

Střední průmyslová škola, Karviná, příspěvková organizace



Blogovací platforma pro Microsoft STC

Maturitní práce

2019/2020

Ondřej Golasowski

Střední průmyslová škola, Karviná, příspěvková organizace



Blogovací platforma pro Microsoft STC

Autor: Ondřej Golasowski, 4. D

Vedoucí práce: Vladimír Hruban

Anotace

Maturitní práce popisuje tvorbu šablony webového blogu pro redakční systém Wordpress. Může sloužit i jako příručka a návod pro zájemce o tvorbu vlastní šablony. Práce je členěna na dva hlavní celky. Prvním celkem je teoretická část, která obsahuje popis využívaných technologií a jejich porovnání. Druhý celek tvoří praktická část, která popisuje samotnou tvorbu blogu pomocí technologií popsanych v teoretické části.

HTML, CSS, PHP, blog, Wordpress, šablona

The thesis describes a process of creating a custom Wordpress template. It serves also as a guide. The thesis is divided into two main parts. The first is purely theoretical, it explains used technologies and their comparison. The second is practical, the development itself is described there.

HTML, CSS, PHP, blog, Wordpress, template

Prohlášení

Prohlašuji, že jsem svou maturitní práci vypracoval samostatně. K práci jsem použil literaturu a prameny, uvedené v seznamu.

Kód návrhu blogu bude po obhajobě maturitní práce uvolněn jako open-source pod licencí GNU GPLv3. Souhlasím s tím, aby moje maturitní práce byla využívána na Střední průmyslové škole Karviná, příspěvkové organizaci, v Microsoft STC, ale i kdekoli jinde v souladu s touto licencí.

V Karviné dne:

Podpis:

Poděkování

Děkuji vedoucímu práce Vladimírovi Hrubanovi za umožnění vzniku nového blogu pro Microsoft STC, za poradenství při zpracování projektu a samotné maturitní práce.

Děkuji paní učitelce Mgr. Renátě Kondělkové za rady týkajících se formální úpravy dokumentu maturitní práce.

Dále bych chtěl poděkovat kolegovi Petrovi Kučerovi za připomínky ke grafickému návrhu, spolupráci při opravě kritických chyb a nastavení SSO a workflow.

Nakonec bych rád poděkoval všem svým kolegům z Microsoft STC a všem ostatním, kteříž se na vývoji podíleli ať už přímo pomocí, či nepřímo podněty během interního testovacího procesu a hlášením nalezených chyb na produkci.

Obsah

Úvod	8
1 Teoretická část.....	9
1.1 Základní pojmy	9
1.1.1 Blog.....	9
1.1.2 Vzhled blogu.....	9
1.1.3 Správa blogu	10
1.2 Systém pro správu obsahu (CMS)	10
1.2.1 Dělení CMS	10
1.2.2 Nejznámější CMS pro blogy.....	10
1.3 Zásady návrhu a vývoje webových stránek	11
1.4 Technologie webového vývoje	12
1.4.1 HTML	12
1.4.2 CSS	14
1.4.3 JavaScript.....	16
1.4.4 PHP	19
1.5 Nástroje pro webový vývoj.....	19
1.5.1 Textový editor.....	19
1.5.2 Verzování.....	20
1.5.3 CI/CD.....	20
1.6 Hosting.....	21
1.6.1 XAMPP.....	21
1.6.2 Microsoft Azure	21
2 Praktická část.....	21
2.1 Grafický návrh blogu	21
2.1.1 Seznam článků	22
2.1.2 Článek	22
2.2 Příprava vývoje	23
2.2.1 Groupware.....	23
2.2.2 Správa kódu	24
2.2.3 Lokální testovací prostředí (XAMPP)	25
2.2.4 Hosting.....	27
2.2.5 CI/CD.....	28

2.3	Vlastní šablona.....	29
2.3.1	Princip šablony.....	30
2.3.2	PHP šablony.....	30
2.3.3	Styly	35
2.3.4	Obrázky.....	36
2.3.5	JavaScript.....	36
2.3.6	Ostatní soubory	36
2.4	Konfigurace Wordpressu	37
2.4.1	Zvolení šablony.....	37
2.4.2	Konfigurace pluginů	37
2.4.3	Přizpůsobení šablony	38
2.4.4	Přidání rubrik a štítků.....	39
2.4.5	Vytvoření příspěvku.....	39
2.5	Po dokončení šablony	40
	Závěr.....	42

Úvod

Úkolem maturitní práce je vytvořit webovou blogovací platformu pro program Microsoft Studentské Trenérské Centrum. Jde o program firmy Microsoft pro studenty středních škol. V rámci programu studenti vypracovávají odborné články se zaměřením na produkty a technologie firmy Microsoft.

Blog vyžaduje originální a poutavý vzhled, který je zároveň v souladu s grafickým jazykem Microsoftu. Dále se požaduje přehlednost pro čtenáře a také jednoduchost správy a případného dalšího přizpůsobování blogu. V neposlední řadě je potřeba zajistit správnou funkčnost na nejpoužívanějších platformách tak, aby byl blog dostupný co nejvíce čtenářům.

V dalších kapitolách jsou popsány využívané technologie. Pro návrh kostry byl využit značkovací jazyk HTML5, pro styly CSS3, pro interaktivní prvky jazyk JavaScript, pro propojení s back-endem jazyk PHP. Jsou popsány i využívané nástroje, především hosting, technologie CD/CI, verzování. V krátkosti jsou popsány základy kooperace na projektu. To vše je pak aplikováno v praktických ukázkách, a to především v souvislosti s použitým redakčním systémem Wordpress.

Přílohy: kopie produkční verze kódu blogu; dokumentace; návrh blogu v PDF.

1 Teoretická část

V teoretické části je popsána problematika tvorby blogu postupně od základů až ke konkrétním technickým podrobnostem. Nejprve je osvětlena základní problematiku blogů, jejich vzhledu a správy. Dále správa blogů pomocí redakčního systému, se zaměřením na redakční systém Wordpress. Kapitola Zásady webdesignu pak zahajuje odbornější část, která je věnována návrhu webu, potřebným technologiím (jazyky), nástrojům (například textový editor či verzovací nástroje) a také samotnému hostingu webu.

1.1 Základní pojmy

1.1.1 Blog

Blog je speciálním typem webové stránky, která slouží k publikaci příspěvků, které mohou obsahovat různé typy multimédií jako jsou zvuky a videa. Příspěvky bývají zobrazovány chronologicky. (1)

Od běžné webové prezentace (například osobního webu či prezentace firmy) se liší především v tom, že je jeho obsah častěji aktualizován. Blogovací služby nabízí již od základu software, pomocí kterého lze blog vytvořit i bez znalosti kódování webů. Takovou službu nabízí například blog.cz.

Podkategorií blogů jsou mikroblogy. Příspěvky jsou podstatně kratší než u blogů. Používají se pro sdílení osobních i odborných zpráv, novinek, pocitů a podobně. Často obsahují odkaz na jinou webovou stránku, kde může být problematika více rozepsána. (1) Příkladem mikroblogovací služby je Twitter, kde je možné na příspěvky i reagovat a hodnotit je. Tak vznikají různé diskuse. (2)

1.1.2 Vzhled blogu

Při tvorbě blogu je možné využít již existujících platforem, které nabízí bezplatný hosting a také i bezplatné šablony blogu. Pro laika je to velkou výhodou; může rychle vytvořit blog bez odborných znalostí. Bezplatné šablony přinášejí však mnoho nevýhod. Používá-li je mnoho dalších uživatelů, stávají se nezajímavými, neupoutají uživatele. Nebývají dobře optimalizované pro telefony, popřípadě je jejich grafický návrh proveden nekvalitně. Pro základní potřeby jednotlivce stačí, chce-li však vlastník blogu zaujmout, měl by blogu poskytnout vlastní osobitý styl.

Řešením je tvorba vlastní šablony. První možností je využít WYSIWYG editor. Jde o anglickou zkratku „What You See Is What You Get“, doslovně přeloženo „Co vidíš, to dostaneš.“ Editory tohoto typu okamžitě zobrazují finální vzhled dokumentu již při jeho úpravách. Zástupci jsou například: Wix Web Builder (online nástroj pro tvorbu webů) a BoldGrid (tvorba šablon pro Wordpress). Využití takových nástrojů se nedoporučuje z několika důvodů: generují neefektivní kód, šablona většinou nejde zcela přizpůsobit a některé nástroje umožňují export pouze na své hostingové služby. (3)

Další možností je kódování vlastní šablony pomocí HTML, CSS a dalších technologií. Tato možnost vyžaduje sice v porovnání nejvíce technických znalostí, oproti tomu nabízí 100% přizpůsobitelnost a nezávislost na platformě a poskytovateli služeb.

1.1.3 Správa blogu

Blog se dá spravovat třemi různými způsoby.

První možností je vytvoření statického webu. Do složky se vloží HTML, CSS a případně JavaScript soubory. Webový server pak uživatelům posílá exaktní kopie těchto souborů. Tato možnost je nejméně efektivní ze všech. Při vytvoření nové stránky je nutno některé části webu kopírovat (hlavička, patička), při vytvoření nového příspěvku se musí psát dokola stejný kód. Výhodou je nenáročnost na webový server – není vyžadována podpora PHP a při vývoji stačí obyčejný textový editor.

Druhou možností je využití preprocesoru, který se spouští před vložením na hosting. Příkladem takového nástroje je Jekyll. Umožňuje použití jednoduššího značkovacího jazyka, například Markdown. Na tomto základě pak vygeneruje HTML a CSS, které je možno přímo zkopírovat na server. (4)

Poslední a nejpoužívanější možností je využití redakčního systému. Takový software může být součástí hostingové služby, kdy nabízí škálu bezplatných šablon a WYSIWYG editor. Software tohoto typu je však možné si i vytvořit či stáhnout a vložit na libovolný jiný webový hosting.

1.2 Systém pro správu obsahu (CMS)

Systém pro správu obsahu (z angličtiny CMS = content management system) se používá ke správě webových stránek a webových aplikací. Pro CMS existují další používané termíny – redakční či publikační systém. (5)

CMS umožňuje zpravidla správu dokumentů pomocí WYSIWIG editoru, správu komentářů a zobrazení statistik. (5)

1.2.1 Dělení CMS

CMS se dělí dle několik kritérií:

- **dle ceny** – placené, zdarma,
- **dle platformy** – PHP, C#...,
- **dle zaměření** – blogy, wiki, fóra...

1.2.2 Nejznámější CMS pro blogy

Wordpress

CMS je na svém webu WordPress.org popisován takto: „WordPress je software určený pro kohokoliv, klade důraz na přístupnost, výkon, bezpečnost a jednoduchost použití.“ Wordpress začal jako čistě blogovací systém, nyní však umožňuje vytvářet i jednoduché osobní weby, e-shopy, rozsáhlé firemní prezentace a další. Jde o otevřený software, je možné program používat jakkoliv, upravovat jej a šířit jak jej samotný, tak i jeho kopie na základě listiny práv pro Wordpress. (6) Dále v praktické části.

Výhody	Nevýhody
rozsáhlá dokumentace, velká komunita	pro základní weby příliš pokročilý
mnoho pluginů	základní instalace může být zranitelná
uživatelsky přívětivé	žádná zaručená oficiální podpora

Tabulka 1.1 – výhody a nevýhody CMS Wordpress (7)

Joomla

Joomla byla vydána později než Wordpress. Na svých stránkách uvádí široké využití. Systém se dá použít nejen na klasické webové stránky a blogy, ale například i pro intranetové sítě, rezervační systémy a další rozsáhlé aplikace. (8)

Systém je napsaný v jazyce PHP a používá databázi MySQL.

Výhody	Nevýhody
mnoho pluginů (k listopadu 2019 je to přes 7800)	uživatelsky nepřívětivá administrace
aktivní komunita, mnoho dokumentace	nedostatek kvalitních šablon
	pro základní weby může být zbytečně pokročilý

Tabulka 1.2 – výhody a nevýhody CMS Joomla (7)

Drupal

Drupal nabízí také poměrně široké využití. Kromě blogů umožňuje tvořit také e-shopy a jiné komplexnější systémy. Jde o modulární systém, dbá na přehlednost a otevřenost svého kódu. (9)

Drupal byl naprogramován v jazyce PHP, a kromě MySQL se dá oficiálně propojit i s PostgreSQL a SQLite.

Výhody	Nevýhody
možnost komunitní podpory na internetu i osobně	pro jednoduché weby příliš pokročilý
přes 6 000 modulů	málo kvalitních bezplatných šablon
mnoho možností komerční podpory	složitý způsob tvorby vlastních šablon

Tabulka 1.3 – výhody a nevýhody CMS Drupal (7)

1.3 Zásady návrhu a vývoje webových stránek

Při návrhu webové stránky je nutno dodržovat mnohé zásady nejen při grafickém návrhu, ale i během vývoje. V mnoha případech je nutno dbát na specifické požadavky, avšak existuje několik zásad, které jsou platné téměř vždy a na nichž se shoduje mnoho designerů.

Zásada, jež je jmenována často jako první, je jednoduchost. V dnešní době je k dispozici mnoho zdrojů informací a pakliže není daná stránka přehledná a požadované informace snadno dohledatelné, uživatel často přejde na stránku jinou, kde si informace může dohledat snadněji.

S předchozím bodem souvisí přehlednost a intuitivnost navigace (hlavní nabídky, menu...). Je nutné klást důraz na správný výběr kategorií, dle kterého se příspěvky a stránky řadí, aby bylo používání blogu co nejjednodušší.

V neposlední řadě je nutno udělat dobrý dojem, a to především pokud stránka či blog reprezentují produkt s velkou konkurencí. Dobrý dojem lze vyvolat použitím správných barev, moderních prvků a jednotností návrhu. Pro výběr barevné palety existují generátory. Moderní prvky lze převzít z mnoha designových jazyků. V roce době tvorby blogu, roce 2019, jsou trendem ploché prvky bez prostorových efektů, jednoduché, ale pestré barevné kombinace, důraz na kontrast a střídme využití barevných přechodů.

Posledním bodem je nutnost důrazu na kompatibilitu mezi platformami. Je důležité přizpůsobit kód, aby se zobrazoval konzistentně a správně mezi prohlížeči, a to nejen na počítači, ale i na mobilních zařízeních – tabletech, telefonech. Webová stránka přizpůsobena pro běh na různých velikostech a orientacích displeje je označována jako responzivní. Z toho se odvozují dva přístupy při návrhu i kódování webu. „Desktop first“ znamená vývoj primárně pro počítače a následné přizpůsobení pro telefony. „Mobile first“ je tedy přístupem opačným. Projekt v této práci byl tvořen prvním jmenovaným přístupem.

1.4 Technologie webového vývoje

Základní webová stránka se skládá z několika souborů. První, nejdůležitější soubor, je dokument napsaný ve značkovacím jazyce HTML. Tento soubor určuje strukturu a samotný obsah webové stránky. Takový soubor umí otevřít webový prohlížeč přímo, i bez webového serveru. Častým způsobem je však využívání souborů PHP. Tento typ souboru kombinuje značky jazyka HTML se skriptem v jazyce PHP. PHP soubor se nejprve musí zpracovat webovým serverem, který spustí PHP kód, provede potřebné procesy a vygeneruje soubor v již čistém HTML, který pošle klientovi – webovému prohlížeči. Tento přístup se používá často u redakčních systémů, kdy PHP slouží k dosazení dat z databáze do HTML šablony. (10)

Dalším soubor, který nesmí chybět, je soubor s kaskádovými styly, CSS. Kaskádové styly umožňují nastavit vzhled celého HTML dokumentu, kupříkladu způsoby zarovnání, barvy a velikosti textu, přechody, animace... (11)

Posledním hojně využívaným typem je soubor se skripty v jazyce JavaScript. Tyto skripty se na rozdíl od skriptů v jazyce PHP spouštějí až na straně klienta. Mohou sloužit k triviálním operacím, jako je otevírání menu, až k obsluhování plnohodnotné webové aplikace. (12)

V následujících odstavcích jsou jmenované technologie popsány podrobněji.

1.4.1 HTML

HTML je odkazovací a značkovací jazyk. Jedná se o způsob, jakým je možné dát prostému textu význam tak, aby jej byly schopny především webové prohlížeče interpretovat jako formátovaný text s různými vlastnostmi. Dále slouží k odkazování na další stránky. Používá se především pro tvorbu formátovaných dokumentů na WWW. Nejnovější verze je HTML5. (10)

Syntaxe

Princip HTML tkví ve vytvoření dokumentu pomocí značek (tagů), které se dělí na dva typy:

- párové značky – mají otevírací a uzavírací značku, obsah se píše mezi značky, například: `<a>`, `<p></p>`,
- nepárové značky – mají pouze otevírací značku (`<input>`, `
`...), u XHTML a JSX je vyžadována takzvaná samozavírací značka (`<input/>`, `
` ...).

Pomocí párových značek se označuje text, dáváme mu význam nejen pro uživatele (prostřednictvím formátování), ale i pro počítač a vyhledávače (zdůrazníme, která část je důležitá, která naopak nikoliv).

Pomocí nepárových značek se může přidávat ať už grafické prvky (odřádkování, horizontální čára...), či funkční prvky (vkládání metadat, CSS, formulářový vstup...).

Většina značek umožňuje vložit i atribut, což je nějaká vlastnost dané značky. Atribut se vkládá ve formátu `nazev-atributu="vlastnost"`.

Základní kostra

V HTML se základní kostra ve většině případů neliší. Vždy se používá `DOCTYPE` pro vyznačení typu dokumentu (`html` pro HTML5), značku `html`, která uzavírá celý zbytek dokumentu a pak dvě hlavní části:

- `head` – hlavička webu, zde se píše informace pro vyhledávač, dané informace nejsou viditelné přímo ve stránce (buď vůbec, nebo v případě titulku je text v popisku karty prohlížeče), slouží ke konfiguraci. Pomocí značky `meta` se nastavuje třeba znaková sada a takzvaný `viewport`, pomocí kterého se pracuje s jednotkami `vh` (výška) a `vw` (šířka) značící procento z daného rozměru klienta (okno prohlížeče). Pomocí `link` můžeme připojit externí kaskádové styly. Pomocí `title` nastavujeme titulek stránky.
- `body` – obsah webu, zobrazuje se na výsledné stránce. V HTML5 existuje mimo jiné i velké množství takzvaných sémantických značek, které nemají žádné vestavěné stylování, ale umožní vystihnout funkci a důležitost částí na stránce. Ve starších verzích se používala značka `div`, nyní existují konkrétnější značky určující třeba:
 - hlavičku – `<header></header>`,
 - hlavní část stránky `<main></main>`,
 - sekci – `<section></section>`,
 - článek – `<article></article>`,
 - doplňující obsah, většinou tzv. sidebar – `<aside></aside>`,
 - patičku – `<footer></footer>`.

```

<!DOCTYPE html>
<html lang="cs-CZ">
<head>
  <meta charset="UTF-8">
  <link rel="stylesheet" href="style.css"/>
  <title>Titulek webu</title>
</head>
<body>
  <header></header>
  <main>
    <section>
      <article>
        <h1></h1>
        <p></p>
      </article>
    </section>
  </main>
  <aside></aside>
  <footer></footer>
</body>
</html>

```

Ukázka kódu 1.1 – základní kostra HTML

1.4.2 CSS

CSS neboli kaskádové styly slouží k nastavení grafických vlastností HTML elementů, umí prvky řadit, umisťovat, zabarvovat, měnit jejich velikost, nastavovat animace a přechody, měnit efekty. Nejnovější verzí je třetí verze, tedy CSS3. (11)

Základní pojmy a syntaxe

Soubor CSS se skládá z bloků tvořených třemi částmi:

- selektor – určuje, pro které elementy se dané vlastnosti aplikují,
- vlastnost – nastavení prvku, jehož hodnota se má změnit,
- hodnota – hodnota dané vlastnosti.

Syntaxe je následující:

```

selektor{
  vlastnost: hodnota;
}

```

Ukázka kódu 1.2 – základní syntaxe CSS

Existuje více druhů selektorů:

- typový – vybírají se všechny elementy daného typu:

```
p{  
  color: red;  
}
```

Ukázka kódu 1.3 – typový selektor

- třídí – volí se tzv. třídu, kterou můžeme přidělit více elementům na stránce, na které se vlastnosti v třídě aplikují:

HTML

```
<p class="cerveny">Text</p>
```

CSS

```
.cerveny{  
  color: red;  
}
```

Ukázka kódu 1.4 – třídí selektor

- id – zvolíme si jednoznačný identifikátor, který se může přiřadit jen jednomu elementu na stránce:

HTML

```
<p id="cerveny">Text</p>
```

CSS

```
#cerveny{  
  color: red;  
}
```

Ukázka kódu 1.5 – ID selektor

SCSS

SCSS je nadstavba jazyka CSS, která umožňuje psát přehledněji a efektivněji. Mezi nejznámější funkce se řadí podpora proměnných (užitečné hlavně v době, kdy CSS neposkytovalo nativní podporu proměnných), vnořování a takzvaných mixinů, což jsou vlastnosti, které se mohou expandovat na více vlastností – vhodné pro podporu více prohlížečů, či jednoduchou práci s modelem Flexbox a responzivitou. Příklady využití SCSS jsou uvedeny v praktické části.

SCSS není podporován přímo prohlížeči, proto se musí zkompilevat do CSS. Existují automatické kompilátory, které po každém uložení soubor hned překompilují, aby bylo možné sledovat změny živě.

Začlenění do dokumentu HTML

Existuje několik způsobů začlenění stylů do dokumentu HTML.

- Externí CSS – Nejpoužívanější způsob vloží CSS do HTML dokumentu pomocí značky `link`.
- Inline CSS – Jedná se o způsob psaní CSS přímo do dokumentu HTML. Využívá se značka `style`.
- Atribut style – Poslední možností je přiřazení elementu atribut `style`. Styl se aplikuje jen na daný element.

1.4.3 JavaScript

JavaScript je objektově orientovaný dynamický skriptovací multiparadigmatický jazyk. Ačkoliv jazyk patří mezi tzv. C-like jazyky, tedy jazyky podobné jazyku C, a obsahuje ve svém názvu název jazyka Java, je od těchto jazyků principiálně odlišný.

V roce 1997 byl JavaScript standardizován pod názvem ECMAScript. Od standardizované verze byl odvozen například ActionScript využívaný v Adobe Flash. (12)

JavaScript se využívá především ve webových stránkách. Spektrum použití je velice rozsáhlé – od jednodušších operací (otevírání menu) přes složitější (řízení animací a kontrola formulářů) až po vytváření komplexních aplikací. (12) Tyto aplikace však nemusí běžet jen v rámci webových stránek, jelikož existují frameworky, které umožňují vývoj desktopových aplikací pomocí webových technologií. Příkladem může být Electron – obsahuje renderovací jádro založené na Chromiu a prostředí Node.js. Pomocí HTML, CSS a JS tedy můžeme vytvořit aplikaci, která vypadá a chová se jako aplikace nativní. (13)

Kromě webových stránek a nativních klientských aplikací se JavaScript používá i na straně serveru, často v kombinaci s běhovým prostředím Node.js. (12)

Začlenění do dokumentu HTML

JavaScript je možné podobně jako CSS do HTML vložit několika způsoby. Prvním způsobem je využití značky `script`, do níž můžeme kód psát přímo. Ten můžeme vložit do značky `head`, nebo do značky `body`. Prohlížeče podporují i vložení přímo do značky `html`, to se však nedoporučuje, jelikož je to sémanticky nesprávné. Validátor vykáže chybné použití. Pomocí stejné značky můžeme vložit i odkaz na soubor s příponou `.js`, kde se píše samotný skript.

Pro větší přehlednost je dobrou praxí skripty umisťovat do externích souborů. Porovnání přístupu vyobrazuje ukázka.

```
<html>
  <body>

    <script>
      //kód
    </script>

    <script src="skript.js"></script>

  </body>
</html>
```

Ukázka kódu 1.6 – ukázka umístění JS kódu v HTML

Základní syntaxe

JavaScript nemá žádné vestavěné vstupně-výstupní funkce, tato funkcionality je vždy závislá na konkrétní implementaci. Většina implementací však obsahuje objekt `console`. Program „Hello World“ tak může vypadat následovně:

```
console.log("Hello World!");
```

Ukázka kódu 1.7 – „Hello World“ v JS

Jako pokročilejší příklad jsem vytvořil program využívající funkce a proměnné. Jedná se o program pro výpočet součtu dvou čísel. Nejprve definuji proměnné `a` a `b`, definuji funkci `secti` s dvěma parametry a poté funkci volám s argumenty již definovaných proměnných. Návrátová hodnota funkce se vypíše do konzole.

```
let a = 5;
let b = 3;

function secti(x, y){
  return x + y;
}

console.log(secti(a, b));
```

Ukázka kódu 1.8 – funkce v JS

Nejdůležitější funkce pro webový vývoj umožňují manipulovat s HTML stránkou pomocí takzvaného DOM (Document Object Model). Jde o objektovou reprezentaci stránky. JavaScript umí elementy vybírat, upravovat je, nastavovat jim atributy.

Pro vybírání elementů existují následující základní funkce:

- `document.getElementById(id)` – vybírá jediný na základě id,
- `document.getElementsByTagName(znacka)` – vybírá všech elementy se stejným názvem značky,
- `document.getElementsByClassName(trida)` – vybírá všechny elementy se stejným názvem třídy.

Následující příklad využívá funkci pro výběr pomocí id a funkci `innerHTML` sloužící k přepsání kódu uvnitř daného elementu.

```
<html>
  <body>

  <p id="ukazka"></p>

  <script>
    document.getElementById("ukazka").innerHTML = "Hello World!";
  </script>

  </body>
</html>
```

Ukázka kódu 1.9 – přepsání kódu uvnitř elementu pomocí `innerHTML`

Pomocí JavaScriptu je možné nastavovat jednotlivým elementům atribut `style`, tedy CSS přímo v HTML, nebo CSS třídu či id. Je možné využít příkazy v ukázce:

```
<html>
  <body>

  <p id="ukazka"></p>

  <script>
    document.getElementById(id).style.vlastnost = "hodnota";
    document.getElementById(id).classList.toggle("trida");
  </script>

  </body>
</html>
```

Ukázka kódu 1.10 – nastavování CSS pomocí JS

První příkaz nastaví konkrétní vlastnost v atributu `style`, druhý příkaz nastavuje třídu o daném názvu v uvozovkách – je-li elementu přiřazena, smaže ji a naopak. Druhá jmenovaná funkcionality se hojně využívá při vytváření tlačítka pro otevírání nabídky.

1.4.4 PHP

PHP je skriptovací programovací jazyk. Ačkoliv umožňuje tvořit konzolové a desktopové aplikace, je využíván především pro tvorbu dynamických webových stránek. Jeho zastoupení mezi jazyky používanými pro serverové aplikace se pohyboval za rok 2019 kolem 79 %. Jeho oblíbenost mezi webovými vývojáři dokazuje i poměrně zažitá zkratka LAMP. Jedná se o první písmena technologií využívaných spolu s PHP. L pro operační systém Linux, a pro webový server Apache, M pro databázový systém MySQL, P pro PHP, Python či Perl. (14) V PHP je napsáno mnoho rozsáhlých projektů, například sociální síť Facebook a také redakční systém Wordpress.

Základní syntaxe

Pro spuštění PHP skriptu se běžně používá webový server s podporou PHP. Po načtení PHP souboru v prohlížeči se veškerý PHP kód zpracuje a klientovi se zašle soubor s čistým HTML.

Základní syntaxi demonstruji na složitějším příkladu programu „Ahoj, světe“, kdy využívám i proměnné. Nejprve složím řetězec ze tří částí pomocí operátoru tečky, uložím do proměnné a její obsah pak vypíšu na obrazovku.

```
<?php
    //Složení řetězce a vložení do proměnné.
    $promenna = "Ahoj" . ", " . "svete!";
    //Výpis obsahu proměnné do dokumentu.
    echo $promenna;
?>
```

Ukázka kódu 1.11 – program „Ahoj, světe!“ v PHP

V praktické části je popsána konkrétní syntaxe potřebná k práci s Wordpressem.

1.5 Nástroje pro webový vývoj

Pro efektivní vývoj je nutno využívat správné nástroje. Následující kapitola popisuje software běžně využívaný v praxi.

1.5.1 Textový editor

Textový editor je základem pro tvorbu kódu. Mezi základní textové editory patří i program Poznámkový blok vestavěný v operačních systémech Windows. Takto jednoduché programy však neposkytují takový komfort, jako textové editory specializované pro programování. Příkladem může být Visual Studio Code, což je open-source textový editor od firmy Microsoft. Nabízí navíc zvýrazňování syntaxe, kontrolu syntaxe, práci ve více oknech a rozšiřitelnost mnoha bezplatnými pluginy, mezi které se řadí v projektu využívaný kompilátor SCSS. (15)

1.5.2 Verzování

U většiny komplexnějších softwarových projektů se využívá správa verzí, zkráceně verzování. Systémy správy verzí umožňují navracet se ke starším verzím, vytvářet větve verzí nezávislé na hlavním vývoji (za účelem testování experimentálních funkcí), jednotlivé verze označovat pro produkci, popřípadě snadno spolupracovat na projektu v týmu a pracovat na projektu na více zařízeních (stolní počítač doma, notebook v práci a na cestách...). (16)

Pro projekt byl vybrán systém git. Využívá totiž lokálního i centrálního úložiště a je velice rychlý. Jedinou nevýhodou je složitější rozhraní oproti populárnímu systému Subversion.

Služba GitHub poskytuje bezplatné úložiště pro privátní i veřejné repozitáře. Repozitář projektu je proto uložen právě na službě GitHub.

Ačkoliv je git poměrně komplexní program, pro účely projektu stačily především tyto základní příkazy:

- `git add .` – přidání kódu do indexu (dočasná úschova kódu před uložením do repozitáře),
- `git commit -m "Komentář k verzi."` – přidání kódu z indexu do repozitáře,
- `git push` – odeslání lokálního repozitáře do vzdáleného na serveru,
- `git pull` – stažení dat ze vzdáleného repozitáře.

Při prvním připojení ke vzdálenému repozitáři je nutné se ověřit. Ověření může probíhat dle typu spojení buď přes SSH pomocí certifikátu, nebo přes HTTPS pomocí jména a hesla. Pro jednoduchost se v projektu používá ověření pomocí HTTPS. Spolupracující kolega tedy mohl být jednoduše přidán ve webovém rozhraní GitHubu k projektu a on sám se mohl ověřit údaji k přihlášení ke službě.

1.5.3 CI/CD

Zkratka CI/CD představuje Continuous Integration a Continuous Delivery. Jedná se o způsoby práce na kódu zajišťující plynulost a pokud možno automatizaci procesů. Kombinují se s agilními metodikami vývoje kódu. Ty se vyznačují operativním plánováním a přizpůsobováním projektu i během jeho vývoje. (17)

Continuous Integration označuje pravidelné (každodenní) sjednocování kódu spolupracovníků do hlavní větve.

Continuous Delivery pak výsledky sjednocení průběžně doručuje na produkci, tedy přímo k zákazníkům.

Azure Pipelines

Jedním z nástrojů CI/CD je služba Azure Pipelines, která je součástí Azure DevOps. Umožňuje snadnou integraci s git repozitářem (který je i vestavěný), popřípadě vybranými externími službami, například službou GitHub. (18)

Konfigurace služby je blíže popsána v praktické části.

1.6 Hosting

Ačkoliv webové prohlížeče umožňují čtení lokálních HTML souborů bez nutnosti běhu webového serveru, PHP soubory již vyžadují server s PHP preprocesorem. Proto bylo nutno vybrat takový server nejen pro běh na produkci – Microsoft Azure, ale i pro samotný vývoj – Apache ze sady XAMPP. Provozu webové aplikace na webovém serveru se říká hosting.

1.6.1 XAMPP

XAMPP je balíček nástrojů používaných pro vývoj webových aplikací. Byl vybrán ze dvou důvodů – jedná se o populární open-source řešení doporučované na mnoha fórech a také zahrnuje všechny potřebné programy pro potřeby vývoje blogu na Wordpressu – webový server Apache, interpret PHP a databázový server MariaDB (náhrada za MySQL). (19)

1.6.2 Microsoft Azure

Nástroj XAMPP samozřejmě není vhodný k hostování aplikací na produkci. Existuje mnoho služeb, které nabízejí hosting. Ať už hosting virtuálního stroje, kde je možné si nainstalovat a nakonfigurovat vlastní webový server a databázi, či hosting zaměřený přímo pro webové aplikace, kdy dojde k instalaci potřebných nástrojů automaticky.

Pro účely hostingu blogu byla vybrána cloudová služba Microsoft Azure z prostého důvodu: nabízí jednoduché správcovské rozhraní, možnost vzdálené správy přes PowerShell a rychlé ruční či automatické škálování.

2 Praktická část

Vše, co bylo rozebráno v teoretické části, je konkrétně využito v části praktické. Tato kapitola je rozdělena do několika částí – první část je věnována grafickému návrhu, druhá přípravě celého vývojového ekosystému, třetí je věnována šabloně, konkrétním souborům a částem kódu, čtvrtá popisu konfigurace redakčního systému pro produkční provoz. V poslední části se krátce shrnují kroky provedené po dokončení šablony. Ačkoliv jsou již mimo rozsah práce, uvádím je pro kompletnost.

2.1 Grafický návrh blogu

Prvním krokem při vývoji každé webové stránky je tvorba grafického návrhu. Pro účely návrhu blogu jsem vybral nástroj Microsoft PowerPoint. Ačkoliv jde primárně o nástroj pro tvorbu prezentací, vzhledem k jednoduchosti ovládání a širokého spektra vestavěných objektů je možné jej využít i pro jednoduché návrhy.

Návrh byl vytvořen v rámci hackathonu¹ Microsoft OneWeek

Vycházel jsem z grafického jazyka firmy Microsoft s názvem UI Fabric (20), abych zachoval jednotnost a soulad s firemními stránkami Microsoftu. Jako primární barva byla zvolena hlavní barva programu – modrá. Konkrétní odstíny pak byly odvozeny z

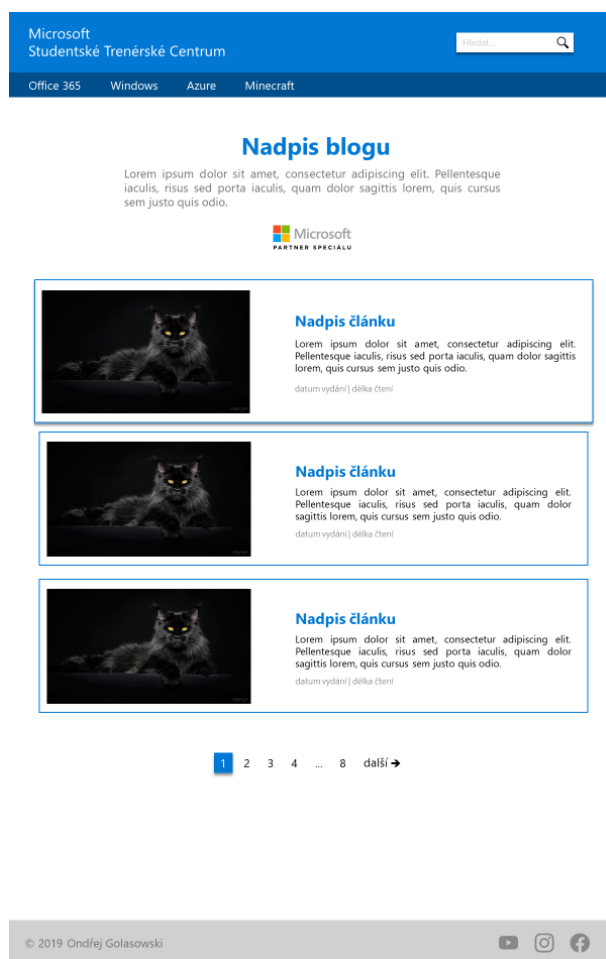
¹ „akce, při níž programátoři, případně ve spolupráci s grafiky a webdesignéry, intenzivně pracují na zadaném softwarovém projektu“ (25)

grafické knihovny Microsoftu Office UI Fabric. Byly vytvořeny dva návrhy, od nichž se pak při tvorbě kódu odvozovaly všechny ostatní stránky.

Záhlaví a zápatí mají všechny stránky stejné. V záhlaví je logo programu, vyhledávač a nabídka kategorií. V zápatí pak copyright text a odkazy na sociální síť. V hlavní části se pak stránky v některých aspektech liší. Ty jsou přesněji popsány v následujících kapitolách. Návrhy v plné velikosti jsou k dispozici v příloze v souboru „Návrh.pdf“.

2.1.1 Seznam článků

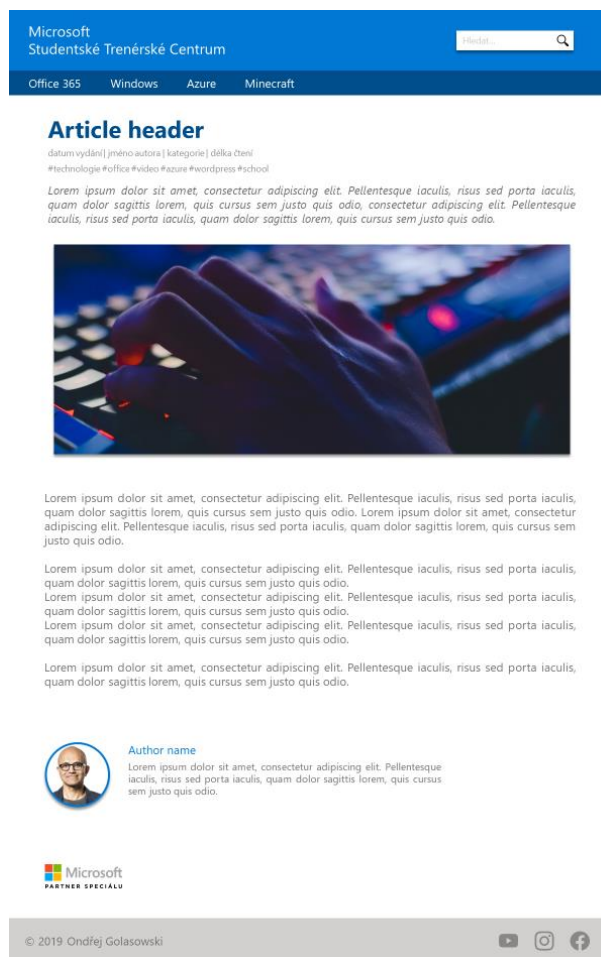
První návrh zachycuje zobrazení seznamu úryvků jednotlivých článků. Byl navržen univerzálně tak, aby jej bylo možné použít pro zobrazení seznamu všech článků, článků z dané rubriky, a i článků jednotlivých štítků. Samotný seznam je uveden nadpisem, popisem a logem speciálu Microsoft. Následuje seznam náhledů. Každý náhled obsahuje nadpis, úryvek, metadata a obrázek. Na základě pravidel UI Fabric byl jako efekt vybraného článku zvoleno zvětšení a stín. Na konci hlavní části je vyobrazen výběr aktuální stránky. Návrh je vyobrazen na obrázku Obrázek 2.1 – návrh seznamu článků na následující straně.



Obrázek 2.1 – návrh seznamu článků

2.1.2 Článek

V druhém návrhu je zobrazeno rozložení jednoho článku při jeho rozkliknutí. V hlavičce se nachází nadpis článků, metadata a samotný článek. Na konci je medailonek autora s odkazem na jeho profil.



Obrázek 2.2 – návrh článku

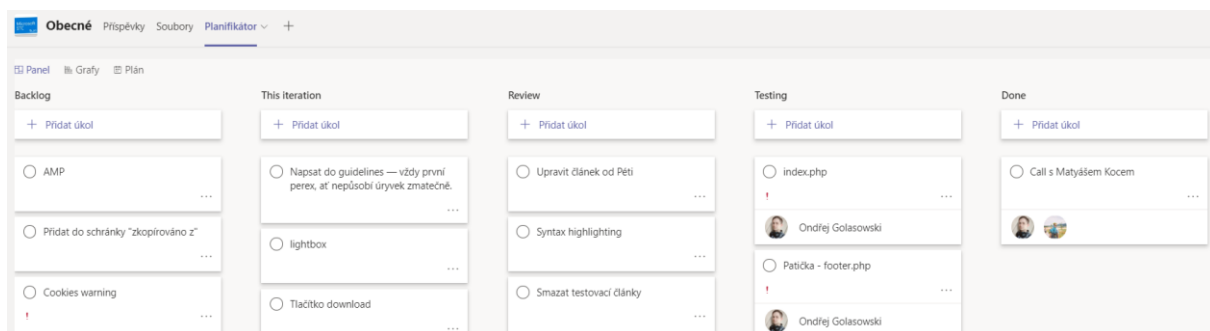
2.2 Příprava vývoje

2.2.1 Groupware

Groupware je software určený pro spolupráci v týmech. Z důvodu zjednodušení komunikace s vedoucím práce a kolegy, který měl za úkol zpracovat některé části projektu, bylo vhodné mít k dispozici vhodný software. Byl vybrán program Microsoft Teams, kde byl v rámci STC tenantu² vytvořen tým. V týmu je možno psát zprávy, posílat přílohy a využívat mnoho dalších aplikací, které byly do Teams integrovány.

Příkladem aplikace spolupracující s Teams je Microsoft Planner. Ta slouží k rozdělování a plánování úkolů. V Planner byly vytvořeno vytvořeno 5 kontejnerů, které reprezentují kategorie úkolů. Do první kategorie s názvem Backlog se řadí požadované funkce k implementaci. V druhé kategorii, This iteration, jsou funkce právě implementované. Třetí kategorie, Review, představuje hotové funkce čekající na zhodnocení od vedoucího projektu. Čtvrtý sloupec je pro funkce, které je nutné otestovat. V posledním sloupci jsou dokončené funkce vhodné pro spuštění na produkci. Následující obrázek zachycuje náhled úkolů.

² označení pro identifikaci firmy či školy s licencí balíku Office 365



Obrázek 2.3 – náhled úkolů v aplikaci Planner

2.2.2 Správa kódu


Klíčovou součástí vývojového ekosystému je správa samotného kódu. Prvním krokem je nastavení vzdáleného repozitáře.

Ve službě GitHub se repozitář tvoří poměrně snadno. Po přihlášení stačí v seznamu repozitářů kliknout na tlačítko „New“ a nakonfigurovat základní nastavení.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?
[Import a repository.](#)

Owner Repository name *

 andreondra /

Great repository names are short and memorable. Need inspiration? How about [super-memory?](#)

Description (optional)

☒ **Public**
Anyone can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

☐ **Initialize this repository with a README**
This will let you immediately clone the repository to your computer.

Add .gitignore: None Add a license: None ⓘ

[Create repository](#)

Obrázek 2.4 – prvotní nastavení repozitáře

Repozitář byl nakonfigurován jako soukromý. Jeho uvolnění proběhne až po obhajobě maturitní práce. Repozitář se nemusel inicializovat, proběhne nahrání lokálního repozitáře.

Dalším krokem je vytvoření lokálního repozitáře a nahrání na server. Používá se tato sada příkazů:

1. `git init` – vytvoření lokálního repozitáře,
2. `git remote add origin <adresa serveru>` - přidání vzdáleného repozitáře pod názvem origin (zvyklost).

Nyní se může nahrát lokální repozitář pomocí příkazů popsanych v teoretické části. Po prvotním přihlášení se údaje uloží a již nic nebrání plnohodnotné synchronizaci práce.

2.2.3 Lokální testovací prostředí (XAMPP)

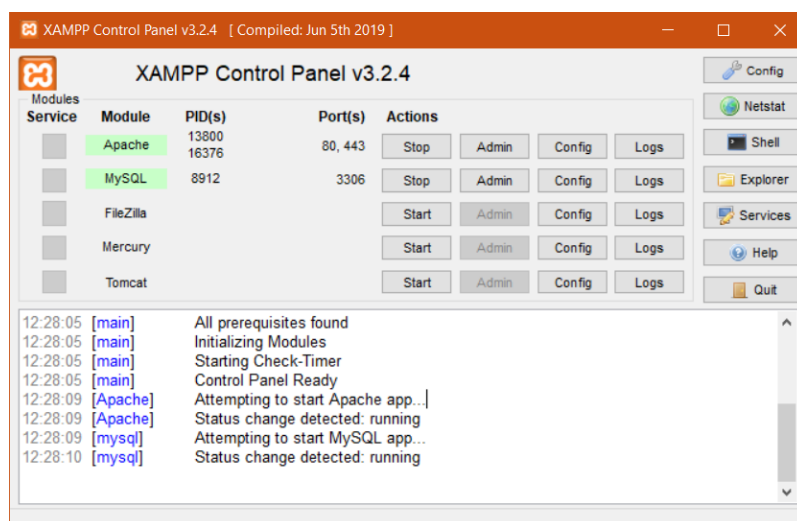
Ačkoliv je možné díky CI/CD využívat pouze vzdáleného webového serveru a tímto způsobem web testovat, není to zcela vhodné pro vývoj. Při testování lokálním jsou změny vidět okamžitě a není třeba je pokaždé nahrávat do repozitáře potažmo do hostingu.

Nasazení v lokálním prostředí probíhá podobně jako na klasickém hostingu, koneckonců XAMPP využívá Apache. Ve výchozím nastavení má Apache jako kořenovou složku htdocs, která se nachází v místě instalace balíčku XAMPP. Vývojář má v tomto případě tři možnosti:

1. zkopírování projektu do složky htdocs,
2. vytvoření symbolického odkazu ve složce htdocs – vytvoření „tunelu“ ve složce htdocs do jiné složky v počítači,
3. vytvoření virtuálního hostitele v konfiguračním souboru – nasměrování Apache do jiné než kořenové složky.

V projektu je využívána druhá možnost, především kvůli jednoduchosti a bez nutnosti zbytečně data kopírovat. Symbolický odkaz (tzv. symlink) se ve Windows tvoří příkazem `mklink /D <start> <cíl>`. Jako start zvolím název, přes který budu k webové stránce přistupovat. Jako cíl zvolím skutečné umístění složky s Wordpressem. Příkaz tedy může vypadat takto: `mklink /D C:\xampp\htdocs\stcblog D:\projekty\stcblog`. Při tvorbě odkazu je nutno dbát na dvě věci – v případě tvorby na disku C: je důležité spustit cmd.exe jako správce a pokud obsahuje cesta k odkazu mezery, tak je potřebné cestu uzavřít do uvozovek.

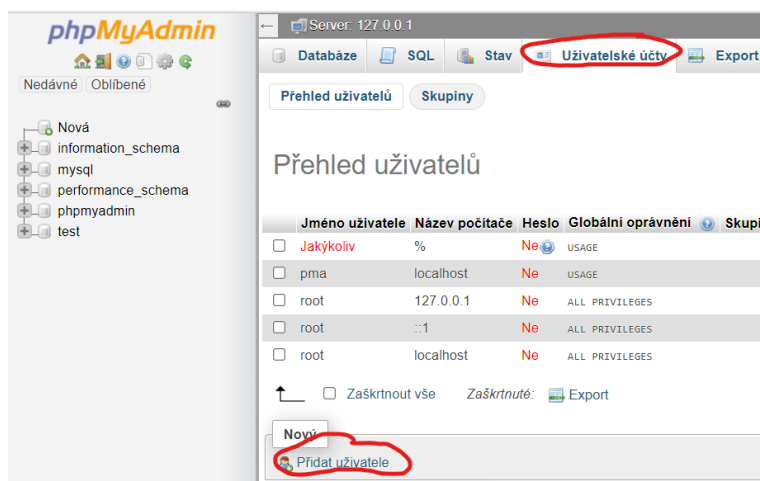
Do složky zvolené v přechodícím odstavci je samozřejmě nutné nahrát samotný Wordpress. Ten je možno stáhnout z oficiálních stránek, rozbalit a nakopírovat. Po spuštění aplikace `xampp-control.exe` je možno spustit Apache a MySQL.



Obrázek 2.5 – rozhraní XAMPP po spuštění Apache a MySQL

Prvním krokem instalace je vytvoření uživatele a databáze v MySQL. XAMPP již v základu obsahuje phpMyAdmin, což je interaktivní grafické rozhraní pro práci s databází. To ulehčí mnoho práce.

Po zadání adresy <http://localhost/phpmyadmin/> do prohlížeče mám možnost vytvořit uživatele společně s databází stejného jména a automaticky přidělenými oprávněními. Na následujícím obrázku je znázorněn postup – zvolení karty Uživatelské účty, vytvoření uživatele.



Obrázek 2.6 – vytvoření uživatele v phpMyAdmin

Následuje zadání údajů o uživateli a potvrzení vytvoření databáze zaškrtnutím checkboxu. Akci je nutno potvrdit tlačítkem vpravo dole. Další obrázek tento proces ilustruje.

Obrázek 2.7 – zadání údajů o uživateli

Po zadání údajů je možné pokračovat v nastavení Wordpressu. Nejprve je nutno v konfiguračním souboru wp-config.php nastavit údaje pro připojení k databázi. Jedná se o název databáze (v lokálním prostředí localhost), uživatelské jméno pro připojení, heslo, adresu a znakovou sadu (ve výchozím nastavení utf8mb4). Vše vysvětlují komentáře v daném souboru. Kromě toho je nutno nastavit tzv. salts, pomocí něž se generují cookies. Postup je taktéž popsán přímo v souboru.

Posledním krokem je nastavení Wordpressu pomocí průvodce přímo na lokální adrese (localhost/nazev_symlinku). Proces je to vskutku jednoduchý, vše je česky okomentováno.

The screenshot shows the 'Vítejte!' (Welcome!) screen of the WordPress installation wizard. It includes a welcome message, a section for 'Základní informace' (Basic information) with fields for website name, username, password, and email, and a 'Důležité upozornění' (Important notice) about password strength. At the bottom is a button to 'Instalovat WordPress' (Install WordPress).

Vítejte!

Prvním uvítacím krokem je pro uživatele WordPressu už samotná jednoduchá instalace. Stačí pouze vyplnit všechny potřebné informace a za okamžik budete moci naplno využívat ten nejznámější redakční systém na světě.

Základní informace

Zadejte prosím následující informace. Nemusíte se ničeho obávat, všechno lze později v administraci webu jednoduše změnit.

Název webu

Uživatelské jméno

Uživatelská jména mohou obsahovat pouze alfanumerické znaky (číslíce, velká a malá písmena anglické abecedy), mezery, podtržítka, pomlčky, tečky a symbol @.

Heslo [Zobrazit](#)

Důležité upozornění: Zvolené heslo budete potřebovat pro přihlášení do administrace webu, uložte si ho prosím na bezpečném místě.

Potvrďte heslo ☐ Potvrďte použití slabého hesla

Emailová adresa

Raději si ještě jednou přezkontrolujte zadanou emailovou adresu, protože na ni bude zasláno administrátorské heslo.

Dostupnost pro vyhledávače ☐ Zakázat prohledávání a indexování obsahu webu

Záleží však pouze na vyhledávačích, zda budou zvolené nastavení respektovat.

[Instalovat WordPress](#)

Obrázek 2.8 – nastavení Wordpressu pomocí průvodce

Po nastavení dojde k vyzvání zadání jména a hesla a pak se zobrazí samotné rozhraní Wordpressu. K podrobné konfiguraci se vrátí v následujících kapitolách.

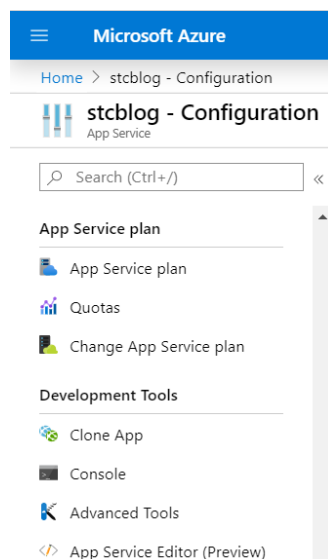
2.2.4 Hosting

Agilní vývoj předpokládá nasazování na produkční prostředí (nebo alespoň testovací) již během vývoje. Pro tyto účely je vhodné založit hosting hned na začátku, aby se mohlo rovnou napojit i CI/CD.

Pro jednoduchost jsem využil šablonu nasazení Wordpress do služeb Azure. Šablona využívá Azure App Service, webový server na virtuálním stroji s operačním systémem dle výběru (Windows Server, nebo Linux). Databázi MySQL šablona nakonfiguruje autonomně.

Šablona funguje velmi jednoduše, v README.md dané šablony na GitHubu je tlačítko, po kliknutí na něj dochází k přesměrování na Azure a po zadání základních údajů, jako je jméno aplikace, dojde k nasazení. Pak je možno na portálu Azure (<https://portal.azure.com/>) najít odkaz na webovou aplikaci (vždy ve tvaru <https://nazevaplikace.azurewebsites.net/>) a na ní již běží nasazená instance systému Wordpress i s nakonfigurovaným wp-config.php. Jediné, co je nutno ručně nastavit v souboru, jsou zmiňované salts, ostatní údaje se převezmou z konfigurace aplikace.

Azure App Service naštěstí obsahuje vestavěný on-line editor založený na VS Code – App Service Editor.



Obrázek 2.9 – App Service Editor v Azure

Po otevření editoru se zobrazí seznam souborů v aplikaci a je možno salts zadat stejně jako při konfiguraci v lokálním prostředí.

Pak zbývá jen na adrese aplikace nastavit Wordpress v průvodci a je možno Wordpress plně využívat.

2.2.5 CI/CD

Jak již bylo zmíněno v teoretické části, pro CI/CD byl vybrán nástroj Azure Pipelines v Azure DevOps, především kvůli jeho snadné integraci s repositářem na službě GitHub a službou Azure.

Po přihlášení na adrese <https://dev.azure.com/> je možno vytvořit nový projekt služby DevOps. Je nutno vypsát základní informace podobné údajům při tvorbě repositáře na GitHubu. Projekt bude soukromý, CD/CI není třeba pro projekt této povahy zveřejňovat.

Create new project

×

Project name *

Description

Visibility

Public

Anyone on the internet can view the project. Certain features like TFVC are not supported.

Private

Only people you give access to will be able to view this project.

Advanced

Version control

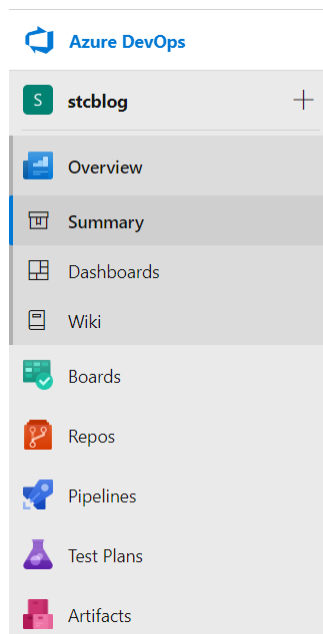
Git

Work item process

Basic

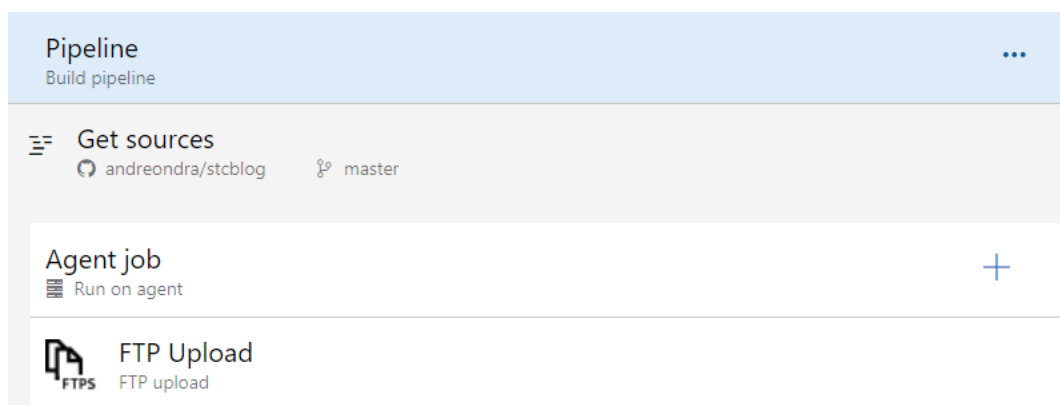
Obrázek 2.10 – tvorba projektu v Azure DevOps

Ačkoliv služba DevOps nabízí mnoho nástrojů, v rámci projektu se využívá pouze Pipelines. Je však možné celý vývojový proces převést do této služby. Místo aplikace Planner je možno využít Boards, místo repozitáře na GitHubu je možné použít Repos a je možné tvořit i autonomní testování.



Obrázek 2.11 – přehled služeb DevOps

Výsledná Pipeline vypadá následovně. Nejprve načte obsah z repozitáře ze služby GitHub a poté veškerý obsah nahraje pomocí FTP na webový server. Repozitář se nakonfiguruje automaticky, stačí Azure DevOps připojit k účtu na GitHub. Pak se přidá jednoduchý úkol – nahrání na FTP pomocí vestavěné funkce FTP Upload. Údaje pro připojení k FTP jsou k dispozici na portálu Azure v záložce Deployment Center.



Obrázek 2.12 – výsledná Pipeline

2.3 Vlastní šablona

Wordpress obsahuje vestavěné šablony, které je možno použít. Existuje i obrovské množství bezplatných i placených šablon. Žádná ze jmenovaných možností však nenabídne takový rozsah přizpůsobení jako tvorba šablony vlastní.

Tvorba šablony začíná vytvořením složky v adresáři s šablonami. Šablony se nachází v podsložce wp-content/themes. Do této složky vkládáme soubory s kódem šablony.

2.3.1 Princip šablony

Pro správné vytvoření šablony je nutné pochopit, jak Wordpress s šablonou pracuje. Na rozdíl od statického webu, kdy jsou data i kostra stránky v jednom souboru (uživatel se tedy posílá kopie souboru ze serveru), u dynamických webů jsou data uložena v databázi a kostra stránky je definována v použité šabloně. Při přístupu uživatele na danou stránku se data z databáze zkopírují na určená místa v šabloně (na základě PHP funkcí) a uživateli se posílá takto složený HTML soubor.

Wordpress má pro každý typ stránky vlastní soubor s šablonou. Aby však vývojáři nemuseli tvořit pokaždé všechny soubory pro všechny případy (jako v případě tohoto projektu), existuje hierarchie, podle které Wordpress postupně zjišťuje přítomnost šablonových souborů a pokud je nenajde, postupně se dostane až na index.php, ze kterého mohou vycházet stránky všech typů. Tomuto se říká „fallback“ (ustupovací, záložní) systém. Pro lepší pochopení je přiložen následující obrázek, na kterém je vyobrazeno, jaké soubory Wordpress postupně zkouší.



Obrázek 2.13 – hierarchie šablonových souborů

Ve vyšším rozlišení je k dispozici interaktivní verze diagramu na <https://wp hierarchy.com/>. Podrobné informace podává oficiální dokumentace <https://developer.wordpress.org/themes/basics/template-hierarchy/>.

2.3.2 PHP šablony

Pro základní funkčnost celé šablony jsou třeba jen dva soubory – soubor index.php a style.css. V následujících odstavcích jsou popsány veškeré části šablony (jednotlivě také nazývány jako šablony), které blog obsahuje. Šablonových souborů existuje ještě více, avšak

pro účely blogu nebyly potřebné. V rámci kapitoly jsou popsány i potřebné funkce pro implementaci daných částí šablony.

V každé podkapitole je popsán kód. Ukázka je uvozena číslem řádku, na které se kód nachází. V první podkapitole jsou popsány veškeré funkce v souboru, v dalších podkapitolách jsou popsány jen funkce, které ještě nebyly zmíněny.

index.php

Soubor index.php je hlavním souborem celé šablony. Používá se pro zobrazení hlavní strany. Také pokud Wordpress nenajde specifický soubor pro danou funkci, vždy využije soubor index.php. Například: pokud ve složce s šablonou neexistuje soubor category.php, Wordpress využije pro zobrazování kategorií soubor index.php.

V index.php je využito několik funkcí a cyklus pro zobrazení článku, který je v oficiální dokumentaci Wordpressu nazýván The Loop (21).

```
4      echo get_bloginfo('template_directory') . get_css_name();
```

První funkce vrací metadata blogu, v tomto případě je jako argument umístění samotné šablony. Druhá funkce vrací verzi stylů, jde o vlastní funkci definovanou ve functions.php. Obě návratové hodnoty jsou pomocí operátoru tečka spojeny v jeden řetězec a vloženy do stránky pomocí `echo`. Používá se pro zjednodušení zadávání cest a pro zajištění funkčnosti i po změně vnější struktury složek. Využití nalézá nejen při vkládání stylů, ale i při vkládání obrázků a dalšího externího statického obsahu.

```
6      <?php bloginfo( 'charset' ); ?>
```

Funkce `bloginfo()` funguje podobně jako `get_bloginfo()`, akorát nevrací řetězec, nýbrž hodnotu posílá přímo na výstup. Argument `charset` vyžaduje aktuální znakovou sadu pro správnou konfiguraci HTML dokumentu.

```
7      <?php bloginfo('name'); ?>
```

Získá se název blogu.

```
8      <?php wp_head(); ?>
```

Vkládá další potřebný kód Wordpressu, například skripty pluginů či vlastní kaskádové styly.

```
11     <?php get_header(); ?>
```

Vloží obsah souboru header.php.

```
15     <?php echo get_bloginfo('name'); ?>
```

Vloží název blogu z databáze.

```
16     <?php echo get_bloginfo('description'); ?>
```

Vloží popis blogu z databáze.

```
20     <?php if ( have_posts() ) : while ( have_posts() ) : the_post(); ?>
```

Jedná se o první řádek cyklu The Loop. `have_posts` zjišťuje, zdali jsou k dispozici příspěvky, `the_post()` načte další příspěvek z databáze a umožní funkcím přistupovat k jednotlivým datům daného příspěvku. Funkce `the_permalink()` vkládá odkaz na

příspěvek (aby se na něj dalo z náhledu prokliknout), `the_post_thumbnail([300, 200])` vloží náhledový obrázek v rozlišení 300×200 pixelů, `the_title()` vkládá titulek článků, `the_excerpt()` úryvek z článku, funkce `get_the_date()` vrací datum publikování a `reading_time()` je opět vlastní funkce, která na základě počtu slov vrací odhadovanou délku čtení.

```
37 <?php endwhile; endif; ?>
```

Uzavírá cyklus a podmínku.

```
40 <?php the_posts_pagination([
```

Funkce vkládá na dané místo přepínač stránek, pokud počet článků překročí mez nastavenou ve Wordpressu. Na dalších řádcích následuje konfigurace textu pro tlačítka – předcházející a následující strana.

```
47 <?php get_footer(); ?>
```

Vkládá obsah souboru footer.php.

category.php

category.php zobrazuje náhledy článků z dané kategorie. Od souboru index.php se liší pouze ve dvou funkcích.

```
6 <?php echo get_bloginfo('name'); ?> | <?php single_cat_title(); ?>
```

V titulku je kromě názvu blogu nově obsažen i název kategorie pomocí funkce `single_cat_title()`. Tato funkce je použita i na 14. řádku jako nadpis stránky.

```
15 <?php echo category_description(); ?>
```

Vrací popis kategorie. Ten lze každé kategorii nastavit přímo v rozhraní Wordpressu.

tag.php

tag.php se také moc neliší od hlavního index.php. Používá se pro zobrazení náhledů článků dle určitého štítku.

```
7 <?php bloginfo('name'); ?> | #<?php single_tag_title(); ?>
```

V titulku stránky se tentokrát zobrazuje název štítku pomocí funkce `single_tag_title()`. Křížek před funkcí je pouze pro dekorativní účely (pro vytvoření tzv. hashtagu).

```
15 <?php single_tag_title(); ?>
```

Zobrazuje název štítku, používá se pro název stránky. Popis štítku zobrazen na rozdíl od kategorie není.

single.php

Tato šablona slouží k zobrazení jednotlivých článků (např. při jejich rozkliknutí na hlavní straně). Chybí zde The Loop; není potřeba, jelikož se zobrazuje právě jeden článek.

```
6 <?php echo get_bloginfo('name'); ?> <?php wp_title('| ', true, 'left'); ?>
```

V šabloně single.php se v titulku zobrazuje název článku pomocí funkce `wp_title()`. První parametr je separátor, druhý povolení k zobrazení a třetí umístění separátoru.


```
14 <?php the_post(); the_title(); ?>
```

14. řádek má dva úkoly – načíst požadovaný článek z databáze a vložit na místo kódu titulek článku.

```
19 <?php echo the_category( ' ' ); ?>
```

Vrací kategorii článku.

```
21 <?php if( get_the_tags() != '' ): ?>
```

Zjišťuje, jsou-li k dispozici štítky. Pokud ano, tak pomocí funkce `the_tags()` tyto štítky vypíše. `endif;` pak uzavírá podmínku. Podmínka je použita z prostého důvodu – aby na stránce nevznikalo prázdné místo v případě neexistence štítků.

```
30 <?php the_content(); ?>
```

Vrací obsah článku.

page.php

Funguje podobně jako `single.php`, Wordpress tuto šablonu využívá pro zobrazování statických stránek (místo příspěvků). Pro `single.php` byl zvolen jednodušší vzhled bez metadat, aby bylo umožněno tvořit většinu vzhledu stránky právě v administrátorském rozhraní Wordpressu.

Používá stejné funkce jako `single.php`.

404.php

Wordpress tuto šablonu použije tehdy, naskytne-li se chyba 404, to znamená, že daná stránka neexistuje.

```
16 <?php echo get_theme_file_uri( 'assets/img/cat.svg' ); ?>
```

Umožňuje přeložit relativní adresu na absolutní tak, aby se obrázek správně vložil do šablony.

search.php

Jedná se o šablonu, která se používá pro zobrazení výsledků vyhledávání. Obsahuje The Loop pro zobrazení všech nalezených příspěvků.

Liší se od `index.php` v několika záležitostech, nejmarkantnějším rozdílem je bezpochyby skript sloužící pro zobrazení počtu nalezených článků. Začíná na řádce 15. Skript načte z databáze objekt do proměnné `$allcount` se všemi příspěvky a do proměnné `$count` uloží počet nalezených příspěvků. V návěští `switch` se pak na základě počtu zobrazí správně vyskládaná věta i s případným počtem nalezených příspěvků. `<?php echo get_search_query(); ?>` pak vloží hledaný výraz. Tato funkce se nachází i v titulku.

searchform.php

S šablonou `search.php` souvisí `searchform.php`. Slouží totiž k návrhu vyhledávacího pole, do nějž se zadává text k vyhledání.

Obsahuje tag `form` s atributem `action` nastaveným na kořenový skript pomocí `<?php echo home_url('/'); ?>`, aby se vyhledávaný řetězec zpracoval. Ve `form` je vložené tlačítko pro vyhledávání (ikona v odkazu) a `input`. Jeho hodnota (atribut `value`) je nastavena

na `<?php echo get_search_query() ?>` pro zachování vyhledávaného dotazu v případě zobrazení na stránce `search.php` (aby uživatel mohl dotaz upravit a nemusel jej psát znovu).

Samotné odeslání dotazu a animace obstarává skript v jazyce JavaScript popsany v dalších kapitolách.

author.php

Jak již může název napovídat, `author.php` se používá pro zobrazení stránky s informacemi o autorovi článku. Obsahuje opět The Loop pro vypsání všech článků daného autora. Kromě toho obsahuje speciální funkce pro zobrazení informací o autorovi.

```
7 <?php the_author(); ?>
```

Zobrazí jméno autora. Podle toho, jak si autor zvolí, se zobrazí buď celé jméno, nebo přezdívká.

```
15 <?php echo get_avatar( get_the_author_meta( 'ID' )); ?>
```

Zobrazí profilový obrázek autora.

```
18 <?php if(!(empty(get_the_author_meta('description')))): ?>
```

Zjistí, jestli autor vyplnil svůj popis. Pokud ano, tak vloží nový odstavec s informacemi pomocí funkce `<?php echo get_the_author_meta('description') ?>`.

```
21 <?php echo get_the_author_meta('user_email'); ?>
```

Vkládá e-mail autora.

header.php

Obsahuje hlavičku stránky, která se vkládá pomocí funkce `get_header()`. Obsahuje i vyhledávací formulář.

```
8 <?php get_search_form(); ?>
```

Vrací obsah souboru `searchform.php`. Tedy vyhledávací formulář.

```
16 <?php wp_nav_menu(); ?>
```

Vrací menu vytvořené ve Wordpressu.

footer.php

Obsahuje patičku stránky, která se vkládá podobně jako hlavička pomocí funkce `get_footer()`.

Obsahuje několik vlastních funkcí, které načítají určitá data z databáze. Vše jde nakonfigurovat z prostředí Wordpressu tak, aby si šablonu mohl částečně upravit i člověk neznalý programování.

```
2 <?php echo get_theme_mod('settings_footer_copyright', '');?>
```

Načte copyright text.

```
4 <?php if(esc_url(get_theme_mod('settings_footer_logo1')) != ""):?>
```

Zjistí, existuje-li obrázek pro první sociální síť. Pokud ano, tak vloží odkaz na danou síť pomocí funkce `<?php echo esc_url(get_theme_mod('settings_footer_link1'))?>` a do něj obrázek pomocí funkce:

```
<?php echo esc_url(get_theme_mod('settings_footer_logo1'))?>.
```

```
15 <?php wp_footer(); ?>
```

Vloží interní kód Wordpressu, především skripty. Je nutno jej psát do patičky těsně před uzavřením tagu `body`.

functions.php

Soubor `functions.php` již není šablona, nýbrž vlastní funkce definované vývojářem. Všechny funkce jsou podrobně okomentovány v daném souboru.

První funkce, `remove_empty_p()`, je převzatá od Ryana Hamiltona. Smaže všechny prázdné odstavce, aby na stránce nevznikaly přebytečné mezery.

Druhá funkce, `get_css_name()`, přidává k adrese uložení stylů (`style.css`) i jejich verzi. Bez této funkce by prohlížeče při aktualizaci stylů používali nadále starou verzi uloženou v mezipaměti, což je samozřejmě nežádoucí.

Třetí funkce a čtvrtá funkce přidávají další funkce do nástroje Customizer. Tento nástroj umožňuje uživateli blogu přizpůsobit některé aspekty šablony bez nutnosti zasahovat do blogu. (22) Funkce `stcblog_customize_register()` přidá nejprve sekci s názvem „Footer“ (patička) pomocí `add_section()`, poté vytvoří nové nastavení šablony pro ukládání copyright textu pomocí `add_setting()` a nakonec vloží ovládací prvek pomocí `add_control()`, do kterého bude uživatel copyright text psát. V další části volá funkci `stcblog_add_socials()`, což je vlastní funkce, která podobným způsobem vygeneruje nastavení a ovládací prvky pro vkládání ikon sociálních sítí a odkazů na ně.

Poslední funkcí je `reading_time()`, která na základě počtu slov v článku vrátí řetězec s odhadovanou dobou čtení.

2.3.3 Styly

Nedílnou součástí šablony jsou kaskádové styly. Vzhled blogu byl přizpůsoben pro správné zobrazení v nejpoužívanějších prohlížečích – mimo jiné prohlížeče založené na jádru Blink (Microsoft Edge Beta, Google Chrome) a WebKit (Safari prohlížeče na iOS, iPadOS).

Veškeré styly se musí ukládat do souboru `style.css`. V rámci vývoje by to však bylo nepraktické, proto je využit preprocesor SCSS, který styly ze složky `assets\scss` zkompiluje do CSS. Preprocesor je nastaven tak, aby všechny styly s názvem začínajícím na podtržítko ignoroval a zkompiloval pouze ostatní soubory. Jediný soubor, který v názvu nemá podtržítko, je `style.scss`. Do tohoto souboru se pomocí `@import` vkládají všechny ostatní soubory. Tím je zajištěno, že se soubory sloučí ve správném pořadí a vše se zkompiluje do jednoho souboru, který se pak automaticky vloží do kořenové složky – `style.css`. Proveďte se také tzv. minifikace, což je odstranění mezer a znaků nových řádků tak, aby se soubor co nejvíce zmenšil a zvýšila se tak rychlost načítání blogu.

Pro každou stránku existuje samostatný SCSS soubor. Kromě toho jsou ve složce `assets\scss` i další soubory.

Ve složce `fabric` je uložena grafická knihovna Fabric UI, která se během projektu využívá.

Ve složce `normalize` jsou uloženy styly, které sjednocují zobrazení na všech prohlížečích.

Soubor `_global.scss` obsahuje styly aplikované pro všechny typy stránek. V souboru `_variables.scss` jsou uloženy používané proměnné.

Soubor `_mixins.scss` obsahuje mixiny. Vytvořil jsem si funkci `for-size()`, do které je možno vkládat styly určené pouze pro zařízení o daném rozlišení a funkci `flexbox()`, která zjednodušuje práci s CSS modelem flexbox. Mixiny se do ostatních souborů vkládají pomocí `@include`.

2.3.4 Obrázky

Složka `img` obsahuje několik obrázků využívaných v šabloně. Prvním je `cat.svg`, který se používá na stránce `404.php`. Další je `favicon.ico`, slouží jako ikona stránky pro prohlížeče. Dále `mslogo.png`, což je logo Microsoftu vložené do šablony. Poslední dva soubory – `stclogo_small.svg` a `stclogo_std.svg` jsou loga STC – menší a větší varianta automaticky zvolená dle velikosti obrazovky.

2.3.5 JavaScript

V rámci projektu byl vytvořen skript v JavaScriptu s názvem `main.js`. Je využita knihovna jQuery pro zajištění snadnější manipulace s DOM. Funkce jQuery lze poznat podle znaku `$`. Obsahuje dvě vlastní funkce.

Na začátku je funkce `$(document).ready()`, což je funkce, která zavolá tzv. callback anonymní funkci ve svém parametru hned, jakmile se celá stránka načte.

První funkcí je `menu()`, která přidáním třídy `active` do sekce s menu zobrazí jeho obsah. Kromě toho přidá třídu `active` i samotnému tlačítku, aby se změnil jeho vzhled a indikovalo se otevření menu.

Funkce `searchForm()` slouží jednak k tomu, aby se při kliknutí na tlačítko ve vyhledávacím formuláři spustilo vyhledávání a také k tomu, aby se vyhledávací pole otevřelo (zvětšilo) při najetí myši (`mouseenter`) a při aktivním psaní (`focuson`).

2.3.6 Ostatní soubory

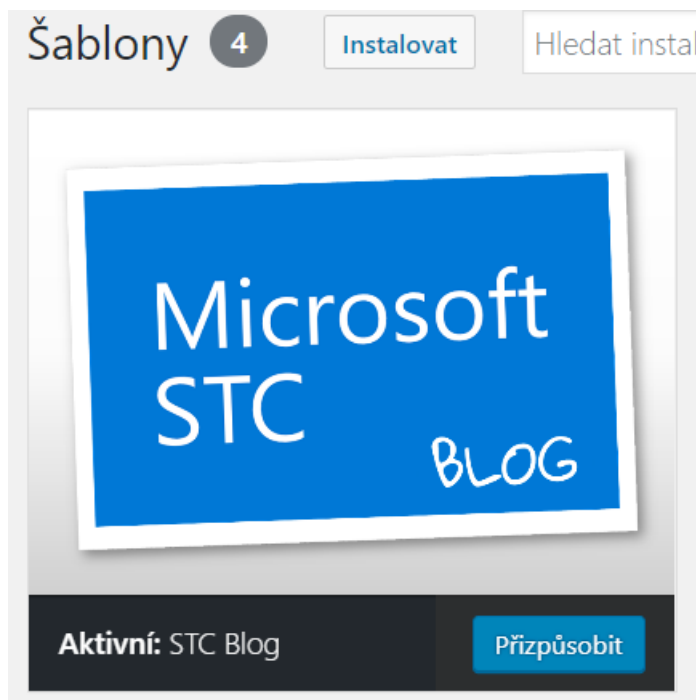
Ve složce `.vscode` je konfigurační soubor. Pomocí něj je nakonfigurován plugin pro kompilaci SCSS.

V kořenovém adresáři jsou další soubory. Soubor `azure-pipelines.yml` je automaticky vytvořený soubor pro běh CI/CD. Soubor `screenshot.png` je logo šablony zobrazované v rozhraní Wordpressu. Soubor `README.md` slouží k uchovávání informací o projektu ve značkovacím jazyce Markdown. Repozitáře jej zobrazují jako popis hned pod výčtem souborů. V souboru `LICENSE` je uložena licence projektu.

2.4 Konfigurace Wordpressu

2.4.1 Zvolení šablony

Prvním krokem při konfiguraci je zvolení správné šablony. Stačí v levém menu kliknout na Vzhled a Šablony. Šablona blogu se pak zobrazuje takto:



Obrázek 2.14 – šablona v seznamu šablon

2.4.2 Konfigurace pluginů

Následuje instalace pluginů, tedy přídatných modulů. Wordpress má vestavěný obchod s pluginy, takže instalace je velmi jednoduchá. Přímo v rozhraní je třeba kliknout na Pluginy a Instalace pluginů. Otevře se nabídka s vyhledáváním.

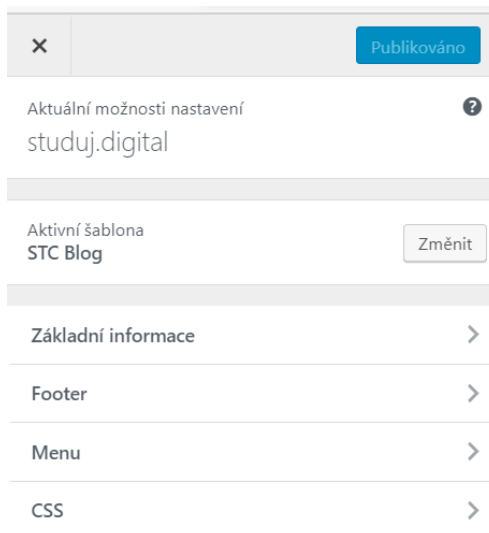
V produkčním Wordpressu byly nainstalovány tři hlavní pluginy. Prvním je SyntaxHighlighter Evolved, který umožňuje vkládat bloky kódu se zvýrazňováním syntaxe. Nevyžaduje žádnou další konfiguraci.

Druhý plugin umožňuje přihlašování pomocí technologie SSO (Single Sign On). Tato technologie umožňuje přihlašování na různých stránkách pomocí stejných účtů. Projekt využívá plugin pro propojení se službou Azure Active Directory, což je zjednodušeně řečeno mimo jiné úložiště účtů používaných v rámci Office 365 služeb. Nebylo tedy nutné všechny účty ručně kopírovat; kolegové z STC se mohou do administrátorského rozhraní blogu připojit stejně jednoduše, jako do služeb Office 365. Konfigurace pluginu je k dispozici v repozitáři vývojáře: <https://github.com/psignoret/aad-ss0-wordpress>.

Třetí plugin zjednodušuje schvalování příspěvků. Umožňuje autorovi poslat článek svému nadřízenému, který to zase může odeslat na jazykovou korekturu, až se článek nakonec dostane ke schválení týmu pro sociální síť, kdy se může publikovat. Tento plugin byl konfigurován kolegou, informace ohledně schvalovacího procesu jsou k dispozici v příloze v rámci dokumentace k blogu.

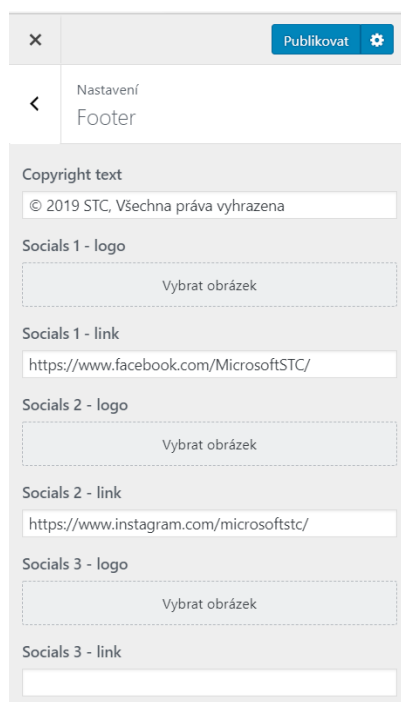
2.4.3 Přizpůsobení šablony

Již bylo zmíněno v rámci popisu souboru functions.php, že je možné nějaké aspekty šablony upravit přímo v rozhraní pomocí Customizera. Wordpress nabízí nějaké vestavené možnosti. První možností je vyplnění základních informací. Zde se píše název a popis webu. To, kde se tyto informace zobrazí, již záleží na šabloně. Šablona blogu tyto informace zobrazuje v titulku a na straně index.php. Je možno nastavit i favicon. Druhou možností je vytvoření vlastního menu pomocí jednoduchého rozhraní. Je možno odkazovat na příspěvky, rubriky (kategorie), štítky i statické stránky nebo jinam do internetu. Třetí možností je přidání vlastního kódu CSS.



Obrázek 2.15 – možnosti přizpůsobení blogu

V rámci projektu byly přidány i další dvě možnosti přizpůsobení, které se nachází v nabídce Footer. Je možno zadat copyright text i tři ikony a odkazy na sociální sítě.

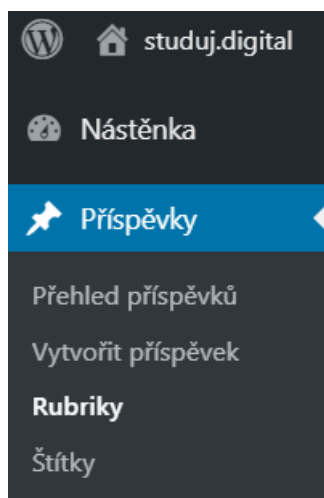


Obrázek 2.16 – vlastní možnosti přizpůsobení (Footer)

2.4.4 Přidání rubrik a štítků

Rubriky a štítky se tvoří po vybrání příslušných možností v nabídce Příspěvky, ve kterých je možno vyplnit název a popis. Články se pak před publikací zařadí do rubriky a přiřadí se jim jednotlivé štítky.

Rubriky lze vnímat jako kategorie a štítky jako klíčová slova.



Obrázek 2.17 – rubriky a štítky

2.4.5 Vytvoření příspěvku

Příspěvek se tvoří kliknutím na Vytvořit příspěvek v nabídce Příspěvky. Zobrazí se hlavní část, se kterou se pracuje podobně jako s textovým procesorem. V pravé části je možno nastavit další parametry článku jako zařazení do rubriky, přiřazení štítků, zvolení autora a přiřazení náhledového obrázku.

Dokument **Blok** ×

Publikování ^

Viditelnost [Veřejná](#)

Publikovat [Okamžitě](#)

☐ Zvýraznit tento příspěvek

☐ Čeká na schválení

Autor Ondřej Golasowski ▼

Rubriky ^

☐ Azure

☐ IT Pro

☐ Programming

☐ Minecraft

☐ Nezařazené

☐ Office 365

☐ Windows

[Vytvořit novou rubriku](#)

Štítky ^

Vytvořit nový štítek

Náhledový obrázek ▼

Stručný výpis příspěvku ▼

Nastavení komentářů ▼

Obrázek 2.18 – parametry článku

2.5 Po dokončení šablony

Pro zajištění co nejvyšší kvality blogu bylo nutné po dokončení šablony provést několik dalších věcí.

Především bylo nutné zakoupit doménu. Byla využita služba domena.cz. Pak stačilo nasměrovat DNS a záznamy na server Azure. V rozhraní Azure bylo pak potřebné povolit využití vlastních domén. V současné době má blog dvě oficiální domény: <https://studuj.digital/> a <https://studujdigital.cz/>.

Dalším krokem bylo zajištění certifikátů pro šifrovaný HTTPS přenos. V Azure bylo nastaveno automatické obnovení certifikátu ze služby Let's Encrypt.

Za účelem snadného hlášení chyb byl vytvořen formulář na platformě Microsoft Forms. Formulář obsahuje pole pro popis chyby, pro definici platformy, na které se chyba objevila a pro nahrání snímků obrazovky. Všechny nahlášené chyby se automaticky posílají do kanálu v aplikaci Microsoft Teams pro zprůhlednění a sjednocení všech informací týkajících se blogu pro vývojáře.

Na závěr byla vytvořena dokumentace s informacemi a pravidly pro kolegy. Dokumentace byla vytvořena ve spolupráci s kolegou Petrem Kučerou, který do dokumentace

připsal informace o procesu schvalování článků. V rámci pilotního provozu byl také veden webinář, jehož náplní byl popis ovládání Wordpressu a představení základních pravidel a nastavení.

Stále probíhají úpravy a vylepšení, již však na veřejném a plně funkčním produkčním provozu. To umožňuje mimo jiné i technologie CI/CD. Jedná se o zařazení analýz a statistik užívání webu. Dále se vylepšuje vzhled a přidávají se případné další funkce na základě zpětných vazeb kolegů a dalších uživatelů.

Závěr

Cílem práce bylo především vytvořit funkční blog s vlastním návrhem. Tohoto cíle bylo zcela dosaženo, práce byla dokončena. Proběhlo interní testování a na základě úspěšných výsledků bylo provedeno vydání do produkce.

Blog je veřejný a již se do něj přidávají články. Ohlasy kolegů jsou pozitivní, chyby byly v průběhu vyladěny a nyní blog slouží svému zamýšlenému účelu.

V rámci tvorby šablony vznikla i rozsáhlá dokumentace ve formátu absolventské i maturitní práce. Proto doufám, že se práce dále využije i jako výukový materiál pro všechny zájemce o vlastní Wordpress šablonu.

U projektu jsem se naučil mnoho dovedností s programovacím jazykem PHP a redakčním systémem Wordpress.

Ted' už existuje jedno místo, na které mohou kolegové všechny své články publikovat a získávat tak cenné reference.

Seznam obrázků

Obrázek 2.1 – návrh seznamu článků	22
Obrázek 2.2 – návrh článku	23
Obrázek 2.3 – náhled úkolů v aplikaci Planner	24
Obrázek 2.4 – prvotní nastavení repozitáře	24
Obrázek 2.5 – rozhraní XAMPP po spuštění Apache a MySQL	25
Obrázek 2.6 – vytvoření uživatele v phpMyAdmin	26
Obrázek 2.7 – zadání údajů o uživateli.....	26
Obrázek 2.8 – nastavení Wordpressu pomocí průvodce	27
Obrázek 2.9 – App Service Editor v Azure	28
Obrázek 2.10 – tvorba projektu v Azure DevOps	28
Obrázek 2.11 – přehled služeb DevOps	29
Obrázek 2.12 – výsledná Pipeline	29
Obrázek 2.13 – hierarchie šablonových souborů.....	30
Obrázek 2.14 – šablona v seznamu šablon.....	37
Obrázek 2.15 – možnosti přizpůsobení blogu	38
Obrázek 2.16 – vlastní možnosti přizpůsobení (Footer).....	38
Obrázek 2.17 – rubriky a štítky	39
Obrázek 2.18 – parametry článku.....	40

Seznam tabulek

Tabulka 1.1 – výhody a nevýhody CMS Wordpress (7)	11
Tabulka 1.2 – výhody a nevýhody CMS Joomla (7).....	11
Tabulka 1.3 – výhody a nevýhody CMS Drupal (7)	11

Seznam ukázek kódu

Ukázka kódu 1.1 – základní kostra HTML	14
Ukázka kódu 1.2 – základní syntaxe CSS	14
Ukázka kódu 1.3 – typový selektor	15
Ukázka kódu 1.4 – třídní selektor.....	15
Ukázka kódu 1.5 – ID selektor	15
Ukázka kódu 1.6 – ukázka umístění JS kódu v HTML	17
Ukázka kódu 1.7 – „Hello World” v JS.....	17
Ukázka kódu 1.8 – funkce v JS	17
Ukázka kódu 1.9 – přepsání kódu uvnitř elementu pomocí innerHTML.....	18
Ukázka kódu 1.10 – nastavování CSS pomocí JS	18
Ukázka kódu 1.11 – program „Ahoj, světe!” v PHP	19

Reference

1. **Příspěvatelé Wikipedie.** Blog. *Wikipedie: Otevřená encyklopedie*. [Online] 4. září 2019. [Citace: 17. listopad 2019.]
2. **Twitter.** About. *Twitter*. [Online] 2019. [Citace: 17. listopad 2019.] <https://about.twitter.com/>.
3. **Bernheim, Laura.** Home > How-To7 Best WYSIWYG Web Builder Reviews. *HostingAdvice.com*. [Online] 29. říjen 2019. [Citace: 17. listopad 2019.] <https://www.hostingadvice.com/how-to/wysiwyg-web-builder/>.
4. **The Jekyll Team.** Docs. *Jekyll*. [Online] 2019. [Citace: 20. listopad 2019.] <https://jekyllrb.com/docs/>.
5. **Bechyňová, Marta.** 7. Redakční systémy. *Stránky k výuce informatiky*. [Online] 2019. [Citace: 20. listopad 2019.] <http://www.ivt.mzf.cz/seminar/7-redakcni-systemy/>.
6. **Wordpress.** O nás: Naše mise. *WordPress.org*. [Online] 2019. [Citace: 20. listopad 2019.] <https://cs.wordpress.org/about/>.
7. **Chapman, Cameron.** TOP 10 CONTENT MANAGEMENT SYSTEMS. *webdesignerdepot.com*. [Online] 31. říjen 2011. [Citace: 20. listopad 2019.] <https://www.webdesignerdepot.com/2011/10/top-10-content-management-systems/>.
8. **Open Source Matters, Inc.** About Joomla! *Joomla!* [Online] 2019. [Citace: 20. listopad 2019.] <https://www.joomla.org/about-joomla.html>.
9. **Příspěvatelé Wikipedie.** Drupal. *Wikipedie: Otevřená encyklopedie*. [Online] 30. červenec 2019. [Citace: 20. listopad 2019.] <https://cs.wikipedia.org/w/index.php?title=Drupal&oldid=17520861>.
10. —. Hypertext Markup Language. *Wikipedie: Otevřená encyklopedie*. [Online] 12. listopad 2019. [Citace: 21. listopad 2019.] https://cs.wikipedia.org/w/index.php?title=Hypertext_Markup_Language&oldid=17844805.
11. —. Kaskádové styly. *Wikipedie: Otevřená encyklopedie*. [Online] 15. říjen 2019. [Citace: 21. listopad 2019.] https://cs.wikipedia.org/w/index.php?title=Kask%C3%A1dov%C3%A9_styly&oldid=17731237.
12. —. JavaScript. *Wikipedie: Otevřená encyklopedie*. [Online] 24. duben 2019. [Citace: 27. září 2019.] <https://cs.wikipedia.org/w/index.php?title=JavaScript&oldid=17176949>.
13. **Electron.** Electron. *Electron*. [Online] 2019. [Citace: 30. listopad 2019.] <https://electronjs.org/>.
14. **Příspěvatelé Wikipedie.** LAMP. *Wikipedie: Otevřená encyklopedie*. [Online] 13. únor 2018. [Citace: 2. prosinec 2019.] <https://cs.wikipedia.org/w/index.php?title=LAMP&oldid=15850163>.

15. **Microsoft.** Visual Studio Code. *Visual Studio Code*. [Online] Microsoft, 2019. [Citace: 31. prosinec 2019.] <https://code.visualstudio.com/>.
16. **Wikipedie, Příspěvatelé.** Verzování. *Wikipedie: Otevřená encyklopedie*. [Online] 31. říjen 2019. [Citace: 31. prosinec 2019.] <https://cs.wikipedia.org/w/index.php?title=Verzov%C3%A1n%C3%AD&oldid=17798883>.
17. **Pittet, Sten.** Continuous integration vs. continuous delivery vs. continuous deployment. *Atlassian CI/CD*. [Online] 2019. [Citace: 31. prosinec 2019.] <https://www.atlassian.com/continuous-delivery/principles/continuous-integration-vs-delivery-vs-deployment>.
18. **Microsoft.** Azure Pipelines. *Microsoft Azure*. [Online] 2019. [Citace: 31. prosinec 2019.] <https://azure.microsoft.com/cs-cz/services/devops/pipelines/>.
19. **Friends, Apache.** XAMPP. *Apache Friends*. [Online] 2019. [Citace: 31. prosinec 2019.] <https://www.apachefriends.org/index.html>.
20. **Microsoft.** Office UI Fabric. *Microsoft Developer*. [Online] 2019. [Citace: 26. prosinec 2019.] https://developer.microsoft.com/en-us/fabric#.
21. **Wordpress.** The Loop. *Wordpress.org*. [Online] Wordpress, 2019. [Citace: 29. prosinec 2019.] https://codex.wordpress.org/The_Loop.
22. —. Theme Options – The Customize API. *Wordpress.org*. [Online] Wordpress, 2019. [Citace: 31. prosinec 2019.] <https://developer.wordpress.org/themes/customize-api/>.
23. **Janovský, Dušan.** Něco málo o PHP. *Jak psát web*. [Online] 2019. [Citace: 21. listopad 2019.] <https://www.jakpsatweb.cz/php/>.
24. **Q-Success.** Historical trends in the usage of server-side programming languages for websites. *W3Techs*. [Online] 2019. [Citace: 2. prosinec 2019.] https://w3techs.com/technologies/history_overview/programming_language.
25. **Příspěvatelé Wikipedie.** Hackathon. *Wikipedie: Otevřená encyklopedie*. [Online] 2. říjen 2018. [Citace: 26. prosinec 2019.] <https://cs.wikipedia.org/w/index.php?title=Hackathon&oldid=16504117>.