Univerzita Hradec Králové
Fakulta informatiky a managementu

Frameworky pro tvorbu CMS systémů

Bakalářská práce

Autor: Tomáš Budina
Studijní obor: Informační management – kombinovaně

Vedoucí práce: prof. Ing. Vladimír Bureš, Ph.D., MBA

Hradec Králové Červen 2021

Prohlášení:

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

Budina Tomáš

V Hradci Králové dne 6.6.2021

Poděkování:

Děkuji vedoucímu bakalářské práce prof. Ing. Vladimíru Burešovi, Ph.D., MBA za jeho odborné a metodické vedení práce.

Anotace

Hlavním cílem práce je analýza vybraných PHP frameworků a ukázka implementace CMS systému pro vybraný PHP framework. Teoretická část obsahuje úvod do CMS systémů, jejich rozdělení a výběr populárních CMS systémů založených na PHP frameworku. Dále popis týkající se všech použitých technologií, obecně frameworky, typy a jejich vybrané funkcionality. Praktická část pak obsahuje analýzy a porovnání vybraných PHP frameworků dle daných kritérií a nadále návrh designu, návrh databáze a adresářovou strukturu spojenou s implementací CMS systému pro jeden vybraný framework. V implementaci jsou představeny jednotlivé požadavky a všechny moduly CMS systému a popis funkčnosti.

Annotation

Title: Frameworks for development of CMS systems

The main goal of the work is the analysis of selected PHP frameworks and a demonstration of the implementation of a CMS system for a selected PHP framework. The theoretical part contains an introduction to CMS systems, their division and selection of popular CMS systems based on PHP framework. Further description concerning all used technologies, general frameworks, types and their selected functionalities. The practical part then contains analyzes and comparisons of selected PHP frameworks according to the given criteria, as well as design proposal, database design and directory structure associated with the implementation of a CMS system for one selected framework. The implementation presents the individual requirements and all modules of the CMS system and a description of functionality.

Klíčová slova: PHP, Nette, CMS, Framework, analýza PHP frameworků, implementace CMS

Keywords: PHP, Nette, CMS, Framework, analysis of PHP frameworks, CMS implementation

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Činnost v podnicích je často velice komplikovaná. Bez ohledu na to, jaký produkt nebo službu podnik poskytuje, je zapotřebí mít toho stále hodně na paměti, a to ještě předtím, než začne podnik uvažovat o tom, jak jeho web souvisí se všemi aspekty daného podnikání. Každá taková společnost, která dnes působí, je odlišná. Všechny čelí společné jedné výzvě, aby si v dnešním obchodním prostředí udržely konkurenceschopnost. K tomu potřebují spolehlivý a efektivní způsob ukládání a přístupu k informacím. Toto je místo, kde vstupují do hry systémy ERP. Věci mohou být ještě více matoucí díky neustálému používání zkratek jako CRM, ERP, CMS a PIM. Co všechny tyto kombinace písmen znamenají, s čím souvisí, a jak lze pomocí jednotlivých částí webu integrovat všechny různé části podnikání?

Systémy ERP mají na starost integrovat všechny aspekty podniku do jednoho komplexního informačního systému, ke kterému mají přístup jednotlivci v celé organizaci. Díky efektivnímu softwaru ERP mohou vlastníci podniků a vedoucí pracovníci automatizovat a zefektivnit zdlouhavé úkoly všem zaměstnancům a pomoci zaměstnancům stát se produktivnějšími a úspěšnějšími v jejich rolích a získat přehled o vnitřním fungování jejich operací v reálném čase.

Mnoho firem používá řešení pro správu vztahů se zákazníky a různé druhy programů, ke zvládání interakcí se svými zákazníky, včetně udržování kontaktních údajů, sledování prodeje a poskytování zákaznické podpory. Pomocí nástrojů CRM lze sledovat všechna zákaznická data a udržovat nezbytnou komunikaci, která je třeba, aby zákazníkům byly poskytnuty nejlepší služby.

Většina podnikových společností používá ke správě svých webových stránek systém pro správu obsahu, protože CMS poskytuje nástroje pro dynamickou správu obsahu a zahrnuje funkce, jako je eCommerce, personalizace, cílený obsah, pracovní postup a další. Pomocí tohoto nástroje je možné upravovat webové stránky a produkty eCommerce prostřednictvím administračního centra a používat výše uvedené funkce k oslovení zákazníků. Je nezbytný web řízený CMS. V tomto případě by použití CMS kromě všech informací o produktu eCommerce ovládalo a ukládalo, jak obsah, tak aktiva pro web.

Aby firma uspěla v dnešním stále konkurenceschopnějším a složitějším obchodním prostředí, musí zavést systémy, které ji umožní snadné usměrnění, automatizaci a škálování. Ve světě, kde produktivita závisí na vzájemně propojených věcech, si podniky uvědomily nutnost mít soudržný systém pro své obchodní informace. Je to řešení, které eliminuje datové toky a neustálou potřebu předávat data a informace tam a zpět mezi odděleními.

Individuální úspěch každého obchodního sektoru je vzájemně propojen, takže je logické, že by měl být také propojen jejich pracovní tok. Jediný systém pro podnikání umožňuje zaměstnancům produktivněji sdílet a používat data a zajišťuje, aby byla každá část organizačního stroje správně synchronizována.

Tato bakalářská práce se zaměřuje na detaily CMS systému pro správu webového obsahu.

2 Metodika práce

2.1 Cíl bakalářské práce

Daným cílem bakalářské práce je analýza vybraných PHP frameworků podle vybraných kritérií a ukázka implementace CMS systému pro jeden vybraný PHP framework. Popsat nejdůležitější funkce vybraných frameworků a jejich vlastnosti. Na základě kritérií poté zhodnotit frameworky podle dané škály hodnocení. K praktické části připadá na starost tedy vytvoření jednoduchého CMS systému, ve kterém se promítnou základní funkce CMS. Bakalářská práce seznámí čtenáře se základními funkcemi porovnávaných frameworků a poskytne pomoc při výběru CMS systémů pro možný podnik.

2.2 Kritéria porovnání

Níže jsou uvedená kritéria, která byla zvolena k důvodu porovnání vybraných frameworků. Vybrané frameworky byly tedy porovnány podle zvolených kritérií a tato kritéria byla poté vyhodnocena. U všech kritérií je popsán důvod výběru a jeho základní definice. Jedná se konkrétně o údaje nalezené v dokumentacích, na jejich oficiálních stránkách, nebo také na diskusních fórech oněch daných frameworků. Každé kritérium je hodnoceno ve škále 1-5 kde 1 je velmi uspokojivé a 5 zcela nedostatečné. Pokud dané kritérium splňuje vše, je mu uděleno hodnocení velmi uspokojivé, pokud nesplňuje požadavky, je mu dáno hodnocení zcela nedostatečné. Všechna kritéria byla testována na systému macOS Big Sur verze 11.1, na verzi PHP 8.2.

Systémové požadavky

Pro správný chod každého frameworku je potřeba dbát pozornost na minimální a doporučené požadavky. Požadavky každého frameworku se zcela mohou lišit. Dále se v této kapitole podívám na kompatibilitu napříč jednotlivými verzemi PHP.

Licence

Jako jedno z dalších kritérií je zvolena licenční politika daného frameworku. Toto kritérium se zaměří na jednotlivé licence při vytváření projektu.

Dokumentace a uživatelská podpora

Další kapitola se zabývá dokumentací u frameworků. Jakým způsobem jsou dokumentace zpracovány, a zdali se v nich jedná o snadnou orientaci. Do této kapitoly byla zároveň zahrnuta přítomnost uživatelské podpory, kterou může být například diskusní fórum. To může být pro budoucí uživatele, nebo vývojáře dostupné jako zdroj informací, při řešení daných problémů.

Instalace

Toto kritérium se zaměří na doporučenou cestu, jak začít nový projekt a jakým způsobem mohu stáhnout a nainstalovat daný framework.

Pravidelné aktualizace

Jedno z mála kritérii je zaměřeno na aktuálnost vybraných frameworků. V jakých časových intervalech jsou aktualizace vydávány, a jak dlouho podpory mají starší verze frameworku. A co je potřeba pro migrace na novější verzi.

Ladění kódu a odchytávání chyb

Přehledné zpracovaní a vizualizace chyb, je určitě užitečným pomocníkem pro všechny vývojáře PHP. I toto kritérium je bráno jako jedno z nejdůležitějších.

Testování kódu

V dnešní době agilního vývoje se aktualizace systémů klidně vydávají každých 14 dní a v tomto tempu je nemožné vše řádné otestovat manuálně. To ale neznamená, že manuální testování úplně a definitivně skončilo. Jen se objem testování celkově přesunul na stranu automatizovaných testů. A právě z toho důvodu je i toto kritérium velice důležité, a proto je zaměřeno na možnost psaní a spouštění jednotlivých testů v aplikaci.

Šablonovací systém

Jedna z mnoha výhod šablonovacích systémů by mělo být ušetření a zpříjemnění práce. Další výhoda by měla být zabezpečení šablon před různými zranitelnostmi jako je například XSS.

Routování a směrování

Routování je jednou z hlavních komponent frameworku. Pomocí routování lze navrhovat URL adresy. Ty díky tomu můžou být přehlednější a přispívají například k lepším SEO výsledkům ve webové aplikaci. Framework také využívá router pro generování URL v šablonách, kdy router sestaví z jednotlivých parametrů výslednou URL.

Podpora databází

Jedním z důležitých kritérií je podpora databází. Skládání SQL dotazů, snadné získávání dat a efektivnost dotazů bez zbytečného přenášení dat. Nastavení a navázání spojení s databází. Jakým způsobem framework umožňuje pokládat dotazy. A také především množství podporovaných databází.

Balíčky a rozšíření

Jedná se o všechny možné addony, moduly, pluginy a rozšíření pro daný framework. V kritériu se zaměřím na údržbu a aktuálnost jednotlivých balíčků. Dále na počet celkových a denních stáhnutí, z čehož zároveň vyplývá aktuálnost balíčků.

Zabezpečeni

Zabezpečení je určitě jedno z nejdůležitějších a hlavních témat a je na denním pořádku. Právě proto je potřeba stanovit i toto kritérium, jakým způsobem si dokáže framework poradit se zabezpečením a jaké možnosti ochrany před různými zranitelnostmi před potenciálními útočníky nabízí.

REST API

Rozhraní REST API je způsob, jakým dva počítačové systémy komunikují přes HTTP protokol podobným způsobem jako webové prohlížeče a servery.
Aplikace implementující rozhraní REST API bude definovat jeden nebo více koncových bodů URL s doménou, portem, cestou nebo řetězcem dotazů – například: https://localhost/users/
Na libovolném koncovém bodě, který mapuje na operace vytváření, čtení, aktualizace a mazání aplikací (CRUD), lze použít různé metody HTTP: GET, POST,

PUT, DELETE. Tímto se lze jednoduše dotázat na danou adresu, která opětovně vrátí všechny uživatele ve formátu JSON.

3 Co je CMS a jak funguje?

Při výběru systému pro správu obsahu (CMS) pro realizaci podnikání je mnoho faktorů, které mohou pomoct při pochopení rozdílů mezi různými typy, jejich funkcemi a cenovými modely. CMS je aplikace, která se používá ke správě a publikování webového obsahu a umožňuje více uživatelům přispívat, vytvářet, upravovat, publikovat, aniž by museli prosit o pomoc vývojáře (1). Poskytuje také pracovní postup správy verzí a vytváření, aby byly velké globální weby více konzistentní. CMS v zásadě usnadňuje správu webových stránek lidem na všech dovednostech a technických úrovních. Prostřednictvím tohoto uživatelsky přívětivého rozhraní CMS vezme všechny vstupy a promění je ve fungující web pro návštěvníky stránek. Existují dva typy CMS: open source a proprietární. Mezi důležité funkce řešení pro správu obsahu patří.

1. Zabezpečení

Zajištění bezpečnosti podnikání před kybernetickými útoky je neuvěřitelně důležitým krokem. Útoky nejen přerušují kontinuitu všech podnikání, ale můžou stát také obrovské částky. Nedávná zpráva společnosti McAfee naznačuje (8), že v roce 2020 mohlo v důsledku počítačové kriminality dojít ke ztrátě až 1 trilionu dolarů. Když se podíváme na srovnání s rokem 2018 tato částka je skoro dvounásobná (8). Cloudové a online systémy CMS jsou stále častějším cílem kybernetických útoků. To má za následek řadu potenciálních bezpečnostních problémů, jako je narušení integrity dat, neoprávněný přístup k soukromým datům a škodlivé kódy a skripty. Většina CMS přichází s poměrně robustní sadou bezpečnostních funkcí, jako je pokročilé dvou fázové ověřování, přísná oprávnění, brány firewall a ochrana před útoky malwaru (2).

2. Vícejazyčná podpora

Systém pro správu webového obsahu, který podporuje snadné vícejazyčné a vícekanálové přepínání, nejenže tuto práci mnohem usnadňuje uživatelům, ale také umožňuje místním správcům značek a správcům obsahu provádět lokalizované

kampaně na kanálech, které nejlépe vyhovují jejich trhům, při zachování globální značky. (1)

3. Uživatelská přívětivost

Zmocnění zaměstnanců na podporu snah o globalizaci začíná zjednodušením procesu, který je k tomu využíván. Používání systému pro správu obsahu, který je intuitivní pro koncového uživatele a který umožňuje zaměstnancům rychle znovu použít komponenty daného CMS systému co nejjednodušeji, jako jsou obrázky, návrhy a zkušenosti, povzbudí týmy, aby převzaly vlastnictví místních zkušeností – zejména pokud to také umožňuje všem uživatelům používat systém v jejich preferovaném jazyce (2). Systém, který podporuje přímé pracovní toky schvalování, usnadní život manažerům značek a zase podpoří pokračující úsilí a dokonalost v procesu globalizace.

4. Testování a experimentování

Pro další podporu globálních týmů při lokalizaci značky na nové trhy je zásadní, aby mohly rychle vyhodnotit výsledky svého úsilí a na základě této zpětné vazby podniknout samostatná opatření. Nejjednodušší způsob, jak to udělat, je zajistit, aby CMS, který je využíván, měl vestavěné experimentální funkce pro snadné testování obsahu a prvků prostředí – ať už na počítači, mobilu nebo na jiných platformách (1).

5. Personalizace

Systém CMS, který umožňuje automaticky přizpůsobit prvky digitálního obsahu, jako jsou kampaně, obsah nebo produktové mřížky, poskytne větší agilitu globálním týmům, protože mohou snadno vytvářet různé varianty obsahu z webu z jednoho globálního systému (2). Rovněž to umožní podnikům větší kontrolu nad globální značkou a podpoří globalizaci ve velkém.

6. Analýza

Systém pro správu obsahu, který má vestavěný analytický modul, jakým může být například nástroj Google Analytics (9), takže marketingové týmy, tvůrci obsahu a

manažeři značek mohou snadno zjistit trendy návštěvníků a příležitosti ke zlepšení digitálního obsahu na základě jednotlivých údajů o návštěvnících a nejvíce navštěvovaných stránkách. Je ještě lepší, když systém dokáže poskytnout tyto informace na osobu, což poskytne ještě mnohem větší přesnost při optimalizaci obsahu poskytovaného všem návštěvníkům z celého světa (3).

7. Škálovatelnost

Firmy, které provozují své CMS v cloudu, budou moci své globalizační úsilí škálovat mnohem rychleji, přičemž vývojové týmy budou moci zavést aktualizace digitálního prostředí do celého světa pouhými několika kliknutími, přičemž využijí dobu provozuschopnosti a neustálá vylepšení nabízená cloudem od poskytovatelů (3).

3.1 Rozdělení CMS

3.1.1 Open source CMS

Software CMS s otevřeným zdrojovým kódem si lze stáhnout bez žádných úvodních nákladů a dalších závazků. Neexistují žádné licence ani poplatky za aktualizaci ani žádné smlouvy (4). U open source CMS však bude nutné zaplatit za:

1. Technickou pomoc při instalaci a nastavení
2. Přizpůsobení k rozšíření softwaru nad rámec základní nabídky
3. Kompatibilní šablony, doplňky a pluginy (i když mohou být k dispozici bezplatné verze)
4. Školení zaměstnanců
5. Podporu, včetně pravidelné aktualizace softwaru

Open source CMS lze naistalovat a spravovat na webovém serveru. Zatímco většina řešení funguje ihned po stáhnutí a nahrání na webový server, k dispozici je bezpočet přizpůsobení, které splňují různé obchodní potřeby, například doplňky pro webové stránky elektronického obchodování, nástroje, které můžou pomoci optimalizovat obsah pro vyhledávače nebo přizpůsobit motivy a rozvržení designu.

3.1.2 Proprietární CMS

Proprietární nebo komerční software CMS vyrábí a spravuje jedna společnost (4). Používání takového CMS obecně zahrnuje:

1. Nákup licenčního poplatku za používání softwaru
2. Placení měsíčních nebo ročních poplatků za aktualizace nebo podporu
3. Možná úhrada dalších nákladů za přizpůsobení a aktualizace, stejně jako za školení a průběžnou technickou nebo uživatelskou podporu.

Obvykle lze přizpůsobit proprietární CMS s vloženými funkcemi, i když to může být za příplatek. Pokud je to možné, ihned po stáhnutí a nahrání vyhledat takové řešení, které splňuje všechny požadavky (4). Pokud dojde k implementaci proprietárního CMS s existujícím webem nebo back-endovým systémem, neopomenout a uvědomit si, že to může vyžadovat další rozsáhlou vývojovou práci a tím pádem I větší časovou vytíženost.

3.2 Populární CMS systémy napsané v jazyce PHP

1. Wordpress

WordPress je bezplatný a otevřený systém pro správu obsahu napsaný v jazyce PHP a spárovaný s databází MySQL. Mezi funkce patří architektura podpůrných modelů a systém šablon, které se ve WordPressu označují jako Témata. WordPress byl původně vytvořen jako systém publikování blogů, ale vyvinul se tak, aby podporoval další typy webového obsahu, včetně tradičnějších e-mailových konferencí a fór, mediálních galerií, webů pro členství, systémů pro správu učení a online obchodů. WordPress používá více než 60 milionů webů (5). Je jedním z nejpopulárnějších používaných systémů pro správu obsahu. Jednou z největších výhod WordPressu je, že jde o bezplatný software s otevřeným zdrojovým kódem. Vynaložení financí za hosting je jednou z podmínek, to se odráží na ceně za používání softwaru WordPress. Další z výhod WordPressu je jeho rozšiřitelnost. Pokud se nejedná o vývojáře, lze snadno upravit svůj web díky obrovskému ekosystému motivů a pluginů (5).

2. Joomla

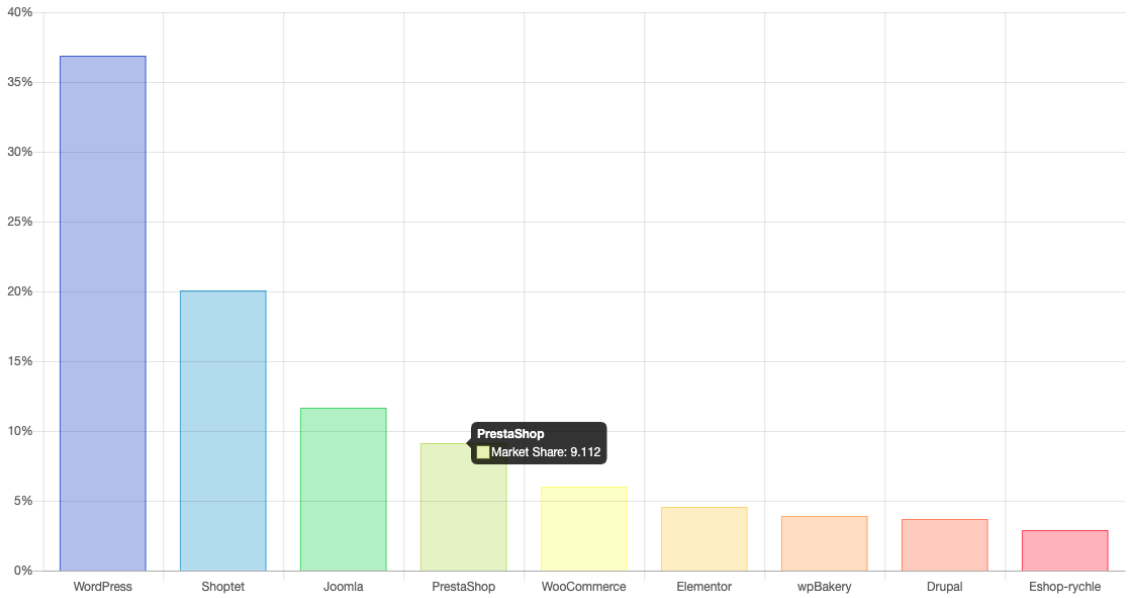
Joomla je otevřený systém pro správu obsahu. Pomáhá vytvářet výkonné dynamické webové stránky a aplikace (6). Má intuitivní rozhraní, pomocí kterého je možné využívat jeho funkce naplno. Joomla si za poslední desetiletí získala obrovskou popularitu a úspěšně se rozrostla v jeden z celosvětově nejpoužívanějších systémů. Joomla je napsána v PHP a používá databázi MySQL k ukládání dat při použití objektově orientovaných programovacích technik (6).

3. Drupal

Drupal se řadí mezi další ze softwarů pro správu obsahu. Používá se k výrobě mnoha webů a aplikací, které jsou používány každý den. Drupal má skvělé standardní funkce, jako je snadné vytváření obsahu, spolehlivý výkon a vynikající zabezpečení. Co ho však odlišuje, je jeho flexibilita; modularita je jedním z jeho hlavních principů. Jeho nástroje pomáhají vytvořit všestranný strukturovaný obsah, který dynamické webové prostředí potřebuje (7). Projekt Drupal je open source software. Kdokoli si jej může stáhnout, používat, pracovat na něm a sdílet ho s ostatními. Je postaven na principech, jako je spolupráce, globalismus a inovace. Neexistují žádné licenční poplatky (7). Drupal bude vždy zdarma. Drupal je napsaný v PHP a distribuovaný pod GNU General Public License.

4. Magento (e-commerce)

Jedná se opět o otevřenou platformu se zaměřením pro elektronický obchod napsaný v jazyce PHP. Používá několik dalších rámců PHP, jako jsou Laminas a Symfony. Zdrojový kód Magento je distribuován pod Open Software License (OSL) v3.0. Magento poskytuje online obchodníkům flexibilní systém nákupního košíku a kontrolu nad vzhledem, obsahem a funkčností jejich online obchodu. Magento nabízí výkonné nástroje pro marketing, optimalizaci pro vyhledávače a správu katalogů. Schopnost Magenta škálovat umožňuje obchodům s pouze několika produkty a jednoduchými potřebami snadno expandovat na desítky tisíc produktů a složité vlastní chování bez změny platforem. Nabízí celou řadu modulů a témat, která mohou snadno zlepšit zážitek zákazníka (10).



Obrázek 1: Poměr CMS systémů v ČR v roce 2021 (11)

3.3 Podle jakých kritérií si vybrat to správné CMS?

1. Rozhodnout se o své vizi

   Nejdůležitější otázkou, kterou si lze klást na začátku projektu CMS, je „Čeho lze dosáhnout?“ S kritickým pohledem na to, kde se vize nachází a jaké vize dosáhnout, lze začít mapovat, jaký obsah, data, nástroje a integrace již fungují, a které jsou potřeba k vytvoření konečné vize.

2. Důležitost pochopit všechny funkce CMS

   Po určení toho, čeho lze při implementaci CMS dosáhnout, je čas začít vybírat systém správy obsahu, který bude v souladu s touto vizí. Určité vlastnosti CMS mohou pomoci posunout CMS projekt rychleji a zrychlit každodenní úkoly (2).

3. Přemýšlet o nákladech

   Rozhodující součástí každého projektu CMS jsou související náklady. To jistě platí pro podnikový systém pro správu obsahu, který může činit poměrně vysokou částku. Pochopení celkových poplatků spojených s takovým projektem znamená, že nedojde k překvapení v podobě nechutných finančních překvapení (4).

4. Vypočíst si návratnosti investic

CMS je investice, a proto by měla zaručit návratnost. Dává tedy smysl, nejprve zvážit, jak vypočítat návratnost investice, jak lze rozpočet pro tento projekt odůvodnit a kolik financí lze na tento projekt uvolnit (4).

5. Určit si svůj rozpočet

   Po odhadu výnosů lze zformulovat představu o tom, jaký rozpočet by byl oprávněný, přiměřený tomuto rozsahu projektu.

4 Použité technologie

4.1 PHP

PHP (zkratka pro PHP: Hypertext Preprocessor) (12) je skriptovací jazyk, který se obecně používá při vývoji webů „na straně serveru“ (12). Zde je důležité nejprve pochopit, co je skriptovací jazyk. Skriptovací jazyky (skupina programovacích jazyků včetně PHP i jazyků jako JavaScript a Ruby) jsou podmnožinou kódovacích jazyků používaných k automatizaci procesů, které by jinak bylo nutné provést v kódu webu krok za krokem pokaždé, když k nim dojde. Skriptovací jazyky jako PHP se liší od značkovacích jazyků jako HTML a CSS v tom smyslu, že zatímco HTML a CSS určují rozložení a vzhled webových stránek, skriptovací jazyky říkají statické webové stránce (vytvořené pomocí HTML a CSS), že má „provést“ konkrétní akci.

4.2 Nette Framework

Nette Framework je open-source framework určený k vytváření webových aplikací v jazyce PHP. Původním autorem je český programátor David Grudl, ale další vývoj nyní udržuje organizace Nette Foundation. Nette je bezplatný software vydávaný pod novou licencí BSD a GNU GPL verze 2 nebo 3 (13). Blíže se na něho zaměříme v praktickém porovnání.

4.3 Nextras ORM

Jedná se o objektový přístup k databázi. Je navržena speciálně pro efektivitu a známá pro její jednoduchost a použití. Podporuje databáze, jakými jsou MySQL, PostreSQL a MS SQL Server (14).

4.3.1 Entita

Entitou se dá nazvat tzv. datová přepravka, která v zásadě obsahuje data pro každý jeden řádek tabulky. Každá entita musí implementovat rozhraní Nextras Orm Entity. Orm má také předdefinovanou třídu Nextras\Orm\Entity\Entity, která implementuje rozhraní a poskytuje další užitečné funkce. Data jsou přístupná prostřednictvím vlastností. Lze anotovat všechny vlastnosti, které by měly být k dispozici. Vlastnosti jsou definovány anotacemi Phpdoc @property (14).

```
/**
 * @property int                $id {primary}
 * @property string             $name
 * @property DateTimeImmutable $born
 * @property string|null        $web
 * @property-read int           $age
 */
class User extends Nextras\Orm\Entity\Entity
{
}
```

Vazby mezi jednotlivými tabulkami jsou poté řešeny následovně a opět za pomocí @property.

```
@property OneHasMany|Comments[] $comments          {1:m
Comments::$user}
```

Pomocí Orm můžeme definovat 4 typy vztahů.

1: m – jeden má mnoho: autor má mnoho knih.

m: 1 - mnoho má jednoho: kniha má jednoho autora.

m: m – mnoho má mnoho: kniha má mnoho značek, značka je spojena s mnoha knihami.

1: 1 – jeden má jeden: uživatel má jedno nastavení, reference pro související entitu je uložena pouze na straně, která je označena jako hlavní.

4.3.2 Repozitář

Pomocí repozitáře lze připravit data a vytvářet z nich různé kolekce. Jako přidat podmínky pro filtrování, seřazení anebo načíst entity z kolekce. Nad repozitářem nelze definovat jednotlivé MySQL dotazy k tomu bude sloužit následující Mapper (14).

4.3.3 Mapper

Mapper nebo také mapovač je poslední vrstva ORM, která komunikuje s databází. Na rozdíl od úložiště je mapovač specifický pro úložiště, a ještě více také pro databázi. Všechno specifické pro databázi by mělo být implementováno pouze v této vrstvě mapovače. V mapovači tedy lze vytvářet specifické MySQL dotazy, které nejde vytvořit pomocí Orm. Orm přichází se dvěma výchozími a předdefinovanými mapovači: prvním je ArrayMapper, který pracuje právě s polem PHP, a druhý je DbalMapper, který používá abstrakční vrstvu databáze Nextras Dbal (14).

4.4 Ajax

Ajax je známá zkratka pro asynchronní vykonávání JavaScriptu a XML. Hlavní myšlenkou Ajaxu je, že data lze načíst bez obnovení celé stránky. Ajax je obvykle zaměňován za typ programovacího jazyka, ale ve skutečnosti se jedná o rozšíření a implementaci JavaScriptu. Termín „Ajax“ je spojen pouze se speciálními metodami (funkcemi) JavaScriptu, které umožňují načtení věcí bez obnovení celé stránky. Ajax se stal důležitou součástí moderního webu. Díky tomu je komunikace se serverem mnohem pohodlnější.

4.5 jQuery

jQuery je knihovna, která umožňuje rychlejší a snazší vytváření webových stránek a webových aplikací. S jQuery lze často napsat jeden řádek kódu, aby bylo dosaženo k tomu, co by zabralo 10-20 řádků běžného kódu v JavaScriptu. jQuery je sám napsán v JavaScriptu a je dodáván ve formě jediného souboru .js, na který je odkazováno z webové stránky. Kód JavaScript poté přistupuje do knihovny voláním různých funkcí jQuery. Je zdarma ke stažení, používání je licencováno dvojí licencí v rámci licencí MIT a GPL (15). Další silnou stránkou jQuery je, že usnadňuje psaní JavaScriptu, který funguje v mnoha různých prohlížečích. Nekompatibilita mezi populárními prohlížeči, jako jsou Chrome, Firefox a Safari, znamená, že pro každý prohlížeč je nutností často napsat různé bloky kódu JavaScript. S jQuery však lze jednoduše zavolat příslušnou funkci a nechat jQuery zabývat se spuštěním kódu v různých prohlížečích.

4.6 Bootstrap

Bootstrap je kombinace HTML, CSS a Javascript navržená jako open-source framework pro vytváření čistých uživatelských rozhraní. Bootstrap vytvořil a stále aktivně podporuje tým Twitter. Bootstrap se stal jedním z nejpopulárnějších front-endových frameworků a open source projektů na světě. Jednoduše řečeno, Bootstrap je obrovská sbírka opakovaně použitelných a univerzálních částí kódu, které jsou napsány v CSS, HTML a JavaScript. Jelikož se také jedná o framework, všechny základy jsou již položeny pro responzivní vývoj webu a všichni vývojáři musí vložit kód do předdefinovaného mřížkového systému. Součástí balíčku je

několik bezplatných nástrojů, které návrhářům umožňují vytvářet běžnější komponenty webového rozhraní a také reagovat na ně, což zvyšuje univerzálnost rámce. Obvykle ušetří nutnost psát dlouhé řetězce zejména kódu CSS. Je tím získáno více času a kapacity na práci na designu samotných webových stránek.

4.7 GoogleAnalytics API

Sada protokolů a nástrojů určených k extrakci dat z účtu Google Analytics do vlastních skriptů nebo programů pro automatizovanější a efektivnější vytváření přehledů a analýz. Pomocí, kterých lze například vytvářet vlastní grafy a nadále zpracovávat data (9).

4.8 MySQL

Databázový software MySQL je open-source, používá GPL licenci (16). Důležité je, že je nabízen ve dvou různých edicích: open-source MySQL Community Server, lze stáhnout, přistupovat ke zdrojovému kódu a používat zdarma. Proprietární MySQL Enterprise edition a další komerční produkty, které vyžadují roční předplatné a zahrnují profesionální podporu a mnoho dalších výhod. MySQL je systém správy relačních databází založený na SQL – Structured Query Language (16). Aplikace se používá pro širokou škálu účelů, včetně datových skladů, elektronického obchodování a aplikací pro protokolování. Nejběžnější použití pro MySQL je však pro účely webové databáze. Lze jej použít k uložení čehokoli od jediného záznamu informací až po celý inventář dostupných produktů pro online obchod. Ve spojení se skriptovacím jazykem, jako je PHP, je možné vytvářet webové stránky, které budou v reálném čase interagovat s databází MySQL, aby uživateli webu rychle vypsaly a zobrazily dané informace uložené v této databázi (16).

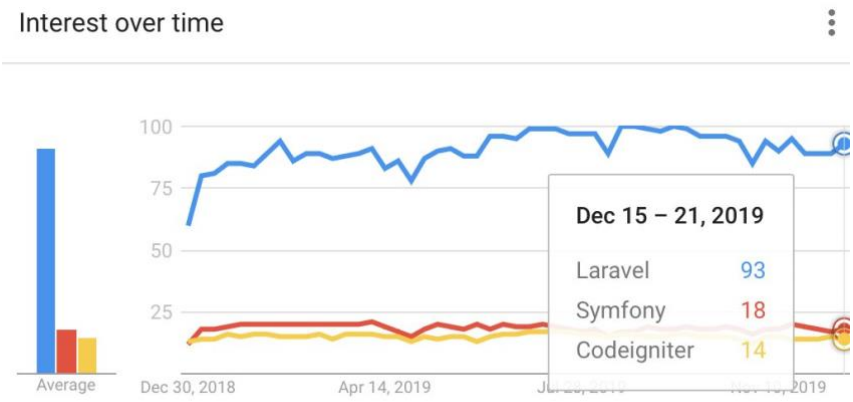
4.9 PhpStorm

PhpStorm je IDE, které je navrženo speciálně ke spolupráci s PHP, a které zároveň podporuje všechny typy nástrojů souvisejících s PHP a také nástroje pro ladění a testování. Je tedy speciálně navrženo pro jazyk PHP, vývojářům, kteří chtějí vytvářet webové aplikace a projekty související s PHP. PhpStorm je přenosná platforma IDE pro PHP, postavená společností JetBrains se sídlem v České republice. PhpStorm je

napsán v jazyce Java. Pro větší efektivitu tohoto IDE je možno využít z mnoha pluginů a rozšíření vytvořených pro PhpStorm nebo je možno napsat vlastní pluginy. IDE se také připojuje k externím zdrojům, jako je XDebug. Zahrnuje plnohodnotný editor SQL s upravitelnými výsledky dotazů. PhpStorm dokáže navíc pracovat s více PHP frameworky, jako jsou právě Nette, Symfony, Laravel, Drupal, Magento, WordPress, Joomla a mnoho dalších.

5 Analýza a porovnání PHP webových frameworků

Je těžké získat definitivní seznam všech PHP frameworků. Na světě jich momentálně existuje více jak 40 (17), ale některé z nich jsou lépe popsány jako systémy pro správu obsahu a nepochybně jich je mnohem více. Za účelem porovnání byly vybrány tři následující frameworky. Jako první byl zvolen framework Laravel (18). Momentálně je nejpoužívanějším frameworkem na trhu, k jeho podpůrným použitím lze využít přes 15 tisíc balíčků pro projekt. Druhým frameworkem je Symfony (19). I u tohoto frameworku lze nalézt a použít přes 4 tisíce podpůrných balíčků. I přes momentální oblíbenost Laravelu má Symfony stále aktivní bázi vývojářů, okolo 600 tisíc vývojářů. Třetím a zároveň posledním frameworkem je Nette (20). I přestože se před Nette tlačí framework Codeignite a celkově upadá byl vybrán tento framework, má poměrně širokou bázi vývojářů a podporovatelů předně v České republice a na Slovensku a k tomu můžeme stále nalézt několik tisíc podpůrných balíčků. Tato kapitola je určena na porovnání výše vybraných frameworků dle daných kritérií.



Obrázek 2: Popularita Frameworků (17)

5.1 Symfony

Systémové požadavky

Před tím, než lze začít vytvářet první nebo upravovat existující aplikace je doporučeno zkontrolovat si doporučené systémové požadavky pro aktuální verzi frameworku. Pro aktuální verzi frameworku je doporučena instalace PHP 7.2.5 nebo

vyšší. Také je doporučeno mít nainstalovaný nástroj Composer, který se používá pro instalaci balíčků PHP (19).

Symfony také poskytuje nástroj ke kontrole, zda váš systém splňuje doporučené požadavky (19).

```
symfony check:requirements
```

Licence

Framework Symfony je publikován pod licencí MIT (21).

„Použití Softwaru bez omezení, mimo jiné včetně práv používat, kopírovat, upravovat, slučovat, publikovat, distribuovat, sublicencovat a / nebo prodávat kopií Softwaru a umožnit osobám, kterým je Software k dispozici za tímto účelem, za následujících podmínek: pokud Výše budou zahrnuty ve všech kopiích nebo podstatné části Softwaru."" (21).

Dokumentace a uživatelská podpora

Symfony má hned několik zdrojů informací. Jednou je celkem výborně zpracovaná dokumentace, která důkladně popisuje instalaci frameworky a jeho jednotlivé části. Orientace v manuálu je jednoduchá bez výraznějších problémů. Jelikož je momentálně komunita okolo symfony velká, lze vyhledat také rady na stackoverflow pro symfony (22), nebo komunikovat přes Slack (23). Dokumentace je naplněna všelijakými příklady. I když je framework celosvětově rozšířený, absence vícejazyčnosti je.

Instalace

Pokud jsou splněny všechny výše uvedené požadavky. Pomocí příkazu se vytvoří nová Symfony aplikace. Pro potřebu si lze vybrat ze dvou příkazů (24). Přes první příkaz lze vytvořit tradiční webovou aplikaci:

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

Pro vytvoření microaplikace nebo API aplikace lze použít následují příkaz:

```
composer create-project symfony/skeleton newProject
```

Bez ohledu na to, jaký příkaz se použije, oba vytvoří nový adresář s názvem projektu, do kterého se stáhnout základní adresáře a soubory, potřebné pro aplikaci, tak aby šlo začít vytvářet Symfony aplikaci (24).

Pravidelné aktualizace

Nové verze přicházejí většinou každých 6 měsíců a jejich podpora končí zhruba po 2 měsících po vydání nové verze. Oprava menších chyb probíhá přibližně jednou týdně každou středu. Zde se jedná především o opravy menších chyb.

Ladění kódu a odchytávání chyb

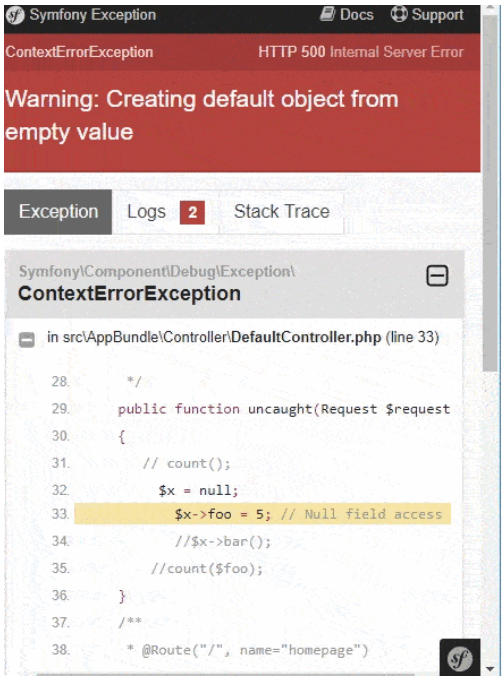
Nástroj k ladění kódu a odchytávání/monitorování chyb Symfony framework nedodává. K dispozici jsou ale doplňky, jakými jsou například Rollbar (25) nebo Sentry (26).

Instalace je velice jednoduchá, přes příkazový řádek stačí použít příkaz:

```
composer require rollbar/rollbar-php-symfony3-bundle
```

a poté zaregistrovat doplněk v `AppKernel::registerBundles() (25).`

Tímto dosáhneme potřebného monitorování chyb viz. Obrázek.



Obrázek 3: Odchytávání chyb Symfony

Testování

Symfony se integruje s nezávislou knihovnou PHPUnit (27), která poskytuje bohatý rámec testování. Zároveň má i vlastní vynikající dokumentaci (28). Psaní testů v Symfony se neliší od psaní standardních testů v PHPUnit. Lze předpokládat například, že v adresáři src/Util/ aplikace je třída pojmenována jako Calculator:

```
<?php

// src/Util/Calculator.php
namespace App\Util;

class Calculator
{
    public function add($a, $b)
    {
        return $a + $b;
    }
}
```

K otestování této funkce budeme postupovat vytvořením CalculatorTest souboru ve složce test/Util.

```
<?php

// tests/Util/CalculatorTest.php
namespace App\Tests\Util;

use App\Util\Calculator;
use PHPUnit\Framework\TestCase;

class CalculatorTest extends TestCase
{
    public function testAdd()
    {
        $calculator = new Calculator();
        $result     = $calculator->add(30, 12);

        // assert that your calculator added the numbers correctly!
        $this->assertEquals(42, $result);
    }
}
```

K provedení testu poté stačí zavolat příkazem (28):

```
php bin/phpunit tests/Util/CalculatorTest.php
```

Poté už lze vidět výsledek testu a zdali test prošel s očekáváným výsledkem.

Šablonovací systém

Symfony je dodáváno se šablonovacím systémem Twig. Ve twigu je kladen důraz na dvě hlavní funkce. Dedičnost šablony a bloky, nebo sekce. Obě funkce umožňují definovat základní šablonu. Jedná se o soubory s příponou twig (29).

```
<!DOCTYPE html>
<html>
<head>
    <title>Welcome to Symfony!</title>
</head>
<body>
<h1>{{ page_title|upper }}</h1>

{% if user.isLoggedIn %}
    Hello {{ user.name }}!
```

```
{% endif %}

        {# ... #}
</body>
</html>
```

Syntaxe Twigu je založena na těchto třech konstrukcích:

| - slouží pro zobrazení filterů,

{{...}}, slouží k zobrazení obsahu proměnné nebo výsledku vyhodnocení výrazu,

{% ...%}, slouží ke spuštění nějaké logiky, například podmíněné nebo smyčky,

{# ... #}, používá se k přidávání komentářů do šablony. V šablonách Twig nemůžete spustit čistý kód PHP, ale Twig poskytuje nástroje pro spouštění logiky v šablonách jako jsou například různé filtry pro velké písmena pro titulek `{{ page_title|upper }}` (29).

Routování a směrování

Anotace hraje důležitou roli v konfiguraci aplikace Symfony (30). Anotace zjednodušuje konfiguraci deklarováním v samotném kódování. Anotace není nic jiného než poskytování meta informací o třídě, metodách a vlastnostech. Směrování značně využívá anotace.

```
/**
 * @Route("/student/home")
 */
public function homeAction()
{
    // ...
}

/**
 * @Route("/student/about/{id}")
 * @param int $id
 */
public function aboutAction()
{
}
```

Zde směrování provádí dva kroky (30). Po přesměrování na URL adresu http://localhost/student/home, inicializuje se první trasa a pokud se shoduje, provede se funkce homeAction (). Pokud dojde k přesměrování na adresu URL http://localhost/student/about/2, bude se shodovat druhá trasa a poté se provede funkce aboutAction (). Ve druhém případě je navíc i parametr id.

Podpora databází

Symfony poskytuje nástroje, které jsou potřeba k používání databáze v aplikaci. K tomu využívá sadu knihoven Doctrine ORM (31). ORM mapuje záznamy z databáze na objekty v kódu. Za tímto účelem, dochází k vytvoření modelů pro každý typ záznamu (nebo každou tabulku) v databázi. Tyto nástroje podporují relační databáze jako jsou MySQL a PostgreSQL a také NoSQL databázi MongoDB (31).

Příklad použití doctrine orm pro nalezení všech uživatelů v databázi:

Doctrine:

```
$users = $repository->findAll();
```

MySQL příkaz:

```
'SELECT * FROM users';
```

Zde lze vidět značně zjednodušený zápis.

Balíčky a rozšíření

Co se týče balíčků a dalších rozšíření na internetu, nebyla nalezena žádná oficiální stránka s podporou pro stažení, sdílení nebo vyhledávání daných balíčků. I přesto se dá několik balíčků na internetu stále najít, a to především na GitHubu.

Zabezpečení

Ochrana aplikace Symfony před SQL injekcemi je poměrně jednoduchá. Všechny vstupy používané aplikací musí být filtrovány. Pole formuláře, parametry adresy URL, ale také soubory cookie nebo záhlaví http, pokud některá z nich zpracovává aplikace. Jedním z poznatků z bezpečnostních auditů, které bylo provedeno i u Symfony, je to, že drtivá většina aplikací provádí ověřování vstupů. Některé parametry jsou ale někde docela často opomíjeny. Pokud dojde k použití Doctrine s databází (která je dodávána se standardní edicí nebo symfonií), je využita dostatečná ochrana proti SQL injekcemi, přinejmenším při použití „klasických“ dotazů na objekty. Pokud je použita definice přes DQL s vlastními dotazy, je potřeba se ujistit, že jsou předány externí hodnoty (pocházející ze vstupů uživatele) jako zástupné symboly pomocí metody setParameter (32). Tímto způsobem se z vlastních dotazů stanou „parametrizované dotazy“, čímž se zabrání injekcím SQL.

Ochrana skriptování napříč weby v Symfony. Escapování dat je také nutností na výstupu. Šablony jako je Twig, v tomto ohledu dělají dobrou práci, protože umožní ono „escapování“ dat (32).

REST API

Symfony nemá snadno dostupné řešení pro rychlé vytváření rozhraní REST API, ale má skvělé balíčky třetích stran, jakými jsou FOSRestBundle (33) a JMSSerializerBundle (34). Zvažme minimální funkční příklad s FOSRestBundle a JMSSerializerBundle. Po jejich instalaci a zapnutí v AppKernel lze v konfiguraci balíčku nastavit, zda použít formát JSON a že to nemusí být zahrnuto v požadavcích na adresu URL. V konfiguraci směrování lze určit, že tento URL odkaz bude implementovat REST API (19). A k tomu zvolit příslušnou http metodu. S FOSRestBundle není potřeba deklarovat směrování pro každou metodu; řídit se podle konvence s názvy metod řadiče a JMSSerializerBundle automaticky převede všechny funkce na JSON (33) (34).

5.2 Laravel

Systémové požadavky

Laravel má hned několik systémových požadavků. Je zapotřebí se ujistit, že spravovaný server splňuje následující požadavky. Verze PHP by měla být vyšší než 7.2.5. Dále je zapotřebí mít nainstalované následující PHP rozšíření. BCMath, Ctype, Fileinfo, JSON, Mbstring, OpenSSL, PDFO, Tokenizer a XML. Dále je vhodné mít nainstalovaný nástroj Composer (18).

Licence

Stejně jako předchozí framework Symfony i Laravel je licencován pod MIT licencí (21).

Dokumentace a uživatelská podpora

I začínající vývojář PHP dokáže porozumět všem konceptům a využít všechny možnosti poskytované Laravelem. Každá funkce je popsána s příklady kódu a s křížovými odkazy na všechny zdroje potřebné k pochopení materiálu. Dokumentace

je krásně rozdělena na několik oddílů. Jediný mínus je opět absence vícejazyčnosti v dokumentaci. Pokud si vývojář nedokáže poradit s nějakou chybou, je pro něj možné získat pomoc v podobě komunitního fóra Laracast (35). Dále existuje i možnost certifikace pro potvrzení úrovně odborných znalostí s Laravelem a existuje spousta možností, jak se podílet na dalším vývoji Laravelu (18).

Instalace

Nový Laravel projekt lze vytvořit přímo pomocí Composeru. Po vytvoření aplikace je zapotřebí spustit místní vývojový server Laravelu Artisan (18).

```
composer create-project laravel/laravel example-app
cd example-app
php artisan serve
```

Pravidelné aktualizace

Vydání nových verzí se koná každý rok okolo měsíce září. Jejich podpora po vydání nové verze je na další 2 roky a podpora zabezpečení je na další 3 roky. Vydání oprav a menších chyb se koná každý týden.

Ladění kódu a odchytávání chyb

Nástroje k ladění kódu a odchytávání/monitorování chyb stejně jako Symfony framework nedodává. Opět je potřeba použít nástroje jako Rollbar (25) nebo Sentry (26).

Testování

Laravel bere ohled i na testování. Podpora testování stejně jako u Symfony je realizována pomocí PHPUnit (27). Chceme-li vytvořit nový testovací případ, vytvoříme ve složce tests/Unit soubor Test.php. Jakmile je soubor vytvořen, můžeme definovat testovací metody jako jsme zvyklí z PHPUnit. Pro spuštení testu zavoláme příkaz php artisan test.php (27).

Šablonovací systém

Laravel používá jako šablonovací systém Blade. Blade neomezuje v používání prostého kódu PHP v šablonách. Na rozdíl od Symfony a Nette, které používají filtry.

Ve skutečnosti jsou všechny šablony Blade kompilovány do prostého kódu PHP a ukládány do mezipaměti, dokud nejsou upraveny (36).

```
<!DOCTYPE html>
<html>
<head>
    <title>Welcome to Laravel!</title>
</head>
<body>
<h1>{{ ucfirst(page_title) }}</h1>

@if (user.isLoggedIn)
Hello {{ user.name }}!
@endif

{{-- Komentar v sablone --}}
</body>
</html>
```

Syntaxe Bladu je založena na těchto konstrukcích:

{{...}}, slouží k zobrazení obsahu proměnné nebo výsledku vyhodnocení výrazu,

@if, @foreach, slouží ke spuštění nějaké logiky, například podmínky nebo smyčky,

{{-- ... --}}, používá se k přidávání komentářů do šablony (36).

Routování a směrování

Směrování je jedním ze základních konceptů v Laravelu. Směrování v Laravelu umožňuje směrovat všechny požadavky aplikace na příslušný Controller (18). Hlavní a primární trasy v Laravelu potvrzují a přijímají URI. Všechny trasy aplikace jsou zaregistrovány v souboru app/routes.php (37).

Ukázka směrování pro úvodní stránku:

```
Route:: get ('/', function () {
    return 'Welcome to index';
});
```

Směrování na stránku s uživateli:

```
Route:: get ('/users', function () {
    return view('laravel');
});
```

V mnoha případech v rámci aplikace nastane situace, kdy bude potřeba zachytit parametry odeslané dopředu prostřednictvím adresy URL. Pro efektivní využití těchto předaných parametrů se musí v Laravelu změnit nastavení routy (37).

```
Route :: get ('user/{id}', function ($id) {
    echo 'Id uzivatele '.$id;
});
```

Podpora databází

Laravel velmi usnadnil pracování s databází. Nakonfigurovat databázi lze v souboru config/database.php, zde je potřeba nakonfigurovat potřebné informace k databázi, jako název databáze, heslo, jméno uživatele a adresu k připojení. Laravel v současné době podporuje následující 4 databáze MySQL, Postgres, SQLite,SQL Server (38). Dotaz do databáze lze spustit pomocí klasického jazyka SQL, vestavěného query Builderu anebo Eloquent ORM (38).

Příklad použití query Builderu pro nalezení uživatele s příjmením Budina:

```
$user = DB::table('users')->where('prijmeni', 'Budina')->first();
```

Příklad použití Eloquent ORM pro stejný případ:

```
$user = Users::where('prijmeni', 'Budina')->first();
```

Balíčky a rozšíření

Pro Laravel můžeme nalézt adresář všech možných a dostupných balíčků a rozšíření, tento adresář se nachází na webu Packalyst (39). Zde je k dispozici více než 14 tisíc komunitních balíčků, které umožňují najít nejlepší řešení pro každý případ (39).

Zabezpečení

Laravel poskytuje robustní Query builder a Eloquent ORM pro skladání SQL dotazů. A díky nim je většina dotazů ve výchozím nastavení chráněna v aplikacích Laravel proti SQL injekcemi. Ale vývojáři obvykle dělají chyby, když předpokládají, že Laravel chrání před všemi injekcemi SQL, i když existují některé možnosti útoku, které Laravel nemůže chránit (36).

Ochrana proti skriptování XSS, Laravel je stejně jako u všech frameworků chráněn svým šablonovacím systémem Blade. Laravel nabízí nativní podporu, která chrání kód před útoky XSS. Tato funkce se spustí automaticky a chrání databázi v procesu. I zde ale může dojít k chybě na straně vývojáře. Abychom se, jakkoliv vyhnuli útokům XSS, měli bychom používat syntaxi dvojitých složených závorek v šablonách Blade: ({{$promenna}}) a vyhýbat se použití {$promenna} (36).

REST API

Nejprve je zapotřebí nadefinovat směrování. To lze udělat v sekci API kde se nachází definice směrování, zde lze i vypnout některé výchozí komponenty middlewaru a zapnout další. Sekce API je umístěna v souboru route/api.php (40).

```
<?php

// routes/api.php
Route::resource('/posts', 'BlogController');
```

Poté přejdeme na definici daného kontroléru, kde si zvolíme pro funkci vybranou HTTP metodu s příslušnou odpovědí. V Symfony používáme FosRestBundle, který zabalí chyby do JSON formátu. V Laravelu toto bohužel musíme udělat sami.

5.3 Nette

Systémové požadavky

Od verze Nette 3.0 jsou požadovány minimální systémové požadavky pro správnou funkci PHP verzi 7.1.0 a vyšší a další požadavky na prostředí webového serveru. Htaccess zabezpečení a rozšíření Reflection, SPL, PCRE, ICONV, PHP tokenizer, PDO, Multibyte String, Memcache, GD a Bundled GD. Zdali používaný server splňuje minimální požadavky si lze ověřit pomocí Requirements checkeru (20).

Licence

Nette framework lze používat pod dvěma licencemi, tou první je BSD licence, která je doporučena pro většinu projektů, je lehčí na pochopení a neplatí zde žádné restrikce, které by nás omezovali na tom, co můžeme dělat s frameworkem. Můžeme volně použít Nette framework v komerčních projektech. Druhou licencí je GPL (13).

Dokumentace a uživatelská podpora

Nette má právě velké zastoupení i mezi českými a slovenskými vývojáři. Lze zde tedy najít mnoho návodů a vyřešených problémů v češtině a zároveň je dokumentace udělána i v angličtině. Dokumentace je pojata vcelku jednoduše, můžeme zde nalézt u každého tématu několik ukázkových příkladů (20). Pro uživatelskou podporu existuje i Nette Forum (41), které je opět lokalizované na češtinu i angličtinu.

Instalace

Instalace je velice jednoduchá, framework lze stáhnout buď manuálně, ale doporučená cesta je stáhnout framework použitím Composeru a to přes tento příkaz (20):

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

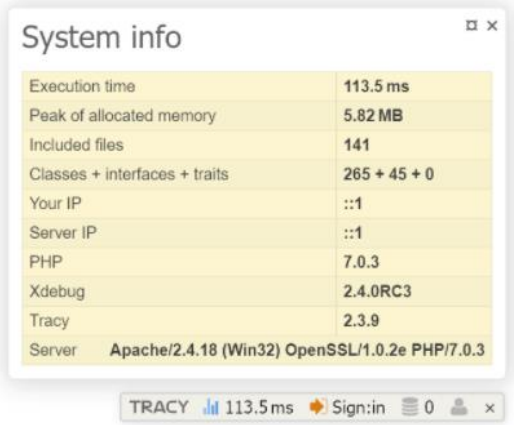
Pravidelné aktualizace

Jednotlivá doba mezi vydáním verzí se dá spíše odhadovat a nelze čekat přesně daný termín, v průměru se jedná přibližně o rok a půl. Každá verze je po vydání nadále spravována v délce jednoho roku a kritické bezpečnostní chyby po dobu dvou let. Menší chyby a opravy jsou vydávány i třeba každé dva dny.

Ladění kódu a odchytávání chyb

Velká výhoda je přehledné zpracování chybových hlášení a systémové informace. K tomu je využíván nástroj Tracy Debugger (20). Je to velice užitečný pomocník všem vývojářům PHP, který pomáhá s jasnými chybami vizualizace a protokolování a výpisem proměnných.

Tracy Bar viz. obrázek je skvělý zdroj neocenitelných informací o webových stránkách. Může měřit čas, zaznamenávat paměť, počítat požadavky na databázi a monitorovat další nejrůznější důležitá data.



Obrázek 4: Tracy Bar

Testování

Nette používá pro testování svůj vlastní nástroj a tím je Nette Tester. Pomocí něho je maličkost otestovat jakýkoliv PHP kód. Lze jednoduše nainstalovat přes Composer příkazem:

```
composer require nette/tester --dev
```

Poté si už lze vytvořit testovací třídu a spustit test (20).

Šablonovací systém

Nette používá jako svůj šablonovací systém Latte. Latte syntaxe je o něco kratší a jednodušší oproti dvou zmíněným frameworkům, a to především díky tomu, že šablona nenutí používat dvoje závorky {…}.

```
<!DOCTYPE html>
<html>
<head>
    <title>Welcome to Nette!</title>
</head>
<body>
<h1>{page_title |capitalize}</h1>

{if $user->isLoggedIn()}
    Hello {user.name}!
{/if}

        {* Komentar v sablone *}
</body>
</html>
```

Syntaxe Latte je založena na těchto konstrukcích:

{…}, slouží k zobrazení obsahu proměnné nebo výsledku vyhodnocení výrazu,

Tagy jako {if}{/if}, nebo {foreach}{/foreach}, slouží ke spuštění nějaké logiky, například podmínky nebo smyčky,

{* Komentář *}, tento zápis se použije pro přidávání komentářů do šablony.

Další důležitou komponentou jsou takzvané n:atributy, kde dochází k dalšímu ulehčení zápisu a ušetření místa v syntaxi. Tagy {foreach}{/foreach} pomocí n:atributů je možné poté nahradit například následujícím zápisem (20).

```
<p n:foreach="$items as $item">{$item}</p>
```

Routování a směrování

Router hraje v aplikaci Nette velice důležitou roli. Díky routeru aplikace zjistí, který Presentér a akce má být spuštěna (20). A také používá router ke generování adres URL v šabloně.

Jednotlivé adresy se zapisují do souboru RouterFactory, a poté je zapotřebí ho ještě připojit do konfiguračního souboru následujícím způsobem:

```
services:
   router: App\Router\RouterFactory::createRouter
```

Jednotlivé cesty se dají definovat následujícím způsobem. Takhle můžeme vytvořit URL adresu pro úvodní stránku.

```
$router->addRoute('<presenter>/<action>', 'Homepage:default');
```

URL adrese můžeme také přiradit parametr jako u předchozích frameworků, v tomto případě budeme chtít vypsat detail uživatele, proto musíme v definici uvést v zobáčcích parametr id (20).

```
$router->addRoute('users/<id>', 'User:detail');
```

Dost často budeme používat také vytváření odkazů v šablonách, k tomu slouží v Latte atribut n:href. Následující ukázka vytvoří URL adresu http://localhost/user/detail/1

```
<a n:href="User:detail $id">detail uživatele</a>
```

Podpora databází

Nette momentálně dovoluje pracovat s 6 druhy databází. MySQL, PostreSQL, Sqlite 3, Oracle, MS SQL, ODBC (42). Konfigurace databáze se provádí v souboru config/local.neon. Nette dovoluje výběr ze 3 možností, jak psát jednotlivé dotazy. Tou první je Database Core, který tvoří základní přístup k databázi. Dotazy vytváříme metodou query(), která vrátí výsledek. Pro ukázku sestavíme dotaz pro vyhledání všech uživatelů.

```
$database->query('SELECT * FROM users');
```

Druhým způsobem je použití Database Explorer. Zde skládáme dotazy pomocí metody table() a můžeme skládat další dotazy za pomocí klauzule select(), where(), orderBy() (43). Pro ukázku použijeme stejný dotaz jako v předchozím případě.

```
$users = $explorer->table('users');
```

A posledním třetím způsobem je použití balíčku Nextras ORM (14). Jedná se především o objektový přístup k databázi. Opět stačí nainstalovat pomocí Composeru a připojit balíček do konfiguračního souboru. Můžeme využít hned

několik metod jako například findAll(), findBy().Pro ukázku použití využijeme stejného dotazu (14).

```
$users=$this->orm->users->findAll()->fetchAll;
```

Balíčky a rozšíření

Co se týče balíčků a rozšíření pro Nette Framework je k dispozici komunitní stránka Contributte (44). Zde lze nalézt hned několik různých balíčků. V tuto chvíli má 19 členů, denní počet stažení balíčků je 12 tisíc a celkový počet je 7,5 milionu (44).

Zabezpečení

Nette používá jako svoji základní vrstvu pro přístup k databázi Database Core. Ta je proti útokům chráněná, pokud ale vývojář špatně použije zápis syntaxe a dotazy bude skládat jako řetězce, vznikne zranitelnost. Bohužel k tomuto stále někdy dochází.

Obrana proti Cross Site Scriptingu je velice důležitá, proto Nette přichází s automatickým escapováním. Používá svoji unikátní technologii kontextově sensitivní escapování, vše se děje kvůli tomu automaticky a všechny výstupy v šabloně ošetři. A vývojář nemusí nic manuálně ošetřovat (45).

REST API

Nejlehčí implementace REST API v Nette je za použití Ublaboo/Api-routeru (46). Pomocí Composeru stačí nainstalovat balíček a poté ho ještě zaregistrovat v konfiguračním souboru config.neon.

```
extensions:
    apiRouter: Contributte\ApiRouter\DI\ApiRouterExtension
```

Api routy poté definujeme v RouterFactory a to následujícím způsobem. První zvolíme URL koncovou adresu, v našem případě /users a také http metodu GET. A spojíme s příslušnou funkcí, která nám vrátí výsledek ve formátu Json (46).

```
$router[] = new ApiRoute('/user', 'ApiRouter', [
    'methods' => ['GET' => 'getUsers', 'POST']
]);

public function actionGetUsers()
{
    $this->sendJson(['Tomáš' => 'Budina']);
}
```

Když přejdeme na adresu http://localhost/users získáme výpis všech uživatelů.

5.4 Zhodnocení porovnání

Systémové požadavky – všechny frameworky mají stejné požadavky a zároveň podporují nejnovější verze jazyka PHP 7. a vyšší. Všude lze udělit hodnocení velmi uspokojivé.

Licence – každý framework je open source a jsou šířeny pod licencí MIT nebo GPL. Tudíž i zde musím udělit pouze velmi uspokojivé.

Dokumentace a uživatelská podpora – všechny frameworky mají kvalitně zpracovanou dokumentaci, u Symfony a Laravelu na rozdíl od Nette vázne absence multijazyčnosti. Symfony a Laravelu dávám hodnocení 3 dostačující a Nette 2 uspokojivé.

Instalace – instalace pro Symfony a Nette je velice jednoduchá, a proto dostanou hodnocení uspokojivé. U Laravelu se instalace trochu komplikuje kvůli nutnosti spuštění místního vývojového serveru Laravelu Artisan. Proto mu udělím hodnocení dostačující.

Pravidelné aktualizace – aktualizace u Symfony a Laravelu probíhají v řádů několik měsíců, kdežto i Nette se dá doba spíše odhadovat v řádech několika let. Z tohoto důvodu prvním dvou zmíněným dám hodnocení velmi uspokojivé a Nette nedostačující.

Ladění kódu a odchytávání chyb – Nette využívá svůj vlastní nástroj Tracy Debugger, to je velká výhoda oproti Symfony a Laravelu, ti musí použít nástroj třetích stran, které nemusí být například dostatečně zabezpečené. Nette dostane hodnocení velmi uspokojivé a ostatní nedostačující.

Testování kódu – Nette opět využívá svůj vlastní nástroj Nette Tester, a ostatní frameworky používají základní vestavený PHPUnit testy. Z toho důvodu musím udělit hodnocení pro Nette jako velmi uspokojivé a ostatní dostačující.

Šablonovací systém – každý framework využívá svůj šablonovací systém. Každý má své výhody a jiný příklady použití. Zde musím udělit hodnocení všem velmi uspokojivé.

Routování a směrování – velká výhoda Symfony je vytváření URL adres jen za použití anotací. Nette naopak umožnuje vytvářet adresy i v šablonách za použití

n:href atributů. Symfony dostane hodnocení velmi uspokojivé, Nette a Laravel uspokojivé.

Podpora databází – každý framework podporuje stejné databáze. Jediná výhoda Symfony je, že dokáže pracovat i s NoSQL databází MongoDB. Symfony získalo uspokovojivé, Laravel a Nette dostačujicí.

Balíčky a rozšíření – co se týče balíčků Nette a Laravel disponují svou vlastní komunitní stránku, kde lze nalézt hned několik balíčků. Symfony tuto možnost nemá, ale i pro Symfony můžeme nalézt balíčky například na Githubu. Nette a Laravel získávají velmi uspokojivé a Symfony nedostačující.

Zabezpečení – každý framework je chráněn proti SQL injection a XSS. Dostanou proto velmi uspokojivé.

REST API – u Laravelu lze použít vestavěný nástroj pro realizaci restové API. U Nette lze použít snadno balíček. A u Symfony je tato realizaci podstatně těžší. Laravel získal velmi uspokojivé, Nette uspokojivé a Symfony dostačující.

Tabulka 1: Hodnocení

Číslo kritéria/ Framework	1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	11.	12.	13.
Symfony	1	1	3	2	1	4	3	1	1	2	4	1	3
Laravel	1	1	3	3	1	4	3	1	2	3	1	1	1
Nette	1	1	2	2	4	1	1	1	2	3	1	1	2

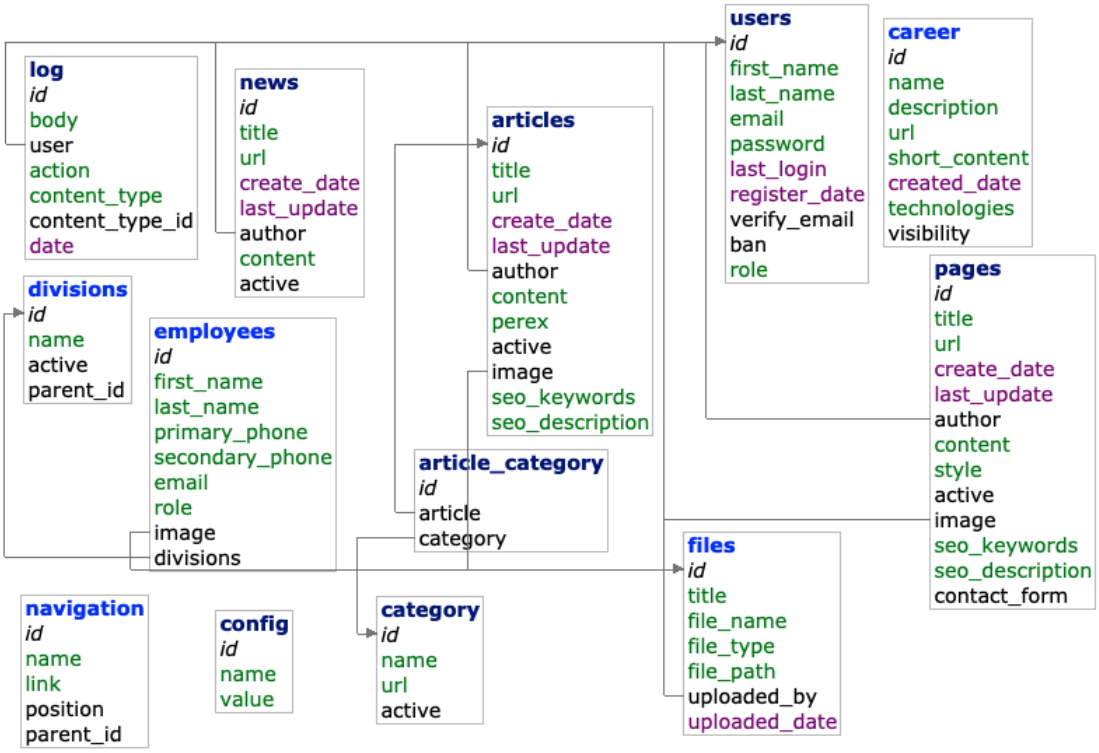
1 – velmi uspokojivé, 2 – uspokojivé, 3 – dostačující, 4 – nedostačující, 5 – zcela nedostačující.

Z hodnocení vyplývá, že nejlépe si vede framework Nette s 22 body, za ním se drží Laravel, který má 25 bodů, a nakonec Symfony s 27 body.

6 Implementace CMS

6.1 Návrh databáze

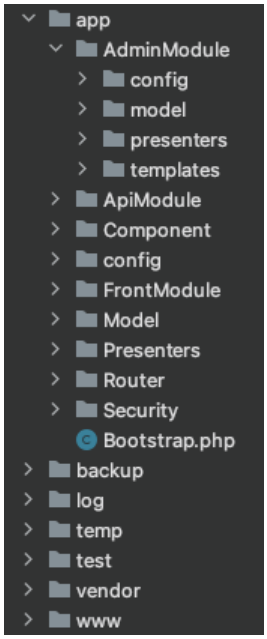
Pro aplikaci byla použita databáze MySQL ve verzi 10.5.8.



Obrázek 5: Diagram databáze

6.2 Adresářová struktura

Kódy aplikace jsou tříděny do jednotlivých složek a modulů. V aplikaci se nachází tři moduly. Admin modul, Api modul a Frontend modul. Admin modul má na starosti logiku v administrační části. Api modul slouží k ovládání integrace s Google Analytics a Frontend modul ovládá logiku na webové stránce to co vidí většinou až návštěvník a další grafické věci.

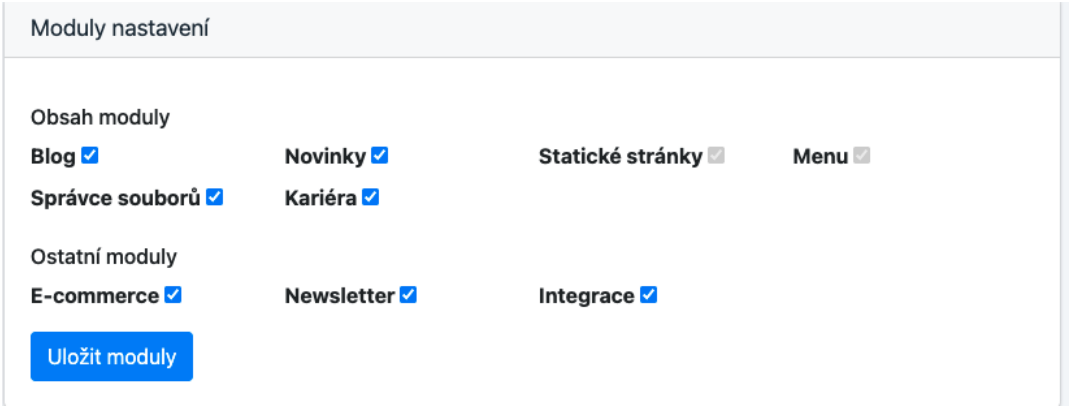


Obrázek 6: Adresářová struktura aplikace

Jak lze vidět na obrázku, jednotlivé moduly jsou děleny do složek podle svého názvu. V těchto složkách lze dále nalézt příslušné služby, šablony a logiku daného modulu.

6.3 Uživatelský návod

Aplikace slouží jako CMS pro správu webového obsahu. Jednou z mála podmínek dobrého CMS je nastavení kvalitního SEO a jednoduchého ovládaní. Administrace je tvořena z několika prvků, které jsou popsány níže, z těchto prvků je možné poté sestavit funkční webové stránky pro koncového návštěvníka webu. Po přihlášení do administrace lze vidět po levé straně menu s danými prvky. Ovládání administrace je praktické a zcela uživatelsky přívětivé. Do administrace lze vstoupit pomoci URL adresy https://domenovejmeno.cz/manage, zde se lze přihlásit pomocí přihlašovacích údajů. Pokud dojde k úspěšnému přihlášení dojde k přesměrování na uvítací stránku administrace. Aplikace je nadále rozdělena do jednotlivých modulů podle obsahu, které může administrátor s rolí superadmin vypínat a zapínat podle potřeby. S těmito moduly se seznámíme níže.



Obrázek 7: Ukázka nastavení modulů

6.3.1 Logování akcí v systému

Každá provedená akce, vytvoření, úprava a smazání údajů je zavedena do logu, které si může příslušný administrátor zobrazit. Tím pádem je zamezena jakákoliv špatná manipulace ze strany editora, nebo ztráta dat. Pro tuto akci byla vytvořena třída LogService a v ní byla vytvořena veřejná funkce createLog s povinnými parametry. Tuto funkci si stačí v dalších třídách přidat opětovně přes konstruktor následovně. Na ukázce můžeme vidět zápis v Nette pomocí takového zápisu.

```
public function __construct(
    Orm $orm,
    User $user,
    Config $config,
    LogService $logService,
    CategoryService $categoryService
) {
    parent::__construct($orm, $user, $config);
    $this->logService      = $logService;
    $this->categoryService = $categoryService;
}
```

6.3.2 Nastavení

Jak už bylo zmíněno výše v úvodu, nastavení slouží k definování základních informací o webových stránkách, nastavení názvu webu, loga a především SEO. Dále lze v této sekci přepínat jednotlivé moduly obsahu, který slouží k modulárnosti aplikaci. Pomocí těchto zvolených modulů viz. Obrázek 7 lze zobrazovat na webových stánkách dané informace. V nastavení lze dále zálohovat databázi. Zálohovat databázi lze po stisknutí tlačítka, následně se uloží všechna data a stáhnou se v balíčku zip. V této sekci lze nastavit i emailový server pro přijímání zpráv z newletteru od zákazníků.

6.3.3 Uživatelé

Tato sekce slouží k ovládání uživatelů v administračním rozhraní. Je možné vypnout uživatele a tím pádem mu znemožnit přístup, změnit mu základní údaje, změnit heslo anebo jeho roli. Momentálně jsou zavedeny 4 uživatelské role. Superadmin, admin, editor a analytik. Na základě těchto rolí má uživatel přístup jen do určitých sekcí. Pro kontrolu oprávnění na straně serveru, můžeme využít rozhraní Nette\Security\Identity. Pomocí tohoto rozhraní lze kontrolovat příslušné role uživatele a zdali má povolen přístup do aplikace. V Nette poté můžeme ověřit, zdali má uživatel přístup do dané sekce pomocí několika zápisů.

Zdali má uživatel povolen přístup do sekce Blog.

```
$user->isAllowed('Admin:Blog','default')
```

Zdali má uživatel roli superadmin.

```
$user->isInRole('admin')
```

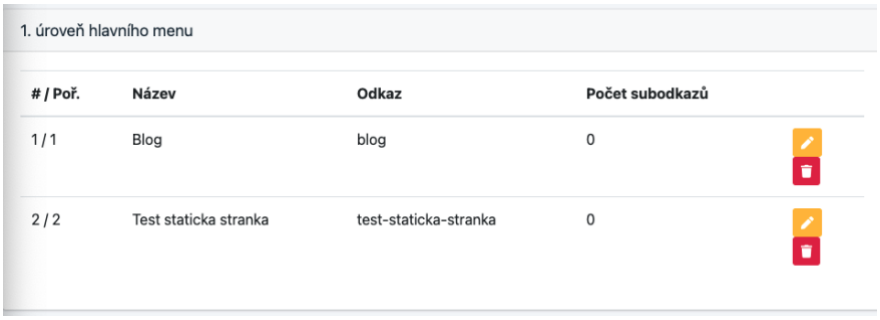
Nebo zdali je uživatel přihlášen.

```
$user->->isLoggedIn()
```

Obrázek 8: Identita uživatele

6.3.4 Menu

Pomocí sekce menu lze definovat hlavní navigace na stránce. Lze definovat jednotlivé úrovně menu jejich URL adresu a název a dále jejich pod odkazy. Z důvodu použití routování v Nette frameworku, lze definovat a poté vytvářet jednotlivé odkazy v menu. U úrovní je možné prohazovat i jednotlivé pořadí podle jakého se v hlavním menu zobrazí na stránkách.



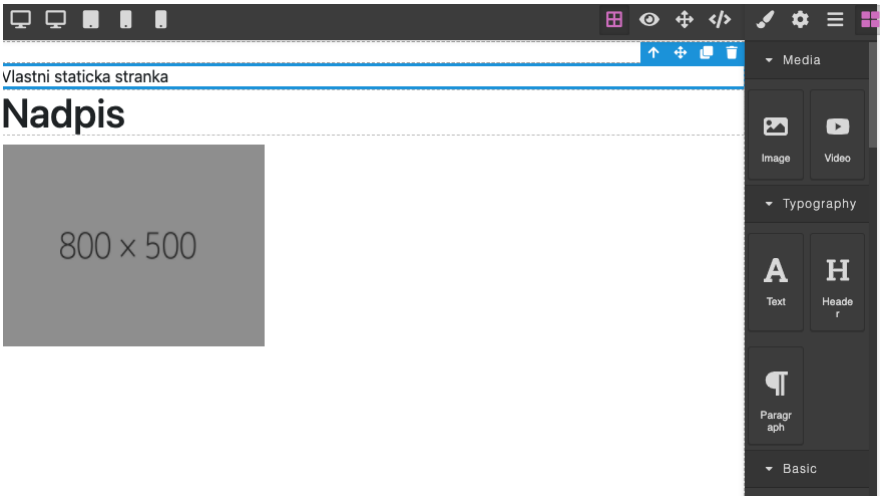
Obrázek 9: Ukázka skládání navigace

Navigace je poté v. latte šabloně skládána následujícím způsobem.

```
{foreach $menuItems as $item}
    <li class="nav-item">
        <a class="nav-link"
href="/{$item['link']}">{$item['name']}</a>
    </li>
{/foreach}
```

6.3.5 Statické stránky

Modul statických stránek má na starost definování jedné webové stránky. Při vytváření statické stránky je třeba vyplnit název, URL adresu, viditelnost na webu a SEO klíčová slova. URL adresu je zapotřebí nastavit z důvodu vytvoření adresy, na kterou budeme odkazovat v hlavním menu. Viditelnost lze přepnout podle potřeby, zdali chceme danou stránku zneaktivnit. To samé platí i pro SEO a klíčová slova. Ty jsou zapotřebí především jako optimalizace pro vyhledávače. Proto aby se naše stránka zobrazovala na přednějších místech ve vyhledávání. Danou statickou stránku si můžeme sami vzhledově přizpůsobit v editoru, a tento vzhled bude vidět na stránkách. K tomu je přiložen editor pro úpravu textu, obrázku a zarovnání do bloků.



Obrázek 10: Editor vzhledu statické stránky

6.3.6 Články

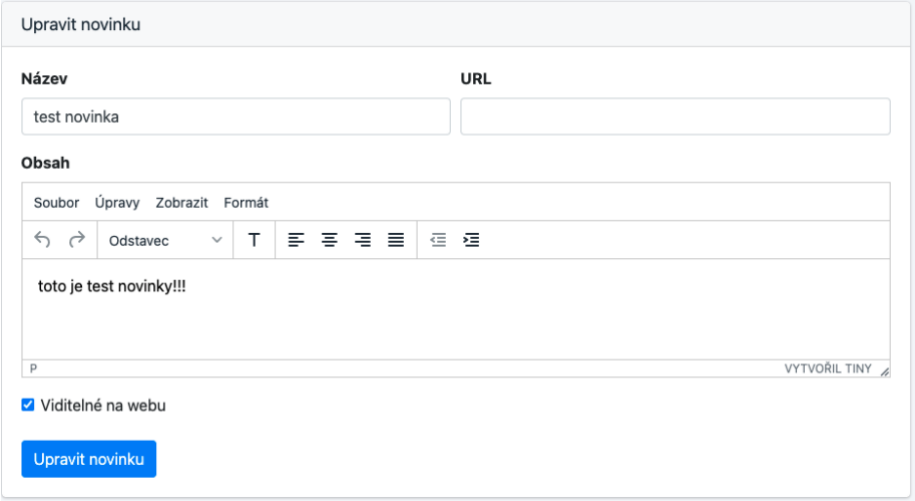
V sekci článků je možné vytvořit články, které můžeme zobrazit na stránkách. Při vytváření článků je potřeba vyplnit několik hlavních údajů, jakými jsou název, URL adresa, viditelnost, perex, hlavní obsah, SEO a do jaké kategorie článek spadá. Pro tyto potřebné údaje platí to samé jako u předešlé kategorie statických stránek. Články pomocí výběru kategorie můžeme dále filtrovat. Je možné vybrat i více kategorií najednou. Článek vytvoříme jednoduchým SQL dotazem. Jak už bylo několikrát řečeno u Nette je možné použít balíček Nextras ORM, které nám psaní SQL dotazů velice usnadní. Všechny hodnoty získáme v HTML formuláře a ty potom nadále zpracováváme právě v tomto dotazu.

```
$article                = new Articles();
$article->title         = $values['title'];
$article->url           = $values['url'];
$article->image         = ($values->media != '' ? $this->orm->files-
>getById($values->media) :
    ($image ? $this->orm->files->getById($fileId) : null));
$article->perex         = $values['perex'];
$article->content       = $values['content'];
$article->createDate    = new \DateTime('now');
$article->lastUpdate    = null;
$article->author        = $this->orm->users->getById($this->user-
>getId());
$article->active        = $values['active'];
$article->seoKeywords   = $values['seoKeywords'];
$article->seoDescription = $values['seoDescription'];

$this->orm->persistAndFlush($article);
```

6.3.7 Novinky

Pomocí novinek lze vytvářet jednoduché informativní informace, které jsou zobrazovány na hlavní stránce webu. U každé novinky je potřeba vyplnit název, obsah a viditelnost, URL adresa není uvedena jako povinný údaj. Pokud je ale vyplněna, je možné se prokliknout a přejít do detailu novinky.



Obrázek 11: Úprava novinky

6.3.8 Správce souborů

Modul správce souborů slouží k ukládání dat nebo případně obrázků a jiných souborů, které lze nadále využít například v sekci statických stránek, pro vložení obrázku do stránky. Jako budoucí rozšíření považuji implementace AWS S3 klienta pro nahrávání souborů na úložiště od Amazonu.

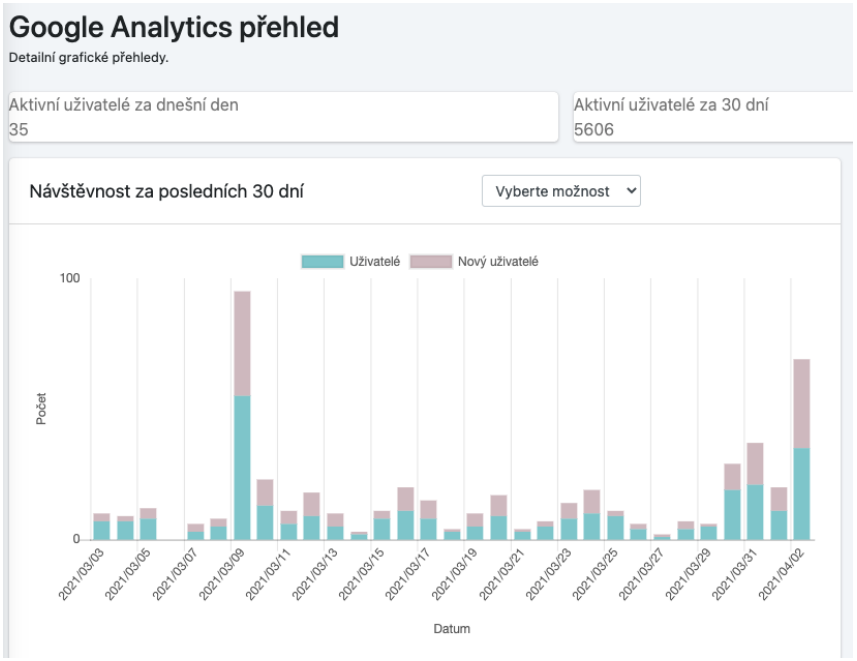
6.3.9 Integrace Google Analytics

Podle výsledků porovnání lze v Nette velice jednoduše aplikovat připojení k REST API, a to je právě využito u připojení s Google Analytics. Hlavní podmínkou pro správné nastavení Google Analytics je zjištění a uložení měřícího kódu, který dodává k této službě sám Google, po úspěšném vložení údajů. Nadále je zapotřebí uložení 3 potřebných údajů a těmi jsou přístupový token, klientova identita a číslo pohledu. Po uložení těchto údajů se aplikace postará o spojení s Google Analytics a lze tak vidět informace o návštěvnících stránek v reálném čase přímo v aplikaci. Po uložení všech potřebných údajů, můžeme přistoupit k vytažení dat z Google Analytics. Připojit k aplikaci REST API je v Nette velice jednoduché. K tomu slouží v Nette a

celkově v PHP cURL knihovna. Pomocí této knihovny lze po připojení k dané koncové adrese z různou http metodou vytáhnout určená data. Níže je ukázka kódu pro inicializaci cURL a získání dat.

```
$curl = curl_init();
curl_setopt_array($curl, array(
    CURLOPT_URL            =>
"https://analyticsreporting.googleapis.com/v4/reports:batchGet",
    CURLOPT_RETURNTRANSFER => true,
    CURLOPT_ENCODING       => "",
    CURLOPT_MAXREDIRS      => 10,
    CURLOPT_TIMEOUT        => 0,
    CURLOPT_FOLLOWLOCATION => true,
    CURLOPT_HTTP_VERSION   => CURL_HTTP_VERSION_1_1,
    CURLOPT_CUSTOMREQUEST  => "POST",
    CURLOPT_POSTFIELDS     => json_encode($postField),
    CURLOPT_HTTPHEADER     => array(
        "Authorization: Bearer " . $gaAccessToken,
        "Content-Type: application/json"
    ),
));
```

Poté tato získaná data už jenom zapíšeme do příslušného grafu.



Obrázek 12: Integrace Google Analytics

7 Závěr

Cílem práce bylo uvést do problémů CMS systémů a seznámit se základními funkcemi. A dále poskytnout pomoc při výběru CMS systémů případnému podniku.
V první části práce byl popsán úvod do CMS systému, jakým způsobem se rozděluje, jeho funkce a nejznámější používané systémy. Dále byly popsány použité technologie, které byly využity, jak u porovnávání vybraných frameworků, tak k implementaci v praktické části.
Další část práce má za cíl analýzu pro vybrané PHP frameworky podle kritérií a implementovat CMS systém pro jeden z analyzovaných PHP frameworků. Na základě kritérií bylo poté provedeno hodnocení podle dané škály.
V praktické části byl sepsán jednoduchý uživatelský návod k implementaci vlastního CMS systému. Pro aplikace byl zvolen PHP framework Nette. Zde je ukázána i práce s frameworkem. Aplikace je tříděna do několika modulů. Slouží jako CMS pro správu webového obsahu. Kde je dbáno především na SEO optimalizaci stránek a přívětivé prostředí. V aplikaci je možné dále přidávat články, statické stránky, novinky a vytvořit si či editovat hlavní navigaci na stránkách dle vlastního uvážení.
Podle vyhodnocení v teoretické a poté v praktické části by bylo možné zvolit jakékoliv z popisovaných frameworků. Jako nejlepší volbou se jeví Nette. Jedná se především o českou podporu. Nástroje jako Laděnka (Tracy) a podpora pro testování, jsou neodmyslitelnou součástí tohoto frameworku. Zároveň disponuje i kvalitně sepsanou dokumentací a má velikou podporu rozšiřovacích balíčků pro správu. Toto hodnocení se promítlo i ve výsledcích porovnání frameworků.

8 Seznam použité literatury

Publikace

1. **ŠTRÁFELDA, Jan.** Redakční systém (CMS). [Online] [Citace: 7. listopadu 2021.] https://www.strafelda.cz/cms.

2. **ČÍŽKOVÁ, Gabriela.** Co by vám měl CMS umožnit? [Online] 2013. [Citace: 7. listopadu 2021.] http://www.web-integration.info/cs/blog/co-by-vam-mel-cms-umoznit/.

3. **PALLI, Ivan.** What Is CMS and How to Choose It for Your Website? [Online] [Citace: 9. listopadu 2020.] https://sitechecker.pro/what-is-cms/.

4. **PROCHÁZKA, Tomáš.** Jaký zvolit CMS, aby se vám obsah snadno spravoval. [Online] [Citace: 9. listopadu 2021.] https://vceliste.cz/blog/cms-pro-spravu-obsahu/.

5. **ŠESTÁKOVÁ, Lucie.** *WordPress: vlastní web bez programování.* Brno: Computer Press, 2013. ISBN 978-80-251-3832-8.

6. **RAHMELL, Dan.** *Beginning Joomla.* místo neznámé: Computer Press (CP Books), 2010. ISBN: 978-80-251-2714-8.

7. **POLZER, Jan.** *Drupal – Podrobný průvodce tvorbou a správou webů.* místo neznámé: Computer Press (CP Books), 2008. ISBN: 978-80-251-1946-4.

Internetové zdroje

8. **SMITH, Zhanna Malekos, Eugenia LOSTRI, LEWIS, James A.** The Hidden Costs of Cybercrime. *McAfee.* [Online] 2020. [Citace: 7. listopadu 2020.] https://www.mcafee.com/enterprise/en-us/assets/reports/rp-hidden-costs-of-cybercrime.pdf.

4. **Google.** Reporting API v4. [Online] [Citace: 7. listopadu 2020.] https://developers.google.com/analytics/devguides/reporting/core/v4/basics.

10. **Magento.** Magento Documentation. [Online] [Citace: 20. listopadu 2020.] https://devdocs.magento.com/guides/v2.4/architecture/archi_perspectives/framework.html.

11. **WhatCMS.** Zastoupení redakčních systémů v ČR. *WhatCMS.* [Online] 2021. [Citace: 20. listopadu 2021.] https://whatcms.org/Tech_Reports.

12. **PHP.** PHP Documentation. [Online] [Citace: 20. listopadu 2020.] https://www.php.net/docs.php.

13. **GRUDL, David.** Licenční politika. [Online] [Citace: 20. listopadu 2020.] https://nette.org/cs/license.

14. **COMMUNITY, Nextras.** Nextras ORM Components. [Online] 2013. [Citace: 22. listopadu 2020.] https://nextras.org/orm/docs/4.0/.

15. **JQuery.** JQuery License. [Online] [Citace: 23. listopadu 2020.] https://jquery.org/license/.

16. **MySQL.** MySQL documentation. [Online] [Citace: 25. listopadu 2020.] https://dev.mysql.com/doc/.

17. **KUKHNAVETS, Pavel.** 10 PHP Frameworks to Keep an Eye on. [Online] 2019. [Citace: 2. prosince 2020.] https://welldoneby.com/blog/top-php-frameworks-to-keep-an-eye-on/.

18. **OTWELL, Taylor.** Laravel Documentation. [Online] 2011. [Citace: 5. prosince 2020.] https://laravel.com/docs/8.x.

19. **Symfony.** Symfony Documentation. [Online] [Citace: 5. prosince 2020.] https://symfony.com/doc/current/index.html.

20. **Nette.** Nette dokumentace. [Online] [Citace: 6. prosince 2021.] https://doc.nette.org/cs/3.1/.

21. **POTENCIER, Fabien.** Symfony Code License. [Online] [Citace: 6. prosince 2020.] https://symfony.com/doc/current/contributing/code/license.html.

22. **Symfony.** Questions tagged. [Online] [Citace: 7. prosince 2020.] https://stackoverflow.com/questions/tagged/symfony.

23. —. Symfony Devs. [Online] [Citace: 8. prosince 2020.] https://symfony-devs.slack.com.

24. —. Symfony Documentation Instalation. [Online] [Citace: 8. prosince 2020.] https://symfony.com/doc/current/setup.html#creating-symfony-applications.

25. **Rollbar.** Rollbar Extension. [Online] [Citace: 8. prosince 2020.] https://rollbar.com.

26. **Sentry.** Symfony Error and Performance Monitoring. [Online] [Citace: 9. prosince 2020.] https://sentry.io/for/symfony/.

27. **Testing framework for PHP.** [Online] [Citace: 9. prosince 2020.] https://phpunit.de.

28. **BERGMANN, Sebastian.** PHPUnit Manual. [Online] Revize 2020. [Citace: 2. ledna 2021.] https://phpunit.readthedocs.io/en/9.5/.

29. **Symfony.** Symfony Documentation Templates. [Online] [Citace: 5. ledna 2021.] https://symfony.com/doc/current/templates.html#creating-templates.

30. —. Symfony Documentation Routing. [Online] [Citace: 5. ledna 2021.] https://symfony.com/doc/current/routing.html.

31. —. Symfony Documentation Database. [Online] [Citace: 5. ledna 2021.] https://symfony.com/doc/current/doctrine.html.

32. —. Symfony Documentation Security. [Online] [Citace: 5. ledna 2021.] https://symfony.com/doc/current/templates.html#output-escaping.

33. **FOSRestBundle.** [Online] [Citace: 20. ledna 2021.] https://github.com/FriendsOfSymfony/FOSRestBundle.

34. **JMSSerializerBundle.** [Online] [Citace: 20. ledna 2021.] https://github.com/schmittjoh/JMSSerializerBundle.

35. **TUDS.** Laracast Developer Forum. [Online] [Citace: 20. ledna 2021.] https://laracasts.com/discuss.

36. **OTWELL, Taylor.** Laravel Documentation Views. [Online] [Citace: 5. února 2021.] https://laravel.com/docs/8.x/blade.

37. —. Laravel Documentation Routing. [Online] [Citace: 8. února 2021.] https://laravel.com/docs/8.x/routing.

38. —. Laravel Database. [Online] [Citace: 10. února 2021.]

39. **Packalyst.** [Online] [Citace: 12. února 2021.] https://packalyst.com.

40. **OTWELL, Taylor.** Laravel Documentation REST API. [Online] [Citace: 12. února 2021.] https://laravel.com/docs/8.x/requests.

41. **Nette Forum.** [Online] [Citace: 18. února 2021.] https://forum.nette.org/cs/.

42. **Nette. Nette Database Core.** [Online] [Citace: 18. února 2021.] https://doc.nette.org/cs/3.1/database-core.

43. —. Nette Database Explorer. [Online] [Citace: 18. února 2021.] https://doc.nette.org/cs/3.1/database-explorer.

44. **VERCEL, FELIX.** Contributte extensions for Nette Framework. [Online] [Citace: 1. března 2021.] https://contributte.org.

45. **Nette.** Nette dokumentace zabezpečení. [Online] [Citace: 2. března 2021.] https://latte.nette.org/cs/safety-first.

46. **VERCEL.** Ublaboo API router. [Online] [Citace: 3. března 2021.] https://ublaboo.org/api-router/.

9 Přílohy

V příloze se nachází CD disk s danou aplikací a její databází.

Univerzita Hradec Králové
Fakulta informatiky a managementu

Zadání bakalářské práce

Autor:	**Tomáš Budina**
Studium:	I1900324
Studijní program:	B0688A140001 Informační management
Studijní obor:	
Název bakalářské práce:	**Frameworky pro tvorbu CMS systémů**
Název bakalářské práce AJ:	Frameworks for development of CMS systems

Cíl, metody, literatura, předpoklady:

Cílem práce je analýza vybraných PHP frameworků a implementace CMS systému. Teoretická část obsahuje úvod do CMS systémů, a dále popis týkajících se technologií, obecně frameworky, typy a jejich funkcionality. Nadále obsahuje postup tvorby webové aplikace. Praktická část pak obsahuje zahrnutí analýzy vybraných PHP frameworků a implementaci CMS systému pro jeden vybraný framework.

Osnova:

1. Úvod do CMS systémů
2. Použité technologie
3. Postup tvorby webové aplikace
4. Analýza a porovnání PHP webových frameworků
5. Implementace CMS
6. Závěr

https://doc.nette.org/cs/3.0/

https://symfony.com/doc/current/index.html

Garantující pracoviště:	Katedra informačních technologií, Fakulta informatiky a managementu
Vedoucí práce:	prof. Ing. Vladimír Bureš, Ph.D., MBA
Datum zadání závěrečné práce:	21.10.2019