Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Wordpress modul pro Czech - American TV

Plzeň 2017 Jan Šedivý

Místo této strany bude zadání práce.

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 1. května 2017

Jan Šedivý

Poděkování

Tímto bych chtěl poděkovat panu Ing. Martinu Dostalovi, Ph.D. a panu Ing. Jiřímu Vaňkovi za věcné rady a odborné vedení této práce. Dále bych rád poděkoval za spolupráci panu Johnu Honnerovi z Czech-American TV.

Abstract

This bachelor's thesis focuses on development of a WordPress plugin for Czech-American TV, that visualizes data with the use of Google maps. Map content can be filtered by keywords. The plugin also includes interface for data administration. The work describes the content management system WordPress and its main components. Mostly those parts of WordPress that are related to developing plugins for this system are described in this work. Next part of the thesis deals with the development of the WordPress plugin. Plugin is implemented with an emphasis on using the existing API provided by WordPress. That increases the probability of correct functionality even with the arrival of new WordPress versions.

Abstrakt

Cílem této bakalářské práce je vytvoření WordPress pluginu pro organizaci Czech-American TV, který slouží k vizualizaci dat s využitím Google mapy. Obsah na mapě lze filtrovat pomocí klíčových slov. Součástí pluginu je také rozhraní pro administraci těchto dat. Teoretickou část práce tvoří popis systému pro správu obsahu WordPress a jeho základních komponent. Popisovány jsou převážně ty části WordPressu, které souvisejí s tvorbou rozšíření pro tento systém. Praktická část práce se zabývá problematikou tvorby WordPress pluginu pro Czech-American TV. Plugin je implementován s důrazem na využívání existujícího API poskytovaného systémem WordPress. Tím je zvýšená pravděpodobnost správné funkčnosti pluginu i s příchodem nových aktualizací WordPressu.

Obsah

1 Úvod

Systém pro správu obsahu WordPress (WP) je v oblasti tvorby webových aplikací velice rozšířený. Ačkoliv je označován jako redakční systém, díky jeho vysoké přizpůsobitelnosti a možnosti přidávání rozšíření (takzvaných pluginů) v něm lze vytvořit daleko rozsáhlejší webové aplikace.

V teoretické části této práce bude čtenář seznámen se systémem pro správu obsahu WordPress. Budou popsány jednotlivé komponenty tohoto systému se zaměřením na možnosti rozšiřování funkcionality WordPressu pomocí pluginů. Stručně budou vysvětleny i další technologie, které budou využité pro vytvoření zadaného pluginu.

Webová aplikace organizace Czech-American TV tvořená systémem WordPress je obsahově zaměřena na poskytování informativních článků a videí o české kultuře jejím anglicky mluvícím návštěvníkům. Součástí tohoto obsahu jsou i příspěvky popisující zajímavá místa v jednotlivých krajích České republiky. Potřeby této organizace týkající se tvorby pluginu budou analyzovány a na základě této analýzy budou navržena vhodná řešení, kterými lze plugin implementovat tak, aby splňoval požadované funkcionality. Tato řešení budou navrhována se snahou využití funkcí poskytovaných aplikačním rozhraním pro tvorbu WP pluginů.

Praktickou částí této práce je vytvoření WordPress pluginu, který bude příspěvky popisující zajímavá místa v České republice vizualizovat s využitím Google mapy. Plugin zároveň přidá možnost kategorizace těchto příspěvků v podobě klíčových slov. Mapa umožní filtrování příspěvků pomocí těchto klíčových slov a pomocí regionů, které jsou v aplikaci již zavedené. Součástí pluginu bude také administrační rozhraní, ve kterém bude administrátor aplikace moci spravovat lokace těchto příspěvků a které bude integrováno do stávající administrace WordPressu. Vytvářené výstupy pluginu budou stylizovány tak, aby odpovídaly stávajícímu motivu webové aplikace.

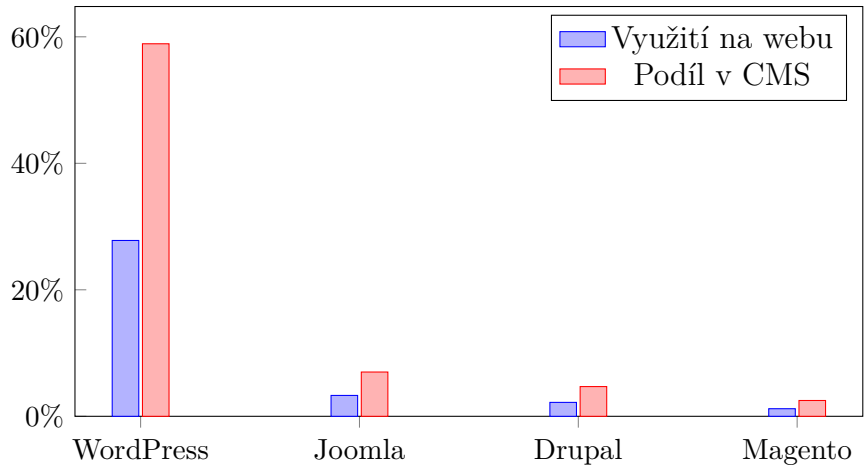
Implementovaný plugin bude otestovaný pro ověření správné funkcionality. V rámci tohoto testování budou prováděny systémové testy a bude zjišťována podpora pluginu mezi nejpoužívanějšími webovými prohlížeči. Z výsledků získaných tímto testováním budou zhodnoceny dosažené výsledky.

2 WordPress (WP)

V této kapitole bude teoreticky popsán systém pro správu obsahu WordPress (dále také jako WP). Budou popsány jeho jednotlivé komponenty se zaměřením na možnosti rozšiřování funkcionality WordPressu pomocí pluginů.

WordPress je open source systém pro správu obsahu (označovaný zkratkou CMS z anglického Content Management System). Vznikl v roce 2003 jako rozšíření blogovacího systému b2/cafelog. Původně pouze blogovací systém se s postupem času rozvinul v plnohodnotný, vysoce přizpůsobitelný redakční systém. V současné době jde podle [1] o nejrozšířenější systém pro správu obsahu s 58,9% podílem v oblasti všech dostupných CMS na trhu. Použitá statistická data zahrnují prvních deset milionů nejnavštěvovanějších internetových stránek. Systém pro správu obsahu WordPress tvoří přes čtvrtinu těchto internetových stránek, konkrétně ke dni 25. 4. 2017 šlo o 27,8% a je výrazně více používaný oproti ostatním CMS, viz graf 2.1.

Stejně jako původní systém b2/cafelog, WordPress je napsaný v jazyce PHP s využitím relačního systému řízení báze dat MySQL [2]. PHP je skriptovací jazyk, kterým je na straně serveru vytvářen dynamický výstup, který je interpretován a výsledek posílán klientovi [3]. Současný model systému WordPress sestává ze třech hlavních komponent: jádra systému, motivů a pluginů.



Graf 2.1: Porovnání využití systému WordPress s dalšími CMS [1].

2.1 Jádro systému WordPress

Jádro je tvořeno sadou souborů, které jsou nutné pro běh celého systému. Obsahují řadu funkcí, které lze rozdělit do několika hlavních skupin:

- **Příspěvky, uživatelský obsah** - Vytváření, ukládání, získávání a interakce s převážnou většinou uživatelského obsahu v systému.
- **Metadata** - Data doplňující obsah, kategorie a uživatelem vytvořené taxonomie.
- **Motivy** - Funkce podporující motivy.
- **Akce, Filtry a pluginy** - Framework pro obsluhu rozšíření systému.
- **Uživatelé** - Vytváření a správa uživatelů a řízení přístupu.

V nejnovější verzi systému WordPress (verze 4.7) je pro správnou funkčnost vyžadován hostitelský server, který podporuje PHP verze 7 a MySQL ve verzi 5.6 a vyšší.[1] Součástí jádra je administrace systému, ve které lze spravovat příspěvky, uživatele a další nastavení. Ačkoliv je WordPress open-source systémem a jeho soubory je tak možné libovolně zobrazovat i upravovat, je doporučováno nezasahovat do funkcí jádra pro úpravu funkcionality systému. Pro rozšiřování funkcionalit systému existuje řada API poskytovaných systémem WP [2].

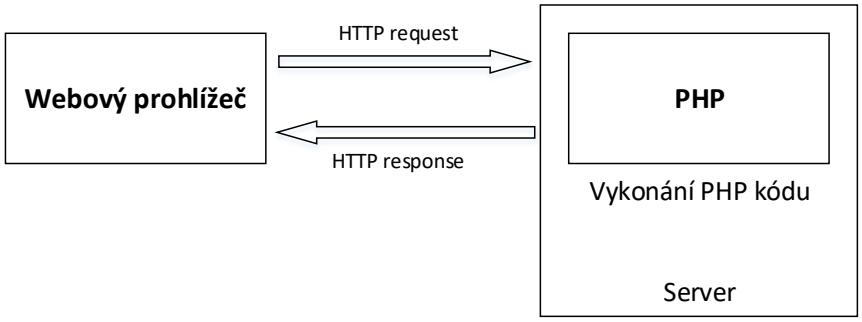
2.1.1 WordPress API

WordPress API (z anglického Application programming interface) je sada předdefinovaných funkcí, které lze využívat při vývoji šablon a pluginů. Umožňují přidávání kódu či získávání externího obsahu WordPressu bez zásahu do jádra systému. Jde o primární metodu rozšiřování funkcionality WordPressu. Jádro WP obsahuje řadu API pro využití ve vývoji pluginů. Jsou rozděleny podle užití, následuje výčet některých z nich [2]:

Plugin API - Využívané pro vývoj vlastních pluginů. Obsahuje prvky pro interakci pluginu se systémem WordPress. Na toto API a jeho funkčnost je podrobněji zaměřena sekce Pluginy v této kapitole.

Widgets API - Umožňuje vytvářet a spravovat widgety. Widgety slouží ke zobrazování vlastních informací pluginu. Mohou být zobrazeny na jakémkoliv postranním panelu, který je definován v šabloně.

[1] https://wordpress.org/about/requirements/



Obrázek 2.1: Princip HTTP requestu [4].

Shortcode API - Shortcode, neboli zkratky, jsou makra přidaná do obsahu WP. Plugin může zkratku zpracovat a v daném místě vykonat přiřazenou funkci. Zkratky umožňují přijímat parametry a tím upravovat výsledný výstup. Kromě implicitních zkratek (například shortcode [gallery] pro zobrazení galerie v obsahu) lze vytvářet vlastní zkratky.

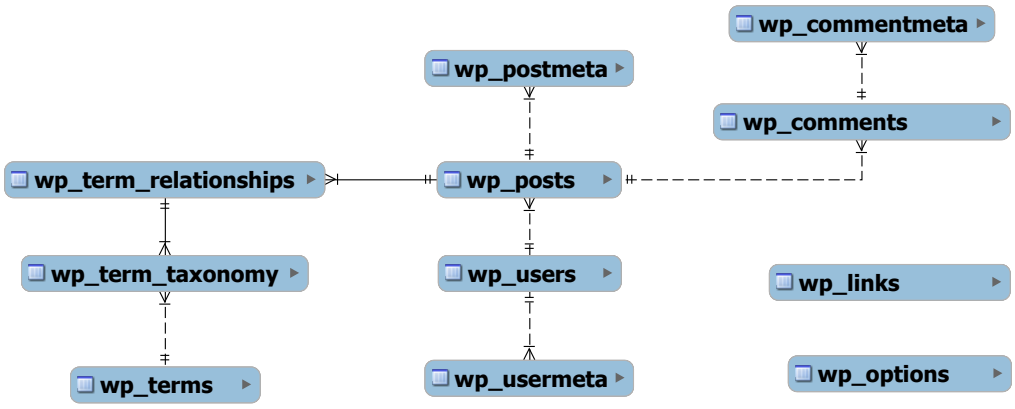
Settings API - Umožňuje vytvářet a spravovat stránky nastavení pro šablony a pluginy v administračním rozhraní WP. Výhodou tohoto API oproti vlastnímu řešení nastavení pluginu či šablony je existence bezpečnostních prvků, Settings API automaticky ošetřuje veškerá data nastavení, která uživatel ukládá.

REST API - Používané pro odesílání HTTP requestů (obrázek 2.1) z WordPressu. Jde o standardizovanou metodu získávání obsahu externí URL. API vezme poskytnutou URL adresu a testuje jí sérií PHP metod pro odesílání HTTP requestů. V závislosti na hostovacím serveru, na kterém systém běží, použije první metodu, kterou považuje za správně nastavenou a provede požadovaný HTTP request.

Options API - Používané pro ukládání dat nastavení do databáze. API poskytuje základní funkce pro jednoduché vytváření, získávání, upravování a odstraňování dat nastavení.

2.1.2 Databáze WP

WordPress používá k ukládání informací systém řízení báze dat MySQL. Do tabulek databáze jsou ukládána veškerá data obsahu (tabulky `wp_posts`, `wp_comments`) a uživatelů (tabulka `wp_users`). Dále jsou v databázi tabulky



Obrázek 2.2: Zjednodušené schéma databáze WP.

pro ukládání nastavení (`wp_options`) a další data doplňující informace o obsahu (tabulky `wp_postmeta`, `wp_terms` a `wp_term_relationships`). V základu tvoří databázi WordPressu 11 tabulek, které jsou vyobrazeny včetně vazeb mezi nimi ve zjednodušeném schématu na obrázku 2.2. Kompletní schéma databáze s atributy těchto tabulek je v příloze A. Struktura databáze je v porovnání s rozsahem systému WordPress minimalistická a je zaměřena na vysokou přizpůsobivost se snahou zajištění zpětné kompatibility se staršími verzemi systému. Změny v databázi jsou tvořeny tak, aby ovlivňovaly aktivní pluginy a šablony co nejméně [2]. Veškeré záznamy změn v databázi jsou dostupné v online manuálu.[2]

wp__comments - Obsahuje veškeré komentáře k obsahu ve WP. Jednotlivé komentáře jsou propojeny s články pomocí cizího klíče odkazujícího na ID článku. Metadata všech komentářů jsou ukládána do tabulky `wp_commentmeta`.

wp__posts - Do této tabulky se ukládají veškeré příspěvky, stránky, přílohy příspěvků či revize příspěvků. Přílohy přípěvků jsou v databázi ukládány jako samostatný záznam v tabulce obsahující informace o příloze a id rodičovského objektu, ke kterému je příloha přiřazena. K tabulce `wp_posts` se vztahuje tabulka `wp_postmeta`. Ta obsahuje metadata každého příspěvku, vázaná pomocí cizího klíče s ID příspěvku. V této tabulce lze navíc ukládat ke každému článku vlastní dodatečné informace v podobě klíče a jemu přiřazené hodnoty.

[2]https://codex.wordpress.org/Database_Description#Changelog

wp_term_taxonomy - Tato tabulka utváří taxonomie z termínů uložených v tabulce `wp_terms`. Jednotlivé termíny taxonomií jsou souvisejícím příspěvkům přiřazovány v tabulce `wp_term_relationships`. Veškeré termíny všech taxonomií používaných v systému jsou uchovávané v tabulce `wp_terms`.

wp_options - Ukládá všechna nastavení systému, která lze nastavit v administračním rozhrani WordPressu. Mimo to ukládá také informace o aktivovaných pluginech či šablonách.

2.1.3 Taxonomie WP

Taxonomie WP (dále jen taxonomie) je definována jako způsob seskupování podobných prvků. Tím je v podstatě přidán relační rozměr systémovému obsahu. V případě WordPressu jsou používány kategorie a tagy k seskupování příspěvků. Seskupením těchto příspěvků je definována jejich taxonomie. Jednotlivé prvky taxonomie se ve WP nazývají termíny.

Základní instalace WordPressu nabízí čtyři různé taxonomie. Taxonomie Kategorie umožňuje seskupovat přidané příspěvky řazením do různých kategorií, definovaných při vytváření příspěvku. Jedná se o hierarchickou taxonomii. To znamená, že kategorie může být rodičem jiné kategorie. Kategorie jsou využívány převážně ke strukturální organizaci obsahu. Druhou základní taxonomií je taxonomie s názvem Tagy. Jde o taxonomii podobnou kategoriím, nejedná se však o taxonomii hierarchickou. Primárním využitím tagů je určení klíčových slov příspěvku. Stejně jako u kategorií se tagy definují při vytváření příspěvku. Příspěvek může být přiřazen k více kategoriím a příspěvku může být přiřazeno více tagů. Třetí základní taxonomií je taxonomie odkazů. Díky této taxonomii lze seskupovat dohromady podobné odkazy. Poslední taxonomií, která je obsažena v základní instalaci WordPressu, jsou formáty příspěvků. Jde o meta informaci příspěvku, která může být využita šablonami pro uzpůsobení prezentace obsahu příspěvku. Tato taxonomie na rozdíl od předchozích základních taxonomií, které umožňují přidávání nových termínů, obsahuje pevně dané termíny, které nelze nijak upravovat [2, 5].

Kromě základních taxonomií nabízí WordPress vytváření a definování vlastních taxonomií. Při registrování nové vlastní taxonomie jsou vždy vyžadovány dva parametry specifikující každou taxonomii ve WordPressu. Prvním z nich je název taxonomie, pod kterým je ukládána do databáze a pod kterým se na taxonomii odkazuje v kódu. Druhým je typ objektů, ke kterým se taxonomie vztahuje. Dále je možné určit řadu dalších parametrů vytvářené

taxonomie. Například lze určit zda-li se jedná o hierarchickou taxonomii či nehierarchickou, zda lze na vytvořenou taxonomii odkazovat v menu administračního rozhraní, možnosti editace taxonomie, případně určit některé další parametry, které vytvářenou taxonomii popisují podrobněji [2, 6].

2.2 Motivy

Motivy, nazývané také jako šablony, umožňují úpravu zobrazování informací koncovému uživateli bez zásahu do hlavních funkcí systému. Konkrétně jde o PHP soubory sestávající z HTML kódu a funkcí WordPressu pro výpis obsahu koncovému uživateli. Šablony mohou být doplněny také o CSS a JavaScript soubory. Volitelnou možností u šablon je využití souboru funkcí. Jde o soubor pojmenovaný `functions.php` uložený v adresáři šablony. Takový soubor se tváří jako plugin, a pokud je v adresáři šablony přítomný, je automaticky načten během inicializace systému WordPress. Lze ho použít například pro zařazení kaskádových stylů či JavaScriptu do šablony nebo pro funkce, které se vyskytují na více místech v šabloně [7].

Součástí každého motivu je také takzvaná smyčka WP. Ta se vztahuje k tomu, jak WordPress rozhoduje, který obsah zobrazí na právě zobrazované stránce systému. Základní smyčka kontroluje, zda pro danou zobrazenou stránku existuje v systému obsah, který může zobrazit, a který splňuje globální nastavení systému (kolik je zobrazováno příspěvků na stránku a podobné nastavení). Pokud takový obsah existuje, je na stránce zobrazován [2, 8]. Rozhodování probíhá na základě předané adresy. WordPress prohledává hierarchickou strukturu motivu dokud nenalezne odpovídající soubor šablony [9]. Funkce smyčky spadají do kategorie označované pojmem Šablonová značka či Template tag.

Template tagy jsou používané pro dynamické zobrazování informací či jiné úpravy obsahu. Některé značky lze použít pouze uvnitř smyčky šablony, většinu ostatních je však možné používat kdekoliv v šabloně, případně i v pluginu. Ve WordPressu je dostupné velké množství značek, které rozlišujeme na dva základní druhy [8]:

- Include tagy
- Podmiňovací tagy

Include tagy jsou funkce používané pro načítání souborů šablony, které mohou vracet obsah načteného souboru do šablony. Takovým tagem je například `get_header`, který vrátí obsah souboru představující hlavičku šablony.

Podmiňovací tagy, v originálním znění Conditional tags, jsou funkce používané v šablonách či pluginech pro pozměnění zobrazeného obsahu na základě podmínek, které zobrazovaná stránka splňuje. Příkladem takové funkce může být například podmiňovací značka `is_page(ID)`, která zjišťuje, zda je právě zobrazován příspěvek s daným ID. Po zakomponování do podmínky tak lze utvářet obsah specifický zvoleným příspěvkům.

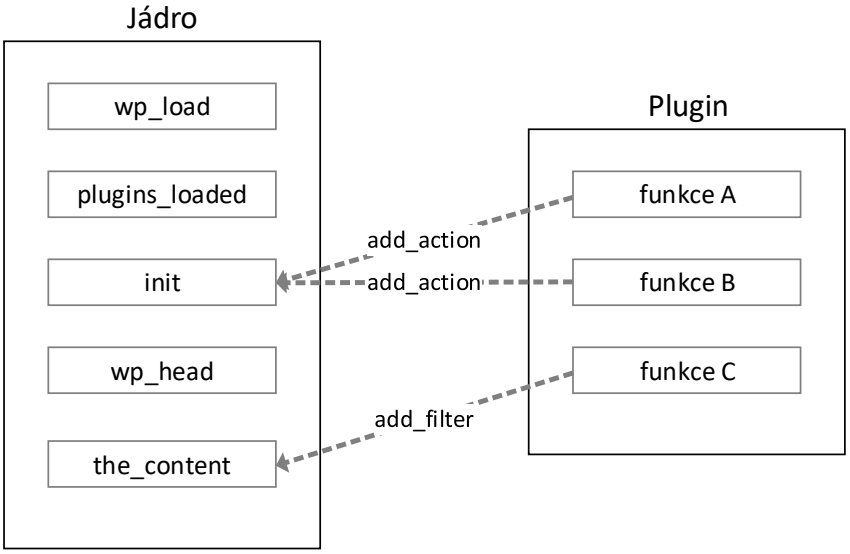
2.3 Pluginy

Jde o kolekce kódu, které rozšiřují základní funkce redakčního systému WordPress. Pluginy ve WordPressu jsou psané s využitím WordPress Plugin API, které je založené na událostmi řízené architektuře [10]. Událostmi řízená architektura, z anglického názvu event-driven architecture (EDA), je architektonický styl ve kterém je jedna či více komponent systému vykonána v reakci na obdržení jedné či více notifikací o provedení události [11]. Ve WP se událost označuje anglickým termínem Hook.

2.3.1 Hook

Hook je specifická událost jádra WP, ke které lze připojit vlastní funkce [10]. Jde o primární metodou interakce pluginu s obsahem WordPressu. Použitím hooků lze spouštět funkce ve specifický čas v průběhu vykonávání základních funkcí systému. Tím lze upravit funkcionalitu systému s docílením požadovaných výstupů [2]. K jednomu hooku lze přiřadit více funkcí, pro každou funkci je však nutné zaregistrovat ji k danému hooku zvlášť. Znázornění hookování funkcí pluginu k událostem jádra je na obrázku 2.3, kde funkce pluginu A a B jsou volány při události jádra nazvané `init` a funkce C v pluginu je volána při události `the_content`.

Hooky se rozdělují na dva druhy: akce (action) a filtry (filters). Rozdílem mezi akcemi a filtry je, jak daný hook WordPress zpracovává. Při přiřazení vlastní funkce k filtru musí tato funkce přijímat vstupní parametr a vracet nějaký výstup, jelikož je určena k modifikování či filtrování výstupu, který je následně posílán do databáze či zobrazován koncovému uživateli na obrazovku. Funkcím, které jsou přiřazené k akcím, nejsou předávány žádné vstupní parametry a jakýkoliv výstup této funkce je jádrem ignorován [10].



Obrázek 2.3: Příklad hookování funkcí pluginu.

Funkce `add_action` pro přiřazení vlastní funkce pluginu k akci jádra, stejně jako funkce `add_filter` pro přiřazení k filtru, obsahuje dle kódu 2.1 čtyři parametry. První parametr je název akce nebo filtru, ke kterému je funkce přiřazena. Druhý parametr je název funkce, která je v reakci na daný hook volána. Tyto dva parametry jsou povinné. Dále lze třetím, volitelným parametrem, specifikovat prioritu dané akce či filtru. Pokud je k jednomu hooku přiděleno více funkcí, systém určí pořadí vykonání funkcí právě podle priorit, které jsou k jednotlivým funkcím přiděleny. Posledním, taktéž volitelným parametrem, je počet argumentů, kterých může volaná funkce nejvíce přijmout [2].

```
add_action('hook_name', 'function_name', [priority], [
    accepted_args]);
```

Kód 2.1: Parametry funkce add_action.

V současné době WordPress obsahuje okolo dvou tisíc různých hooků, které lze při vývoji pluginů a šablon využívat.[3] Kromě těch však WordPress nabízí i možnost vytváření vlastních hooků. Ty umožňují ostatním vývojářům plugin jednoduše upravovat či rozšiřovat. Stejně jako standardní hooky se dají dělit na akce a filtry. Jsou také vytvářeny a volány stejně jako hooky jádra WP. Jelikož jakýkoliv plugin může vytvářet vlastní hooky, je třeba volit unikátní jména pro zamezení jmenných konfliktů. Je-li totiž v systému

[3]http://adambrown.info/p/wp_hooks/hook

více hooků se stejným jménem, mohou vznikat chyby, jejichž zdroj je těžko dohledatelný [12].

2.3.2 Struktura pluginu

Plugin je tvořený ze souborů obsahujících PHP funkce, které mohou případně být napojeny na hooky jádra či jiných pluginů. Dále může být doplněný o soubory kaskádových stylů, JavaScriptu, obrázků či jazykové soubory, které slouží pro překlad konečného výstupu do různých jazyků. Každý plugin musí ve své adresářové struktuře obsahovat alespoň jeden PHP soubor. Jméno tohoto souboru by mělo být odvozeno z názvu pluginu aby byl uživatel WordPressu schopný poznat, jaký plugin kopíruje do adresáře pluginů v systému. Jelikož jsou všechny pluginy ukládány do stejného adresáře, název souboru pluginu by měl být unikátní, aby nedocházelo ke konfliktům s jinými pluginy. Takový soubor je považovaný za hlavní soubor pluginu. Plugin může být tvořený jediným PHP souborem obsahující všechny funkce. Druhým případem je plugin tvořený více soubory a adresáři, které jsou všechny uložené v jednom rodičovském adresáři. V takovém případě pro tento rodičovský adresář platí stejná pravidla pro název adresáře jako pro název souboru v případě pluginu tvořeného jediným souborem [2, 10].

Každé rozšíření systému WordPress musí obsahovat validní hlavičku pluginu. Hlavička ve formě komentáře se musí nacházet na začátku PHP souboru. Není třeba, aby byla hlavička uvedena v každém souboru rozšíření. Stačí, pokud bude uvedena v hlavním PHP souboru pluginu, přes který se v systému inicializuje. Tím se zajistí, že se WordPress k adresářové struktuře, ve které je soubor s hlavičkou uložený, bude chovat jako k pluginu. V případě, že je plugin tvořen jediným souborem, bude se chovat k tomuto souboru jako k pluginu. Možný způsob zápisu hlavičky pluginu je znázorněn v kódu 2.2.

```
<?php
/*
Plugin Name: WordPress Plugin
Description: Zde je krátký popis pluginu
Version: 1.1
Author: Jan Novák
Author URI: http://www.novak.cz
License: GPL
*/
?>
```

Kód 2.2: Hlavička pluginu.

Jediným povinným prvkem hlavičky je parametr Plugin Name, který obsahuje název rozšíření. Ačkoliv ostatní parametry nejsou povinné, vyplnění alespoň základních z nich, jako je popis pluginu, verze, či autor pluginu, je doporučované zejména z důvodu odlišení od ostatních pluginů v sekci pro správu rozšíření v systému. Právě v administrátorské části systému v sekci pro správu rozšíření lze jednotlivé pluginy přidávat či odebírat ze systému, u některých lze případně upravovat jejich nastavení [2].

Jakákoliv souborová struktura splňující tyto základní požadavky pluginu, která je umístěna do systému WordPress v adresáři `/wp-content/plugins/` bude zobrazena v seznamu pluginů v administračním rozhraní WordPressu. Samotné uložení do zmíněného adresáře ale neznamená, že je plugin zapojený do systému. Každý plugin musí být pro inicializování do systému aktivován v seznamu pluginů [10].

2.3.3 Zabezpečení pluginu

Jedním z hlavních kroků při vytváření pluginů je zajištění bezpečnosti proti nežádoucím útokům a zneužívání systému skrze plugin. Pokud plugin takovou bezpečnostní díru obsahuje, může se pomocí ní stát zranitelný celý systém. WordPress v základu nabízí některé zabudované bezpečnostní prvky, které je doporučené využívat [2].

Takovým bezpečnostním prvkem je například kontrola uživatelských pravomocí pro provedení požadované akce pluginu. WordPress využívá koncept rolí uživatelů, kde má každý uživatel v systému přiřazenu některou z rolí zavedených v systému. Každá role k sobě má přiřazen soubor způsobilostí. Ty určují, které akce může uživatel s danou rolí v systému provádět. Způsobilostí může být například možnost publikování příspěvků, editace uživatelů, přidávání pluginů a podobně. Výchozí instalace WordPressu nabízí několik základních předdefinovaných rolí:

- Administrator - má přístup k veškerým administračním funkcím
- Editor - může spravovat a publikovat veškeré příspěvky
- Author - může spravovat a publikovat pouze vlastní příspěvky
- Contributor - může psát a spravovat pouze vlastní příspěvky, bez možnosti publikace
- Subscriber - může spravovat pouze svůj vlastní účet

Kromě těchto rolí nabízí i možnost vytvoření vlastních rolí s vybranými způsobilostmi. Z bezpečnostního hlediska pak obsahuje také několik funkcí

pro zjišťování, zda má určitý uživatel jistou roli či způsobilost. Veškeré tyto funkce vracejí boolean hodnotu toho, zda má uživatel požadovanou roli či způsobilost. Role a způsobilosti se těmto funkcím předávají jako parametry a lze jim předat výchozí i vlastně vytvořené role a způsobilosti [6, 13].

Dalším bezpečnostním prvkem pluginu jsou takzvané WP Nonce. Označení Nonce je zkratka z anglického "number used once", volně přeloženo jako číslo použité jednou. Používání Noncí je kritickým prvkem ochrany WP a pluginů proti CSRF útokům [2, 6]. CSRF, celým názvem Cross Site Request Forgery, někdy označované také jako XSRF, je druh útoku na webovou aplikaci, při kterém je prohlížeč oběti donucen přesměrovat pomocí HTML a JavaScriptu na nebezpečnou URL, která reprezentuje nechtěnou transakci. Tato URL je vkládána jako odkaz na obrázek či jiný HTML tag, který se tváří jako součást aplikace a který automaticky danou URL načítá [14].

Nonce jsou ve WP používány u odesílání požadavků (ukládání nastavení, odesílání formulářů, Ajaxové požadavky) pro ochranu před neautorizovaným přístupem pomocí generování tajného klíče. Tento klíč je generovaný ještě před odesláním formuláře a je přidán do formuláře jako skryté pole. Při odeslání formuláře je spolu s daty předán i klíč. Na straně serveru je poté jako první kontrolována jeho správnost. Pokud se nonce neshodují, WP přestane požadavek zpracovávat a zobrazí chybovou zprávu [2].

V neposlední řadě patří mezi formy zabezpečení pluginu kontrola veškerých dat, která jsou získávána od koncového uživatele a ukládána do databáze. Nesprávně kontrolovaná vstupní data mohou vést k SQL Injection útokům a dalším neočekávaným chybám [2, 6]. Zranitelnost vůči SQL Injection útokům vychází z používání uživatelských vstupů pro vytváření SQL dotazů zpracovávaných aplikací. Tím je útočníkovi umožněn neautorizovaný přístup k datům v databázi a jejich poškození. Tomu lze zabránit jejich vhodnou validací a sanitizací [15].

Validace je proces ověření, že data obdržená od koncového uživatele jsou ve formátu, ve kterém jsou očekávána. Validace je prováděna před ukládáním daných dat do databáze [6].

Sanitizace je proces ošetření vstupních dat od uživatele před jejich uložením do databáze. Mezi způsoby ošetření dat lze zařadit kontrolu správného kódování (ve WP implicitně nastaveno na UTF-8), odstranění HTML tagů či odebrání nevhodných znaků například z emailové adresy [6].

Samotný jazyk PHP obsahuje funkce pro validaci a sanitizaci dat, WordPress pak pro tyto účely obsahuje několik dalších pomocných funkcí [6].

2.3.4 Best practices

Při vyvíjení pluginů do WordPressu je vhodné dodržovat určité doporučované praktiky (neboli best practices), které zajišťují lepší organizaci kódu a podporují jeho funkčnost v rámci jádra WP a ostatních pluginů aktivovaných v systému [16].

Ve WordPressu jsou veškeré proměnné, funkce a třídy implicitně definované v globálním jmenném prostoru (namespace). To znamená, že je možné jedním pluginem přepisovat a přistupovat k proměnným, funkcím či třídám, které jsou vytvořené jiným pluginem (proměnné definované uvnitř funkcí nebo tříd tímto nejsou ovlivněny). Díky tomu může uvnitř WP docházet ke jmenným kolizím. Jmenné kolize nastávají, pokud více pluginů používá stejné jméno pro proměnnou, funkci či třídu. Pro zamezení tohoto problému lze použít metody, které jsou popsány níže.

- Přidávání předpony v podobě unikátního identifikátoru ke každé proměnné, funkci nebo třídě tvořené pluginem. Tím je zamezeno možnému přepisování vlastních proměnných a nechtěnému volání vlastních funkcí a tříd jiným pluginem, aktivovaným v systému.

- Kontrola, zda-li je již daná proměnná, funkce nebo třída implementována ve WP. Pro tyto účely nabízí jazyk PHP několik funkcí. Existenci proměnné lze kontrolovat funkcí `isset()`, existenci funkcí a tříd pak pomocí PHP funkcí `function_exists()` a `class_exists()`.

Snadnějším řešením jmenných kolizí je pak použití objektově orientovaného programování a tříd. Stále je nutné zajistit, aby název třídy nekolidoval s jinou třídou, o zbytek však bude postaráno samotným PHP.

Mezi další doporučované praktiky pro vývoj pluginů patří správná organizace kódu a souborů pluginu. Kořenový adresář pluginu by měl obsahovat pouze hlavní soubor pluginu, popřípadě soubor s funkcemi pro odstranění pluginu a jeho dat v souboru uninstall.php. Veškeré další soubory by měly být organizovány do podsložek kdykoliv je to možné. To platí například pro soubory JavaScriptu, kaskádových stylů či jiné soubory s funkcemi a třídami pluginu [16].

3 Další použité technologie

V této kapitole jsou stručně popsány další technologie, které byly spolu s WP použity pro vývoj tohoto pluginu. Jsou zde stručně vysvětleny technologie HTML, CSS a JavaScript, které tvoří základ pro vytváření webových aplikací. Dále je vysvětleno aplikační rozhraní Google Maps API, jehož funkce jsou v pluginu použity pro interaktivní mapu.

3.1 HTML, CSS, JavaScript

Hypertext Markup Language (HTML) je značkovací jazyk používaný pro vytváření webových aplikací. HTML dokumenty sestávají ze stromové struktury elemetů a textu. Elementy jsou označované otevíracím tagem (například `<body>`) a případně ukončovacím tagem. Elementy mohou obsahovat atributy, které upravují jejich chování.

Kaskádové styly CSS jsou používané pro úpravu interpretace HTML dokumentů. S příchodem standardu CSS3 nyní CSS nabízí úroveň dynamické interaktivity dříve podporované pouze JavaScriptem. U jakýchkoliv HTML elementů tak lze kromě rozměrů, barev či rozmístění přidávat například animované přechody nebo transformace.

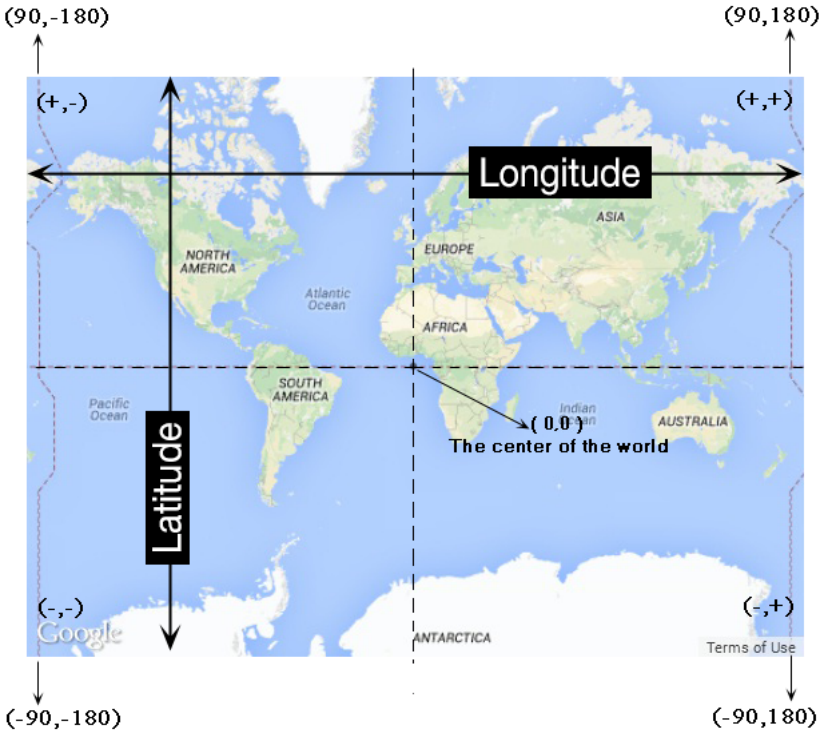
JavaScript je skriptovací jazyk používaný v HTML, jehož chod je zajištěný uvnitř webového prohlížeče na straně klienta. Spolu s CSS umožňuje vytvářet dynamický obsah webových stránek bez nutnosti získávání celé nové stránky ze serveru. JavaScript je čím dál více používaný pro Ajax (zkratka pro Asynchronous JavaScript and XML). Jde o sadu metod vestavěných do JavaScriptu, které umožňují přenos dat mezi webovým prohlížečem a serverem v pozadí. Příkladem používání této technologie jsou například Google mapy, kde jsou nové sekce mapy získávány ze serveru bez nutnosti obnovení stránky. Použitím Ajaxu je možné výrazně snížit množství přenášených dat. Zároveň byla použita knihovna jQuery, která přidává k JavaScriptu lepší kompatibilitu mezi prohlížeči, snadnější manipulaci s HTML elementy či funkce pro jednodušší používájí Ajaxu [3, 17].

3.2 Google Maps API

Aplikační rozhraní Google Maps API poskytuje využití mapových služeb od společnosti Google ve vlastních webových aplikacích pro efektivní zobrazo-

vání vlastních mapových dat. Existuje více druhů API pro různé platformy. Podle platformy se Google Maps API rozdělují na Android, iOS, Web API a Web Service API.[4] V této práci je popisováno a používáno rozhraní Google Maps JavaScript API pro webovou platformu.

Google Maps JavaScript API se skládá z JavaScript souborů obsahujících třídy a funkce, kterými lze nastavovat chování mapy. Základní chování mapy je zajištěno pomocí HTML, CSS a JavaScriptu. Mapové podklady jsou zobrazovány jako dlaždice obrázků načtené v pozadí pomocí Ajaxových požadavků a poté vkládány do HTML elementu div umístěného na stránce. Každým pohybem na mapě API posílá informace o nové pozici a úrovni přiblížení. Na základě těchto informací jsou získány aktualizované mapové podklady [18]. Pro vyjádření lokace ve světě jsou používány souřadnice. Mapy Google i samotné API používají souřadnicový systém World Geodetic System 84 (WGS 84). Stejný souřadnicový systém je používaný například u GPS. Souřadnice jsou vyjadřovány pomocí zeměpisné šířky (latitude) a zeměpisné délky (longitude), znázornění os souřadnic je na obrázku 3.1. Zapisují se ve formě dvojice desetinných čísel, pro oddělení celé části čísla a desetinné části se používá tečka [18].



Obrázek 3.1: Znázornění zeměpisné šířky a délky [19].

[4]https://developers.google.com/maps/get-started/

Pro načtení API na webové stránce je použit script tag s URL, která představuje umístění souboru, který načítá veškeré symboly a definice potřebné pro jeho používání. Tato URL obsahuje také parametr callback, který specifikuje JavaScript funkci volanou po načtení API. Dalšími parametry lze nastavit například jazykovou lokalizaci mapy. Mapa je vložena do identifikovatelného elementu div vytvořením nové instance třídy Map [20].

4 Vývoj pluginu

Do této kapitoly je zahrnuta praktická část práce. Nejprve jsou v první sekci analyzovány potřeby Czech-American TV týkající se vytvoření pluginu pro existující CMS WordPress. V další sekci je navrhnuto, jakým způsobem se budou analyzované potřeby řešit ve vytvářeném pluginu. Poté následuje popis implementace navrhnutých řešení. Poslední částí této kapitoly je testování výsledného pluginu, podle kterého jsou zhodnoceny dosažené výsledky.

4.1 Analýza potřeb

4.1.1 Aktuální stav

Systémem, na který bude vyvíjený plugin napojován, je webová aplikace organizace Czech-American TV, která převážně anglicky mluvícím návštěvníkům nabízí informativní články a videa o české kultuře. Aplikace je dostupná na adrese www.catvusa.com, lokalizovaná je kompletně do anglického jazyka. Je provozována na systému WordPress, který je průběžně aktualizován na nejnovější dostupnou verzi. Uživatelské rozhraní je modifikováno na míru vytvořenou WordPress šablonou, která je postavena na front-end CSS frameworku Bootstrap 3.

Obsahem, na který se bude vytvářený plugin napojovat, jsou příspěvky vlastního typu Fact (Fakta). V těchto příspěvcích jsou popisována zajímavá místa v České republice, mezi které patří například hrady, zámky, muzea či přírodní rezervace. Každý takový příspěvek je vytvářen v administračním rozhraní WordPressu WYSIWYG textovým editorem. Na webovém potálu Czech-American TV je takový příspěvek vždy prezentován na samostatné stránce, na které je zobrazen celý jeho obsah. Příspěvek vždy obsahuje název a obsah v podobě textu. Dalším prvkem každého příspěvku je náhledový obrázek. Takový obrázek není součástí obsahu příspěvků, je brán jako jeho příloha.

Dále je ke každému příspěvku přiřazen některý z krajů České republiky jako termín explicitní taxonomie. Tato taxonomie je již vytvořena a obsahuje všech 14 krajů České republiky. U této taxonomie není předpokládána žádná strukturální změna. Aktuálně jsou v systému dokončené příspěvky pouze z Libereckého, Jihomoravského a Moravskoslezského kraje. Příspěvky ze zbylých regionů České republiky budou doplněny až s implementovaným pluginem a jejich příprava není součástí této práce. Na každý kraj je před-

pokládáno přibližně 20 příspěvků typu Fact. V konečném součtu je tak očekáváno přibližně 300 těchto příspěvků.

V aktuálním systému WP je kromě základních uživatelských rolí vytvořeno několik nových rolí. Zejména pak role Public Relations & Media Team. Uživatelé s touto rolí nebo se základní rolí Editor vytvářejí a spravují příspěvky typu Fact.

4.1.2 Požadované funkce

Systémové požadavky jsou dle [21] rozdělené na funkční a mimofunkční požadavky.
Funkční požadavky:

- Interaktivní mapa, která bude pomocí bodů vizualizovat příspěvky typu Fact.
- Kategorizování příspěvků typu Fact podle klíčových slov.
- Filtrování bodů na mapě pomocí regionů a klíčových slov.
- Administrace dat, zakomponovaná do stávajícího administračního rozhraní.

Mimofunkční požadavky:

- K administraci dat bude mít přístup každý uživatel, který může v systému upravovat příspěvky.
- Veškeré výstupy pluginu budou v anglickém jazyce a budou vzhledově odpovídat stávající šabloně.

Hlavní funkcí pluginu by měla být interaktivní mapa, která je napojena na stávající obsah příspěvků typu Fact. Mapa je na webu umístěna na nově vytvořené samostatné stránce. Odkaz na tuto stránku je umístěný v hlavní navigační liště v podnabídce položky Facts. Na mapě budou zobrazeny body, které znázorňují lokaci míst v ČR, které jsou v příslušném příspěvku zmíněny. Po kliknutí na bod umístěný na mapě se zobrazí souhrnné informace, obsahující název příspěvku a náhledový obrázek, který je v příspěvku obsažen. Název i náhled obrázku budou mít funkci odkazu, který uživatele v novém okně přesměruje na stránku s obsahem příspěvku, se kterým je lokace spjata.

K současné struktuře příspěvků typu Fact je přidána možnost jejich označování pomocí klíčových slov. Jedná se o vazbu typu M:N. To znamená, že

k jednomu příspěvku je možné přiřadit více klíčových slov a klíčové slovo může být přiřazeno u více příspěvků. Klíčová slova bude možné spravovat. Pověřený uživatel bude moci vytvářet nová, mazat existující či upravovat stávající klíčová slova.

Body na mapě bude uživatel moci filtrovat podle krajů České republiky a podle klíčových slov. Oba filtry bude možné kombinovat mezi sebou. To znamená, že si uživatel bude moci na mapě zobrazit například pouze muzea (filtrování podle klíčových slov) v Jihočeském kraji (filtrování podle krajů).

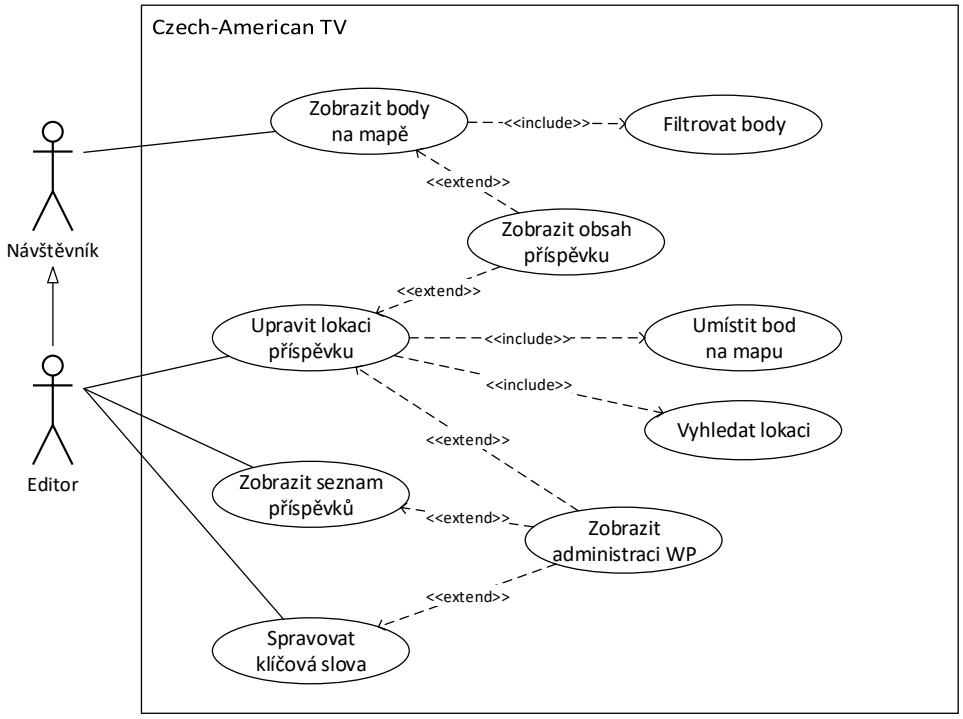
Uživatel přihlášený do WP s editorskými právy bude moci zobrazit editační stránku, která je součástí administrace WP. Tato stránka bude vypisovat seznam příspěvků, které lze na mapě zobrazit. Jednotlivé prvky v seznamu odkazují na editační formulář, kterým je možné upravit souřadnice bodu, kterým je příspěvek vizualizován na mapě. Souřadnice bude možné určit i kliknutím na mapu, umístěné u formuláře.

Veškerý koncový výstup pluginu musí vizuálně zapadat do současného celkového vzhledu webové aplikace tvořeného používanou šablonou. Veškeré vizuální výstupy pluginu musejí být responzivní vůči zobrazovacím zařízením a jejich rozlišením (PC, tablet, chytrý mobilní telefon). Textové výstupy pluginu by měly být v anglickém jazyce. Žádné další mimofunkční požadavky (například výkonnostní) nebyly specifikovány.

4.1.3 Případy užití (Usecase)

Usecase diagram na obrázku 4.1 popisuje rozdělení uživatelů pluginu a jejich možné interakce s pluginem. Diagram je tvořený dle technik UML převzatých z [22].

Uživatele pluginu dělíme na dva typy. Prvním typem je aktér zvaný Návštěvník. Jedná se o uživatele webových stránek Czech-American TV, který není přihlášený do WP pomocí uživatelského účtu, nebo je přihlášený na účtu, na kterém nemá příslušné editační role. Může zobrazit stránku s mapou, na které se zobrazují body znázorňující jednotlivé příspěvky. U mapy může nastavovat filtry a zobrazit stránku s obsahem příspěvku kliknutím na informační okno na mapě. Druhým aktérem je Editor. Jedná se o uživatele, který je přihlášený do systému WordPress s rolí editor či vyšší. Může provádět stejné akce jako běžný uživatel. Navíc může zobrazit administrační rozhraní WP s administrací dat pluginu, kde může pomocí formuláře upravovat umístění jednotlivých bodů na mapě. Při upravování lokace příspěvku může použít mapu pro umístění bodu či nastavení umístění dle zadané adresy. Dále může v administračním rozhraní WP spravovat klíčová slova.



Obrázek 4.1: Usecase diagram.

4.2 Návrh pluginu

4.2.1 Mapa a vizualizace dat

Pro vizualizaci potřebných dat na mapě a zajištění potřebných mapových podkladů je zvoleno aplikační programové rozhraní Google Maps JavaScript API popsané v kapitole 3. Ačkoliv existuje řada alternativních aplikačních rozhraní pro vizualizaci mapových dat, jmenovitě OpenLayers[5], Leaflet[6] či Here Maps[7], rozhraní od společnosti Google je zvoleno na základě dobré podpory a častých aktualizací v podkladových mapách i API samotném [23]. Dalšími důvody pro zvolení tohoto rozhraní je přehledná online dokumentace s praktickými ukázkami využívání API a fakt, že toto API je již používáno v jiné části webu Czech-American TV a tím bude zachováno použití jednotných technologií napříč webovou aplikací.

Pro umístění mapy je třeba vytvořit novou stránku, která bude obsahovat div element, ve kterém bude mapa inicializována. Stránka bude vytvořena

[5]https://openlayers.org/
[6]http://leafletjs.com/
[7]https://developer.here.com/

ručně v administračním rozhraní WP. Odkaz na tuto stránku bude umístěný do stávající kontextové nabídky taktéž ručně v administračním rozhraní WordPressu. Obsah a rozvržení této stránky je navržený na obrázku 4.2. Titulek stránky a její textový obsah, ve kterém bude stručně popsané ovládání mapy a filtrů, budou vytvořené pomocí administračního rohraní. Výstupem pluginu budou až možnosti filtrování a interaktivní mapa. Ten bude na stránce zobrazen pomocí funkce přiřazené k unikátní shortcode značce, která bude umístěna v textovém obsahu stránky. Tímto je zajištěna možnost pozdějších stylistických úprav stránky či případné jednoduché zobrazení mapy s filtry na jiné stránce. Zároveň se v této funkci budou linkovat potřebné souboru JavaScriptu a kaskádových stylů a tím dosaženo, že budou linkovány pouze na stránce s pluginem. Vizuální podoba výstupu pluginu bude stylována pomocí front-end frameworku Bootstrap 3 již využívaného pro styl šablony webu. Tento framework spolu s dalšími vytvořenými styly zajistí grafický soulad pluginu se zbytkem aplikace a potřebnou responzivitu pro zobrazení na mobilních zařízeních.

Mapa bude inicializována s implicitně nabízenými silničními mapovými podklady. Lokalizovaná bude do anglického jazyka a během inicializace centrována tak, aby bylo v okně mapy viditelné celé území České republiky. Při načtení stránky budou na mapě vždy zobrazeny body všech zveřejněných příspěvků, které mají určenou zeměpisnou šířku a zeměpisnou délku, nehledě na jim přidělené kategorie a region (vždy však budou muset mít přiřazen alespoň jeden region a jednu kategorii).

Data budou na mapě vizualizována pomocí značek implicitně nabízených aplikačním rozhraním Google Maps API. Potřebné informace o příspěvku (název příspěvku, náhledový obrázek a odkaz na příspěvek) budou ke každé značce ukládány jako parametry této značky během jejího vytváření. Informace budou zobrazovány ve vyskakovacím informačním okně umístěném uvnitř interaktivní mapy. Obsah okna bude generovaný podle parametrů, které byly značce přiřazeny při jejím vytvoření. Součástí obsahu bude název příspěvku, náhledový obrázek a tlačítko.

Všechny tři prvky budou sloužit jako odkaz na stránku příspěvku. Zobrazené bude vždy nejvýše jedno informační okno v jeden okamžik. Pokud uživatel otevře informační okno u značky v době, kdy je již otevřené okno jiné značky, dojde k automatickému zavření stávajícího okna. Tímto nebude docházet k překrývání mapy velkým množstvím otevřených informačních oken.

Obrázek 4.2: Návrh stránky s výstupem pluginu.

4.2.2 Ukládání souřadnic

Jelikož jsou příspěvky na mapě zobrazovány pomocí dvojice desetinných čísel představující zeměpisné souřadnice bodu, je zapotřebí tato data v rámci pluginu nějakým způsobem přiřazovat k jednotlivým příspěvkům a ukládat do databáze. WordPress nabízí několik metod, jak data vázaná k pluginu ukládat.

- ukládání vlastních nastavení WP
- explicitní taxonomie
- vlastní databázové tabulky
- ukládání vlastních metadat k příspěvkům

Jedním ze způsobů je použití ukládání vlastních nastavení poskytované WordPressem. Jde o mechanismus ukládání unikátně pojmenovaných dat do

databáze WP, které mohou být pluginem získávány zpět. Tento způsob je však vhodný pouze pro uchovávání poměrně malého objemu statických dat. Zároveň tato data nelze přiřazovat k jednotlivým příspěvkům. Pro ukládání souřadnic k příspěvkům tedy není vhodné použít tuto metodu.

Dalším možným způsobem je použití vlastních taxonomií WP. Ačkoliv je možné pomocí taxonomií přiřazovat k příspěvkům uživatelem upravovatelná data, tento postup je využitelný především pro klasifikaci příspěvků a přistupování k příspěvkům asociovaných ke stejným termínům taxonomie.

Dále WordPress nabízí také vytváření vlastních tabulek v databázi, do kterých lze data ukládat. Použití této metody je vhodné pro ukládání dat, která nejsou nijak spojena s konkrétními příspěvky, stránkami či komentáři.

Poslední zmíněnou metodou je ukládání ve formě vlastních metadat příspěvků. Každý příspěvek má jemu přiřazená metadata, ukládána do databáze ve formě klíč - hodnota. Jde o metodu, kterou je vhodné použít pro data asociovaná k individuálním příspěvkům, stránkám či přílohám [24]. Z tohoto důvodu je právě tato metoda zvolená pro ukládání souřadnic k příspěvkům. Ke každému příspěvku typu Fact bude ukládána dvojice nových metadat představující zeměpisnou šířku a zeměpisnou délku jeho lokace. Tato metadata bude možné měnit či odebírat skrze vytvořenou administraci dat.

4.2.3 Kategorizace příspěvků

Plugin bude rozdělovat existující příspěvky do kategorií podle klíčových slov. K tomuto účelu bude využita taxonomie WordPressu, sloužící pro kategorizaci obsahu. WordPress implicitně nabízí taxonomii s názvem Tagy, kterou lze použít pro klíčová slova příspěvků. Tato taxonomie je však společná pro všechny příspěvky, nezávisle na jejich typu. Během zobrazování seznamu klíčových slov pro filtrování by tak nebylo snadné rozlišovat, která klíčová slova souvisejí pouze s příspěvky typu Fact a ostatní klíčová slova, použitá pro jiný obsah.

Proto bude při aktivaci pluginu vytvořena nová taxonomie, která bude použitelná pouze pro příspěvky typu Fact. Název této taxonomie bude obsahovat unikátní předponu pro předejití případných jmenných kolizí. Pro formu klíčových slov je vhodná nehierarchická forma taxonomie, nepředpokládají se totiž žádné vztahy mezi jednotlivými termíny taxonomie. U možností správy taxonomie bude při vytváření taxonomie nastaveno, aby měl každý uživatel, který má pravomoc upravovat příspěvky v systému, následující možnosti:

- přiřazování termínů z této taxonomie k příspěvkům typu Fact
- vytváření nových termínů k taxonomii

- úprava a mazání stávajících termínů taxonomie

K příspěvkům bude možné přiřadit více termínů z této taxonomie. Přiřazovat termíny bude možné během vytváření nového příspěvku typu Fact nebo při editaci již existujícího příspěvku. Při aktivaci pluginu zároveň dojde k vytvoření osmi základních termínů k taxonomii, které byly navrženy ve spolupráci s Czech-American TV po analyzování stávajícího obsahu. U těchto kategorií je očekávané časté použití:

- Castles
- Chateau
- Monuments
- Museum
- Other Landmarks
- Traditions
- Natural Preserves
- Uncategorized

Při vytvoření této taxonomie dojde na editační stránce příspěvků typu Fact k automatickému přidání textového pole pro přiřazování termínů z této taxonomie k příspěvku. Toto textové pole umožňuje přidávat více termínu najednou, obsahuje našeptávač a možnost zobrazení seznamu nejčastěji používaných termínů této taxonomie.

Pokud nastane situace, že by se při vytváření příspěvku nepřiřadila žádná kategorie, je k tomuto příspěvku během ukládání automaticky přidělena kategorie Uncategorized, aby bylo možné tento příspěvek zobrazit při filtrování. Stejně tak k tomu dojde v případě, že by se během editace příspěvku odstranily všechny jeho kategorie a nebyla přidělena žádná nová kategorie.

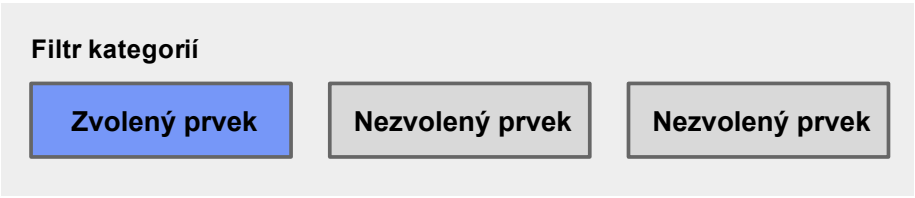
4.2.4 Filtrování dat

Stránka s mapou bude obsahovat také formulář, na kterém si uživatel bude moci vybrat ze seznamu regionů a kategorií, podle kterých se budou body na mapě filtrovat. Prvky seznamu budou jednotlivé termíny daných taxonomií. Ty budou získávány automaticky z WP při každém načtení stránky a budou získávány jen ty termíny, které jsou přiřazené u alespoň jednoho příspěvku. Seznam bude fungovat na principu zaškrtávacích polí vytvořených z HTML

`<input>` tagů typu Checkbox přestylovaných na vzhled tlačítka, které představuje stav zvoleného a nezvoleného prvku seznamu. Zvolený a nezvolený prvek seznamu se budou rozlišovat barvou tlačítka, jako je znázorněno na obrázku 4.3. Důvodem je lepší uživatelská přívětivost oproti standardním zaškrtávacím polím. U seznamu bude umístěna legenda, vysvětlující uživateli význam jednotlivých barevných variant tlačítek.

Součástí seznamu bude také ovládací prvek v podobě dalšího HTML checkboxu, kterým bude možné vybrat rychleji všechny prvky seznamu najednou. Pod seznamem možných kategorií a filtrů bude tlačítko, kterým bude možné filtry uložit. Všechny tyto možnosti filtrování budou umístěny do modálního okna poskytovaného frameworkem Bootstrap. Jde o dialogové okno umístěné na popředí stránky, překrývající stávající obsah. Otevíratelné bude pomocí tlačítka umístěného u mapy (viditelné na obrázku 4.2), zavírat půjde tlačítkem umístěným uvnitř modálního okna.

Uložením filtrů se pomocí JavaScriptu pošle Ajaxový požadavek, obsahující všechny vybrané regiony a kategorie z formuláře. Podle tohoto výběru se získají na straně serveru příslušné příspěvky, které budou jako odpověď na Ajax požadavek vráceny klientovi. Během získávání odpovědi bude zavřeno modální okno s filtry a notifikování uživatele, že dochází k filtrování příspěvků na mapě. Po obdržení odpovědi dojde k odstranění stávajících bodů z mapy a zobrazení nově získaných filtrovaných příspěvků na mapě. Pokud při výběru termínů z obou filtrovaných taxonomií uživatel vybere takové termíny, jejichž kombinace neodpovídá žádnému příspěvku, bude na stránce zobrazena informativní hláška jako zpětná vazba pro uživatele a zároveň bude mapa centrována na pozici, v jaké se nachází při načtení stránky. Veškeré filtrovací akce budou prováděny pomocí JavaScriptu a Ajax požadavků pro možnost filtrování příspěvků na mapě bez nutnosti obnovení stránky. Nastavení filtrů nebude nijak ukládáno a při obnovení stránky se tak jakákoliv nastavení filtrů zruší.



Obrázek 4.3: Rozlišení zvoleného a nezvoleného prvku.

4.2.5 Administrace dat

Dalším požadovaným prvkem pluginu je rozhraní pro administraci dat, ve kterém bude moci uživatel spravovat souřadnice příspěvků typu Fact. Půjde o obsah integrovaný do administračního rozhraní WordPressu. Tímto řešením budou z velké části automaticky ošetřena přístupová práva pro změnu dat uvnitř WP. Vlastním prvkem ošetření přístupu k administraci dat pluginu bude kontrola uživatelských pravomocí, aby k ní mohli přistupovat jen uživatelé s rolí Editor, Public Relations & Media Team čí vyšší. K tomuto účelu poskytuje WP funkci `current_user_can`, která kontroluje, zda má právě přihlášený uživatel přiřazenou danou způsobilost. Tuto funkci lze použít i pro kontrolu uživatelské role, avšak podle [25] může tato praktika vracet nespolehlivé výsledky. Z tohoto důvodu bude kontrola pravomocí probíhat na základě uživatelských způsobilostí. Pro účel administrace dat byla vybrána způsobilost `edit_posts`, představující možnost úpravy příspěvků. Tuto způsobilost mají obě zmíněné role. Administrace dat tohoto pluginu bude rozdělena na dvě hlavní části tvořené dvěma stránkami v administračním rozhraní WP:

1. Seznam příspěvků typu Fact
2. Formulář pro editaci příspěvku typu Fact

Seznam příspěvků typu Fact

Odkaz na tuto stránku bude umístěn jako položka v menu u sekce Fact, která již v administračním rozhraní existuje a obsahuje také odkaz na stránku pro přidávání a editaci příspěvků typu Fact. Další odkaz na tuto stránku bude umístěn jako tlačítko na stránce s mapou, které se bude zobrazovat pouze uživatelům s potřebnými rolemi. Samotná stránka pak bude obsahovat seznam všech příspěvků typu Fact. Ten bude implementovaný třídou `wp_list_table`[8] dostupnou ve WP, která slouží pro generování tabulkových seznamů v administračním rozhraní. Tato třída byla zvolena oproti vlastnímu řešení seznamu příspěvků proto, že nabízí funkce pro jednodušeji implementovatelné pokročilé funkce jako je stránkování seznamu či jeho filtrování.

Samotná třída `wp_list_table` zajišťuje pro řazení seznamu pouze generování hlavičky tabulky s názvy sloupců a posílání parametru názvu sloupce, podle kterého má být seznam seřazován. Funkce `get_posts` pro získávání příspěvků pak obsahuje parametry, kterými lze určit seřazení dat již při jejich

[8] https://codex.wordpress.org/Class_Reference/WP_List_Table

získávání z databáze. Do těchto parametrů však lze zapsat pouze vymezené hodnoty a řazení například podle regionu (který představuje jeden sloupec tabulky) není možné. Řazení seznamu tedy bude řešeno vlastním algoritmem. K tomu bude využita PHP funkce `usort`. Ta umožňuje řazení prvků v poli podle vlastního porovnávací funkce. Příspěvky budou z databáze získávány jako pole objektů, porovnávací funkce pak bude vždy porovnávat ty prvky dvou objektů příspěvků v poli, které byly určeny vybráním sloupce, podle kterého se má seznam seřadit. Pokud nebude vybrán žádný sloupec pro řazení, bude seznam implicitně řazený podle data poslední úpravy příspěvků. Funkce `usort` byla zvolena z důvodu možnosti porovnávání podle variabilních kritérií díky využití vlastní porovnávací funkce.

Formulář pro editaci příspěvku typu Fact

Sloupec s názvem příspěvku v každém řádku tabulkového seznamu bude fungovat jako odkaz na editační formulář daného příspěvku. Formulář bude obsažen na další stránce, vytvořené pluginem v administračním rozhraní WP. Z databáze se při načítání stránky získají informace o příspěvku. Pokud již bude mít přiřazené souřadnice, předvyplní se do formuláře. U formuláře bude také mapa, která bude implementovaná pomocí Google Maps JavaScript API. Na té bude mít uživatel možnost umístit značku, prezentující umístění příspěvku. Při umístění značky či jejím přesunutí bude její lokace automaticky vkládána do formuláře, kterým lze lokaci příspěvku uložit do databáze. Pro vyhledání pozice pomocí uživatelem zadané adresy bude použito Geocoding API, které je součástí Google Maps API. Jedná se o aplikační rozhraní, které umožňuje převádění adresy (jako například "Technická 8, Plzeň 3") na její zeměpisné souřadnice (z předešlého příkladu vrátí zeměpisnou šířku 49.7265837 a zeměpisnou délku 13.3524429). Toto API podle textového řetězce na vstupu vrací pole objektů ve formátu JSON s nejbližší shodou [26]. Pokud bude k hledanému výrazu vrácené neprázdné pole objektů, bude použit první prvek tohoto pole, který by měl představovat nejbližší shodu s hledaným výrazem, jehož souřadnice budou vyplněny do formuláře pro ukládání souřadnic. Použití tohoto API, oproti například vlastní databázi obcí České republiky, poskytuje řadu výhod:

- vyhledávání přesné pozice podle podrobné adresy
- aktuálnost vrácených výsledků
- možnost hledat nejen adresy ale i místa na mapě, například hledané slovo "Sněžka" vrátí zeměpisné souřadnice vrcholu hory Sněžka označeného jako místo na mapách Google

Po odeslání formuláře a uložení souřadnic do databáze jako metadata příspěvku bude uživatel přesměrován zpět na seznam příspěvků, kde se mu zobrazí jednorázová notifikace o výsledku provedení uložení souřadnic do databáze.

4.3 Implementace

4.3.1 Struktura pluginu a inicializace

Během implementace pluginu bylo dbáno na použití doporučených postupů pro vývoj pluginu, takzvaných Best practices. Podle toho byla vytvořena i adresářová struktura pluginu a názvy souborů.

- V kořenovém adresáři pluginu se nachází jediný PHP soubor, který představuje hlavní soubor pluginu.
- Všechny další soubory pluginu jsou logicky rozděleny do adresářů podle typu souboru.
  - Soubory kaskádových stylů jsou v adresáři css a jsou rozdělené na styly veřejné, používané na stránce s interaktivní mapou a na styly používané v administraci dat pluginu.
  - JavaScriptové soubory jsou v adresáři js, opět rozdělené na skripty používané u interaktivní mapy a skripty v administraci dat.
  - Ostatní PHP soubory obsahující veškeré třídy pluginu jsou umístěny v adresáři includes. Soubor `catv_fm_map_container.php` vytváří výstup na stránce s mapou, `catv_fm_database.php` obsahuje třídu s funkcemi pro získávání a ukládání dat do databáze a soubor `catv_fm_functions.php` obsahuje třídu, ve které jsou funkce pro práci s těmito daty a další funkce. Posledním souborem pluginu je `catv_fm_admin_lists.php`, který implementuje třídu `wp_list_table` pro vytvoření seznamu příspěvků v administraci dat

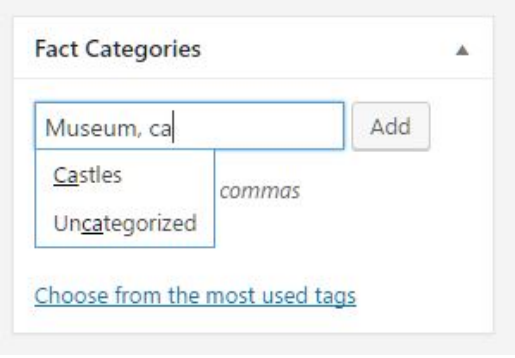
Hlavní soubor pluginu nazvaný `catv_facts_map` zajišťuje připojení pluginu do WP. Na začátku tohoto souboru je umístěná hlavička pluginu, podle které WP rozpozná, že se jedná o adresář pluginu. V hlavičce je specifikován název pluginu a krátký popis. Během aktivování pluginu skrze administrační rozhraní WP dojde k zavolání funkce pluginu, registrované pomocí WP funkce `register_activation_hook`, ve které je WP funkcí

`register_taxonomy` inicializována vlastní taxonomie pro kategorizování příspěvků (kód 4.1). Pro taxonomii byl zvolen unikátní název (první parametr funkce) z důvodu předcházení jmenných konfliktů s jinými taxonomiemi, dále určen typ příspěvků, ke kterým se taxonomie váže (druhý parametr) a nastaveny další parametry podle navržených vlastností. Po inicializaci této taxonomie jsou do ní vloženy základní termíny.

```
register_taxonomy(
    'fm_fact_categories',
    'fact',
    array(
        'hierarchical' => false,
        'label' => __('Fact Categories'),
        'capabilities' => array(
            'assign_terms' => 'edit_posts',
            'edit_terms' => 'edit_posts',
            'delete_terms' => 'edit_posts',
            'manage_terms' => 'edit_posts'
        )
    )
);
```

Kód 4.1: Registrace vlastní taxonomie.

Taxonomie pro kategorie je vytvořena jako veřejná taxonomie. Tímto je v administračním rozhraní WP automaticky vytvořena stránka pro správu této taxonomie a u editace příspěvků typu Fact přidán formulář pro přidávání termínů z této taxonomie k editovanému příspěvku. Na obrázku 4.4 je názorná ukázka přidávání kategorií k příspěvku.



Obrázek 4.4: Přidávání kategorií k příspěvku.

4.3.2 Mapa příspěvků

Interaktivní mapa je zobrazena na místě shortcode značky `[facts-map]`. Tímto způsobem je na místě značky zavolána funkce `catv_fm_init`, kte-

rou jsou na stránce načteny potřebné kaskádové styly a skripty s vlastními funkcemi mapy. Zároveň jsou zde WP funkcí `get_posts` získány z databáze všechny příspěvky typu Fact, které lze zobrazit na mapě. Získané pole těchto příspěvků je pro použití na mapě přiřazeno k JavaScriptové proměnné v JSON formátu pomocí PHP funkce `json_encode`. Mezi potřebnými skripty je načteno i Google Maps JavaScript API s callback funkcí `initializeMap`. Jde o JavaScriptovou funkci, kterou je na stránce mapa inicializována v div elementu s id "mapContainer" vytvořením instance třídy `google.maps.Map` do globální proměnné `map`. Po specifikování vlastností této třídy, představujících povinná nastavení mapy, je ještě do globální proměnné `infoWindow` vytvořena instance objektu `google.maps.InfoWindow`, kterou je inicializováno informační okno. Nakonec dojde ke zobrazení značek všech příspěvků na mapě (kód 4.2).

```
function setMarkers(marker_data) {
  for (i in marker_data) {
        var marker = new google.maps.Marker({
            position: new google.maps.LatLng(
            marker_data[i].lat,
            marker_data[i].lng),
        map: map,
        postName: marker_data[i].post_name,
        postTitle: marker_data[i].post_title,
        postImg: marker_data[i].thumbnail
    });
    var content;
    //zde je vytvoření obsahu do proměnné content
    setInfoWindow(marker, content);
    markers.push(marker);
  }
}
```

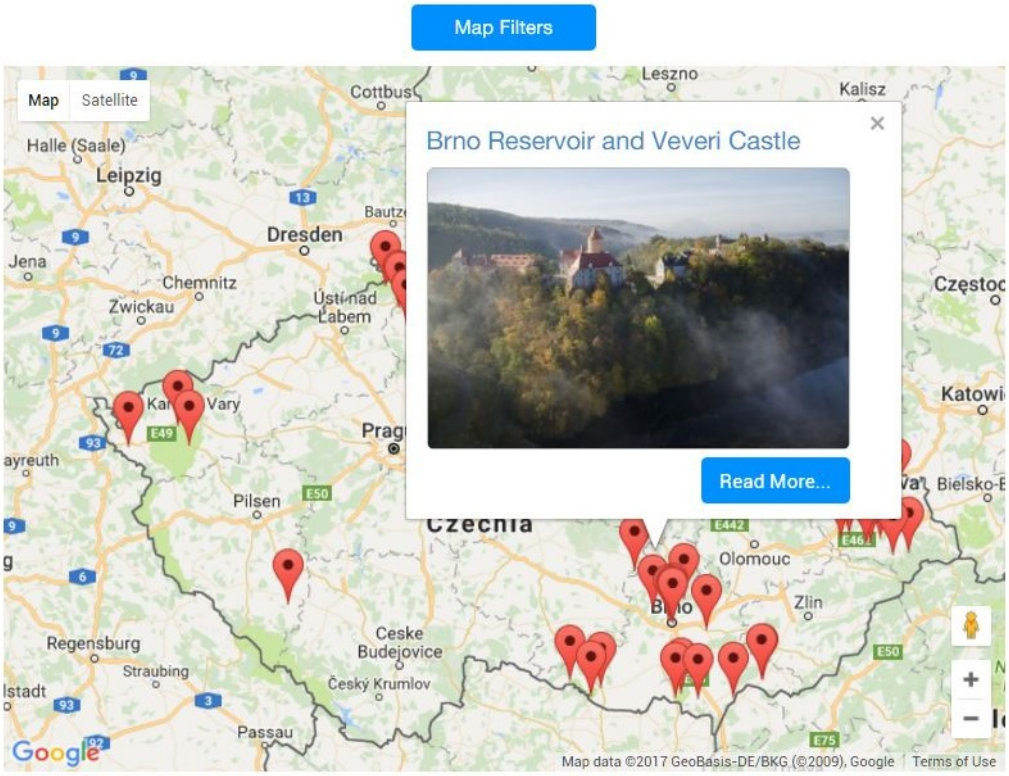
Kód 4.2: Funkce pro zobrazení značek na mapě.

Pro zobrazení těchto značek je procházeno pole všech příspěvků. Ke každému příspěvku je zobrazena značka vytvořením nové instance objektu `google.maps.Marker`, kterému je nastavena proměnná mapy, která byla nastavena při její inicializaci. Dále je určena pozice značky jako objekt `google.maps.LatLng` s dvojicí parametrů představující zeměpisnou šířku a zeměpisnou délku. Ke každé značce jsou přidány informace o příspěvku jako vlastnosti objektu, ze kterých je následně tvořený obsah informačního okna značky. Obsah je vytvořený z HTML kódu uloženého do proměnné `content`, která je následně předána spolu s instancí objektu značky funkci `setInfoWindow` (kód 4.3).

```
function setInfoWindow(marker, content) {
    google.maps.event.addListener(marker, 'click', function() {
        infoWindow.setContent(content);
        infoWindow.open(map, this);
    });
}
```

Kód 4.3: Nastavení informačního okna značky.

Ta k instanci objektu značky přidá event listener poskytovaný aplikačním rozhraním, který bude naslouchat na kliknutí na značku. Obslužná funkce v případě kliknutí na značku nejprve nastaví obsah informačního okna a poté okno zobrazí u značky. Obsahem informačního okna je název příspěvku, náhledový obrázek a tlačítko "Read More...". Všechny tři prvky fungují jako odkaz na stránku s obsahem příspěvku. Ta je otevřena v nové záložce prohlížeče pro zachování nastavení filtrů a pozice na mapě při zobrazení daného příspěvku.



Obrázek 4.5: Ukázka mapy.

Posledním krokem při vytváření jednotlivých značek je přidání instance objektu každé značky do globálně definovaného pole `markers`, které je využívané pro další funkce pluginu. Výsledná inicializovaná mapa se zobrazenými značkami a otevřeným informačním oknem jedné ze značek je ukázána na obrázku 4.5.

4.3.3 Filtry

Filtrování příspěvků je řešeno pomocí Ajaxových požadavků. Pro tyto účely je během načítání skriptů na stránce načten také JavaScriptový soubor pluginu (kód 4.4), ve kterém je funkce pro odeslání Ajax požadavku na server. Součástí načtení tohoto souboru je použití WP funkce `wp_localize_script`, ve které je deklarován JavaScriptový objekt `fm_ajax_script` s proměnnou `ajaxurl`, obsahující cestu k WP souboru `admin-ajax.php`, jímž jsou zpracovávány Ajax požadavky.

```
wp_enqueue_script('catv_ajax_handle',
    plugin_dir_url(__FILE__) . '/js/ajax.js',
    array('jquery'));
wp_localize_script('catv_ajax_handle',
    'fm_ajax_script',
    array('ajaxurl' => admin_url('admin-ajax.php')));
```

Kód 4.4: Načtení JavaScriptového souboru.

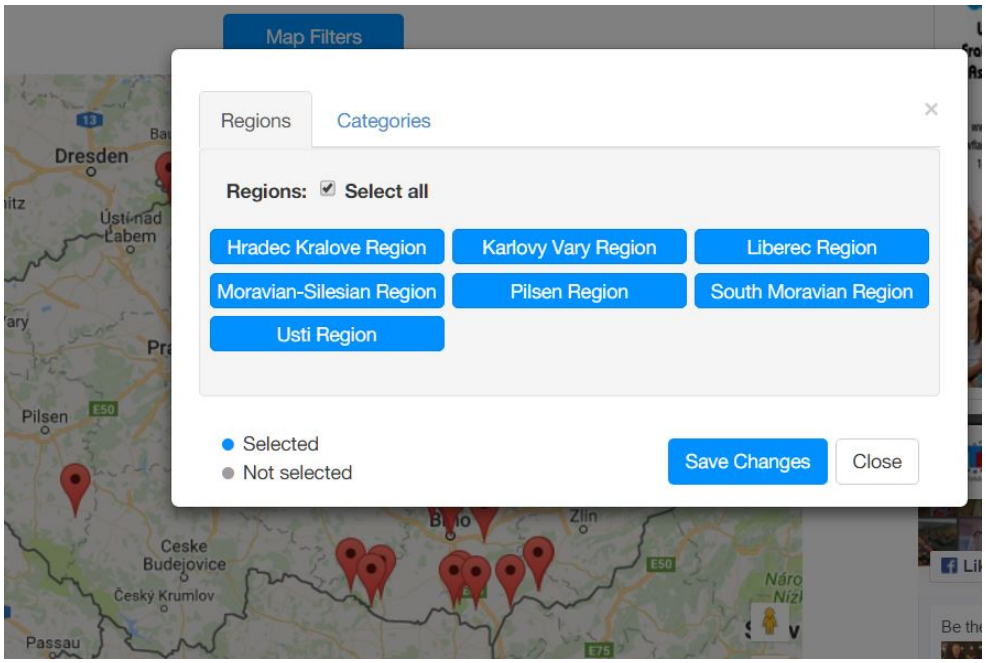
Ovládací prvky filtrů jsou umístěné do modálního okna. Tlačítko umístěné nad mapou (na obrázku 4.2) otevírá modální okno díky přiřazeným atributům `data-toggle="modal"` a `data-target="#filterModal"`. V obsahu okna je vytvořený formulář s metodou post. Uvnitř formuláře je nejprve formou skrytého input tagu se jménem "action" a hodnotou `fm_ajax_hook` zajištěna identifikace formuláře pro vytvoření potřebných WordPress akcí (kód 4.5), které specifikují funkci pluginu volanou při přijatém Ajax požadavku. WordPress rozlišuje Ajax požadavky od nepřihlášených (název této akce obsahuje předponu `wp_ajax_nopriv`) a přihlášených uživatelů. Proto je obslužná funkce tohoto požadavku přiřazována dvakrát.

```
add_action(
    'wp_ajax_fm_ajax_hook',
    'catv_fm_functions::get_filtered_facts');
add_action(
    'wp_ajax_nopriv_fm_ajax_hook',
    'catv_fm_functions::get_filtered_facts');
```

Kód 4.5: Akce pro přidělení obslužných funkcí k Ajax požadavku.

Kromě tohoto tagu je vytvořen další skrytý input tag, do jehož hodnoty je vygenerována WP Nonce. Dále jsou ve formuláři vytvořené samotné seznamy termínů obou taxonomií, jejichž data jsou získána WordPress funkcí `get_terms` s parametrem specifikujícím název taxonomie. Další parametr `hide_empty` v této funkci zajistí, že jsou vráceny pouze používané termíny. Pro každý prvek tohoto pole je vytvořeno vstupní pole formuláře typu checkbox s vlastní CSS třídou , která předělá jeho vzhled do stylu přepínatelného tlačítka. Oba seznamy termínů z filtrovaných taxonomií jsou vizuálně rozděleny do záložkového panelu (obrázek 4.6) vytvořeného třídami frameworku Bootstrap. Ve spodní části modálního okna je umístěné tlačítko pro uložení filtrů, které má atributem `onclick` přiřazenou JavaScriptovou funkci `submit_fm_filters`. Tato funkce slouží k odesílání Ajax požadavku pro filtrování značek na mapě podle uložených filtrů. Nejprve dojde k zavření modálního okna. Poté je jQuery funkcí `post` (kód 4.6) odeslán samotný Ajax požadavek.

K tomu je potřeba u této funkce určit, kam se požadavek odesílá. Jde o proměnnou ajaxurl získanou dříve (kód 4.4). Druhým parametrem funkce jsou data formuláře, která jsou serializována do standardní URL notace ve formě textového řetězce. Možná podoba přenášených dat je tedy například "region=34&category=6". Třetím parametrem funkce post je callback funkce, která je provedena při obdržení odpovědi na požadavek. Posledním parametrem je očekávaný formát dat obdržených v odpovědi.



Obrázek 4.6: Modální okno s nastavením filtrů.

```
jQuery.post(
    fm_ajax_script.ajaxurl, //URL
    jQuery("#fm_filters_form").serialize(), //data
    function (filterData) { //success funkce
        //kód callback funkce
    },
    "json"
);
```

Kód 4.6: jQuery funkce pro odeslání Ajax požadavku.

Obslužná funkce sestává z WordPress funkce `wp_verify_nonce`, která ověří hodnotu WP nonce z formuláře s hodnotou vygenerovanou na serveru, a funkce `get_posts`, která z databáze získává pole objektů příspěvků podle zadaných kritérií (kód 4.7). Prvním je určení typu příspěvku, tedy typ Fact. Dále počet, kolik příspěvků se má z databáze získat. Nastavením této hodnoty na -1 jsou vráceny veškeré nalezené příspěvky splňující podmínky dotazu.

Parametry `tax_query` a `meta_query` rozšiřují původní dotaz o další podmínky. V parametru `tax_query` je určena podmínka, že se mají získat pouze příspěvky, kterým je přiřazen v taxonomii region některý z termínů v poli regions a v taxonomii `fm_fact_categories` některý z termínů v poli categories. Proměnné regions a categories jsou pole s daty získanými od uživatele v Ajax požadavku. Relací `AND` je zajištěno, že budou získávány pouze příspěvky, kterým je přiřazen alespoň jeden z požadovaných regionů a zároveň alespoň jedna z požadovaných kategorií.

Rozšiřující parametr `meta_query` doplňuje původní dotaz podmínkou, kterou je zajištěn výběr pouze těch příspěvků, ke kterým existují v tabulce metadat příspěvků záznamy se souřadnicemi. Tím jsou na Ajax požadavek vraceny pouze ty příspěvky, které lze zobrazit na mapě. Tato funkce je mimo jiné použita pro získání příspěvků i při inicializaci mapy během načítání stránky, pouze s proměnnými `region` a `categories`, které obsahují všechny termíny těchto taxonomií. Pole těchto příspěvků je pak procházeno, a k jednotlivým prvkům WP funkcí `get_post_meta` přidány souřadnice příspěvku. Takto získané příspěvky jsou následně vraceny pomocí PHP funkce `json_encode`, která vrací textový řetězec, obsahující reprezentaci hodnoty poskytnuté proměnné ve formátu JSON. Ten byl v požadavku nastavený jako očekávaný formát dat v odpovědi.

```
$facts = get_posts(array(
    'post_type' => 'fact',
    'numberposts' => -1,
    'tax_query' => array(
        'relation' => 'AND',
        array(
            'taxonomy' => 'region',
            'field' => 'term_id',
            'terms' => $regions
        ),
        array(
            'taxonomy' => 'fm_fact_categories',
            'field' => 'term_id',
            'terms' => $categories
        ),
    ),
    'meta_query' => array(
        array(
            'key' => 'fm_fact_lat',
            'compare' => 'EXISTS'
        ),
        array(
            'key' => 'fm_fact_lng',
            'compare' => 'EXISTS'
        )
    ),
    'post_status' => 'publish'
));
return catv_fm_database::assign_fact_map_data($facts);
```

Kód 4.7: Získání filtrovaných příspěvků z databáze.

Tento textový řetězec je vrácen jako odpověď na Ajax požadavek, která je zpracována callback funkcí. Zde je nejprve zjištěno, zda je vrácená odpověď neprázdná. Pokud ano, jsou z mapy odstraněny stávající značky. Toho je dosaženo tím, že pro každou instanci objektu značky v globálním poli `markers` je nastavena hodnota jejího parametru map na null a toto pole vyprázdněno. Pro zobrazení nově získaných příspěvků je použita funkce `setMarkers`, která je již popsána pro přidávání značek během inicializace mapy. Pokud je odpověď prázdná, je uživateli vypsána do připraveného `div` elementu chybová hláška, že pro jeho výběr filtrů nebyly nalezeny žádné odpovídající příspěvky.

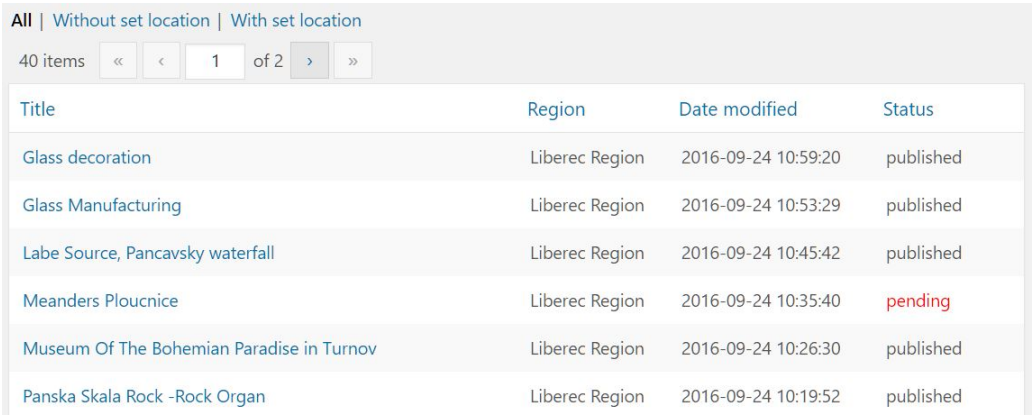
4.3.4 Administrace dat

Přidání stránek pro administraci dat pluginu v administračním rozhraní WP je přiřazené k akci `admin_menu` pomocí funkce `add_submenu_page`. Prvním

parametrem této funkce je určen rodičovský prvek v postranním menu, v tomto případě sekce Facts. Kromě různých vlastností názvu stránek je v této funkci nastavována také způsobilost nutná pro zobrazení této stránky. Ta je, stejně jako potřebná způsobilost pro přiřazení souřadnic, nastavena na `edit_posts`. Posledním parametrem je funkce, která vytváří obsah této stránky. Při této akci jsou zároveň načítány potřebné JavaScript soubory.

Vždy před zobrazením obsahu pro administraci dat je nejprve podmiňovacím tagem `is_admin` kontrolováno, zda je zobrazováno administrační rozhraní WP, aby administrace dat nemohla být přístupná na nežádoucí stránce. Tato funkce však nekontroluje, zda má uživatel požadovaná práva pro zobrazení administrace dat. To je kontrolováno funkcí pluginu, ve které je WP funkcí `current_user_can` zjišťováno, zda má tento uživatel v systému přidělenou způsobilost `edit_posts` či `edit_others_posts`.

Ke zobrazení seznamu příspěvků typu Fact na nově vytvořené stránce administračního rozhraní (na obrázku 4.7) dojde po splnění těchto podmínek. Vytvoření seznamu s jeho funkcemi je ve třídě `catv_fm_admin_lists`, která dědí funkce z WordPress třídy `wp_list_Table`. Celý proces vytvoření a zaplnění seznamu daty a jejich následné seřazení a stránkování probíhá ve funkci `prepare_items`. V té je nejprve definováno asociativní pole, jehož klíče určují, ve kterých sloupcích se zobrazují která data příspěvku. Hodnotou přiřazenou ke klíči je pak určen název sloupce na výstupu. Ke všem sloupcům jsou přidány ovládací prvky pro řazení seznamu.



Obrázek 4.7: Seznam příspěvků v administraci dat.

Formát obsahu sloupců v seznamu je vytvořený funkcí `column_default`. V té je název příspěvku v prvním sloupci změněn na odkaz, kterým je uživatel přesměrován na stránku pro editaci daného příspěvku. Sloupce s regionem příspěvku a datem jeho poslední úpravy jsou ponechané ve formátu, v jakém byly získány z databáze. U posledního sloupce se stavem příspěvku je text

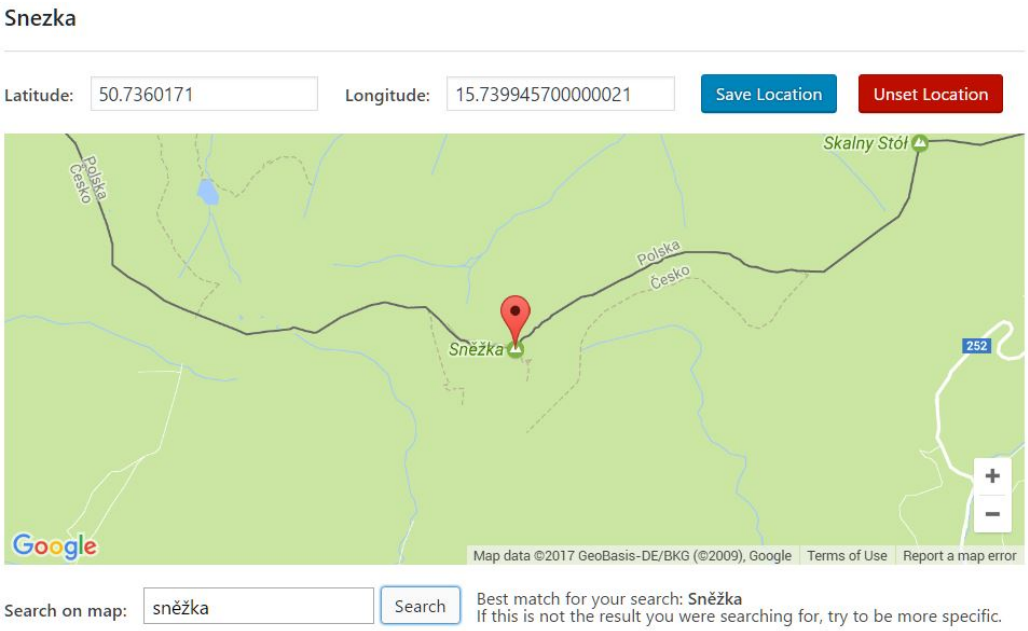
označen červeně, pokud příspěvek nemá stav published (pro upozornění, že tento příspěvek nebude na mapě zobrazen ani po určení souřadnic).

```
function usort_reorder($a, $b)
{
    $orderby = (!empty($_GET['orderby']))
        ? $_GET['orderby']
        : 'post_date';
    $order = (!empty($_GET['order'])) ? $_GET['order'] : 'desc';
    if (property_exists($a, $orderby)) {
        $result = strcmp($a->{$orderby}, $b->{$orderby});
        return ($order === 'asc') ? $result : -$result;
    }
}
```

Kód 4.8: Porovnávací funkce pro řazení seznamu.

Pole s daty pro seznam příspěvků je řazeno PHP funkcí `usort`. Prvním parametrem je pole, které je funkcí řazeno. Druhým parametrem je název porovnávací funkce, podle jejíž návratové hodnoty prvky řadí. Od porovnávací funkce je očekávaná návratová hodnota typu `integer`, podle které funkce řadí dva vybrané prvky (větší než 0, pokud je první parametr větší než druhý, menší než 0, pokud je první parametr menší než druhý a 0 pokud jsou oba porovnávané parametry stejné). V porovnávací funkci (kód 4.8) jsou nejprve určeny implicitní vlastnosti řazení pro případ, že by tyto parametry nebyly zadány uživatelem. K tomu dochází například při prvním načtení stránky. Implicitně jsou data řazena sestupně podle atributu data vytvoření příspěvku. Následuje samotný porovnávací algoritmus, kde je nejprve ověřeno, že prvek obsahuje atribut, podle kterého má být porovnaný. Tyto dva prvky, předávané jako parametry této funkci, jsou porovnané PHP funkcí `strcmp`. Tato funkce vrací výsledek porovnávání ve stejné formě, která je požadována jako návratová hodnota porovnávací funkce pro `usort`. Její výsledek je pouze negován v případě, že je požadováno opačné řazení.

Při zobrazování stránky s formulářem v administraci dat (na obrázku 4.8) je provedena stejná kontrola přístupu jako při zobrazování stránky se seznamem příspěvků. Formulář s nastavenou metodou post má nastavenou akci na `admin-post.php`. V tomto souboru jsou zpracovány formuláře v administračním rozhraní WP. Z tohoto důvodu je ve formuláři vytvořené skryté pole s atributem action, ve kterém je specifikována obslužná funkce formuláře. Dále obsahuje skryté pole s id upravovaného příspěvku a vygenerované pole WP nonce. Obsah stránky související s obsahem příspěvku je získáván podle id upravovaného příspěvku předaného v URL parametru.



Obrázek 4.8: Formulář pro správu souřadnic příspěvku.

Pro vkládání souřadnic je vytvořena dvojice vstupních polí, která musejí být obě vyplněná pro možné odeslání formuláře. Pokud má upravovaný příspěvek specifikované souřadnice, jsou do těchto vstupních polí předvyplněny. Obsahují také kontrolu zadaných dat na straně klienta pomocí regulárního výrazu v atributu pattern. Tím je znemožněno odeslání formuláře, pokud obsah vstupních polí nesplňuje formát desetinného čísla.

Mapa pod formulářem je inicializována stejným způsobem jako na stránce s interaktivní mapou příspěvků. Na této mapě je vytvořena nová instance objektu značky a přidán event listener pro kliknutí na mapu. Pokud je upravovaný příspěvek s již existujícími souřadnicemi, je značka na těchto souřadnicích zobrazena. Listener zpracovává souřadnice kliknutí na mapě a na tomto místě zobrazí značku (případně přesune stávající) a tyto souřadnice vyplní do vstupních polí formuláře.

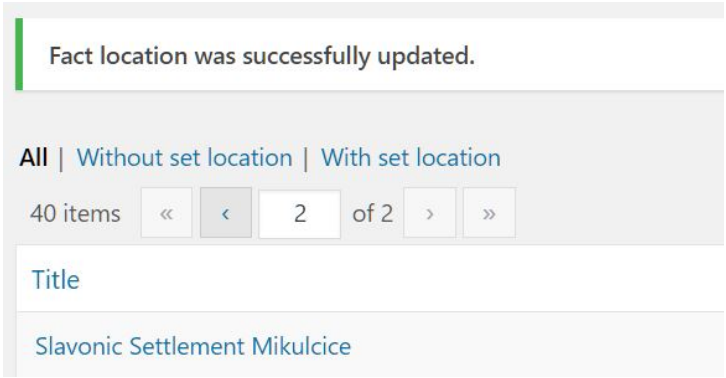
Pod mapou je umístěné vstupní pole pro zadání adresy hledaného místa na mapě. Tlačítko vedle tohoto pole volá funkci `codeAddress`, která předává textový řetězec ze vstupního pole funkci geocode, poskytované Google Maps Geocoding API (kód 4.9). Kromě adresy je specifikován i region na hodnotu ‘cz’. Tímto budou upřednostňovány výsledky z České republiky. Při úspěšně získané odpovědi od aplikačního rozhraní je vybraný první prvek pole s objekty, představující nejbližší shodu s hledaným výrazem.

```
function codeAddress() {
    var address = document.getElementById('fact_address').value;
    geocoder.geocode({'address': address, 'region': 'cz'},
    function (results, status) {
        if (status == 'OK') {
            map.setCenter(results[0].geometry.location);
            map.setZoom(13);
            marker.setPosition(results[0].geometry.location);
            setLatLngFields(results[0].geometry.location);
            /* vypsání nejbližšího výsledku uživateli */
        } else {
            /* notifikování uživatele o nenalezeném místě */
        }
    });
}
```

Kód 4.9: Funkce vyhledání místa na mapě pomocí geocoding API.

Jeho lokace je použita pro nastavení hodnot vstupních polí formuláře pro určení lokace příspěvku a na mapě je vytvořena (či přesunuta stávající) značka na této pozici. Při nesprávné odpovědi je vypsána chybová hláška o nenalezení žádné shody s hledaným výrazem.

Odesláním formuláře dojde ke zpracování poslaných dat obslužnou funkcí. V té je nejprve kontrolována WP nonce formuláře s hodnotou WP nonce vygenerované na serveru. Poté dojde k opětovnému ověření způsobilostí uživatele. Pokud jsou tyto kontroly úspěšné, dojde k sanitizaci vstupních dat WP funkcí `sanitize_text_field` a ke kontrole správnosti formátu ukládaných souřadnic. Pokud jsou obě souřadnice neprázdné a v pořádku projdou kontrolou formátu, jsou WP funkcí `update_post_meta` uloženy do databáze jako dvojice meta dat příspěvku. Tato funkce aktualizuje existující meta data příspěvku s daným klíčem. Pokud příspěvek meta data s tímto klíčem nemá, jsou touto funkcí automaticky vytvořena.



Obrázek 4.9: Zobrazení notifikace o stavu uložení souřadnic.

Poslední částí ukládání souřadnic je vrácení uživatele zpět na seznam příspěvků a zobrazení notifikačního panelu (na obrázku 4.9), který uživatele informuje o úspěšnosti uložení souřadnic. Tato notifikace je vytvářena již při ukládání souřadnic a její informace (typ, obsah) jsou uloženy jako proměnná v relaci uživatele pomocí PHP globální proměnné `$_SESSION`. Při zobrazování obsahu se seznamem příspěvků je pak kontrolováno, zda je proměnná s notifikací nastavená a případně zobrazena. Styl této notifikace je zajištěn CSS styly, které jsou součástí WP. Zobrazením notifikace dojde k odstranění proměnné z relace uživatele, aby při obnovení stránky nedocházelo k opakovanému zobrazování notifikace.

5 Testování

V této kapitole je testován výsledný implementovaný plugin. V rámci testování byly prováděny systémové testy a testována podpora webových prohlížečů. S pomocí získaných výsledků testů jsou poté vyhodnoceny klady a zápory implementovaného pluginu.

5.1 Systémové testy

Systémové testy byly prováděny na lokální kopii webové aplikace Czech-American TV s potřebným obsahem, ve které byl aktivovaný implementovaný plugin. Pro potřeby testování byla vytvořena nová stránka, do které byl plugin začleněn. V rámci těchto testů byly simulovány různé scénáře, které mohou nastat při používání funkcí pluginu (Převzaté z usecase diagramu na obrázku 4.1):

- načtení stránky s interaktivní mapou odkazem jako přihlášený a nepřihlášený uživatel
- použití filtrů na mapě, zvoleny různé kombinace obou typů filtrů
- zobrazení informačního okna a přesměrování na stránku s obsahem příspěvku
- přidávání a správa vytvořené kategorie jako přihlášený uživatel
- zobrazení seznamu příspěvků administrace dat jako přihlášený uživatel
  - řazení seznamu podle různých sloupců
  - stránkování seznamu
- zobrazení formuláře v administraci dat pro editaci souřadnic jako přihlášený uživatel
  - zapsání souřadnic ručně do polí formuláře a následné uložení
  - určení souřadnic kliknutím na mapu a následné uložení
  - nalezení souřadnic vyhledáním adresy a následné uložení
  - nemožnost uložení při zadání chybného formátu souřadnic do formuláře

Command	Target	Value
open	/	
clickAndWait	link=Map of Facts	
verifyText	class=postitle	Map of Facts
verifyElementPresent	id=mapContainer	
verifyElementNotPresent	id=wpadminbar	
verifyElementNotPresent	link=Edit Facts Locations	

Obrázek 5.1: Použitý Selenium test.

U každého scénáře byla ověřena správnost dosažených výsledků manuálním testováním. Dále byl pro testování využit framework Selenium. Jedná se o sadu systémových nástrojů pro testování webových aplikací a podporu automatizace těchto testů. Konkrétně bylo využito rozšíření Selenium IDE[9] do webového prohlížeče Mozilla Firefox, které umožňuje vytváření testovacích případů a jejich následné automatické provádění [27].

Tento způsob testování byl použit u vhodných scénářů. Jedním z těchto testovaných případů bylo otevření stránky s mapou skrze kontextovou nabídku webu a ověření správného zobrazení mapy na této stránce pro uživatele, který není přihlášený do WP. Posloupnost příkazů tohoto testu je zobrazena na obrázku 5.1. Mezi dalšími situacemi testovanými pomocí Selenium IDE bylo správné zobrazení chybové notifikace uživateli při filtrování, kdy nebyl vybrán žádný prvek ze seznamu či zobrazení správné notifikace administrátorovi při uložení validních souřadnic k příspěvku v administraci dat. Zdrojové kódy všech těchto testů jsou na přiloženém DVD. Všechny použité Selenium testy proběhly bez jakýchkoliv chyb. Plugin úspěšně implementuje všechny výše popsané scénáře a funkční požadavky, které jsou od něj očekávány.

[9] http://www.seleniumhq.org/docs/02_selenium_ide.jsp

5.2 Podporované prohlížeče

Další částí testování bylo testování podpory různých webových prohlížečů. Ověřována byla vizuální a funkční stránka výstupů implementovaného pluginu. Pro testování byly zvoleny nejpoužívanější[10] prohlížeče v aktualizovaných verzích dostupné pro operační systém Windows. Konkrétně:

- Google Chrome, verze 56.0.2924.87 (64-bit)
- Mozilla Firefox, verze 53.0 (32 bitů)
- Microsoft Edge, verze 38.14393.0.0

Vizuálně se plugin mezi jednotlivými prohlížeči liší pouze minimálně. Konkrétně jde o různé zobrazování vstupních polí, ať už textových či zaškrtávacích polí. Zaškrtávací pole stylované do vzhledu tlačítka použité u filtrů se však vizuálně neliší. V zásadě však tyto rozdíly nemají vliv na provádění jednotlivých funkcí pluginu. Z tohoto pohledu pak plugin funguje na všech třech testovaných prohlížečích podle očekávání. Lze tak konstatovat, že implementovaný plugin je těmito třemi prohlížeči podporován.

5.3 Zhodnocení dosažených výsledků

Z výsledků testování vyplývá, že výsledný implementovaný plugin úspěšně splňuje veškeré potřeby, které byly Czech-American TV stanoveny. Vytvořením tohoto pluginu a napojením na stávající obsah jejich webové aplikace bude návštěvníkům díky interaktivní mapě umožněno přehlednější vyhledávání informací o zajímavých místech v České republice. S implementovaným filtrováním si budou moci lépe vyhledat jen ty informace, které je zajímají.

Mezi další klady tohoto pluginu lze zařadit zajištění budoucí kompatibility s nově příchozími verzemi systému WordPress díky snaze využívat při vytváření pluginu co nejvíce funkcí, které jsou nabízeny aplikačním rozhraním WP. Výsledný výstup pluginu dle mého názoru stylově zapadá do stávající šablony organizace Czech-American TV a ovládání funkcí pluginu je jednoduché a intuitivní. Dalším kladným prvkem výsledného pluginu je využití Geocoding API, které bylo využito nad rámec stanovených požadavků, pomocí kterého může uživatel v administraci dat určovat jednoduše a rychle pozici příspěvku s vysokou přesností.

Způsobem, jakým je plugin implementován, je momentálně řešena velmi specifická úloha. Plugin lze použít pouze pro vizualizaci příspěvků, které

[10] https://www.w3counter.com/globalstats.php

mají určený typ Fact. Typ příspěvků, které jsou vizualizovány, lze teoreticky změnit přepsáním konstantních proměnných ve zdrojovém kódu pluginu. Tento typ příspěvků musí být ve WP přítomný již při aktivaci pluginu. Zároveň je nutné využívat pluginem vytvořená klíčová slova, aby bylo možné využívat funkce filtrování bodů na mapě. S tím souvisí i možné budoucí vylepšení tohoto pluginu, kde by si uživatel mohl sám bez zásahu do zdrojového kódu určit, jaký druh obsahu by chtěl zobrazovat na mapě, jaké taxonomie by chtěl použít pro možnosti filtrování a podobné, uživatelem přizpůsobitelné vlastnosti.

6 Závěr

V rámci teoretické části této práce byl popsán systém pro správu obsahu WordPress se zaměřením na technologie využívané pro vývoj pluginů pro tento systém. Následně byly analyzovány potřeby Czech-American TV týkající se vytvoření WordPress pluginu. Podle této analýzy byla navržena taková řešení implementace, která budou tyto potřeby splňovat.

Vytvořený plugin umožňuje vizualizaci požadovaného obsahu na Google mapě využitím Google Maps API. Obsah je na mapě možné filtrovat pomocí regionů a klíčových slov. Klíčová slova může administrátor libovolně přidávat, mazat a přiřazovat k příspěvkům zobrazovaným na mapě. Součástí pluginu je také administrace dat, která je dostupná s využitím administračního rozhraní WordPressu. V té je možné k příspěvkům přiřazovat, měnit a odebírat souřadnice, kterými jsou na mapě prezentovány. Při implementaci byl kladen důraz na využití existujícího API pro vývoj pluginů, které je poskytované systémem WordPress. Použitím frameworku Bootstrap vytvořené výstupy pluginu vizuálně zapadají do stávajícího motivu webové aplikace.

Nad rámec stanovených požadavků byly do administrace dat začleněny funkce aplikačního rozhraní Geocoding API, které je součástí Google Maps API. Popis využití těchto funkcí je součástí návrhu administrace dat. Těmito funkcemi je uživateli umožněno vyhledávání pozice na mapě pomocí zadávání adres a míst. Tím je výrazně usnadněno a zrychleno přidávání souřadnic k příspěvkům.

Testování pluginu probíhalo ve formě systémových testů, kde byla ověřována funkčnost vytvořeného pluginu. V rámci tohoto testování byl využit framework Selenium pro částečnou automatizaci těchto testů. Dále byla testována podpora pluginu nejpoužívanějšími webovými prohlížeči. Na základě výsledků těchto testů bylo ověřeno, že plugin splňuje veškeré požadavky, které byly stanoveny organizací Czech-American TV a byly zhodnoceny dosažené výsledky.

Výsledný plugin byl napojený na webovou aplikaci organizace Czech-American TV. Po nenáročné úpravě zdrojového kódu by však mohl být využitelný i v dalších WordPress aplikacích pro úlohy podobného typu. Umožnění těchto úprav koncovému uživateli pluginu bez nutnosti zásahu do zdrojového kódu by mohlo být předmětem dalšího rozšíření této práce.

Přehled zkratek

Ajax	Asynchronous JavaScript and XML
API	Application Programming Interface
CMS	Content Management System
CSRF	Cross Site Request Forgery
CSS	Cascading Style Sheets
EDA	Event-Driven Architecture
GPS	Global Positioning System
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
JSON	JavaScript Object Notation
SQL	Structured Query Language
URL	Uniform Websource Locator
WP	WordPress
WYSIWYG	What You See Is What You Get

Literatura

[1] *Usage of content management systems for websites* [online]. W3Techs, 2017. [cit. 2017/04/25]. Dostupné z: `https://w3techs.com/technologies/overview/content_management/all`.

[2] Williams, B. – Damstra, D. – Stern, H. *Professional WordPress: Design and Development, Second Edition.* John Wiley & Sons, Inc., 2013. ISBN 978-1-118-44227-2.

[3] Nixon, R. *Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5.* O'Reilly Media, 2014. ISBN 978-1-491-91866-1.

[4] *Introduction to PHP* [online]. ZenTut. [cit. 2016/12/08]. Dostupné z: `http://www.zentut.com/php-tutorial/introduction-to-php/`.

[5] *Taxonomies* [online]. WordPress.org. [cit. 2017/03/05]. WordPress Codex. Dostupné z: `https://codex.wordpress.org/Taxonomies`.

[6] Messenlehner, B. – Coleman, J. *Building Web Apps with WordPress.* O'Reilly Media, Inc., 2014. ISBN 978-1-449-36407-6.

[7] *Theme Development* [online]. WordPress.org. [cit. 2016/12/11]. WordPress Codex. Dostupné z: `https://codex.wordpress.org/Theme_Development`.

[8] Hedengren, T. D. *Smashing WordPress: Beyond the Blog.* John Wiley & Sons, Ltd., 2014. ISBN 978-1-118-60075-7.

[9] *Template Hierarchy* [online]. WordPress.org. [cit. 2017/02/28]. Theme Handbook. Dostupné z: `https://developer.wordpress.org/themes/basics/template-hierarchy/`.

[10] Bondari, B. – Griffiths, E. *WordPress 3 Plugin Development Essentials.* Packt Publishing Ltd., 2011. ISBN 978-1-849513-52-4.

[11] Etzion, O. – Niblett, P. *Event Processing in Action.* Manning Publications Co., 2011. ISBN 9781935182214.

[12] *Custom Hooks* [online]. WordPress.org. [cit. 2017/03/28]. Plugin Handbook. Dostupné z: `https://developer.wordpress.org/plugins/hooks/custom-hooks/`.

[13] *Roles and Capabilities* [online]. WordPress.org. [cit. 2017/03/26]. WordPress Codex. Dostupné z: `https://codex.wordpress.org/Roles_and_Capabilities`.

[14] Boodaei, M. – Klein, A. Scrambling HTML to prevent CSRF attacks and transactional crimeware attacks, 2007. US Patent App. 11/714,933.

[15] Kieyzun, A. et al. Automatic creation of SQL injection and cross-site scripting attacks. In *Software Engineering, 2009. ICSE 2009. IEEE 31st International Conference on*, s. 199–209. IEEE, 2009.

[16] *Best Practices* [online]. WordPress.org. [cit. 2017/04/05]. Plugin Handbook. Dostupné z: `https://developer.wordpress.org/plugins/the-basics/best-practices/`.

[17] *A vocabulary and associated APIs for HTML and XHTML* [online]. W3C, 2014. [cit. 2017/04/15]. Dostupné z: `https://www.w3.org/TR/html5/introduction.html`.

[18] Svennerberg, G. *Beginning Google Maps API 3.* Apress, 2010. ISBN 978-1-4302-2802-8.

[19] *Google Maps API V 3 - Tutorial* [online]. w3resource, 2017. [cit. 2017/04/29]. Dostupné z: `http://www.w3resource.com/API/google-maps/`.

[20] *Getting Started* [online]. Google Developers, 2017. [cit. 2017/04/13]. Google Maps JavaScript API. Dostupné z: `https://developers.google.com/maps/documentation/javascript/tutorial`.

[21] Sommerville, I. *Softwarové inženýrství.* Computer Press, 2013. ISBN 978-80-251-3826-7.

[22] Arlow, J. – Neustadt, I. *UML 2 a unifikovaný proces vývoje aplikací.* Computer Press, a.s., 2011. ISBN 978-80-251-1503-9.

[23] Dincer, A. – Uraz, B. *Google Maps JavaScript API Cookbook.* Packt Publishing Ltd., 2013. ISBN 978-1-84969-882-5.

[24] *Writing a Plugin* [online]. WordPress.org. [cit. 2017]. WordPress Codex. Dostupné z: `https://codex.wordpress.org/Writing_a_Plugin`.

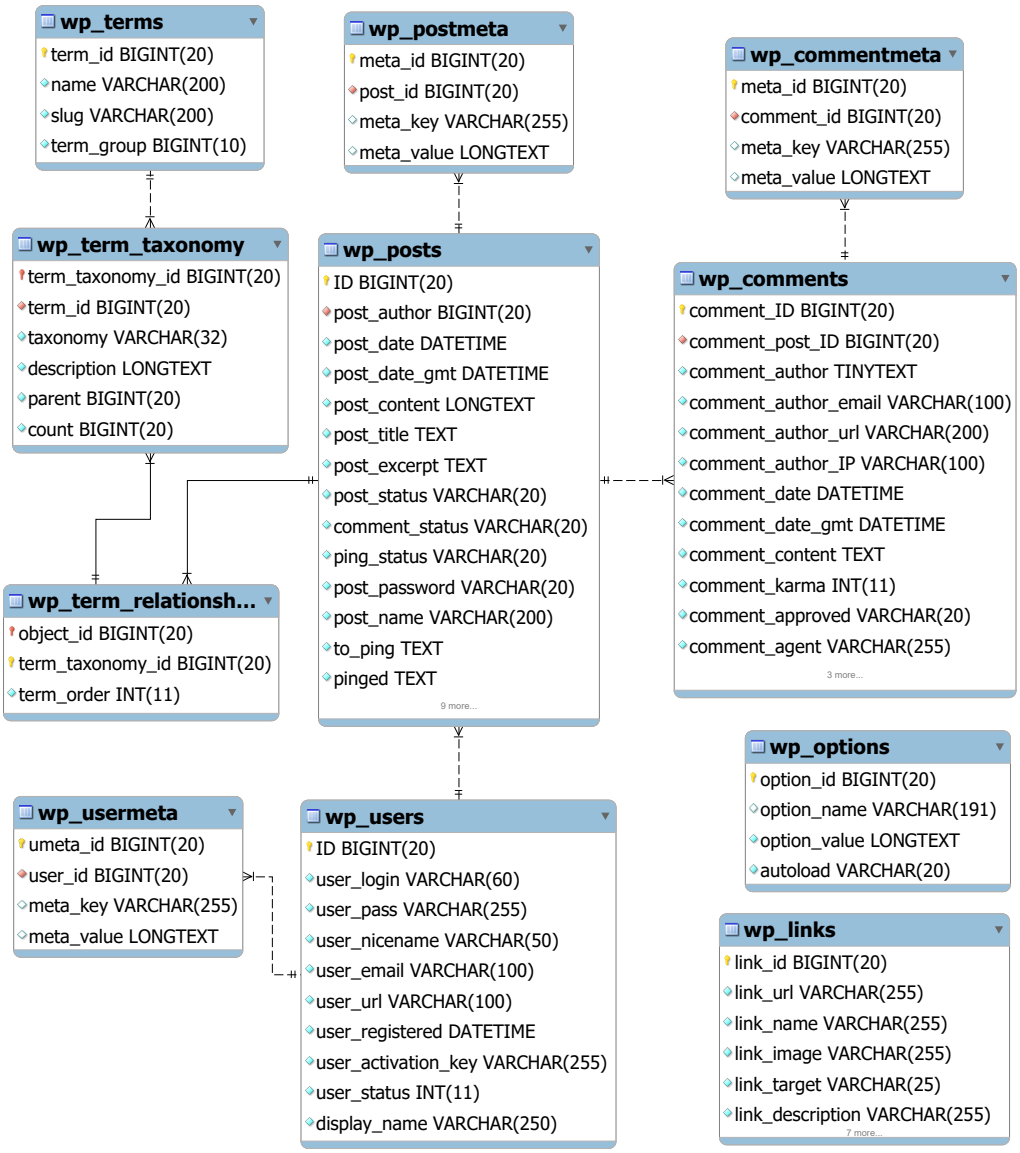
[25] *Function Reference/current user can* [online]. WordPress.org. [cit. 2017/04/28]. WordPress Codex. Dostupné z: `https://codex.wordpress.org/Function_Reference/current_user_can`.

[26] *Developer's Guide, What is Geocoding?* [online]. Google Developers, 2017. [cit. 2017/04/12]. Google Maps Geocoding API. Dostupné z: `https://developers.google.com/maps/documentation/geocoding/intro`.

[27] *Introduction, Test Automation for Web Applications* [online]. Selenium Project, 2017. [cit. 2017/04/28]. Selenium Documentation. Dostupné z: `http://docs.seleniumhq.org/docs/01_introducing_selenium.jsp`.

Přílohy

A Schéma databáze WP



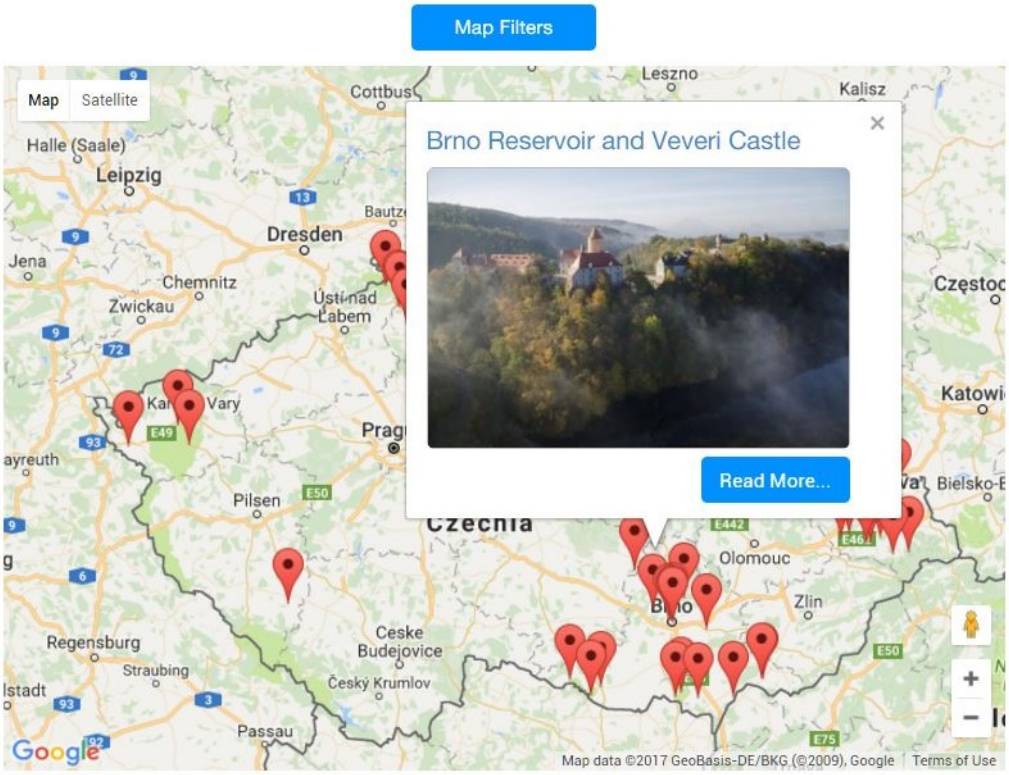
B Uživatelská příručka

Instalace pluginu

Pro instalaci pluginu je potřebná aktualizovaná verze WP, ve které jsou registrované příspěvky typu Fact a kategorie krajů s názvem region. Zip archiv pro instalaci pluginu je umístěný na přiloženém DVD. Tento archiv se nahraje do systému skrze administrační rozhraní WP v sekci Plugins - Add new. Po úspěšném nahrání pluginu je plugin nutné aktivovat. Aktivace pluginu je uživateli nabídnuta po nahrání archivu, případně lze plugin kdykoliv aktivovat/deaktivovat v seznamu pluginů.

Administrace dat je do administračního rozhraní WP přidána automaticky a je dostupná pod záložkou Edit Fact Location v sekci Facts v kontextové nabídce administračního rozhraní. Mapa příspěvků je zobrazena na místě zkratky "[facts-map]", kterou je třeba umístit do textového obsahu stránky, na které se má mapa zobrazit.

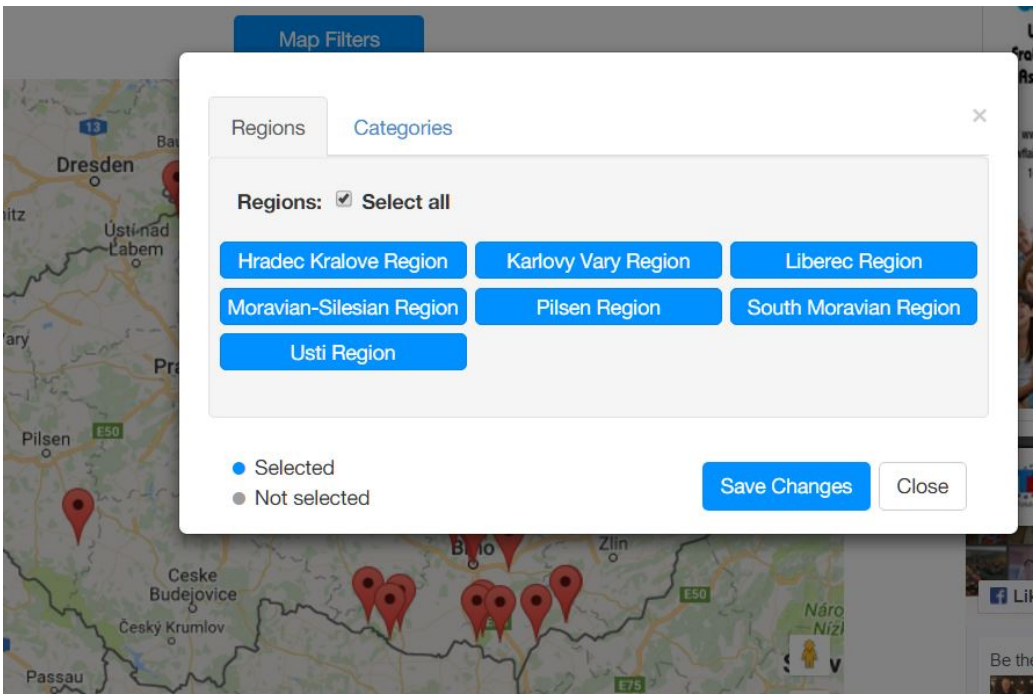
Ovládání mapy



Obrázek B.1: Mapa příspěvků.

Mapu lze posouvat levým tlačítkem myši. Dále lze mapu přibližovat a oddalovat kolečkem myši či ovládacími prvky v pravém dolním rohu mapy. Kliknutím na ikonu značky na mapě se u této značky otevře informační okno (obrázek B.1), které obsahuje informace o příspěvku (Název, obrázek). Tyto informace, spolu s tlačítkem v pravém dolním rohu informačního okna fungují jako odkaz na stránku příspěvku, která se otevře v nové záložce.

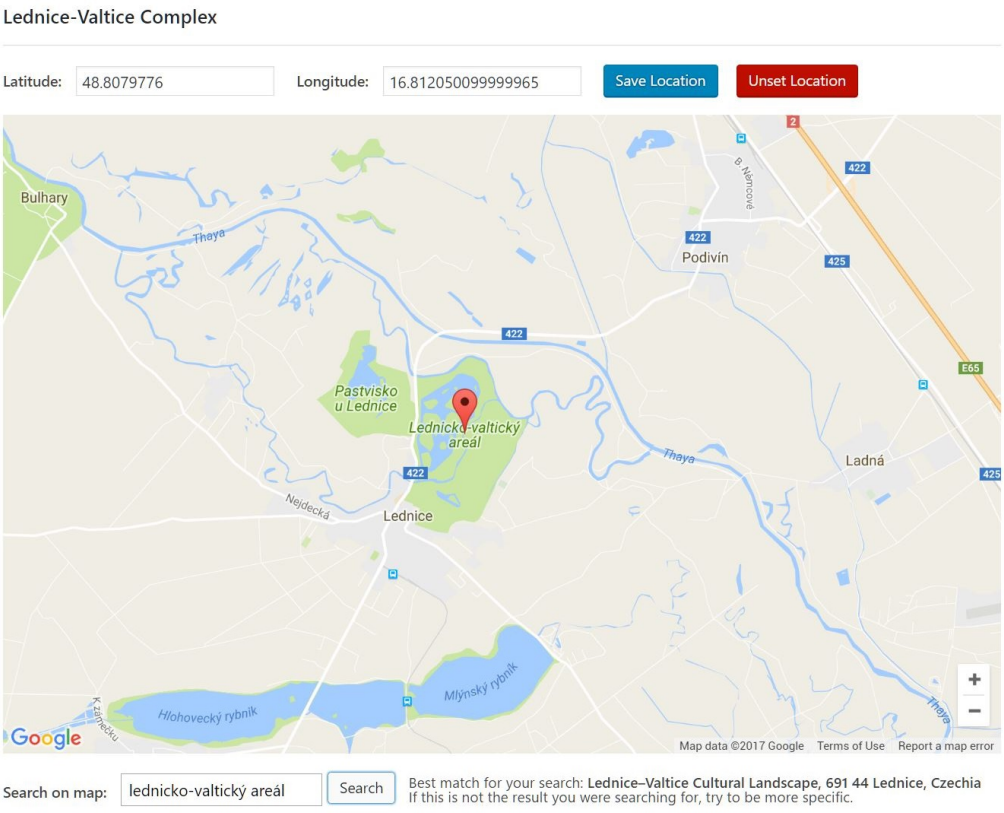
Filtrování bodů na mapě je dostupné tlačítkem "Map Filters" nad oknem mapy. Kliknutím na toto tlačítko dojde k otevření modálního okna (obrázek B.2), ve kterém lze nastavit požadované filtry. V horní části okna se nacházejí záložky pro přepínání mezi regiony a kategoriemi (klíčovými slovy) příspěvků. Termíny zvolené pro filtrování jsou označené modrým pozadím, nezvolené termíny šedým. Záložka s regiony i záložka s kategoriemi obsahuje zaškrtávací pole "Select all", kterým lze rychle označit všechny termíny v seznamu. Po vybrání požadovaných termínů ze seznamů se tato nastavení uloží tlačítkem "Save Changes", čímž dojde k zavření modálního okna a zobrazení filtrovaných příspěvků na mapě. Pokud nejsou pro vybrané nastavení filtrů nalezeny žádné příspěvky, je nad mapou vypsána chybová hláška. Pro zavření modálního okna s filtry bez uložení změn slouží tlačítko "Close" v pravé části okna.



Obrázek B.2: Modální okno s filtry.

Administrace dat

Administrace dat je dostupná pro přihlášeného uživatele, který může v systému upravovat příspěvky. Odkaz se nachází v kontextové nabídce administračního rozhraní WP v sekci Facts. Další odkaz je umístěný na stránce s mapou. Administrace dat obsahuje seznam všech příspěvků typu Fact se stručnými informacemi o nich (Název, region, datum poslední úpravy, stav). Seznam lze řadit podle sloupců kliknutím na příslušný sloupec. Dále lze obsah seznamu filtrovat podle toho, zda má příspěvek přiřazenou lokaci či ne, Ovládací prvky těchto filtrů se nacházejí nad seznamem. Název příspěvku v seznamu příspěvků funguje jako odkaz na formulář pro úpravu jeho lokace (na obrázku B.3).

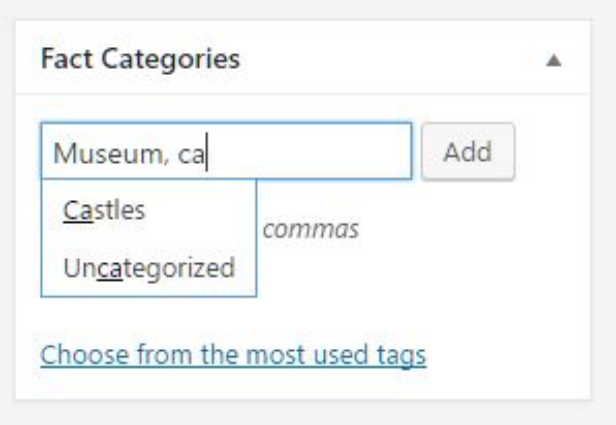


Obrázek B.3: Formulář pro úpravu souřadnic příspěvku.

Formulář obsahuje dvojici vstupních polí pro zadání zeměpisné šířky (Latitude) a zeměpisné délky (Longitude). Jde o desetinné číslo, pro oddělení celé části a desetinné části se používá tečka (tedy například 50.12345). Zeměpisná šířka je číslo v rozmezí -90.0 až 90.0, zeměpisná délka je číslo v rozmezí

-180.0 až 180.0. Pokud již má příspěvek určenou lokaci, jsou tyto souřadnice předvyplněny do formuláře. Určení pozice je možné i kliknutím na mapu pod formulářem, které nastaví souřadnice místa kliknutí do vstupních polí. Dalším způsobem určení pozice je vyhledání pozice. Pro tento účel slouží textové pole pod mapou. Na základě vepsané adresy je nalezena nejbližší shoda se zadanou adresou a nastavení těchto souřadnic do vstupních polí formuláře. Pro ověření nalezeného výsledku je po vyhledání adresy vypsána adresa nejbližší nalezené shody. Uložení souřadnic je provedeno odesláním formuláře tlačítkem "Save Location", při kterém je uživatel vrácen zpět na seznam příspěvků. Nad seznamem je zobrazena notifikace o stavu uložení dat do systému.

Klíčová slova k příspěvku lze přiřazovat během vytváření nového či úpravy existujícího příspěvku typu Fact skrze administrační rozhraní WP. K tomu slouží okno na postranním panelu s názvem Fact Categories (obrázek B.4). Do textového pole se vypisují klíčová slova oddělená čárkou a k příspěvku se přidají tlačítkem Add vedle textového pole. Pod textovým polem jsou zobrazena všechna aktuálně přiřazená klíčová slova. Odebrat klíčové slovo příspěvku lze v tomto seznamu kliknutím na křížek vedle daného klíčového slova. Po přidání klíčových slov je nutné uložit celý příspěvek aby se provedly změny klíčových slov. Všechna klíčová slova, která jsou v systému lze spravovat v administračním rozhraní WP na stránce Fact Categories dostupné ze sekce Fact v kontextové nabídce administračního rozhraní.



Obrázek B.4: Formulář pro úpravu klíčových slov příspěvku.

C Obsah přiloženého DVD

Na přiloženém DVD jsou obsaženy následující adresáře a soubory:

- v adresáři `WP plugin` jsou obsaženy zdrojové kódy pluginu zabalené do Zip archivu.
  - tento archiv je zároveň možné použít pro instalaci pluginu.
- v adresáři `Selenium testy` jsou obsaženy zdrojové kódy použitých Selenium testů.
- v adresáři `LaTeX` jsou zdrojové kódy textu této práce včetně všech obrázků použitých v této práci.
- Text práce ve formátu pdf je v kořenovém adresáři tohoto DVD.