Univerzita Hradec Králové
Fakulta informatiky a managementu
Katedra informačních technologií

Porovnání PHP Frameworků
Bakalářská práce

Autor: Patrik Štípek

Obor: AI3

Vedoucí práce: Mgr. Daniela Ponce, Ph.D.

Hradec Králové duben 2019

Prohlášení:

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně a s použitím uvedené literatury.

V Hradci Králové dne: Patrik Štípek

Poděkování

Chtěl bych poděkovat vedoucí práce paní Mgr. Daniele Ponce Ph.D. za cenné rady, inspiraci a čas, který mi věnovala.

Anotace

Tato bakalářská práce se zabývá představením a porovnáváním PHP frameworků Nette, Laravel a Symfony. Porovnání je zaměřeno na hlavní vlastnosti frameworků a je hodnoceno podle vybraných kritérií. Frameworky jsou představeny a jsou sepsány jejich silné a slabé stránky. Pro praktické porovnání je v každém frameworku vytvořena aplikace, která je následně podrobena několika výkonnostním testům a výsledky těchto testů slouží pro hodnocení.

Klíčová slova

Framework, MVC, PHP, Composer, engine, ORM, CRUD

Annotation

This bachelor thesis deals with the introduction and comparison of PHP frameworks Nette, Laravel and Symfony. The comparison is focused on the main features of frameworks and is evaluated according to selected criteria. Frameworks are introduced and their strengths and weaknesses are written. For practical comparison an application is created in each framework, which is then subjected to several performance tests. The results of these tests are used for evaluation.

Keywords

Framework, MVC, PHP, Composer, engine, ORM, CRUD

Obsah

1. ÚVOD

V posledních letech se webové aplikace rozšířily ve velkém. Dnes na ně člověk narazí skoro pokaždé. S dnešní dostupností informací není problém takovou aplikaci vytvořit. Pro tvorbu webové aplikace je potřeba mít znalost některého z webových programovacích jazyků. Avšak v dnešní době webová aplikace potřebuje spoustu dalších částí, aby byla bezpečná a snadno vyhledatelná na internetu.

Jelikož si člověk chce svou práci co nejvíce ulehčit, tak vytvořil technologie, které mají za úkol zrychlit, ulehčit a ucelit postup při tvorbě aplikace. S postupem času jsou na aplikace kladeny větší nároky, než tomu bylo dřív, a proto vznikly tzv. frameworky, které webovým vývojářům usnadňují práci.

V dnešní době existuje mnoho frameworků pro tvorbu moderních webových aplikací a neustále vznikají další a další. V této práci jsou popsány a zhodnoceny tři takovéto frameworky.

1.1. Cíl bakalářské práce

Cílem bakalářské práce je představit a porovnat PHP frameworky Laravel, Symfony a Nette. Popsat jejich hlavní vlastnosti a najít jejich silné a slabé stránky. Zjistit, jak jsou tyto frameworky bezpečné. V praktické části vytvořit aplikace v každém frameworku, které budou obsahovat základní aplikační funkcionalitu dnešních webových aplikací. Při vývoji aplikace zjistit jejich složitost a uživatelskou přívětivost. Následně podrobit každou aplikaci sérii několika testů. Díky získaným bodům z praktické a teoretické části bude zvolen nejvhodnější framework pro tvorbu webové aplikace. Nicméně práce by měla čtenáře seznámit s těmito frameworky a měla by mu poskytnout základní informace, podle kterých si bude moct zvolit framework, který je pro něj nejvhodnější.

1.2. Struktura práce

Práce je systematicky rozdělena do několika částí. V první části práce jsou sepsány kritéria hodnocení. U každého kritéria je stručně popsán důvod výběru. V další části jsou popsány webové technologie a čtenář je seznámen s pojmem framework. Následně je každý framework představen a u každého jsou popsány hodnotící vlastnosti. V poslední části je

popsána tvorba webové aplikace a tato aplikace otestována. Výsledky z testů jsou zobrazeny v grafech v poslední části.

2. Analýza současných hodnocení

Na internetu existuje spousta prací, diskuzí, článků, nebo i videí, kde jsou popisovány a hodnoceny aktuální PHP frameworky. V některých článcích je vybrán nejlepší framework pro tvorbu aplikace. Hodnocení se dá rozdělit na dvě skupiny. Do první skupiny patří hodnocení jednoho frameworku. Autor si vybere jeden framework, který hodnotí a testuje. Druhá skupina obsahuje hodnocení, ve kterých autor vybere několik frameworků a ty navzájem porovnává.

Jako příklad první skupiny lze uvést práci o frameworku Laravel od Lukáše Fialy [1]. V práci detailně popisuje framework Laravel, který zároveň srovnává s Nette a Symfony.

Porovnávání více frameworků záleží na zadaných kritériích. Ve webových článcích autoři často vyberou 10 frameworků a ty seřadí od jedničky po desítku. Ke každému frameworku napíší základní informace, vlastnosti a někdy i plusy a mínusy. Pořadí většinou vytvářejí podle oblíbenosti, počtu vytvořených projektů a nabízených funkcí.

Jako příklad lze uvést práci od Jiřího Rebendy [2] o porovnání frameworků Phalcon, Nette a Zend. Autor v poměrně rozsáhlé práci hodnotí tři frameworky podle stanovených kritérií.

3. Kritéria porovnání

V této části jsou popsány kritéria hodnocení porovnávaných frameworků. U každého kritéria je popsán důvod výběru a metoda hodnocení. Frameworky jsou za každé kritérium hodnoceny body od 0 do 2, kde 2 je nejvíce bodů. Pokud framework má velké slabiny či nedostatky u daného kritéria, nebo nesplňuje požadavky kritéria, tak nedostane žádný bod, ovšem pokud je u daného kritéria vše v pořádku, tak framework dostane plný počet bodů. U vybraných kritérií je bodové rozmezí od 0 do 4, jelikož některé oblasti jsou důležitější a je potřeba to zohlednit v konečném výsledku. Tyto kritéria mají v popisu poznamenáno, že jsou hodnoceny 4 body.

3.1. Systémové požadavky

Každý framework má své požadavky pro svůj chod. Je zde popsáno, jaké požadavky každý framework potřebuje a zda vyžaduje nějaký speciální HW, nebo SW. Dále se zjišťuje, s jakými verzemi PHP je framework kompatibilní.

3.2. Licence

Jedním z kritérií hodnocení je také dostupnost a možnost použití frameworku, jakou licenci framework používá a za jakých podmínek je možné sw měnit či šířit jeho kopie.

3.3. Dokumentace

Pro pochopení a naučení frameworku a jeho následné použití je velice důležitá dokumentace, ve které jsou popsány všechny vlastnosti a funkce frameworku a jsou zde uvedeny ukázky kódu a návody správného použití.

3.4. Aktualizace

Dalším kritériem je aktuálnost daného frameworku: zda framework vydává pravidelně aktualizace, zda používá nejnovější postupy a technologie ve webovém vývoji.

3.5. Šablonovací systém

Šablonovací systém zvyšuje bezpečnost webové aplikace a měl by udržovat přehledný kód. Omezuje psaní čistého PHP kódu ve views a zamezuje tím vznik takzvaného „špagety kódu“.

Hodnoceno je, zda framework používá šablonovací systém, co šablonovací systém nabízí a jak se s ním pracuje.

3.6. Konzole

Vytvářet aplikaci s pomocí konzole dokáže velmi usnadnit práci. Vývojář může pomocí konzole vytvářet třídy, nebo manipulovat s databází. V tomto kritériu se hodnotí, zda framework umožňuje používat konzoli. Jaké příkazy nabízí a jak se s konzolí pracuje.

3.7. Routování

Jedním z posledních trendů je použití routeru. Ve webové aplikaci lze pomocí routeru definovat cesty mezi controllerem a view. Tyto cesty lze snadno vytvářet, upravovat a jsou znovupoužitelné.

3.8. Složitost

Náročnost na naučení a složitost, je také jedním z kritérií hodnocení. I když je tato část spíše subjektivní, je důležité vědět, jak je náročné začít efektivně a správně vytvářet aplikaci v daném frameworku, ať pro nového, nebo zkušeného programátora.

3.9. Podpora databází

Jedním z hlavních bodů hodnocení je, jak daný framework pracuje s databází. Jestli podporuje použití migrací, jakou techniku používá pro práci s databází, jaké dotazy dovoluje použít a jaké databáze podporuje. Jelikož dnes s databází pracuje skoro každá aplikace nebo i služba a uchovává si v ní data. Tak je důležité, aby framework podporoval větší množství databází, proto toto kritérium je hodnoceno 4 body.

3.10. Bezpečnost

Jedním z hlavních účelů frameworku, je zabezpečit aplikaci před možnými útoky a ulehčit tak práci vývojáři se zabezpečením. Tento bod je jedním z nejdůležitějších kritérií, jelikož vlastník aplikace chce, aby aplikaci měl plně pod kontrolou a každý uživatel si v ní nemohl dělat co chce a hlavně, aby nejenom citlivá data jeho uživatelů, ale i ostatní informace o aplikaci byla v bezpečí a nikdo je nemohl zneužít. Tato část bude hodnocena podle způsobu zabezpečení a rozsahu dostupné ochrany. Toto kritérium je hodnoceno 4 body.

3.11. Autentizace

V dnešní době většina webových aplikací ukládá údaje o svých uživatelích a jen přihlášeným uživatelům dovoluje používat aplikaci. Pro každou takovou aplikaci je důležitá část přihlášení a registrace uživatelů. Proto je jedním z kritérií hodnocení, kde se zjišťuje, jestli framework pomáhá, či dokonce umožňuje vytvořit přihlašování a registraci uživatele v aplikaci.

3.12. Rychlost

Rychlost byla vždy na internetu jedna z nejdůležitějších vlastností webové stránky či aplikace. V dnešní době většina uživatelů opouští webovou stránku, pokud načítání trvá moc dlouho. Proto rychlost frameworků bude jedním z klíčových hodnotících kritérií a bude hodnoceno 4 body. Rychlost frameworku bude měřena v praktické části této práce.

3.13. Rychlost CRUD operací

V dnešní době snad každá webová aplikace používá databázi. Ukládá si tam data o uživateli, nebo o čemkoliv jiném a pro rychlý běh aplikace je důležité, aby i ve velkém provozu, kdy aplikaci může využívat stovky uživatelů, pracovala s databází rychle. Proto patří toto kritérium mezi ty nejdůležitější. V praktické části je několik testů, kde se měří rychlost crud operací v relačních databázích. Následně jsou výsledky porovnány s ostatními frameworky. Toto kritérium je hodnoceno 4 body.

3.14. Pracovní nabídky

V posledním kritériu jsou porovnány pracovní nabídky všech frameworků na českém trhu podle dat z portálu Indeed [3] pro region Praha.

4. Technologie

V této kapitole jsou představeny hlavní technologie, které jsou v této práci používány.

4.1.PHP

PHP (Hypertext Preprocessor) je server-side skriptovací programovací jazyk, který se používá pro tvorbu dynamických webových stránek a aplikací. *PHP vyvinul Rasmus Lerdorf v roce 1994 pod názvem Personal Home Page Tools (PHP Tools). První byla vydána v roce 1995. Umožňovala používat databáze a bylo možné vytvořit dynamické webové stránky.* [4] Verze 7.0 byla vydána 3. prosince 2015. Vydání verze 7.3 je očekáváno dne 6. prosince 2018.

PHP provádí veškeré operace na straně serveru. Proto není možné vidět zdrojový kód v prohlížeči. JavaScript na rozdíl od PHP provádí operace na klientské straně, takže jeho kód v prohlížeči zobrazit lze. Syntaxe jazyka je inspirována několika programovacími jazyky například Perl, C a Pascal. Jazyk je nezávislý na operačním systému, fungování na jiných operačních systémech omezuje jen několik funkcí.

Pro programování v PHP je potřeba mít webový server. Nejlepší řešení je použití serveru Apache. Pro Windows existuje řada aplikací, které vytvoří webový server lokálně a lze je zdarma používat. Nejznámější jsou aplikace XAMP a WAMP. Při vložení aplikace na internet tyto potřeby zajistí hosting. Dnes podporují PHP i hostingy, které jsou zdarma.

Jazyk PHP se stal velice oblíbeným programovacím jazykem. Používají ho i weby jako je Wikipedie nebo Facebook. Server w3techs.com uvádí: „*V dnešní době PHP používá 78.9 % všech webových stránek, které používají server-side programovací jazyk.*„ [5]

4.2. Framework

Framework je softwarová struktura, která slouží jako podpora při programování a vývoji a organizaci jiných softwarových projektů. Může obsahovat podpůrné programy, knihovnu API, návrhové vzory nebo doporučené postupy při vývoji. [6]

Použití frameworku přináší vývojáři přehledný a znovupoužitelný kód. To je důležité zejména při tvorbě aplikace v týmu a u větších aplikací. Pokud framework je stavěn na MVC architektuře (MVC architektura je popsána v bodě 4.5), tak je vývojář nucen psát aplikaci objektově. Využití frameworků má také své nevýhody. Framework vývojáři říká, jak má být aplikace strukturovaná a pokud se nejedná o tzv. lehký framework, tak je framework v některých oblastech velice omezující. Dalším nedostatkem je jeho velikost. Čím více má

framework funkcionalit a komponent, tím zároveň roste i jeho velikost a jeho nároky na paměť.

4.3. Účel

Při vývoji webové aplikace je potřeba naprogramovat spoustu funkcionalit, které jsou potřeba v každé aplikaci a které zaberou spoustu času. Také se pro vývojáře tato část vývoje stane otravnou rutinou. Proto vznikla potřeba vyvinout nástroj, který tyto problémy vyřeší. Řešením je použití frameworku.

MVC framework poskytuje dobře organizovaný, znovupoužitelný a čistý kód. Jeho použití ušetří mnoho času a urychlí vývoj aplikace. Programátor se částečně zbaví potřeby zabezpečení aplikace, jelikož frameworky mají zabezpečení v sobě. MVC frameworky zajišťují oddělení logiky a prezentace, což zpřehledňuje kód a používají aktuální trendy, mezi které patří objektové programování. Dále umožňuje růst aplikace v průběhu času, jelikož frameworky jsou škálovatelné.

Výhodou frameworku je ušetření času. Funkce a časté use case jako například přihlašování uživatelů nebo obsluha formulářů lze (u některých frameworků) automaticky vygenerovat, nebo framework obsahuje zabezpečení formulářů. Vývojář nemusí řešit všechno zabezpečení aplikace. [7]

4.4. Rozdíl mezi knihovnou a frameworkem

Někteří uživatelé ani neví, jaký je rozdíl mezi frameworkem a knihovnou. Hlavní rozdíl je v kontrole nad projektem. Framework kontroluje aplikaci a říká programátorovi jakou bude mít aplikace architekturu a jakým způsobem bude fungovat. Naopak knihovna obsahuje třídy a funkce, které může programátor použít jak chce. Programátor tak má plnou kontrolu nad projektem.

4.5.MVC

MVC (Model View Controller) je návrhový vzor, tedy objektová struktura, jež odděluje data a jejich zpracování od jejich zobrazení. Model představuje datové úložiště, obstarává získání dat a práci s nimi a vrací je controlleru, který si o ně zažádal, controller je následně předá view, který je jakýmsi způsobem zobrazí. [6] I když je tento návrhový vzor už poměrně starý a existují novější alternativy, tak je stále velice používaný. Pomáhá vytvářet objektovou

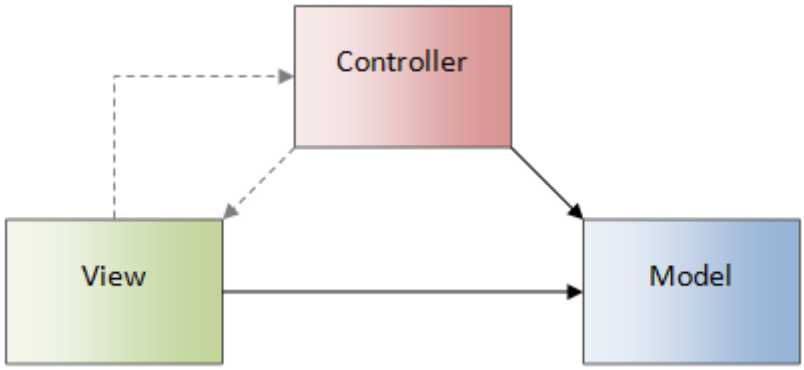
strukturu aplikace a zlepšuje přehlednost kódu. Tím je umožněno aplikaci jednoduše udržovat a rozšiřovat bez velkých dopadů na zbytek aplikace.

Model MVC se rozděluje na 3 části. Model je zobrazen na obrázku 1.

Model – má odpovědnost za stav celé aplikace. Obsahuje data aplikace, logiku a vše, co do ní spadá. Mohou to být výpočty, databázové dotazy. Jeho funkce spočívá v přijetí parametrů zvenku a vydání dat ven. Model neví, odkud data v parametrech přišla a ani jak budou výstupní data zformátována a vypsána. [8]

View – se stará o zobrazení výstupu uživateli. Nejčastěji se jedná o šablonu obsahující HTML stránku, která umožňuje vkládat do šablony proměnné, podmínky a cykly. View není jen šablona, ale zobrazovač výstupu. Obsahuje tedy minimální množství logiky, která je pro výpis nutná. View stejně jako model o datech neví, odkud se vzaly, stará se jen o zobrazení dat uživateli. [8]

Controller – propojuje komponenty (model a view) a drží celý systém pohromadě. Komunikuje jak s modelem a view, tak i s uživatelem, který přes něj ovládá aplikaci. Funkce controlleru je posílat data modelu a následně je přeposlat view. [8]



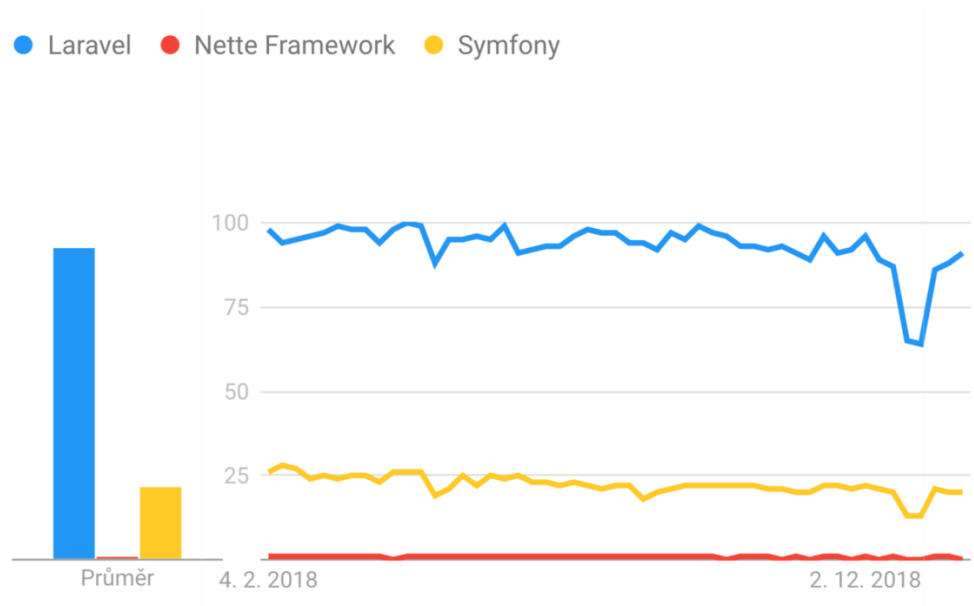
Obrázek 1 Architektura MVC [9]

5. Představení porovnávaných frameworků

Dnes existuje mnoho PHP frameworků, které může kdokoliv použít. Vlastně pořád vznikají nové a nové. V této kapitole jsou představeny tři frameworky, které používají architekturu MVC. Frameworky jsou krátce popsány, jsou vypsány základní vlastnosti a silné a slabé stránky. První framework je poměrně nový, ale velice oblíbený framework Laravel. Druhý framework je český Nette Framework, který je také hodně oblíbený a používaný zejména v České republice a na Slovensku. Třetí framework je Symfony, na kterém je částečně založen první framework a obsahuje mnoho znovupoužitelných komponent.

5.1.Popularita

Podle mnoha webů, které sestavují seznamy nejlepších frameworků dnešní doby, je skoro pokaždé na první příčce Laravel. Symfony se pohybuje mezi prvními třemi. Naopak Nette není ve světě tak rozšířen a mimo rok 2015, kdy byl vyhodnocen na stránce sitepoint.com [10] jako třetí nejpopulárnější framework, v žebříčkách není vidět. Podle dat získaných na stránkách google trends [11], lze na obrázku 2 vidět, jak často byly v průběhu roku 2018 vyhledávány.



Obrázek 2 Poměr vyhledávání v roce 2018 [11]

Nejvíce byl za poslední rok vyhledáván Laravel. Symfony uživatelé vyhledávali zejména ve Francii a francouzsky mluvících zemích. Framework Nette, který má nejmenší počet vyhledávání byl vyhledáván hlavně v Česku a na Slovensku.

5.2. Laravel

5.2.1. Popis

Je open source PHP framework vydaný v roce 2011 Taylorem Otwellem. Laravel byl vyvinut za účelem vývoje aplikací založených na principu MVC. [12] Přestože Laravel je velice nový framework, tak se podle mnoha stránek stal v poslední době nejoblíbenějším PHP frameworkem. Podle počtu vytvořených projektů v roce 2018 lze zjistit, že popularita tohoto frameworku neustále stoupá.

Obsahuje rozsáhlou dokumentaci a velikou základnu vývojářů, která se stále rozrůstá. Oficiální web obsahuje mnoho návodů ve formě screencastů, jimž se říká Laracasty. Na internetu lze nalézt také mnoho návodů a tutoriálů. Disponuje obrovskou škálou schopností, které umožňují rychlý vývoj. Má svůj šablonovací systém s názvem „Blade“, který přispívá ke snadné čitelnosti kódu. Umožňuje použití migrací při práci s databází, které pomáhají při práci v týmu udržovat přehled o databázi.

Umožňuje vygenerování základních funkcionalit webové aplikace, mezi které patří přihlašování, registrace a obnova hesla.

Pořádají se také konference nazvané Laracon. Hovoří se zde o frameworku a dalších technologiích. Konají se každoročně v USA a Evropě.

5.2.2. Stručná historie

Laravel vytvořil Taylor Otwell jako pokus poskytnout pokročilejší variantu frameworku CodeIgniter, která neposkytovala určité funkce, jako je vestavěná podpora pro autentizaci a autorizaci uživatelů. První beta vydání bylo představeno 9. června 2011 a ve stejném měsíci také následovala první verze Laravelu. [13]

Druhá verze byla vydána v září roku 2011 a přinesla různá zlepšení. Mezi hlavní nové funkce patří podpora řadičů, které z Laravelu vytvořily framework plně kompatibilní s MVC, vestavěnou podporu principu inverze řízení (IoC) a šablonovací engine s názvem Blade. Nevýhodou v Laravelu 2 bylo odstranění podpory balíků třetích stran. [13]

Laravel 3 byl vydán v únoru 2012 se sadou nových funkcí včetně rozhraní příkazového řádku (CLI) s názvem Artisan, vestavěnou podporou pro další správu databáze, migrace databází, podpora pro manipulaci s událostmi a balící systém s názvem Bundles. Zvýšení uživatelské základny a popularity se spojilo s vydáním Laravelu 3. [13]

Laravel 4, s označením Illuminate, byl vydán v květnu 2013. V této verzi byl přepsán celý framework a přeměnilo se jeho rozložení na soubor samostatných balíčků distribuovaných prostřednictvím nástroje Composer, který slouží jako správce balíčků na úrovni aplikací. Taková změna rozložení zlepšila rozšiřitelnost Laravelu. [13]

Laravel 5 byl vydán v únoru 2015. Mezi nové funkce v Laravelu patřila abstrakční vrstva nazvaná Flysystem, služba Elixir a zjednodušené externě zpracované autentizace prostřednictvím volitelného balíčku Socialite. [14]

Pro listopad 2018 je aktuální verze 5.7, která byla vydána 6. září 2018. Opravuje chyby z minulé verze, přichází nepovinné emailové ověření, vylepšení testování v konzoli a další. Verze 5.5 a 5.6 jsou verze, u kterých jsou opravovány bezpečnostní chyby. Všechny starší verze jsou již bez podpory. [15]

5.2.3. Licence

Laravel je open source framework pod licencí MIT, která vznikla na Massachusettském technologickém institutu a je velice podobná licenci BSD. *Tato licence umožňuje uživateli se softwarem nakládat téměř libovolně (používat, kopírovat, modifikovat, slučovat, publikovat, distribuovat či prodávat), jedinou podmínkou je zahrnutí textu licence do všech kopií a odvozenin softwaru.* [16] Mezi další software, který využívá MIT licenci je například operační systém Windows 10.

5.2.4. Požadavky

Pro provoz Laravelu je potřeba mít PHP verzi 5.4, nebo vyšší a MCrypt rozšíření. Pro aktuální verzi 5.7 je potřeba mít verzi PHP 7.1.3, nebo vyšší a PHP rozšíření: OpenSSL, PDO, Mbstring, Tokenizer, XML, Ctype a JSON. [17] Dále je potřeba mít nainstalovaný sw pro správu balíčků Composer.

5.2.5. Instalace

Laravel lze nainstalovat několika způsoby. Nejjednodušší způsob instalace je pomocí Composeru.

Pokud je Laravel instalován poprvé, tak ho lze naistalovat přes Composer následujícím příkazem.

```
composer global require "laravel/installer"
```

Následně lze vytvářet nové projekty příkazem, který je zobrazen dále.

```
laravel new project-name
```

Další možnost, jak vytvořit nový projekt je použít příkaz níže.

```
composer create-project --prefer-dist laravel/laravel project-name
```

Tyto dvě možnosti jsou popsány v dokumentaci. [17]

5.2.6. Hlavní vlastnosti

Laravel obsahuje tuny specifických balíků, jednoduchý šablonovací engine Blade, jednotkové testování, ORM, packaging system, RESTful controllers a nyní je Laravel prvním frameworkem, který zavádí směrování abstraktním způsobem. Což zmenšuje potíže s organizací kódu. [18]

5.2.6.1. Artisan

Artisan je command-line interface, který je součástí Laravelu. Poskytuje řadu užitečných příkazů, které mohou výrazně pomoci při vytváření aplikace. Pro zobrazení seznamu všech dostupných příkazů stačí zadat příkaz `list.` [19]

Každý příkaz obsahuje také vysvětlení, které popisuje dostupné argumenty a možnosti příkazu. Pro zobrazení nápovědy, stačí napsat `help` před název příkazu.

Vybrané artisan příkazy:

Příkaz	**Funkcionalita**
make:auth	Vytvoří views, controllery a routes pro přihlášení a registraci uživatelů.
serve	Zprovozní aplikaci na localhostu. Obvykle na portu 8000.
make:controller	Vytvoří novou třídu pro controller.
make:migration	Vytvoří nový soubor pro definování migrace.
migrate	Spouští migraci. Může vytvořit novou tabulku v databázi nebo přidat sloupeček v tabulce.
make:seeder	Vytvoří třídu, která přidává data do databáze.
db:seed	Spustí zanesení databáze.
route:list	Vygeneruje tabulku všech routovacích cest v aplikaci.

Tabulka 1 artisan příkazy

5.2.6.2.Modularita

Laravel poskytuje 20 vestavěných knihoven a modulů, které pomáhají při vylepšování aplikace. Každý modul je integrován Composerem, který usnadňuje aktualizace. [20]

5.2.6.3. Routování

Laravel poskytuje uživateli flexibilní přístup k definování routovacích tras ve webové aplikaci. Routování pomáhá lépe měnit aplikaci a zvyšuje její výkon. [20] Cesty lze vytvořit několika způsoby. Pro jejich zpřehlednění mohou obsahovat i jméno a v aplikaci je lze volat jménem.

Ukázka routeru.

```
<?php

/*
|--------------------------------------------------------------------------
| Web Routes
|--------------------------------------------------------------------------
|
| Here is where you can register web routes for your application. These
| routes are loaded by the RouteServiceProvider within a group which
| contains the "web" middleware group. Now create something great!
|
*/

Route::get('/', function () {
    return view('welcome');
});
Route::get('/home', 'HomeController@index')->name('home');
```

První cesta je automaticky vygenerovaná při vytváření projektu a posílá uživatele na úvodní stránku aplikace. Druhá cesta uživatele směruje na stránku /home a funkcí „name“ je pojmenována. To znamená, že tuto cestu lze volat jménem „home“.

5.2.6.4. Blade

Laravel používá šablonovací engine Blade. Tento engine je jednoduchý šablonovací jazyk používaný k návrhu hierarchických bloků a rozvržení s předdefinovanými bloky, které obsahují dynamický obsah. [20]

Použití enginu Blade ve view je zobrazeno na další stránce.

```
@if (count($posts) > 0)
    @foreach($posts as $post)
            {{ $post->title }}
            {{ $post->text }}
    @endforeach
@endif
```

Blade lze využívat pokud view má příponu .blade.php. PHP kód se píše za znaménko „@“. Na prvním řádku je zobrazena jednoduchá podmínka if. Na dalším řádku následuje cyklus foreach, ve kterém se budou vypisovat hodnoty z proměnné „post“. K proměnným ve view se přistupuje pomocí dvou složených závorek.

5.2.6.5.Databáze

Laravel umožňuje velice jednoduchou interakci s databází přes celou řadu databázových backendů a to s použitím buď surového SQL, nebo fluent query builderu, nebo Eloquent ORM. [21]

Laravel v dnešní době podporuje 4 databáze: MySQL, PostgreSQL, SQLite a SQL Server. [21]

Veškeré konfigurace databáze jsou uloženy v souboru: config/database.php.

5.2.6.6. Eloquent ORM

Laravel obsahuje nástroj pro vytváření SQL dotazů, který pomáhá při dotazování databází pomocí jednoduchých řetězových metod. Poskytuje implementaci ORM (Objektově Relační Mapování) a implementaci ActiveRecord nazvanou Eloquent. [20]

Ukázka dotazu na tabulku uživatele, kde jsou vybrány pouze záznamy, které mají hodnotu „id“ větší než 100. Dále příkazem „paginate“ se nastaví stránkování. To se dále využije ve view, kde se bude zobrazovat vždy jen 15 záznamů.

```
User::where('id', '>', 100)->paginate(15);
```

5.2.6.7. Migrace

Pro zjednodušení práce s databází a možností kontroly verzí databáze, Laravel umožňuje používat tzv. migrace. Migracemi lze definovat databázové tabulky nebo nové sloupečky. Slouží k tomu migrační třídy. Migracemi lze přidat novou tabulku či sloupec do databáze, nebo použitím „rollbacku“ lze vrátit databázi do předchozího stavu. Migrace se spouští příkazem: `php artisan migrate`

Vytvořit novou migrační třídu přes konzoli lze příkazem:

```
php artisan make:migration migrationName
```

Třída obsahuje 2 metody. Metoda „up“ se spustí při migraci a slouží k přidání objektu do databáze. Metoda „down“ se provede při rollbacku a slouží k odstranění změn.

Ukázka metody up() v migrační třídě.

```
public function up()
{
    Schema::create('posts', function (Blueprint $table) {
        $table->increments('id');
        $table->string('title', 255);
        $table->timestamps();
    });
}
```

V této metodě, která se spouští s příkazem `php artisan migrate` framework vytvoří novou tabulku. Tabulka se jmenuje „posts“ a bude obsahovat sloupeček s primárním klíčem „id“, který se bude automaticky inkrementovat. Dále bude mít sloupeček „title“ s velikostí 255 bitů a políčka kdy byl záznam vytvořen a změněn, ty se vytvoří automaticky pomocí příkazu „timestamps()“.

Ukázka metody down().

```
public function down()
{
    Schema::drop('posts');
}
```

Tato funkce, která se spustím příkazem `php artisan rollback`, vymaže tabulku „posts“.

5.2.7. Výhody

Tento framework disponuje velikou základnou programátorů. Na jeho webu je spoustu video tutoriálů. Má také vlastní diskuzní fórum laracast (obdoba stackoverflow). Laravel je velice intuitivní a díky mnoha tutoriálům není těžké ho začít používat. Díky veliké komunitě není problém najít řešení problému s kódem.

Laravel umožňuje vygenerovat celou autentizaci pomocí jednoduchého příkazu. Využívá pro směrování směrovací cesty „routes“, které mohou zjednodušit směrování v aplikaci a zároveň umožňuje chránit aplikaci proti CSRF útokům. Cesty je možné pojmenovávat.

Používá svůj šablonovací engine Blade, který napomáhá udržovat přehledný kód. Obsahuje také migrace, které usnadňují práci s databází.

5.2.8. Nevýhody

Problém je u kompatibility starších verzí. Při příchodu nové verze je potřeba aplikaci přizpůsobit novější verzi. Zejména při přechodu z verze 4 na verzi 5.

V dnešní době v České republice není mnoho pracovních nabídek s tímto frameworkem.

5.3. Nette Framework

5.3.1. Popis

Nette je nejznámější český MVC PHP framework. Autorem frameworku je český vývojář David Grudl. Po vydání vyrostla kolem frameworku velká komunita českých a slovenských PHP vývojářů. Dnes se o vývoj stará organizace Nette Foundation. [22] Nette je hojně využíváno tuzemskými a slovenskými společnostmi. Mezi weby, které využívají Nette patří Root.cz, ČSFD.cz, Ulož.to a další.

Využívá programování řízené událostmi a z velké části je založen na použití komponent. Nette obsahuje velmi silný validační jazyk, dokáže vygenerovat validační JavaScript pro kontrolu formulářů. Snahou tohoto frameworku je eliminovat rizika a umožnit opětovnou použitelnost zdrojového kódu. [22] Stejně jako další frameworky obsahuje zabezpečující prvky proti útokům.

5.3.2. Stručná historie

V roce 2007 na konferenci PHP frameworků David Grudl poprvé ukazuje Nette Framework. O pár měsíců byl vydán ve verzi 0.7. Následovaly verze 0.8 a 0.9.

Verze 2.0 - Po letech vývoje, po alfa a beta verzích a třech release kandidátech, Nette Foundation s hrdostí oznamuje, že vyšla nová verze jednoho z nejpopulárnějších frameworků u nás: Nette Framework 2.0 final. [23]

Nette 2 obsahuje – podporu Dependency Injection, nová databázová vrstva s integrovaným NotORM, přepsaný šablonovací jazyk Latte, přizpůsobitelná Laděnka, nový značkovací jazyk NEON. [23]

Po vydání verze 2.1 v roce 2014 zakladatel David Grudl opustil funkci hlavního vývojáře. Od této doby se o vývoj stará organizace Nette Foundation a komunita vývojářů.

Poslední verze byla vydána v červnu 2016. V roce 2018 je verze 2.4 stále aktuální, tato verze je více jak 2 roky stará a jako jediná je stále podporována. [24]

5.3.3. Budoucnost

Nette 3 - Jelikož podpora Nette 2.4 podle harmonogramu skončila v červnu 2018, tedy po dvou letech od jeho vydání, je hlavní prioritou brzký příchod nové verze Nette 3. Ta bude určená pro PHP 7.1 a bude využívat jeho zásadních novinek, jako jsou: fungování ve striktním režimu, kód bude používat skalární typehinty, metody budou mít návratové typehinty. Stabilní verze následovaná po testovacích RC by mohla vyjít v rozmezí října až listopadu 2018. [25]

Nette 4 - Ihned po vydání Nette 3 by začal vývoj Nette 4. Hlavní novinkou by byla změna vnitřní architektury na tzv. middleware. Komponenta pro zpracování a sanitizaci HTTP požadavku, router, PresenterFactory i samotný presenter by byly vrstvy middleware, mezi které bude možné snadno včlenit další vrstvy, nebo tyhle existující nahradit. Mělo by tak být mnohem snadnější provozování různých aplikací či dokonce frameworků na jednom webu. Každá vrstva by mohla požadavek odmítnout, třeba router by mohl legitimně vyhodit chybu 404. Zároveň by nové řešení mělo být plně kompatibilní se stávajícími presentery, takže bude možné vytvářet aplikace využívající jak nových middleware-presenterů, tak i presenterů současných. [25]

5.3.4. Licence

Nette Framework je šířen jako svobodný software, aby ho kdokoliv mohl používat. Nette umožňuje vývojářům si vybrat, pod kterou licencí budou projekt vytvářet. Na výběr je mezi licencemi New BSD, nebo GPL ve verzi 2, nebo 3. [26]

Licence BSD je doporučena pro většinu projektů, jelikož je snadné ji pochopit a neklade téměř žádná omezení na to, co může vývojář s frameworkem dělat. Lze používat Nette i v komerčních projektech. Vývojář si však může zvolit licenci GPL, pokud se mu lépe hodí pro jeho projekt. [26]

5.3.5. Požadavky

Nette vyžaduje nainstalované PHP verze 5.6 a vyšší. Dále je potřeba mít několik PHP rozšíření. Veškeré požadavky pro provoz Nette jsou sepsány na stránkách dokumentace.

Pro kontrolu splněných požadavků, je možné použít nástroj „Requirements Checker“, který zkontroluje, zda Nette může běžet na daném zařízení.

5.3.6. Instalace

Nette Framework lze stáhnout manuálně, ale doporučená cesta je začít nový projekt s využitím Composeru. [27] Vytvoření nového projektu pomocí Composeru se provede příkazem:

```
composer create-project nette/web-project nette-project
```

5.3.7. Hlavní vlastnosti

5.3.7.1. Latte

Nette používá vlastní šablonovací systém, který umožňuje stejně jako ostatní šablonovací systémy používat ve views dynamický obsah.

Latte je potřeba nejprve přidat do projektu. K tomu stačí spustit příkaz `composer require latte/latte`. Pro podporu tohoto systému je potřeba mít soubor uložen s příponou .latte. Podmínkou je používat PHP verzi 5.4.4, nebo vyšší. V Latte se veškerý PHP kód píše do složených závorek, jak je zobrazeno v ukázce níže.

```
{foreach $posts as $post}
    {$post->title}
{/foreach}
```

V tomto příkladě je napsaný kód, který vypisuje všechny titulky cyklem foreach.

5.3.7.2. N:makra

Nette umožňuje používat cykly a podmínky v Latte přímo v html elementech. Stejně jako to mají některé javascriptové frameworky. Programátorovi se tak zkrátí kód. Všechny n:makra začínají znaky „n:“.

Následující ukázka má stejnou funkci jako ukázka v předchozí kapitole. Cyklem foreach vypíše pro každou proměnnou $post v poli $posts nový odstavec s názvem daného příspěvku.

```
<p n:foreach="$posts as $post">{$post->name}</p>
```

Jako další příklad lze ukázat odkaz definovaný pomocí n:makra.

```
<a n:href="Post:show $post->id">Detail</a>
```

Otevřením tohoto odkazu se odešle id příspěvku do metody show($id), která následně zobrazí detail příslušeného příspěvku.

5.3.7.3. Tracy aneb Laděnka a Tracy Bar

Laděnka zobrazuje chyby v kódu a umožňuje programátorovi snazší opravu. *Dokáže vyhledat chybu přímo v googlu. Laděnka dále obsahuje tzv. Tracy bar, ten zobrazuje základní informace o aktuální stránce. Zobrazuje informace o načtení stránky, jako je doba pro načtení stránky, využití paměti při žádosti, anebo počet zapojených souborů. Také ukazuje počet errorů s cestou k souboru a číslem řádku. Také obsahuje informace o cestě, která byla použita.* [28]

Tracy je potřeba do projektu naimportovat. To lze jednoduše s použitím Composeru. Pro fungování aktuální verze Tracy je potřeba mít verzi PHP 5.4.4, nebo vyšší.

5.3.7.4. Flash zprávy

Nette umožňuje uložit textový řetězec na straně backendu do tzv. flash zprávy, kterou pak lze zobrazit uživateli ve views, například jako informativní zprávu, že uživatel byl úspěšně přihlášen.

V presentéru se uloží zpráva, která se pošle do view. Zpráva je určena k jednomu použití, po vypsání se obsah smaže. Vytvoření zprávy:

```
$this->flashMessage('Flash message.');
```

Ve view se akorát všechny uložené zprávy vypíšou. Ukázka view:

```
{foreach $flashes as $flash}
    <div> {$flash->message} </div>
{/foreach}
```

5.3.7.5. Database connection

Nette Database Core je základní vrstva pro přístup k databázi, tzv. database abstraction layer. [29] Tato vrstva komunikuje mezi aplikací a databází. Programátor tak může jednodušeji pracovat s databází a vytvářet dotazy bez použití SQL. Základem Connection API je PHP rozšíření PDO, které zprostředkovává „*konzistentní rozhraní pro přístup k databázím v PHP*.“ [30]

Po připojení se automaticky zobrazí panel na debugger baru. Ten programátora informuje o všech vykonaných SQL dotazech, době jejich vyřízení, počtu vrácených řádků, dokonce uvádí i jejich EXPLAIN podobu a místo, v kódu, kde byl dotaz volán. Oproti PDO dovoluje Connection uvádět v příkazech exec a query výčet parametrů. Parametry mohou být proměnné, konstantní data, objekt, nebo soubor. Parametry mohou být zadány i v poli. [29]

Dotaz na tabulku uživatelů je zobrazen v ukázce níže. Otazník, za kterým se do dotazu vkládá proměnná chrání dotaz před zranitelností vůči SQL injection.

```
$database->query('SELECT * FROM users WHERE name = ?', $name);
```

Nette Database Core podporuje následující databáze: MySQL, PostgreSQL, SQLite, Oracle, MS SQL a ODBC. [31] S ODBC neumí pracovat Database Explorer, který sestavuje dotazy na databázi.

5.3.8. Výhody

V České republice se koná mnoho kurzů a školení pro tento framework. Existují i pravidelná setkání fanoušků tohoto frameworku a webových aplikací. Jmenují se „Poslední sobota“ a pořádají se již od roku 2008.

Další výhodou tohoto frameworku je rozšířenost po České a Slovenské republice a z toho vyplívající projekty. Nette má oproti ostatním porovnávaným frameworkům nejvíce pracovních nabídek v České a Slovenské republice.

Nette validuje formuláře javascriptem. Ten v případě špatně vyplněného formuláře vypíše chybovou hlášku, kterou lze upravit.

Jako jediný z porovnávaných frameworků Nette obsahuje dokumentaci a návody v češtině. Což určitě ocení čeští vývojáři. Hodně aktivní je také české fórum, kde uživatelé řeší své problémy. Také na internetu lze nalézt mnoho příspěvků v češtině, ale samozřejmě i v angličtině.

5.3.9. Nevýhody

Česká komunita může být nevýhodou pro zahraniční vývojáře, jelikož mnoho řešených problémů, je pouze v češtině.

Nette Framework nebyl vytvořen žádnou velkou společností, která by se postarala o jeho neustálé financování a šíření do ostatních zemí. A to je jeden z důvodů, proč se Nette zatím

nerozšířil v zahraničí. Vzhledem k tomu, že je framework financován převážně z komunity vývojářů, momentálně není jistá jeho budoucnost.

Při vývoji aplikace se informace ukládají do cache aplikace, aby se mohly rychleji načíst. Bohužel, občas se stává, že cache nemá aktuální informace, a proto se nemusí ukázat změny, které vývojář v aplikaci udělal. Proto je nutné vymazat cache, aby se do ní mohly uložit nové informace. Tento proces se stává časem hodně rušivým.

Absence nástroje pro práci s databází. Vývojáři musí celou databázi vytvořit a spravovat sami. Chybí tak možnost sledovat změny provedené v databázi, či změny vracet.

5.4. Symfony

5.4.1. Popis

Symfony je poslední z představovaných frameworků. Využívá architekturu MVC. Framework používá komponenty, které lze naimportovat pomocí Composeru. Tyto komponenty jsou znovupoužitelné. Na tomto frameworku je částečně založený i první z představených frameworků Laravel. Symfony obsahuje (podobně jako Laravel) symfonycasty, což jsou video tutoriály, které obsahují návody, jak pro Symfony, tak i pro další technologie.

Symfony vyvinula francouzská společnost SensioLabs. Symfony dnes používají projekty, mezi které patří: Drupal, Joomla!, i některé PHP frameworky.

Symfony pořádá během roku několik konferencí. Tyto setkání se jmenují Symfonycon a trvají několik dní. Jsou zde probírány témata ohledně Symfony a dalších technologií.

5.4.2. Stručná historie

Symfony vyvinula francouzská společnost SensioLabs. Framework byl nejprve vyvíjen pod názvem Sensio framework, poté se framework přejmenoval na Symfony. První verze byla vydána v říjnu roku 2005 zakladatelem projektu Fabienem Potencierem. [32]

V roce 2003 Fabien strávil nějaký čas otázkou o stávajících vývojových open source nástrojích pro webové aplikace v PHP. Zjistil, že žádný nesplnil dříve popsané požadavky. Když byl spuštěn PHP 5 zjistil, že všechny dostupné nástroje již dosáhly stádia stáří, a nebudou schopny nadále pokrývat všechny vlastnosti nové verze PHP. V praxi to znamenalo, že jejich architektura byla zastaralá a nemohla by reagovat účinně na nové změny. Následně

strávil jeden rok vývojem jádra pro Symfony, svoji práci založil na MVC, ORM a Ruby on Rails šablonovacím systému. [32]

Momentálně Symfony plně podporuje 3 verze. Verze 3.4 je verze s dlouhodobou podporou (LTS) a má být plně podporována do listopadu 2020. Plná podpora verze 2.8 bude končit v listopadu 2018, verze 4.1 v lednu 2019 a ve verzi 4.0 jsou opravovány jen bezpečnostní chyby. Verze 4.2 má vyjít v listopadu 2018. [33]

5.4.3. Licence

Symfony je open source framework pod licencí MIT. *Každý může volně používat, kopírovat modifikovat, sloučit, publikovat, distribuovat, sublicencovat nebo prodávat kopie Softwaru za podmínek, že software bude obsahovat licenční znění a autorská práva (copyright).* [34]

5.4.4. Požadavky

Pro instalaci aktuální verze je potřeba mít vezre PHP 7.1.3, nebo vyšší. Také je potřeba mít nainstalovaný Composer.

Symfony umožňuje zkontrolovat, zda zařízení splňuje veškeré požadavky příkazem:
`composer require symfony/requirements-checker`

5.4.5. Instalace

Pokud zařízení splňuje veškeré požadavky, je možné vytvořit projekt. Symfony umožňuje stáhnout celou adresářovou strukturu, nebo mikro verzi. Pokud vývojář chce mikro verzi projektu, stačí pozměnit příkaz a místo „website-skeleton“ zadat „skeleton“.

```
composer create-project symfony/website-skeleton project-dir
```

5.4.6. Hlavní vlastnosti

5.4.6.1.Twig

Je šablonovací systém pro Symfony. Twig je používán mnoha open-source projekty jako jsou Symfony, Drupal8, eZPublish, phpBB, Piwik, OroCRM a mnoho frameworků podporuje Twig. Například: Slim, Yii, Laravel, Codeigniter a Kohana. [35]

Twig je bezpečný šablonovací systém a umožňuje vývojáři upravit šablony podle sebe. Pro použití je potřeba Twig naimportovat. K tomu stačí použít Composer. Dále je potřeba soubor uložit s příponou .html.twig.

Twig používají tři druhy složených závorek. Dvojité složené ({{...}}) slouží pro vypsání proměnné, složené s křížkem ({#...#}) slouží pro komentáře a složené s procentem ({%...%}) slouží pro příkazy. V ukázce níže je zobrazen cyklus, který vypisuje titulky příspěvků.

```
{% for post in posts %}
        {{ post.title }}
{% endfor %}
```

5.4.6.2. Router

Symfony používá pro směrování v aplikaci router. Router zpracovává požadavky na dané cesty v aplikaci. Používání routeru umožňuje mít přehled nad všemi cesty v aplikaci a jejich snadnou správu a editaci. Cesty lze samozřejmě použít vícekrát, což programátorovi šetří čas a řádky kódu. Definované cesty také vytváří pěknou url adresu. K těmto cestám lze i nastavovat jazyk. V ukázce je zobrazena cesta. Na prvním řádku je název cesty, na dalším je adresa a na posledním řádku je cesta k danému controlleru.

```
users:
    path: /users
    controller: App\Controller\UserController::getUsers
```

5.4.6.3. Anotace

Anotace definují funkcím i objektům charakteristiky. Zapisují se v komentáři za znak „@“. Místo definování všech cest v routeru lze zapsat cestu pomocí anotace @Route() přímo u dané funkce v controlleru. Cestě lze nastavit všechny parametry stejně jako v routeru. Například jazyk nebo název. Anotace je potřeba před použitím do projektu naimportovat. Nejjednodušší způsob je použitím Composeru.

Anotace se používají i u entit, kde udávají, jaký datový typ daná proměnná je a jaký datový typ má sloupeček v databázi. Anotace dále umožňují omezovat přístup k funkcím v controllerech. Slouží pro to anotace @IsGranted() a @Security(). V další ukázce je anotace, která povoluje smazat příspěvek jen administrátorům, nikdo jiný příspěvek smazat nemůže.

```
/**
 * @Route("/delete/{id}", name="deletePost")
 * @Security("is_granted('ROLE_ADMIN')")
 */
```

5.4.6.4. Doctrine ORM & migrace

Symfony neobsahuje komponentu, která pracuje s databází, ale využívá k tomu knihovnou zvanou „Doctrine“, což je nástroj, který umí s databází pracovat. Doctrine je velice silný nástroj a dokáže velmi pomoct se správou a vývojem databáze. Díky Doctrine lze používat migrace a další. Migracemi lze vytvářet nové tabulky nebo přidávat sloupečky do tabulek. Vývojář tak nemusí šahat do databáze přímo nebo psát SQL kód pro vytvoření tabulky. Tuto vlastnost určitě ocení hlavně vývojáři, kteří pracují v týmu. Všechny změny v databázi lze mít tak pod kontrolou a lze se vracet k předchozím verzím. Migrace je potřeba jako spoustu dalších funkcionalit do projektu naimportovat. Stačí k tomu použít Composer. Doctrine dokonce umí vytvořit novou databázi, takže vývojáři se nemusí s databází setkat přímo.

Symfony podporuje následující relační databáze: MySQL, PostgreSQL, Microsoft SQL a také jednu NoSQL databázi. MongoDB, ale pro práci s touto databází je potřeba importovat Doctrine ODM knihovnu. [36]

Doctrine využívá pro skládání dotazů ORM, který umožňuje vytvářet crud operace nad databází bez použití SQL kódu. Tato technika ochraňuje dotazy před SQL injection.

Kód pro vytvoření nové databáze pomocí konzole je zobrazen v ukázce níže. Po spuštění kódu se objeví formulář s dodatečnými informacemi.

```
php bin/console doctrine:database:create
```

5.4.6.5. Bin/console

Jelikož Laravel vychází ze Symfony, tak mají mnoho společného. Podobně jako Laravel má příkaz v konzoli „Artisan“, Symfony má příkaz pojmenovaný „bin/console“. Vývojář tímto příkazem může vytvářet nové třídy, entity, controllery, spouštět migrace, čistit cache a další. Tato funkcionalita je velice užitečná a dokáže vývojáři efektivně pomoct s vývojem aplikace. Lze si také vytvořit svůj vlastní příkaz. V tabulce na další straně jsou vypsány a popsány některé příkazy.

Příkaz	Funkcionalita
Debug:router	Vypíše všechny cesty v aplikaci.
Make:controller	Vytvoří nový controller.
Make:entity	Vytvoří novou entitní třídu.

Tabulka 2 bin/console příkazy

5.4.6.6. Flash messages

Symfony obsahuje integrovanou funkcionalitu pro odesílání zpráv uživateli. Slouží pro zobrazování informačních zpráv. Po zobrazení jsou smazány. Lze je využít pro zobrazení zprávy, že byl vytvořen nový příspěvek.

```
$this->addFlash('success', 'Post created.');
```

Ve view se zprávy vypíší takto.

```
{% for message in app.flashes('success') %}
        {{ message }}
{% endfor %}
```

5.4.7. Výhody

Symfony umožňuje přidávat do projektu jen ty balíčky, které vývojář chce. Aplikace tak na začátku vývoje obsahuje jen základní funkcionalitu. To znamená, že do aplikace se přidávají jen ty balíčky, které jsou potřeba. Aplikace tak neobsahuje plno zbytečných funkcionalit.

Obsahuje širokou nabídku tzv. bundles. Bundles jsou podobné pluginům.

Umožňuje vytvořit tzv. microapp. *Web-skeleton je optimalizován pro tradiční webové aplikace. V případě vytváření microservices, konzolové aplikace nebo rozhraní API, je dobré zvážit použití mnohem jednoduššího skeleton projektu (symfony/skeleton).* [37]

Symfony je jako jeden z mála financován velkou společností. SensioLabs financuje tento framework a stará se o jeho vývoj.

Symfony má dlouhou historii. Za jeho existenci se objevilo několik platforem a frameworků, které používají jeho komponenty. To je známka vysoké použitelnosti jeho částí a komponent.

Za svou existenci si framework vybudoval velikou komunitu vývojářů a je používán mnoha společnostmi.

Obsahuje velmi rozsáhlou dokumentaci s mnoha příklady a ukázkami použití. Na stránkách frameworku existují tzv. SymfonyCasts. Jsou to video tutoriály, díky kterým se lze naučit Symfony, nebo jiné technologie.

5.4.8. Nevýhody

Symfony nepatří do skupiny jednoduchých frameworků. Je potřeba více času na naučení tohoto frameworku a následného použití, než je u jiných frameworků.

Řešení některých problémů jsou ve francouzském jazyce.

6. Zabezpečení

Jedním z důvodů, proč použít při vývoji aplikace framework, je snaha vyhnout se bezpečnostním rizikům a dírám ve zdrojovém kódu aplikace. Existuje mnoho typů útoků na webovou aplikaci. V dnešní době většina webových frameworků tyto bezpečnostní útoky zná a snaží se jim předejít. V následujících bodech jsou popsány nejběžnější útoky a způsoby, jak se proti nim bránit.

6.1.SQL Injection

První typ útoku využívá špatného ošetření vstupu od uživatele při sestavování dotazů a parametrů do jiných systémů. Nejčastějším typem tohoto útoku je SQL Injection – špatná obsluha parametrů při tvorbě SQL dotazů. [38]

Tento typ útoky je častý a pro útočníka velice jednoduchý. Útočníkovi stačí znát syntaxi jazyka SQL a část struktury databáze. Nebo lze použít techniku „pokus omyl“ a zaměřit se na nejčastější názvy. Následně si s databází útočník může dělat co chce. Útočník může získat data, která pro něj nejsou určena a měnit je. V praxi to znamená, že může smazat celou tabulku a tím nadělat velké škody vlastníkovi aplikace.

Zde je ukázka špatného použití (proměnná „input“ je text, který útočník poslal do aplikace):

```
“SELECT * FROM users WHERE id = $input“;
```

Pokud útočník pošle SQL injection v tomto případě “ 22 OR 1 = 1“, může tak získat data ke kterým by neměl mít přístup.

```
“SELECT * FROM users WHERE id = 22 OR 1 = 1“;
```

V tomto případě dotaz byl v pořádku a útočník získá všechna data z tabulky uživatelů.

Pokud útočník bude znát strukturu databáze, nebo jen název jedné tabulky, tak může měnit záznamy, nebo tabulky v databázi. Zde je ukázka SQL kódu, který smaže tabulku “users“.

```
“SELECT * FROM users WHERE id = 22; DROP TABLE users“;
```

Ochrana před tímto útokem, je kontrolovat vstupy. V PHP existuje rozšíření PDO, které dotaz připravý a vrátí výsledek. Na straně databáze lze nastavit práva uživatelům, kteří k databázi přistupují. Nebo v některých případech používat pohledy.

Framework	Zabezpečen
Laravel	Ano
Nette	Ano
Symfony	Ano

Tabulka 3 zabezpečení proti SQL Injection

6.1.1. Laravel

Laravel používá pro práci s databázemi PDO rozšíření, pro efektivnější práci s databází. Právě toto rozšíření zabezpečuje Laravel před tímto útokem. [39]

Jediný případ, kdy je možné úspěšně provést tento útok je, pokud by vývojář vytvářel tzv. „raw“ dotazy a správně tyto dotazy neošetřil. „*Dotazy se skládají do řetězce a aplikace může být zranitelná.*“ [39] Při správném použití dotazů je tedy Laravel zcela chráněn a vývojář nemusí implementovat žádnou ochranu.

6.1.2. Nette

Nette přistupuje k databázi přes Database Core vrstvu. Dotazy na databázi se vytváří pomocí metody query(), která je proti SQL Injection chráněná.

Ale stejně jako u Laravelu, pokud by se dotaz skládal jako řetězec, tak vznikne zranitelnost aplikace. Nette je tedy také zcela chráněno a vývojář si akorát musí dát pozor na správné skládání dotazů.

6.1.3. Symfony

Symfony používá pro práci s databází objektově relační zobrazení (ORM), které chrání aplikaci před útokem.

Lze se však dotazovat databáze pomocí DQL (dotazovací jazyk) a zde je možnost výskytu zranitelného místa. Pokud vývojář používá DQL, je zapotřebí jakoukoliv proměnnou přidávat do dotazu pomocí funkce setParameter(). Symfony stejně jako ostatní frameworky je proti tomuto útoku chráněno.

6.2.Cross Site Scripting (XSS)

XSS je útok, kterým útočník dostane do aplikace svůj kód. Může poslat do aplikace html tagy, nebo javascript.

Při zobrazování vstupů od uživatele je nutné zajistit, aby veškerý výstup nebyl do HTML stránky zapsán přímo, ale jen jako text. Jinými slovy, musí dojít k nahrazení pro HTML klíčových znaků za jejich entitní vyjádření tj. < za <, > za > a & za &. [38]

Nejlepším řešením je používat takové systémy, které při výpisu textu na obrazovku rovnou vše správně převedou na entitní vyjádření. Texty, které se nemají převádět, se pak musí explicitně označovat. [38]

Ukázka, kde útočník vloží do webové aplikace jednoduchý html kód s tučným textem.

XSS

Tento útok sice není nijak nebezpečný, ale lze místo tučného textu vložit do stránky například obrázek.

Nebezpečnější útoky jsou s použitím javascriptu. Pokud není na stránce vypnutý javascript, lze tak se základní znalostí javascriptu vložit do stránky script, který může přesměrovat uživatele na jinou stránku, nebo může získat citlivé údaje.

Zde je ukázka jednoduchého scriptu, který po otevření stránky uživatele přesměruje na jinou stránku.

```
<script> window.location.replace("https://www.seznam.cz"); </script>
```

Ochránit aplikaci lze tím, že při odeslání formuláře se zkontroluje, jestli vstup obsahuje speciální znaky a ty se následně převedou na jejich entitní vyjádření. V PHP existuje funkce „htmlspecialchars()“, která kontroluje vstup a mění znaky.

Framework	**Zabezpečen**
Laravel	Ano
Nette	Ano
Symfony	Ano

Tabulka 4 zabezpečení proti XSS

6.2.1. Laravel

Laravel je chráněn šablonovacím enginem Blade. *Blade veškeré výstupy nejprve pošle do PHP funkce htmlspecialchars, která převede veškeré speciální znaky do HTML entit.* [40]

Stejně jako u předchozího útoku, i zde existuje případ, kdy je aplikace zranitelná. Pokud vývojář použije ve view místo obvyklých dvojitých složených závorek pro přístup k datům

jen tyto: „{!! !!}“, tak je možnost, že aplikace bude zranitelná. U těchto závorek se vypisují surová data a nekontrolují se speciální znaky.

6.2.2. Nette

Nette používá ve svém Latte technologii kontextově sensitivního escapování (Context-Aware Escaping), která ošetřuje veškeré výstupy automaticky. [41] A je tedy proti tomuto útoku zcela chráněno.

6.2.3. Symfony

Symfony používá pro svoji ochranu svůj šablonovací engine Twig, který převádí speciální znaky do HTML entit. [42] Tímto je aplikace chráněna.

Aplikace může být nezabezpečena, pokud vývojář nastaví, aby výstup nebyl převáděn do HTML entit.

6.3.Cross Site Request Forgery (CSRF)

Cross site request forgery je velmi známý a častý útok na webovou aplikaci. *Útok spočívá v tom, že přiměje uživatele navštívit stránku, která skrytě vykoná útok na webovou aplikaci, kde je uživatel zrovna přihlášen. Lze takto například pozměnit nebo smazat článek, aniž by si toho uživatel všiml. Proti útoku se lze bránit generováním a ověřováním autorizačního tokenu.* [41]

Framework	Zabezpečen
Laravel	Ano
Nette	Ano
Symfony	Ano

Tabulka 5 zabezpečení proti CSRF

6.3.1. Laravel

Laravel automaticky generuje CSRF "token" pro každou relaci aktivního uživatele spravovanou aplikací. Tento token slouží k ověření, zda ověřený uživatel je ten, kdo skutečně posílá žádosti aplikaci. [43]

Pro aktivaci ochrany je potřeba přidat do formuláře skryté pole s CSRF tokenem. Díky tomuto poli bude aplikace moct ověřit uživatele. Pole se přidá kódem: `@csrf`

6.3.2. Nette

Nette pro ochranu aplikace generuje a ověřuje autorizační token.

Stačí přidat do formuláře kód: `$form->addProtection();`.

6.3.3. Symfony

Symfony ověřuje uživatele pomocí skrytého tokenu ve formuláři. Token zná jen uživatel a aplikace. Avšak nejdříve je potřeba naimportovat ochranu do aplikace. Ochrana se naimportuje příkazem `composer require symfony/security-csrf`. [44] Poté lze přidat do formuláře políčko s tokenem.

7. Souhrn teoretické části

Architektura – Laravel a Symfony používají architekturu MVC a Nette používá MVP (Model, View, Presenter). Všechny architektury jsou objektové a člení kód na tři části.

Správa balíků – pro instalaci samotného frameworku a přidání dalších rozšíření používají všechny frameworky Composer.

Systémové požadavky – požadavky jsou více méně u všech stejné. Žádný z frameworků nemá speciální požadavek. Všechny podporují používání nejnovějšího PHP verze 7. Všechny frameworky dostaly 2 body.

Licence – všechny frameworky jsou open source. Používají licence, které umožňují software volně používat, měnit a šířit. Všechny frameworky dostaly 2 body.

Dokumentace – každý framework má velice kvalitní dokumentaci. Všechny mají ve své dokumentaci ukázky správného použití kódu. Symfony a Nette obsahují návod pro vytvoření první aplikace. Laravel a Symfony obsahují video tutoriály, ve kterých jsou popsány a vysvětleny dané technologie. Nette také obsahuje dokumentaci v češtině. Laravel a Nette v tomto bodě získaly 1 bod, Symfony 2.

Aktualizace – Laravel a Symfony vydávají pravidelně nové verze a používají ve svém frameworku nejnovější trendy. Oba vydávají také verzi s dlouhodobou podporou. Naopak Nette má v této době aktuální verzi, která je více než 2 roky stará. Nette proto nezískalo žádný bod za tuto část. Laravel a Symfony 2.

Šablonovací systém – každý framework používá svůj vlastní šablonovací systém. Všechny šablonovací systémy jsou velice intuitivní a snadné pro použití. Symfony a Nette používají šablonu, kde je základní html struktura. A v ostatních views jsou akorát potřebné tagy. To nabízí i Laravel. Všechny frameworky dostaly 2 body.

Směrování – všechny frameworky používají pro směrování router. Laravel a Symfony definují cesty v aplikaci v jednom souboru. Symfony umožňuje definovat cesty pomocí anotací. Nette neumožňuje definovat všechny cesty v aplikaci v jednom souboru, ale umožňuje psát odkazy v n:makrech. Laravel a Symfony získaly 2 body a Nette 1.

Použití konzole – Laravel a Symfony umožňuje používat konzoli při práci s aplikací. Díky příkazům artisan a bin/console, lze jednoduše přidávat nové části aplikace, či importovat celou funkcionalitu. Symfony a Laravel získaly 2 body a Nette žádný.

Náročnost – podle zkušeností z vývoje aplikací a také podle mnoha hodnocení na internetu, je nejjednodušší z těchto tří frameworků právě Laravel. K tomuto také přispěl fakt, že Laravel

je v posledních letech nejoblíbenějším PHP framework a právě proto existuje na internetu mnoho návodů a řešených problémů, které určitě vývojáři pomůžou. Obsahuje také možnost importovat základní funkcionality aplikace. Laravel získal 2 body, Nette a Symfony 1.

Databáze – všechny frameworky skládají bezpečné dotazy. Používají pro práci s databází ORM, nebo PDO rozšíření. Dokáží vytvořit dotazy na databázi bez nutnosti použití SQL. Ale také lze psát dotazy s čistým SQL. Laravel umožňuje používat seedery, který zaplňují databázi testovacími daty. Laravel a Symfony umožňují používat migrace. Naopak Nette v této části zaostává. Všechny frameworky podporují relační open source databáze jako jsou: MySQL a PostgreSQL. Symfony po stažení potřebné knihovny dokáže pracovat i s NoSQL databází MongoDB. Laravel získal 4 body, Symfony 3 a Nette 2.

Bezpečnost – u hrozby SLQ injection jsou všechny frameworky chráněny. Ale u nesprávného skládání dotazů existuje možnost, že aplikace bude zranitelná. U tohoto útoku jsou ochráněny stejně. Ochranu před XSS zaštiťují šablonovací systémy, které automaticky nahrazují speciální znaky jejich entitním vyjádřením. Ale u všech lze tuto ochranu vypnout. Také u ochrany před CSRF mají frameworky stejnou metodu ochrany. Přidávají do formuláře políčko s tokenem, který se následně ověřuje. U Laravelu a Nette stačí přidat do formuláře další pole s tímto tokenem, ale u Symfony je nejprve třeba ochranu naimportovat. Nette a Laravel získali 3 body a Symfony 2.

Pracovní nabídky – z nabízených pozic na adrese cz.indeed.com pro město Praha je vidět, že v dnešní době je v České republice největší poptávka po Nette vývojářích. V 10 inzerátech byl vyhledán Nette programátor, nebo programátor se zkušenostmi s Nette 5krát. Symfony a Laravel vývojáři byli poptávání pouze jednou. Nette získalo 2 body. Laravel a Symfony 1.

Za teoretickou část Laravel získal 23 bodů, Symfony 21 bodů a Nette 16 bodů.

8. Praktické porovnání

8.1. Příprava aplikace

Pro vývoj aplikací byla použita aplikace XAMPP, která vytvoří lokální webový server Apache pro běh aplikace a databází. Aplikace byly vytvářeny ve vývojovém prostředí PHPStorm.

8.2. Vytvoření projektu

Při vytváření projektů programátorovi hodně pomůže Composer. Všechny frameworky používají pro správu balíčků právě Composer, ten dokáže naimportovat projekt spuštěním jednoho příkazu.

Laravel projekt se vytvoří příkazem, který je v ukázce níže.

```
composer create-project laravel/laravel applicationLaravel
```

Symfony nabízí programátorovi výběr mezi plnou a mikro verzí aplikace. Pokud bude aplikace sloužit například jako API, programátor zadá „symfony/skeleton“. Pro vytvoření plnohodnotné webové aplikace slouží příkaz níže.

```
composer create-project symfony/website-skeleton applicationSymfony
```

Nette nabízí volbu mezi použitím Composeru a stažením .zip souboru.

```
composer create-project nette/web-project
```

Pro porovnání byly aplikace nainstalovány v aktuálních verzích pro listopad 2018.

Framework	**Verze**
Laravel	5.4.36
Symfony	4.1.7
Nette	2.4.12

Tabulka 6 verze frameworků

8.3. Adresářová struktura

Adresářové struktury frameworků jsou odlišné. Struktury Symfony a Laravelu jsou v některých částech podobné. Nette používá zcela odlišnou strukturu.

Laravel má veškerou logiku uloženou ve složce app/. Ve složce database/ jsou uloženy migrace a zaplňovači databáze (seeders), public/ obsahuje soubor index.php. Složka resources/ obsahuje veškeré views a ve složce routes/ jsou uloženy veškeré cesty aplikace.

App/

Bootstrap/

Config/

Database/

Public/

Resources/

Routes/

Storage/

Tests/

Vendor/

Symfony uchovává migrace, controllery a entity ve složce src/. Ve složce config/ jsou konfigurační soubory a seznam cest. V public/ je uložen soubor index.php. A ve složce templates/ jsou veškeré views.

Assets/

bin/

config/

public/

src/

templates/

tests/

translations/

var/

cache/

log/

vendor/

Nette ukládá veškeré presentéry a views spolu s další logikou ve složce app/. Struktura Nette je oproti předchozím frameworkům menší.

App/

config/

log/

temp/

vendor/

www/

8.4. Vytvoření databáze

Je potřeba vytvořit relační databázi s tabulkami, kam se ukládají data aplikace.

Databázi pro Laravel je potřeba vytvořit pomocí nějakého nástroje, nebo vlastním skriptem, samotný framework to neumí. Po vytvoření databáze je potřeba v souboru config/database.php zadat správné údaje databáze. Následně lze vytvářet a pouštět migrace, které umožňují jednodušší práci s databází. Tabulky lze vytvářet migracemi. Ty tabulky vytvoří s nastavením přesně pro Laravel, nebo je lze vytvořit SQL kódem.

Symfony na rozdíl od Laravelu umožňuje vytvořit novou databázi. Uživatel tak nemusí psát SQL kód pro vytvoření databáze. Nejdříve je potřeba naimportovat „orm-pack“, který dokáže pracovat s databází. Po vytvoření je nutné v souboru .env změnit údaje o databázi. Poté je možné vytvářet a spouštět migrace.

Nette vytvářet databáze neumožňuje. Je potřeba aby databázi vytvořil sám programátor. Databáze se konfiguruje v souboru app/config/config.neon. Nette neobsahuje ani žádnou podporu pro migrace, programátor je tímto odkázán sám na sebe, nebo na nástroje třetí strany.

8.5. Autentizace

Pro vytvoření aplikace, do které se uživatelé mohou přihlašovat a registrovat, je potřeba vytvořit registraci a přihlašování. Kam patří formulář, validace hesla, kontrola existujících uživatelů a vytvoření uživatele, nebo session. Některé frameworky umožňují vygenerovat některé třídy, nebo dokonce celou logiku a ušetří tím programátorovi mnoho času.

Vytvoření přihlášení s pomocí Artisanu je v Laravelu otázka pár minut. Příkazem `php artisan make:auth` se do projektu stáhnou views a controllery pro přihlášení a registraci uživatelů.

Podobně jako Laravel i Symfony umožňuje naimportovat část přihlašování. Použitím CLI a zadáním příkazu `php bin/console make:auth` se vygeneruje formulář pro přihlášení uživatele spolu s controllerem. Registraci uživatele je potřeba dopsat.

Nette na rozdíl od předchozích frameworků neumožňuje vygenerovat ani část logiky. Přihlašování a registraci je potřeba napsat. V dokumentaci stejně jako u ostatních frameworků existuje návod, jak implementovat přihlašování do aplikace.

8.6. Směrování

Laravel používá pro směrování router. Veškeré cesty jsou uloženy v souboru routes/web.php. Ukázka cesty pro hlavní stránku se jménem „home“:

```
Route::get('/home', 'HomeController@index')->name('home');
```

Symfony také využívá pro směrování router. Cesty lze definovat v souboru confir/routes.yaml. Symfony ještě nabízí použití tzv. anotací, které programátorovi umožňují definovat cesty přímo u controllerů. Anotace je nejdříve nutné naimportovat příkazem: `composer require annotations`. Ukázka anotace:

```
/**
* @Route("/create", name="createPost")
*/
```

Nette obsahuje továrnu na cesty, ve které si programátor definuje, v jakém tvaru se odkazy budou psát. Do tohoto souboru se nepíší konkrétní cesty jako u ostatních frameworků. Odkazy se píší pomocí tzv. n:maker, ve tvaru presentér a název funkce. V těchto makrech lze funkcím posílat i hodnoty. Ukázka odkazu:

```
<a n:href="Post:create">Create Post</a>
```

8.7. Omezení přístupu

V aplikaci má mít přístup k příspěvkům jen přihlášený uživatel. Proto je potřeba zamezit přístup k příspěvkům nepřihlášeným uživatelům.

V Laravelu lze zamezit přístup na stránky v routeru. Před cesty ke kterým má mít přístup jen přihlášený uživatel stačí přidat kód `Auth::routes()`, který kontroluje, zda je uživatel přihlášený.

U Symfony je nejprve potřeba naimportovat balíček symfony/security-bundle, který vývojáři umožňuje zabezpečit aplikaci. Následně ve funkcích lze kontrolovat, zda je uživatel přihlášený. Kód vypadá takto:

```
this->denyAccessUnlessGranted('IS_AUTHENTICATED_FULLY');
```

Nebo je možnost omezit přístup k controllerům pomocí anotací.

V Nette se zamezí přístup také v dané funkci tím, že se zkontroluje, jestli je uživatel přihlášený. Kód: `$this->getUser()->isLoggedIn();`

8.8. Benchmarking

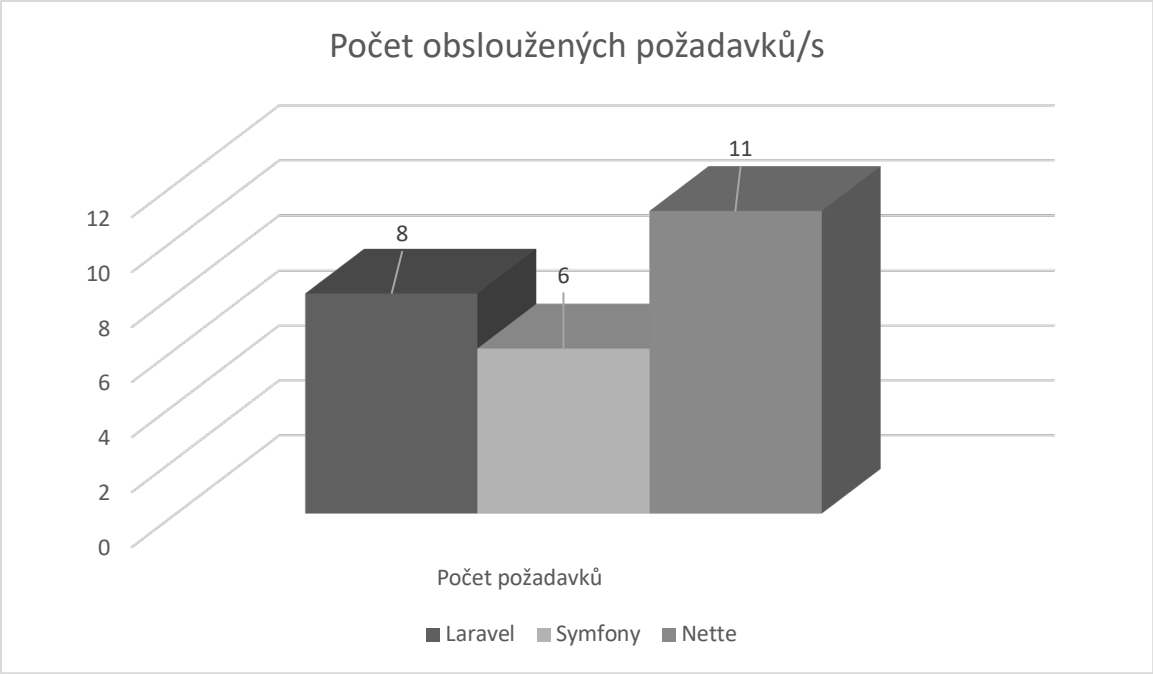
Benchmark je metoda testování, která měří výkon hardwaru, nebo softwaru. Díky těmto testům je možné porovnat daný hw/sw a zhodnotit jejich výkonnost.

K tomuto testování byl použit nástroj Apache ab tool, který dokáže změřit výkon webové aplikace. Například změřením počtu obsloužených požadavků aplikací za vteřinu.

Pro toto testování byly vytvořeny jednoduché „Hello world“ aplikace. V praxi jsou tyto aplikace nepoužitelné, ale frameworky obsahují jen základní kostru a měly by mít nejméně rozšíření, takže lze zjistit, jak jsou rychlé samy o sobě. Aplikace byly nastaveny do produkčního módu a dány na server. Jako hosting byl použit poskytovatel Endora.cz. HW konfigurace serveru: CPU 1x Intel Xeon 2.00GHz, RAM 32 GB.

Je zde znázorněno několik testů. Každý framework byl testován za stejných podmínek a výsledky, které jsou zde znázorněny jsou průměrem výsledků získaných z měření. Testování bylo provedeno ve večerních hodinách.

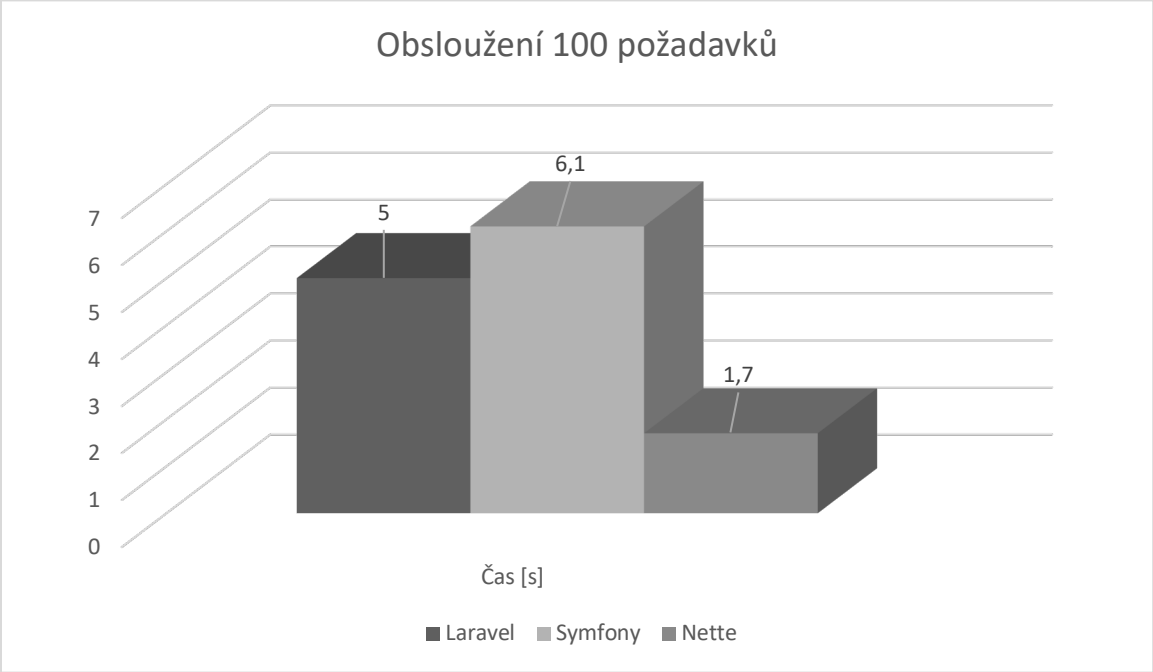
Test, ve kterém se měří, kolik požadavků aplikace dokáže obsloužit za jednu vteřinu.



Obrázek 3 Obsloužené požadavky za vteřinu

V tomto testu zvítězil Nette, který za jednu vteřinu obsloužil 11 požadavků. Laravel má o 3 méně a Symfony je skoro 2krát pomalejší.

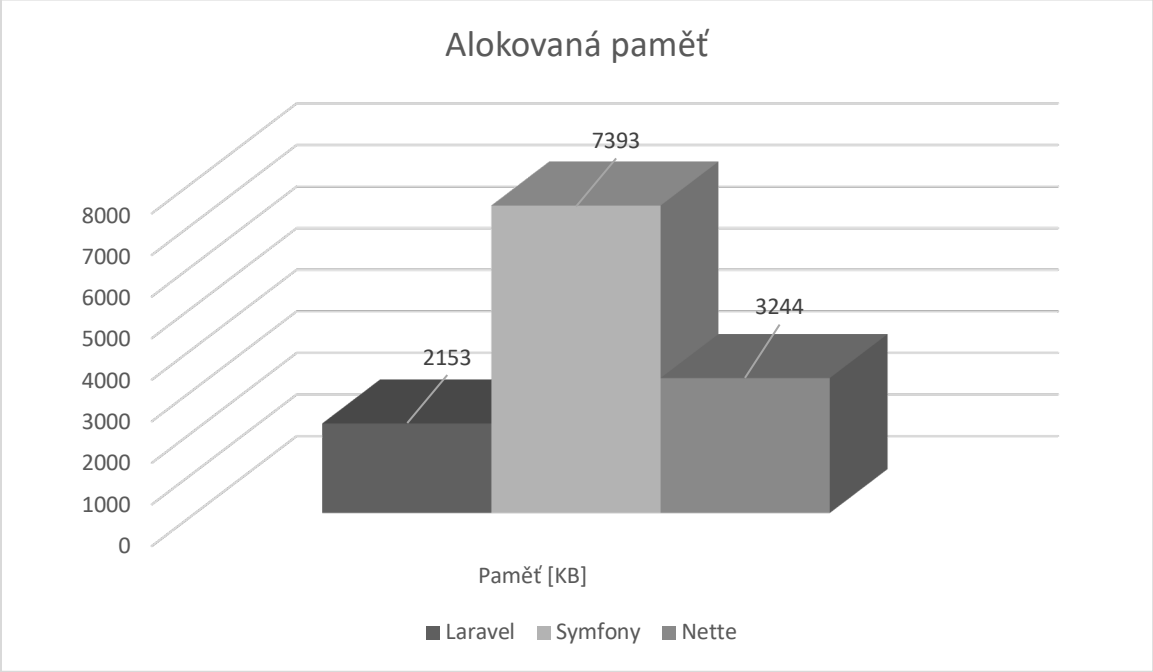
Výsledek testu, ve kterém se zjišťuje za jakou dobu aplikace obslouží 100 požadavků.



Obrázek 4 Obsloužení 100 požadavků

V tomto testu je jasný vítěz Nette, který několikrát překonal oba své soupeře. Laravel s časem 5 vteřin je o více než jednu vteřinu rychlejší jak Symfony.

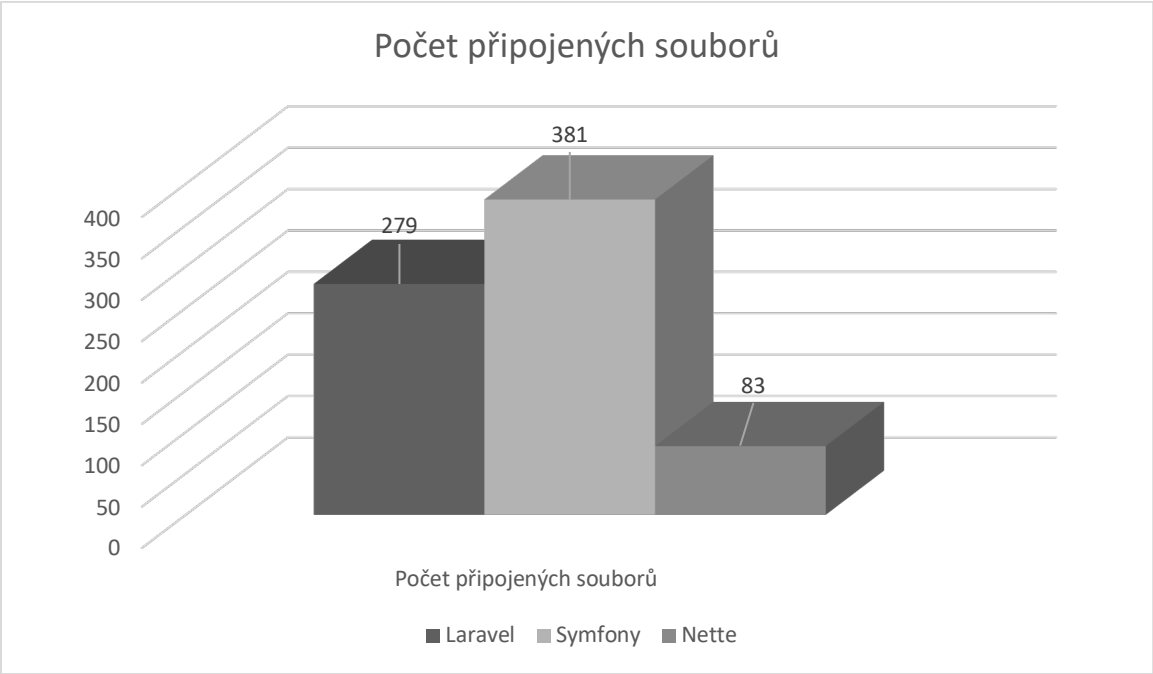
V dalším testu se zjišťuje, kolik paměti aplikace alokuje pro obsloužení požadavku. Tato hodnota byla zjištěna funkcí get_memory_usage().



Obrázek 5 Alokovaná paměť

V tomto testu vychází nejhůře Symfony, který zabírá přes 7 MB paměti. Na podobných hodnotách je Laravel a Nette. Nette, ale v tomto testu má větší výsledek než Laravel. Laravel je tak v tomto testu nejméně paměťově náročný.

V posledním testu se zjišťuje, kolik souborů je zapojeno do obsloužení požadavku. Počet připojených souborů při požadavku na aplikaci bylo zjištěno funkcí get_includet_files().



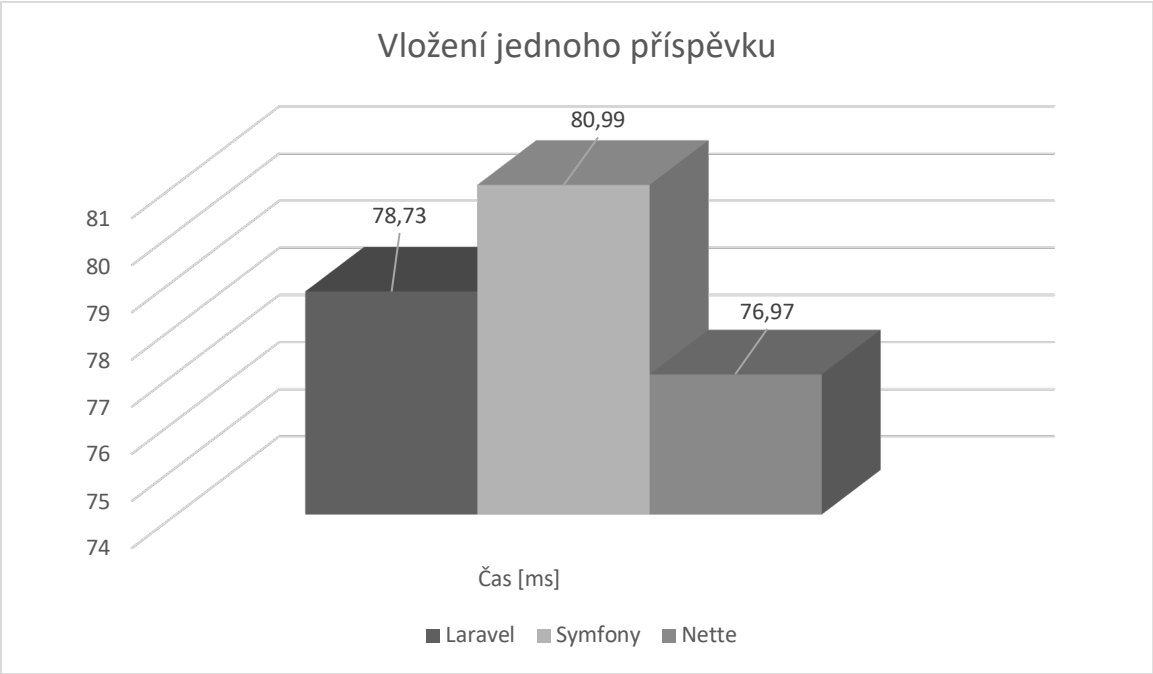
Obrázek 6 Připojené soubory

V tomto grafu je na první pohled vidět, že Nette používá na jeden požadavek nejméně souborů, naopak Symfony jich používá skoro 5krát tolik. Tyto soubory pak ovlivňují rychlost aplikace.

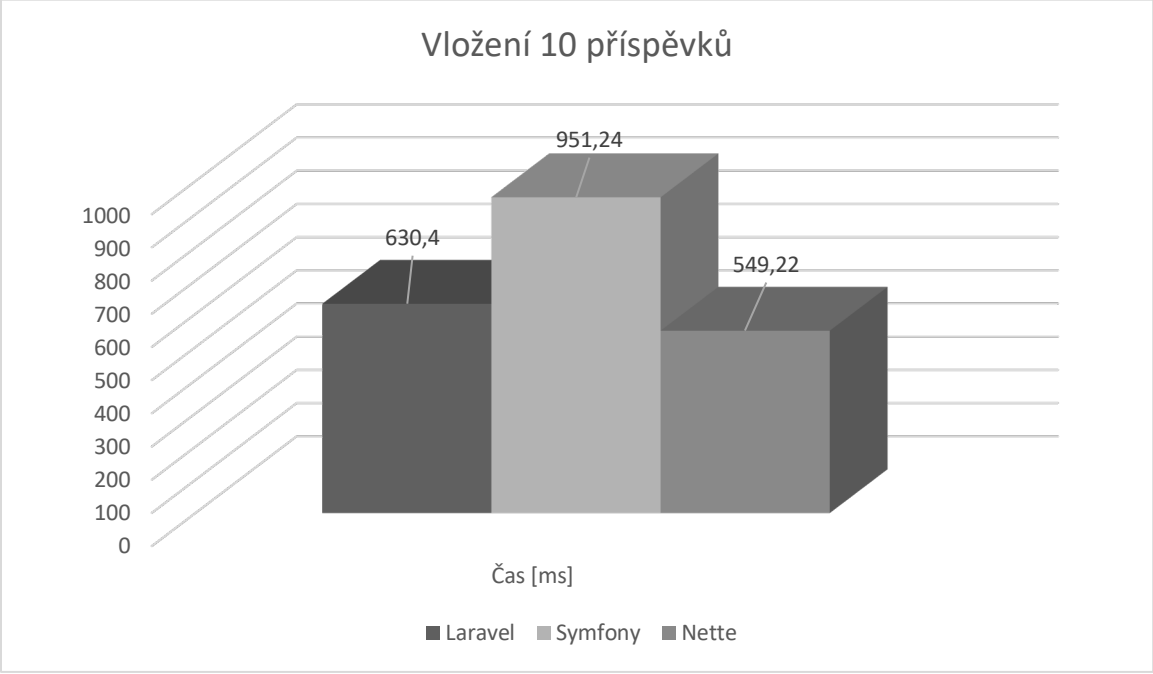
8.8.1. CRUD operace

Databáze pro testování rychlosti práce frameworků s databázemi se skládá ze tří tabulek: uživatelé, příspěvky a komentáře. Uživatel může vytvářet příspěvky a psát k nim komentáře. Příspěvek obsahuje komentáře. Jelikož každý framework pracuje s databází trochu jinak a uchovává si různé hodnoty, tak každý framework má data uložena ve své databázi.

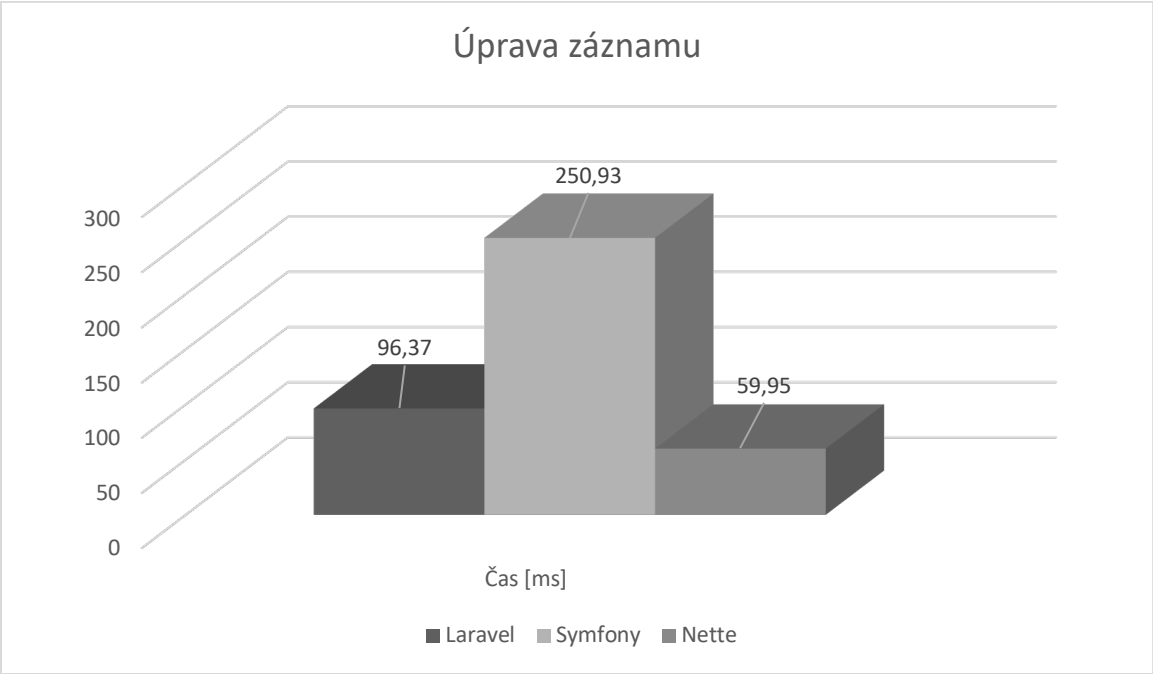
Bylo změřeno, za jaký čas framework uloží, aktualizuje, nebo vybere příspěvky z databáze. V následujících grafech jsou průměrné časy, které byly získány z 20 měření.



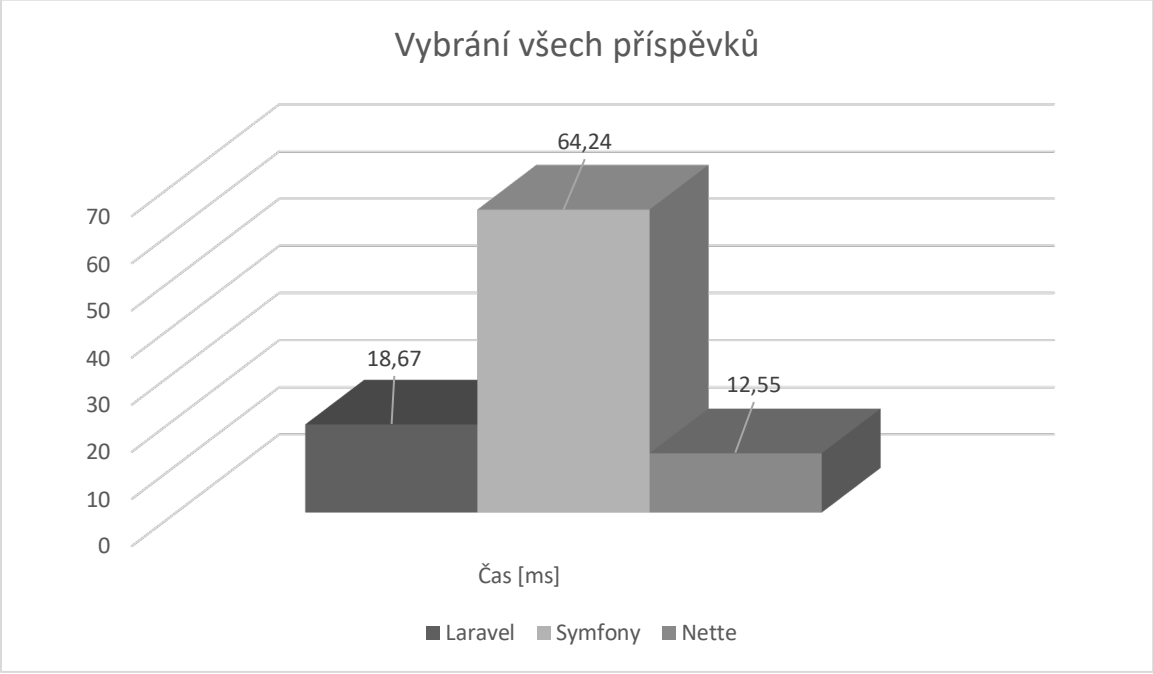
Obrázek 7 Vložení záznamu



Obrázek 8 Vložení 10 záznamů



Obrázek 9 Úprava záznamu



Obrázek 10 Smazání záznamu

8.9. Vyhodnocení Bechmarkingu

Měření benchmarkingu u php frameworků je na internetu nespočet. Avšak většina těchto testů nemá v měření frameworky, které jsou porovnávány v této práci. Proto byly zvoleny výsledky pro porovnání z testu provedeného členy týmu frameworku Phalcon z roku 2017. Výsledky jsou dostupné na blogu Phalconu [45]. Jejich výsledky pomohou při porovnání těchto testů.

V první řadě je nutno podotknout, že jejich aplikace nebyly hostovány na veřejném serveru, ale na vlastním virtuálním serveru, takže se výsledky řádově liší.

V prvním testu, kde se měřil počet nejvíce obsloužených požadavků za jednu vteřinu byl nejlepší Nette, který dokázal obsloužit skoro 2krát více požadavků než Symfony a o 3 více než Laravel. Laravel v testu porazil Symfony a tím se výsledek oproti výsledkům z roku 2017 liší.

V testu v této práci se měří místo 1000 „jen“ 100 požadavků. Ale i při tomto menším množství požadavků je opět nejrychlejší Nette. Který stejně jako v testu z roku 2017 má několikrát kratší čas než Laravel a Symfony. Laravel je v tomto testu oproti výsledků z roku 2017 lepší než Symfony.

Aplikace mají oproti roku 2017 větší paměťové nároky. A v tomto testu vyšel nejlépe Laravel, který patřil v roce 2017 mezi poslední. Za ním je Nette, který měl z těchto tří v roce 2017 nejlepší výsledek. Symfony v tomto testu je zase poslední a oproti testu z roku 2017 se rozdíl mezi ostatními zvětšil.

Počet připojených souborů má stejně jako v roce 2017 nejméně Nette, kde se počet liší o pár souborů. A pořád se zde vyskytuje mezi ostatními veliký rozdíl. Symfony má největší počet připojených souborů a rozdíl mezi testy se liší o několik desítek. Laravelu se také zvýšil počet souborů, ale ne tak dramaticky.

U crud operací je vidět, že nejrychleji s databází pracuje framework Nette, který má nejkratší průměrné časy u všech crud operací. Druhý v pořadí je Laravel, který má podobné výsledky u vložení a vybrání jako Nette. Symfony je ve všech měřeních nejpomalejší a mimo vkládání záznamů do databáze je dokonce několikrát pomalejší než ostatní frameworky.

8.10. Bodové hodnocení

Za vytvoření autentizace v praktické části Laravel získal 2 body, Symfony a Nette 1.
V testech v rychlosti aplikace Nette získalo 4 body, Laravel 3 a Symfony 1. Za CRUD operace získalo Nette 4 body, Laravel 3 a Symfony 1.

Po sečtení bodů ze všech kritérií se na první příčku dostal framework Laravel se 31 body z 36 možných, druhou příčku obsadil český Nette s 25 body a na třetím místě s 24 body skončilo Symfony.

9. Závěr

Cílem práce bylo popsat a porovnat tři vybrané PHP frameworky podle vybraných kritérií, které jsou sepsány ve druhé kapitole. V první části práce byly popsány webové technologie a MVC architektura. Čtenáři bylo vysvětleno, co to je framework. Dále byly popsány výhody a nevýhody frameworku a také byl popsán rozdíl mezi knihovnou a frameworkem.

V další části, byl každý framework představen. U každého byly popsány porovnávané vlastnosti a základní charakteristiky. Následně byly ke každému sepsány plusy a mínusy. Další kapitola se zaobírala nejčastějšími útoky na webovou aplikaci a zjišťovalo se, jaké mají proti těmto útokům frameworky zabezpečení. Ke každému útoku bylo na příkladě (pokud to bylo možné) zobrazena obrana frameworku. V další části byla shrnuta teoretická část práce, kde byla zhodnocena všechna doposud probraná kritéria.

V praktické části byl popsán postup, jakým byly aplikace vytvořeny. Dále se v každé aplikaci vytvořila základní funkcionalita jako je směrování a přihlašování. Byly popsány rozdíly v syntaxi a způsobu použití každého frameworku. V další části byly provedeny praktické testy. V prvním testu se zjišťovalo, jak rychle dokáže framework obsluhovat požadavky a jak je paměťově náročný při obsluze těchto požadavků. Ve druhém testu se ve vytvořených aplikacích měřila rychlost CRUD operací.

Cílem práce bylo představit čtenáři jednotlivé frameworky, popsat jejich základní vlastnosti a porovnat je. Z výsledků porovnání byly čtenáři poskytnuty informace, díky kterým si může snadněji vybrat svůj framework pro tvorbu webové aplikace. Dalším cílem bylo vybrat jeden framework, který je nejvhodnější pro vývoj aplikace. Podle hodnocení v teoretické a praktické části lze zvolit framework Laravel jako ideální framework pro tvorbu webové aplikace v dnešní době. Jelikož disponuje velice kvalitní dokumentací, má silnou základnu vývojářů, používá nejnovější technologie, je poměrně snadné se v něm orientovat a začít vytvářet aplikace. Laravel také získal dobré výsledky v praktických testech.

10. SEZNAM ZDROJŮ

[1]Lukáš Fiala, Framework Laravel, Olomouc 2015, Bakalářská práce, Univerzita Palackého v Olomouci

[2]Jiří Rebenda, Srovnání PHP frameworků Phalcon, Nette a Zend , Praha 2014, Bakalářská práce, Vysoká škola ekonomická v Praze

[3]Indeed [online] Indeed © 2019 [cit. 2-3-2019]. Dostupné z: https://cz.indeed.com/PHP-Program%C3%A1tor-jobs-in-Hlavn%C3%AD-m%C4%9Bsto-Praha

[4]PHP: History of PHP - Manual. Php.net[online]. The PHP Group © 2001-2018. [cit. 06-11-2018]. Dostupné z: http://php.net/manual/en/history.php.php

[5]Usage of server-side programming languages for websites. W3techs. [online]. W3techs © 2019 [cit. 03-03-2019]. Dostupné z: https://w3techs.com/technologies/overview/programming_language/all

[6]PHP frameworky. Programujte.com [online]. Programujte.com, © 2008 [cit. 06-10-2018]. Dostupné z: http://programujte.com/clanek/2008022000-php-frameworky/

[7]Frameworky vs DevStacky. Zdrojak.cz [online]. Devel.cz Lab s.r.o. © 2015. [cit. 16-06-2018]. Dostupné z: https://www.zdrojak.cz/clanky/frameworky-vs-devstacky/

[8]Lekce 1 - Popis MVC architektury. itnetwork. [online]. itnetwork.cz. © 2018 [cit. 06-11-2018]. Dostupné z: https://www.itnetwork.cz/php/mvc/objektovy-mvc-redakcni-system-v-php-popis-architektury

[9]Úvod do architektury MVC. In: Zdroják.cz [online]. Devel.cz Lab s.r.o. © 2009 [cit. 06-10-2018]. Dostupné z: https://www.zdrojak.cz/clanky/uvod-do-architektury-mvc/

[10]The Best PHP Framework for 2015: SitePoint Survey Results, SitePoint Pty [online]. SitePoint Pty © 2019 [cit. 10-04-2019]. Dostupné z: https://www.sitepoint.com/best-php-framework-2015-sitepoint-survey-results/

[11]Porovnání. Google Trends. [online] Google © 2018. [cit. 18-01-2019] Dostupné z: https://trends.google.com/trends/explore?q=%2Fm%2F0jwy148,%2Fm%2F04n23m7,%2Fm%2F09cjcl

[12]Laravel. Wikipedia [online]. San Francisco (CA): Wikimedia Foundation, © 2019 [cit. 08-03-2019]. Dostupné z: https://cs.wikipedia.org/wiki/Laravel

[13]History of Laravel PHP framework, Eloquence emerging. Maksim Surguy - Maks Surguy's [online]. Maks Surguy © 2018. [cit. 06-11-2018]. Dostupné

z: https://maxoffsky.com/code-blog/history-of-laravel-php-framework-eloquence-emerging/

[14]Laravel 5 Essentials. Birmingham: Packt Publishing Limited, 2015. ISBN: 978-1785283017.

[15]Laravel Release Process. Laravel-news [online]. Eric L. Barnes © 2019 [cit. 07-11-2018]. Dostupné z: https://laravel-news.com/laravel-release-process

[16]Jak vyzrát na open source software - 2. část. [online]. © 2016 EPRAVO.CZ, a.s. [cit. 07-11-2018]. Dostupné z: https://www.epravo.cz/top/clanky/jak-vyzrat-na-open-source-software-2-cast-102708.html

[17]Installation. Laravel [online]. Taylor Otwell © 2019 [cit. 03-03-2019]. Dostupné z: https://laravel.com/docs/5.7/installation

[18]11 Best PHP Frameworks for Modern Web Developers in 2018. Coder's Eye [online]. © 2018. [cit. 07-11-2018]. Dostupné z: https://coderseye.com/best-php-frameworks-for-web-developers/

[19]Artisan Console. Laravel [online]. Taylor Otwell © 2019 [cit. 03-03-2019]. Dostupné z: https://laravel.com/docs/5.7/artisan

[20]Laravel Overview. Tutorialspoint [online]. Tutorialspoint © 2018. [cit. 07-11-2018]. Dostupné z: https://www.tutorialspoint.com/laravel/laravel_overview.htm

[21]Database: Getting Started. Laravel [online]. Taylor Otwell © 2019 [cit. 08-03-2019]. Dostupné z: https://laravel.com/docs/5.7/database

[22]Nette Framework. Wikipedia. [online]. San Francisco (CA): Wikimedia Foundation, 2001-2019 [cit. 03-03-2019]. Dostupné z: https://cs.wikipedia.org/wiki/Nette_Framework

[23]Vyšel Nette Framework 2.0 final. Zdroják.cz [online]. Devel.cz Lab s.r.o. © 2018. [cit. 07-11-2018]. Dostupné z: https://www.zdrojak.cz/zpravicky/vysel-nette-framework-2-0-final/

[24]Download. Nette Framework [online]. Nette Foundation © 2018 [cit. 07-11-2018]. Dostupné z: https://nette.org/cs/download#toc-starsi-verze

[25]Co se chystá v Nette?. phpFashion [online]. © 2018. [cit. 07-11-2018]. Dostupné z: https://phpfashion.com/co-se-chysta-v-nette

[26]Licenční politika. Nette Framework. [online]. Nette Foundation © 2018. [cit. 07-11-2018]. Dostupné z: https://nette.org/cs/license#toc-new-bsd-license

[27]Začínáme. Nette Framework. [online]. Nette Foundation © 2018. [cit. 07-11-2018]. Dostupné z: https://doc.nette.org/cs/2.4/quickstart/getting-started

[28]Tracy. Nette Framework. [online]. Nette Foundation, © 2019 [cit. 03-03-2019]. Dostupné z: https://tracy.nette.org/cs/

[29]Nette Database. Nette Framework. [online]. Nette Foundation, © 2017 [cit. 03-03-2019]. Dostupné z: https://doc.nette.org/cs/2.0/database

[30]Introduction. Php.net [online]. The PHP Group © 2001-2019 [cit. 03-03-2019]. Dostupné z: http://php.net/manual/en/intro.pdo.php

[31]Nette Database. Nette Framework [online]. Nette Foundation, © 2019 [cit. 08-03-2019]. Dostupné z: https://doc.nette.org/cs/2.4/database

[32]Symfony in Brief. Symfony [online]. SensioLabs © 2018. [cit. 08-11-2018]. Dostupné z: https://symfony.com/legacy/doc/book/1_0/en/01-Introducing-Symfony#chapter_01_sub_who_made_symfony_and_why

[33]Symfony Roadmap. Symfony [online]. SensioLabs © 2018 [cit. 08-11-2018]. Dostupné z: https://symfony.com/roadmap

[34]Symfony Code License. Symfony [online]. SensioLabs, © 2019 [cit. 08-03-2019]. Dostupné z: https://symfony.com/doc/current/contributing/code/license.html

[35]Introduction. Symfony [online]. SensioLabs © 2019. [cit. 08-03-2019]. Dostupné z: https://twig.symfony.com/doc/2.x/intro.html

[36]Databases and the Doctrine ORM. Symfony [online]. SensioLabs, © 2019 [cit. 08-03-2019]. Dostupné z: https://symfony.com/doc/current/doctrine.html

[37]Installing & Setting up the Symfony Framework. Symfony [online]. SensioLabs, © 2019 [cit. 08-03-2019]. Dostupné z: https://symfony.com/doc/current/setup.html

[38]Bezpečnost na webu – přehled útoků na webové aplikace. Zdrojak.cz [online]. Devel.cz Lab s.r.o. © 2008 [cit. 16-06-2018]. Dostupné z: https://www.zdrojak.cz/clanky/prehled-utoku-na-webove-aplikace/

[39]Database: Query Builder. Laravel [online]. Taylor Otwell © 2018. [cit. 08-11-2018]. Dostupné z: https://laravel.com/docs/5.7/queries

[40]Blade Templates. Laravel [online]. Taylor Otwell, © 2019 [cit. 08-03-2019]. Dostupné z: https://laravel.com/docs/5.7/blade

[41]Zabezpečení před zranitelnostmi. Nette Framework [online]. Nette Foundation © 2018. [cit. 16-06-2018]. Dostupné z: https://doc.nette.org/cs/2.4/vulnerability-protection

[42]How to Escape Output in Templates. Symfony [online]. SensioLabs, © 2019 [cit. 08-03-2019]. Dostupné z: https://symfony.com/doc/current/templating/escaping.html

[43]CSRF Protection. Laravel [online]. Taylor Otwell © 2018. [cit. 08-11-2018]. Dostupné z: https://laravel.com/docs/5.7/csrf

[44]How to Implement CSRF Protection. Symfony [online]. SensioLabs, © 2019 [cit. 08-03-2019]. Dostupné z: https://symfony.com/doc/current/security/csrf.html

[45]Benchamrking Phalcon, Phalcon blong. [conline]. Phalcon © 2019 [cit. 19-02-2019]. Dostupné z: https://blog.phalconphp.com/post/benchmarking-phalcon

11. Seznam obrázků a grafů

12. Seznam tabulek

13. Přílohy

K bakalářské práci jsou přidány 2 CD s vytvořenými aplikacemi a jejich databázemi, které byly použity pro porovnání v praktických testech.

14. Zadání práce

Univerzita Hradec Králové
Fakulta informatiky a managementu
Akademický rok: 2018/2019

Studijní program: Aplikovaná informatika
Forma: Prezenční
Obor/komb.: Aplikovaná informatika (ai3-p)

Podklad pro zadání BAKALÁŘSKÉ práce studenta

PŘEDKLÁDÁ:	ADRESA	OSOBNÍ ČÍSLO
Štípek Patrik	Jungmannova 1507, Hradec Králové - Pražské Předměstí	I1600618

TÉMA ČESKY:

Porovnání PHP frameworků

TÉMA ANGLICKY:

Comparison of PHP frameworks

VEDOUCÍ PRÁCE:

Mgr. Daniela Ponce, Ph.D. - KIT

ZÁSADY PRO VYPRACOVÁNÍ:

Student na základě předchozích analýz dostupných PHP frameworků zvolí kritéria, podle kterých vybere PHP frameworky pro porovnání, stanoví kriteria a postup pro jejich porovnání, analyzuje tyto frameworky, sestaví vyhodnocení a svá zjištění formuluje v podobě doporučení.

Osnova
1. úvod
2. výběr frameworků
3. vlastnosti frameworků
4. praktické porovnání
5. vyhodnocení výsledků
6. závěr

SEZNAM DOPORUČENÉ LITERATURY:

Luke Welling, Laura Thomson: Mistrovství PHP a MySQL. 1. vydání, Brno. Computer Press, 2017, ISBN 978-80-251-4892-1
Martin Bean: Laravel 5 essentials, 1. vydání. Packt Publishing Limited, 2015, ISBN 978-1-78528-301-7
Steve Prettyman: Learn PHP 7, 1. vydání. Apress, 2015, ISBN 978-1-4842-1730-6

Podpis studenta: ..... **Datum:** 30.3.2018

Podpis vedoucího práce: Ponce **Datum:** 30.3.2018