivní hodnotit váhu online materiálů, kterými převládá Laravel oproti fyzickým přednáškám se záznamem o Nette.

5.8 Rychlost implementace nových technologií

Pro příklad srovnání rychlosti implementace nových technologií jsem vybral porovnání implementace podpory nejnovější verze PHP, vzhledem k tomu, že se v porovnání jedná o PHP frameworky.

Nejnovější verzí PHP k 10.2.2019 je verze 7.3, tato verze vyšla 6.12.2018. (71) Tuto verzi PHP ještě neimplementoval ani jeden framework. Aktuální verze Laravel, konkrétně

5.7 z 4.9.2018, podporuje PHP ve verzi 7.1.3. Oproti tomu aktuální verze Nette, jedná se o verzi 2.4 z roku 2016, podporuje PHP ve verzi 7.0 (z roku 2015). Když porovnáme tyto verze PHP, Laravel nabízí o rok novější verzi PHP (PHP 7.1 přišlo v roce 2016), viz kapitoly 3.3.1.8 Historie Nette a 3.3.2.9 Historie Laravel.

PHP již vyšlo ve 20-ti verzích, pro porovnání jsem tak využil počet nových verzí, které daný framework ještě neimplementoval a tím tak určit kolik % z této oblasti jaký framework dostane, viz. tabulka č. 12.

	Nette	Laravel
Neimplementováno verzí PHP	3	2
% podíl	15	10
100 - % podíl neimplementovaných verzí PHP	85	90

Tabulka 12 - Porovnání počtu neimplementovaných verzí PHP

5.9 Výsledek porovnání frameworků

	Nette	Laravel
	Výsledek [%]	Výsledek [%]
Výsledek rychlosti	100	71,75
Výsledek bezpečnosti	100	100
Výsledek Dependency injection	100	100
Výsledek počtu řádků kódu	89,03	100
Výsledek výskytu ve fórech	0,52	100
Výsledek uplatnění na trhu práce	50	57,145
Výsledek zdrojů učení	100	100
Výsledek rychlosti implementace nových technologií	85	90
∑Výsledků	624,55	718,895
Průměr ∑Výsledků	78,07	89,86

Tabulka 13 - Výsledek porovnání frameworků Nette a Laravel

Dle výsledků v tabulce č. 13 vidíme, že ve větším množství testů se lépe umístil framework Laravel. Průměrně z testování za každou oblast získal o necelých 12 % více bodů, za což vděčí převážně velké popularitě a tím i větším výskytu ve fórech a možnostech uplatnění na trhu práce. Tento výsledek však neznamená, že by své využití nenašel i framework Nette, ten získal mnohem lepší hodnocení v rychlosti, a tak v případě aplikace, pro kterou by byla zásadní rychlost, by byl lépe vyhovující.

6. Závěr

V teoretické části této práce byly charakterizovány technologie pro tvorbu webových aplikací, které byly rozděleny na Client Side a Server side technologie. Mezi Client side technologie byly zařazeny technologie HTML a CSS. V rámci Server side technologií byly poté blíže specifikovány databáze a PHP. Dále byly charakterizovány PHP frameworky Nette a Laravel.

V praktické části byly vytvořeny dvě totožné aplikace, kde každá využívala jeden z výše uvedených frameworků. Na těchto aplikacích byla poté testována rychlost jednotlivých frameworků, jejich bezpečnost a počet řádků, které bylo nutné napsat pro zprovoznění aplikace. Déle byla provedena porovnání na základě syntézy teoretických poznatků.

Z porovnání vyplývá, že pokud by vývojáři volili framework jen na základě rychlosti, lepší volbou bude zvolení frameworku Nette, který byl v testování oproti frameworku Laravel rychlejší o téměř 30 %. Z hlediska bezpečnosti oba frameworky obstáli v testu SQL injection a Cross Site scripting. Neznamená to, že by nějaký z těchto frameworků byl neprolomitelný, ale oba nabízejí určitou míru ochrany bez nutnosti zásahu vývojáře. Oba frameworky též nabízejí podporu Dependency injection, byť tuto funkcionalitu framework Laravel nazývá jinak. V čem se však frameworky již liší je množství kódu, které je nutné pro zprovoznění aplikace. Při využití frameworku Laravel je výsledný kód přibližně o 10 % kratší než při použití frameworku Nette. Ve prospěch frameworku Laravel hovoří též počet výskytů na fórech, kam se vývojáři často uchylují, jakmile řeší nějaké problémy s aplikací. Framework Laravel nabízí několikanásobně více vláken či zmínek ve fórech oproti frameworku Nette. Když se podíváme na pracovní uplatnění obou frameworků, zjistíme, že zde to již není tak jednoznačné. Framework Nette má lepší uplatnění na trhu práce v České republice, zato však nemá uplatnění v USA. Framework Laravel nabídne uplatnění jak v USA, tak i v České republice, byť v České republice je nabízeno méně pozic. Při porovnání zdrojů učení bylo zjištěno, že oba frameworky nabízejí několik způsobů, jak se daný framework naučit. Ačkoli jsou tyto způsoby rozdílné, nelze je objektivně zhodnotit, který je vhodnější z důvodu rozdílu v osobních preferencích. Framework Nette například nabízí českou dokumentaci nebo osobní setkání, při kterém se vývojáři mohou sejít a diskutovat o svých problémech s jejich aplikací s jinými vývojáři. Tuto možnost framework Laravel nenabízí, za to zas nabídne video tutoriály a své „Laracast“ videa, což

jsou videa vytvářená přímo Laravelem, která se týkají buď určité problematiky nebo příchodu nové verze frameworku a přehledu nových funkcí v této verzi. Pokud však budeme chtít na frameworku vybudovat aplikaci, kterou budeme chtít dlouhodobě udržovat, bude lepší volbou framework Laravel, který nabídne rychlejší implementaci nových technologií a tím nám umožní využívat nejnovějších možností pro tvorbu webových aplikací.

Oba frameworky nabízí několik výhod i nevýhod. Ani jeden framework není ve všech srovnávaných částech lepší. Pokud srovnáme výsledky čistě na základě číselných porovnání, lépe na tom bude framework Laravel, avšak jsou situace, kdy by bylo lepší použití frameworku Nette, jako například v aplikacích vyžadujících primárně rychlost komunikace s databází.

7. Seznam použitých zdrojů

(1) What Do Client-Side and Server-Side Mean? | Client Side vs. Server Side. *Cloudflare* [online]. b.r. [cit. 2019-02-07]. Dostupné z: https://www.cloudflare.com/learning/serverless/glossary/client-side-vs-server-side/

(2) BROWN, Tiffany, Kerry BUTTERS a Sandeep PANDA. *HTML5 Okamžitě*. Brno: Computer Press, 2014. ISBN 978-80-251-4296-7.

(3) HTML & CSS. *W3.org* [online]. b.r. [cit. 2018-08-13]. Dostupné z: https://www.w3.org/standards/webdesign/htmlcss

(4) Kaskádové styly. *W3.org* [online]. b.r. [cit. 2018-06-15]. Dostupné z: https://www.w3.org/Style/CSS/Overview.cs.html.

(5) Databáze. *Adaptic.cz* [online]. b.r. [cit. 2018-08-22]. Dostupné z: http://www.adaptic.cz/znalosti/slovnicek/databaze/

(6) PANYKO, Tomáš. *NoSQL databáze*. České Budějovice, 2013. Bakalářská práce. Jihočeská univerzita v Českých Budějovicích. Vedoucí práce RNDr. Hana Havelková.

(7) Návrh databáze – NoSQL vs SQL. *Zdroják.cz* [online]. b.r. [cit. 2019-02-07]. Dostupné z: https://www.zdrojak.cz/clanky/navrh-databaze-nosql-vs-sql/

(8) WELLING, Luke a Laura THOMSON. *Mistrovství PHP a MySQL*. Brno: Computer Press, 2017. ISBN 978-80-251-4892-1.

(9) Advantages & Disadvantages. *Cs.iit.edu* [online]. b.r. [cit. 2018-08-15]. Dostupné z: http://www.cs.iit.edu/~cs561/cs425/VenkatashSQLIntro/Advantages%20&%20Disadvantages.html

(10) BORONCZYK, Timothy. *MySQL Okamžitě*. Brno: Computer Press, 2016. ISBN 978-80-251-4737-5.

(11) HOPKINS, Callum. *PHP Okamžitě*. Brno: Computer Press, 2014. ISBN 978-80-251-4196-0.

(12) PHP /základy/. *Tvorba-webu.cz* [online]. b.r. [cit. 2019-02-01]. Dostupné z: https://www.tvorba-webu.cz/php/

(13) Historical trends in the usage of server-side programming languages for websites. In: *W3techs.com* [online]. b.r. [cit. 2018-06-22]. Dostupné z: https://w3techs.com/technologies/history_overview/programming_language

(14) What Is a WAMP, MAMP, or LAMP?. *Safaribooksonline.com* [online]. b.r. [cit. 2018-08-13]. Dostupné z: https://www.safaribooksonline.com/library/view/learning-php-mysql/9781449337452/ch02s01.html

(15) PHP -- Jak začít. *Jakpsatweb.cz* [online]. b.r. [cit. 2019-02-07]. Dostupné z: https://www.jakpsatweb.cz/php/jak-zacit.html

(16) Lekce 4 - Textové řetězce podruhé a pole v PHP. *ITnetwork.cz* [online]. b.r. [cit. 2019-02-01]. Dostupné z: https://www.itnetwork.cz/php/zaklady/zaklady-php-tutorial-escapovani-a-pole

(17) Lekce 5 - Asociativní pole v PHP a obsluha formulářů. *ITnetwork.cz* [online]. b.r. [cit. 2019-02-07]. Dostupné z: https://www.itnetwork.cz/php/zaklady/zaklady-php-asociativni-pole-obsluha-formularu

(18) PHP - Formuláře. *Tvorba-webu.cz* [online]. b.r. [cit. 2019-02-07]. Dostupné z: https://www.tvorba-webu.cz/php/formulare.php

(19) Lekce 6 - Podmínky v PHP. *ITnetwork.cz* [online]. b.r. [cit. 2019-02-07]. Dostupné z: https://www.itnetwork.cz/php/zaklady/zaklady-php-podminky-if-else

(20) CYKLUS. *IT SLOVNÍK.CZ* [online]. b.r. [cit. 2019-02-07]. Dostupné z: https://it-slovnik.cz/pojem/cyklus

(21) While. *Php.net* [online]. b.r. [cit. 2019-02-07]. Dostupné z: http://php.net/manual/en/control-structures.while.php

(22) For. *Php.net* [online]. b.r. [cit. 2019-02-07]. Dostupné z: http://php.net/manual/en/control-structures.for.php

(23) Foreach. *Php.net* [online]. b.r. [cit. 2019-02-07]. Dostupné z: http://php.net/manual/en/control-structures.foreach.php

(24) History of PHP. *Php.net* [online]. b.r. [cit. 2018-06-24]. Dostupné z: http://php.net/manual/en/history.php.php

(25) What happened to PHP 6?. *Phproundtable.com* [online]. b.r. [cit. 2018-06-24]. Dostupné z: https://www.phproundtable.com/episode/what-happened-to-php-6

(26) PHP7: deset věcí, které o něm potřebujete vědět. *Interval.cz* [online]. b.r. [cit. 2018-06-24]. Dostupné z: https://www.interval.cz/clanky/php-7-deset-veci-ktere-o-nem-potrebujete-vedet/

(27) Co je to PHP. *Helpmark.cz* [online]. b.r. [cit. 2018-08-15]. Dostupné z: https://www.helpmark.cz/slovnikpojmu/49-php

(28) Je PHP jazyk pro amatéry?. *Root.cz* [online]. b.r. [cit. 2018-08-15]. Dostupné z: https://blog.root.cz/mystik/je-php-jazyk-pro-amatery/

(29) PHP frameworky. *Programujte.com* [online]. b.r. [cit. 2018-08-18]. Dostupné z: http://programujte.com/clanek/2008022000-php-frameworky/

(30) Most Used PHP Frameworks in 2017. *Medium.com* [online]. b.r. [cit. 2018-06-29]. Dostupné z: https://medium.com/@vishva.eleganzit/most-used-php-frameworks-in-2017-73572e562fe9

(31) David Grudl: Přednáška na VUT Brno 2011. In: *Youtube.com: Kanál: Nette Framework* [online]. b.r. [cit. 2018-08-18]. Dostupné z: https://www.youtube.com/watch?v=0yPsGysbj2U&t=2632s

(32) Should You Use a PHP Framework? Five Pros and Cons. *Envatotuts+* [online]. b.r. [cit. 2019-02-07]. Dostupné z: https://code.tutsplus.com/tutorials/should-you-use-a-php-framework-five-pros-and-cons--cms-28905

(33) *Nette.org* [online]. b.r. [cit. 2018-08-20]. Dostupné z: https://nette.org/

(34) The Best PHP Framework for 2015: SitePoint Survey Results. *Sitepoint.com* [online]. b.r. [cit. 2018-08-20]. Dostupné z: https://www.sitepoint.com/best-php-framework-2015-sitepoint-survey-results/

(35) *Posobota.cz* [online]. b.r. [cit. 2018-08-20]. Dostupné z: https://www.posobota.cz/

(36) Začínáme. *Nette.org* [online]. b.r. [cit. 2018-08-23]. Dostupné z: https://doc.nette.org/cs/2.4/quickstart/getting-started

(37) Download. *Nette.org* [online]. b.r. [cit. 2018-08-23]. Dostupné z: https://nette.org/cs/download

(38) Requirements Checker. *Nette* [online]. b.r. [cit. 2019-02-07]. Dostupné z: https://doc.nette.org/cs/2.4/requirements

(39) MVC aplikace & presentery. *Nette* [online]. b.r. [cit. 2018-08-23]. Dostupné z: https://doc.nette.org/cs/2.2/presenters

(40) Nette Database vs dibi. *Planette* [online]. b.r. [cit. 2019-02-01]. Dostupné z: https://pla.nette.org/cs/nette-database-vs-dibi

(41) TRACY. *Nette* [online]. b.r. [cit. 2019-02-01]. Dostupné z: https://tracy.nette.org/cs/

(42) David Grudl: Nette Framework poprvé na konferenci PHP frameworky 2007. *Youtube.com: Kanál: Nette Framework* [online]. b.r. [cit. 2018-08-23]. Dostupné z: https://www.youtube.com/watch?v=VPU1dETzCJI

(43) Sbohem a šáteček, Nette 2.0 & PHP 5.2. *Phpfashion.com* [online]. b.r. [cit. 2018-08-23]. Dostupné z: https://phpfashion.com/sbohem-a-satecek-nette-2-0-php-5-2

(44) Přejděte na Nette 2.1. *Phpfashion.com* [online]. b.r. [cit. 2018-08-23]. Dostupné z: https://phpfashion.com/prejdete-na-nette-2-1

(45) Nette Revolution 2.2. *Phpfashion.com* [online]. b.r. [cit. 2018-08-23]. Dostupné z: https://phpfashion.com/nette-revolution-2-2

(46) Nette 2.3 bude trošku citlivka. *Phpfashion.com* [online]. b.r. [cit. 2018-08-23]. Dostupné z: https://phpfashion.com/nette-2-3-bude-trosku-citlivka

(47) Nette 2.3.0 beta for testing. *Forum.nette.org* [online]. b.r. [cit. 2018-08-23]. Dostupné z: https://forum.nette.org/en/22055-nette-2-3-0-beta-for-testing

(48) David Grudl - následník Nette/Object. In: *Youtube.com: Kanál: Nette Framework* [online]. b.r. [cit. 2018-08-23]. Dostupné z: https://www.youtube.com/watch?v=WOEmbEL9N6c

(49) Nette Framework. *Wikipedie* [online]. b.r. [cit. 2019-02-07]. Dostupné z: https://cs.wikipedia.org/wiki/Nette_Framework#Nev%C3%BDhody_Nette_Framework

(50) Co je Laravel?. *Laravelblog.cz* [online]. b.r. [cit. 2018-08-22]. Dostupné z: http://laravelblog.cz/co-je-laravel/

(51) Laravel vs. Nette—tři měsíce poté. *Blog.ikw.cz* [online]. b.r. [cit. 2018-08-22]. Dostupné z: https://blog.ikw.cz/laravel-vs-nette-t%C5%99i-m%C4%9Bs%C3%ADce-pot%C3%A9-78f1200497c9

(52) Release Notes. *Laravel.com* [online]. b.r. [cit. 2018-08-22]. Dostupné z: https://laravel.com/docs/5.7/releases

(53) Installation. *Laravel* [online]. b.r. [cit. 2019-02-07]. Dostupné z: https://laravel.com/docs/5.7

(54) Database: Getting Started. *Laravel* [online]. b.r. [cit. 2019-02-01]. Dostupné z: https://laravel.com/docs/5.7/database

(55) Controllers. *Laravel* [online]. b.r. [cit. 2019-02-01]. Dostupné z: https://laravel.com/docs/5.7/controllers

(56) Whoops is coming back in Laravel 5.5. *Laravel News* [online]. b.r. [cit. 2019-02-01]. Dostupné z: https://laravel-news.com/whoops-laravel-5-5

(57) Directory Structure. *Laravel* [online]. b.r. [cit. 2019-02-01]. Dostupné z: https://laravel.com/docs/5.7/structure#the-public-directory

(58) Artisan Console. *Laravel* [online]. b.r. [cit. 2019-02-07]. Dostupné z: https://laravel.com/docs/5.7/artisan

(59) Kde se Laravel, tu se Laravel…. *Medium.com* [online]. b.r. [cit. 2018-08-22]. Dostupné z: https://medium.com/@laravue/kde-se-laravel-tu-se-laravel-c315b5fea541

(60) Release Notes. *Laravel.com* [online]. b.r. [cit. 2018-08-22]. Dostupné z: https://laravel.com/docs/5.2/releases

(61) Release Notes. *Laravel.com* [online]. b.r. [cit. 2018-08-22]. Dostupné z: https://laravel.com/docs/5.3/releases

(62) Laravel 5.4 přichází!. *Laravelblog.cz* [online]. b.r. [cit. 2018-08-22]. Dostupné z: http://laravelblog.cz/laravel-5-4-prichazi/

(63) Release Notes. *Laravel.com* [online]. b.r. [cit. 2018-08-22]. Dostupné z: https://laravel.com/docs/5.5/releases

(64) Laravel 5.6 byl zveřejněn!. *Codefirst.cz* [online]. b.r. [cit. 2018-08-22]. Dostupné z: https://codefirst.cz/clanky/6/laravel-5-6-byl-zverejnen

(65) Advantages and Disadvantages of Laravel. *Softwaredeveloperindia* [online]. b.r. [cit. 2019-02-07]. Dostupné z: https://www.software-developer-india.com/advantages-and-disadvantages-of-laravel/

(66) SQL vs. XXS Injection Attacks Explained. *Keirstenbrager* [online]. b.r. [cit. 2019-02-10]. Dostupné z: https://keirstenbrager.tech/sql-vs-xxs-injection-attacks-explained/

(67) David Grudl: Dependency Injection [Posobota 38]. In: *YouTube: Kanál: Nette Framework* [online]. b.r. [cit. 2019-02-11]. Dostupné z: https://www.youtube.com/watch?v=ODcfsRpQ0Pw&t=1931s

(68) Dependency Injection. *Nette* [online]. b.r. [cit. 2019-02-11]. Dostupné z: https://doc.nette.org/cs/2.4/dependency-injection

(69) Service Container. *Laravel* [online]. b.r. [cit. 2019-02-11]. Dostupné z: https://laravel.com/docs/5.7/container

(70) Výsledky: Technologie na českém webu. *Zdroják.cz* [online]. b.r. [cit. 2019-02-11]. Dostupné z: https://www.zdrojak.cz/clanky/vysledky-technologie-na-ceskem-webu/

(71) PHP 7.3 features and release date. *Symfony Finland* [online]. b.r. [cit. 2019-02-11]. Dostupné z: https://symfony.fi/entry/php-7-3-features-and-release-date

7.1 Seznam tabulek

7.2 Seznam obrázků

8. Přílohy

Příloha 1 ……... data.xslx

Příloha 2 ……. Nette.jmx

Příloha 3 …….. Laravel.jmx

Příloha 4 …….. aplikace.zip