VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA PODNIKATELSKÁ
ÚSTAV INFORMATIKY

FACULTY OF BUSINESS AND MANAGEMENT
INSTITUTE OF INFORMATICS

NÁVRH MODULU SYSTÉMU PRO ANALÝZU CHOVÁNÍ NÁVŠTĚVNÍKŮ

PROPOSAL OF A SYSTEM MODULE FOR ANALYZING VISITORS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — TOMÁŠ WAWROSZ
AUTHOR

VEDOUCÍ PRÁCE — Ing. JAN LUHAN, Ph.D.
SUPERVISOR

BRNO 2015

Vysoké učení technické v Brně
Fakulta podnikatelská

Akademický rok: 2014/2015
Ústav informatiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Wawrosz Tomáš

Manažerská informatika (6209R021)

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách, Studijním a zkušebním řádem VUT v Brně a Směrnicí děkana pro realizaci bakalářských a magisterských studijních programů zadává bakalářskou práci s názvem:

Návrh modulu systému pro analýzu chování návštěvníků

v anglickém jazyce:

Proposal of a System Module for Analyzing Visitors

Pokyny pro vypracování:

Úvod
Cíle práce, metody a postupy zpracování
Teoretická východiska práce
Analýza současného stavu
Vlastní návrhy řešení
Závěr
Seznam použité literatury
Přílohy

Seznam odborné literatury:

CONOLLY, T., C. E. BEGG a R. HOLOWCZAK. Mistrovství - databáze: profesionální průvodce tvorbou efektivních databází. 1. vyd. Brno: Computer Press, 2009. 584 s. ISBN 978-80-251-2328-7.
DAWSON, A. Výjimečný webdesign: jak tvořit osobité, přitažlivé, použitelné weby. 1. vyd. Brno: Computer Press, 2012. 344 s. ISBN 978-80-251-3719-2.
KAUSHIK, A. Webová analytika 2.0: kompletní průvodce analýzami návštěvnosti. 1. vyd. Brno: Computer Press, 2011. 456 s. ISBN 978-80-251-2964-7.
KOFLER, M. a B. ÖGGL. PHP 5 a MySQL 5: pru

Vedoucí bakalářské práce: Ing. Jan Luhan, Ph.D.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2014/2015.

L.S.

doc. RNDr. Bedřich Půža, CSc.	doc. Ing. et Ing. Stanislav Škapa, Ph.D.
Ředitel ústavu	Děkan fakulty

V Brně, dne 28.2.2015

Abstrakt

Tato bakalářská práce se zabývá návrhem prototypu modulu, který slouží k analýze webových stránek pomocí teplotních map. Modul byl navržen dle provedené analýzy, která se skládá z analýzy samotného produktu z uživatelského i technického hlediska a analýzy konkurence nabízející podobnou funkcionalitu. Součástí práce je návrh vytvoření funkčního prototypu, návrh uživatelského rozhraní v praxi a návrhy pro rozšíření funkcionality v budoucnu.

Abstract

This bachelor thesis propose a prototype of a module, which serves as s tool for a website analysis via heatmaps. The module has been designed on the base of the analysis. The analysis include an analysis of the product itself from both, customer and technical points of view and an analysis of competitors offering similar functionality. This thesis includes proposal of a functional prototype of the module and design of the user interface for the module.

Klíčové slova

Analýza webových stránek, Software jako služba, Smartsupp, teplotní mapy, PHP, Javascript, Node.js, CoffeeScript

Key words

Website analysis, Software As a Service, Smartsupp, heatmaps, PHP, Node.js, Javascript, CoffeeScript

Bibliografická citace

WAWROSZ, T. *Návrh modulu systému pro analýzu chování návštěvníků.* Brno: Vysoké učení technické v Brně, Fakulta podnikatelská, 2015. 58 s. Vedoucí práce Ing. Jan Luhan, Ph.D..

Čestné prohlášení

Prohlašuji, že předložená bakalářská práce je původní a zpracoval jsem ji samostatně. Prohlašuji, že citace použitých pramenů jsou úplné a že jsem ve své práci neporučil autorské práva (ve smyslu Zákona č. 121/2000 Sb., o právu autorském a o právech souvisejících s právem autorským).

Tomáš Wawrosz

V Brně dne 28. května 2015 .....

Podpis

Poděkování

Tímto bych rád poděkoval panu Ing. Janu Luhanovi, Ph.D. za skvělou spolupráci a rady, které mi poskytnul při psaní této práce.

Obsah

Úvod

Během mého působení ve firmě Smartsupp, s.r.o. vznikl na základě zpětné vazby zákazníků požadavek pro vznik funkce umožňující analýzu webových stránek pomocí teplotních map. Takto vznikl podnět pro vytvoření této bakalářské práce.

Podstatou této bakalářské práce je navrhnout prototyp modulu na základě požadavků a provedené analýzy, který firmě umožní otestovat jeho funkcionalitu a na jehož základě bude umožněno uvést praktické části vytvořené v rámci této bakalářské práce do praxe.

Jak již bylo výše zmíněno, součásti této bakalářské práce je vytvoření analýzy funkcionalit modulu a následně, na základě této analýzy, vytvoření funkčního prototypu modulu. Dále je součásti této práce vytvoření grafického návrhu uživatelského rozhraní tak, jak by mohlo být využitelné v praxi a návrhy pro další rozšíření funkcionalit modulu.

Cíl práce a vymezení problémů

Cílem této bakalářské práce je navrhnutí modulu pro analýzu chování návštěvníků na webu pomocí teplotních map. Modul bude navržen na základě požadavků společnosti Smartsupp, s.r.o. a bude navrhnut pro produkt Smartsupp.

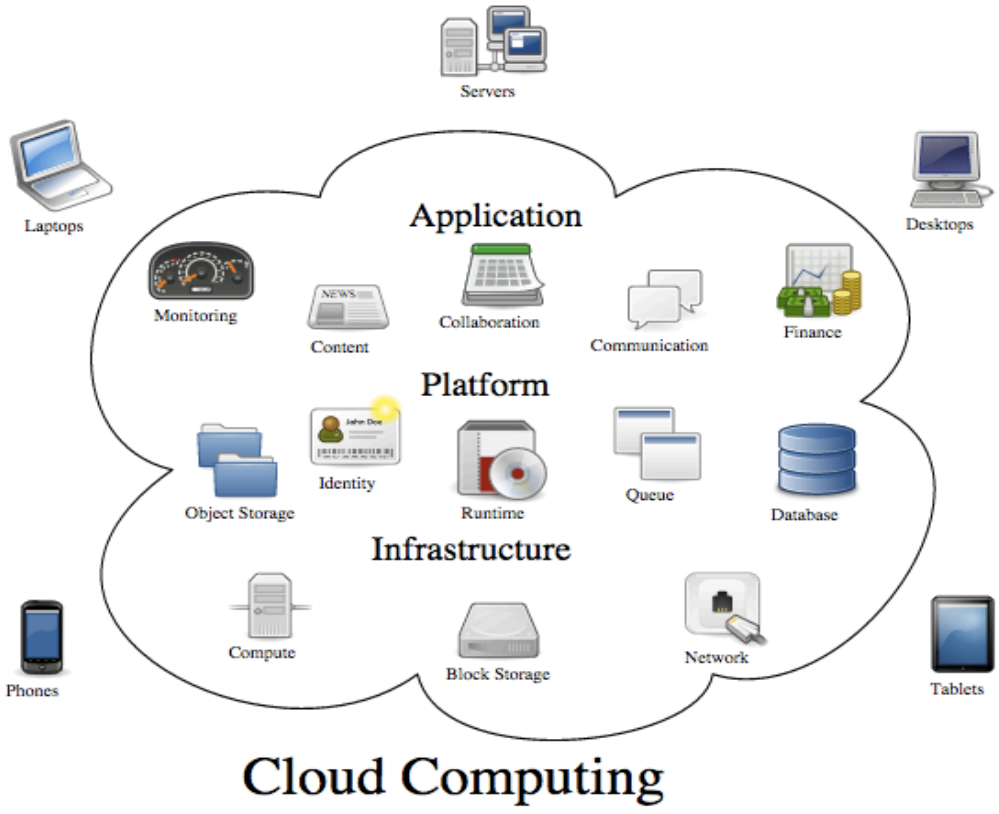
Funkcionalita modulu bude určena na základe analýzy funkčnosti konkurenční služeb tak, aby byl modul konkurenceschopný.

1 Teoretické východiska práce

V této části bakalářské práce se zabývám teoretickými východisky pro problematiku tvorby modulu ve webovém prostředí a popis konkrétních technologií, které využívá firma Smartsupp, s.r.o.

1.1 Cloud computing

Cloud computing je sdílení hardwarových a softwarových prostředků pomocí internetu. Uživatel nemusí vlastnit přímo konkrétní hardware a software, ale pomocí internetu přistupuje ke cloudu poskytovatele služeb (1).



Obrázek 1: Znázornění cloud computingu (2)

Nabídky, které se distribuují pomocí cloud computingu jako služby, mají tyto vlastnosti:

- Nízké bariéry vstupu (jsou dostupné i pro malé a střední podniky)

- Velká škálovatelnost (využívá se tolik prostředků, kolik je potřeba)
- Dostupnost pro více uživatelů
- Nezávislost na přístupovém zařízení (1)

1.1.1 IaaS

Infrastructure as a Service (Infrastruktura jako služba) poskytuje zákazníkovi infrastrukturu (hardware). Zákazník nemusí tedy investovat vysoké částky na pořízení a hostování vlastních serverů, ale pronajme si je od poskytovatele (1).

Infrastrukturu je možné dynamicky škálovat oběma směry v závislosti na požadavcích aplikace. Účtování probíhá většinou podle reálně spotřebovaných prostředků (1).

IaaS umožňuje pronájem mimo jiné těchto zdrojů:

- Úložiště na serveru
- Paměť
- Výkon procesoru (1)

1.1.2 PaaS

Platform as a Service (Platforma jako služba) poskytuje prostředky nutné k vytváření aplikací a služeb výlučně z internetu aniž by byl potřeba instalovat jakýkoliv software (1).

Mezi služby PaaS patří například:

- Návrh aplikací
- Vývoj
- Testování
- Hostování (1)

1.1.3 SaaS

Software as a Service (Software jako služba) je model, v němž je aplikace i s daty hostována na straně dodavatele a zákazník k ní přistupuje pomocí internetu. Zákazní se tedy nemusí starat o správu ani podporu produktu. Na druhou stranu má zákazník malý vliv na to, jak aplikaci změnit, či upravit. Dodavatel služby se stará o veškerou technickou podporu, aktualizace a udržení veškeré infrastruktury v chodu (1).

Výhodou pro zákazníky je rozložení nákladů na pořízení softwaru, kdy zákazník nemusí dopředu platit fixní částku za pořízení produktu, ale platí za to, co skutečně používá (1).

Nevýhodou může být závislost zákazníka na dodavateli, kdy nemusí existovat jednoduchý způsob jak svá data přenést k jinému poskytovateli, případně náklady na přesun jsou zpoplatněny (1).

Příklady aplikací, které mohou využívat modelu SaaS:

- Systémy CRM
- Účetní systémy
- Analýza webu
- Správa webového obsahu
- Správa IT služeb (1)

1.2 Responzivní webdesign

V dnešní době se stále více uživatelů připojuje k internetu pomocí mobilních zařízení. Webové stránky vytvořené primárně pro desktopové počítače se obtížně zobrazují. Web většinou přestane být uživatelsky přívětivý (text není čitelný, aktivní prvky jsou příliš malé apod.) (3).

Zařízení	Zastoupení
Desktop	71,47%
Tablet	8,07%
Mobil	20,46%

Tabulka 1: Podíl přístupů na web podle zařízení v roce 2015 (4)

Jedním ze způsobů, jak tento problém vyřešit, je využít verze webové stránky určené speciálně pro mobilní zařízení. Taková verze většinou běží na vlastní doméně (například http//m.seznam.cz). Nevýhodami tohoto řešení jsou větší náklady na údržbu a vývoj (dvě verze jedné webové stránky) a nemožnost pružně reagovat na nová zařízení (tablety, phablety) (3).

Od zavedení technologií HTML5 a CSS3 do praxe lze zobrazování webové stránky pro různá rozlišení pomocí responzivního designu. Responzivní webová stránka má jednu verzi HTML kódu a zobrazuje se na jedné URL adrese. Díky CSS3 stránka sama rozpozná, pro jaké rozlišení má web vykreslit (3).

Responzivní webdesign má tři úrovně:
Flexibilní strukturu – Elementy(divy) na stránce nemají danou absolutní polohu, ale mohou flexibilně plavat (5).

Flexibilní obrázky – Obrázky nemají danou přesnou velikost. Mohou se flexibilně měnit podle velikosti rodičovského prvku (5).

Media Queries – klíčová vlastnost CSS3 pro responzivní webdesign. Umožňuje aplikování různých sad CSS pravidel podmíněně na velikosti zobrazeného webu (5).

1.3 Teplotní mapy

Teplotní mapa (anglicky heatmaps) je grafické zobrazení hodnot pomocí spojitého barevného spektra. V prostředí internetu mohou zobrazovat interakce návštěvníků na webových stránkách (6).

Pomocí teplotních map je možné testovat chování návštěvníků na webu a následně vylepšovat použitelnost webu (zvýraznění aktivních prvků, přeskládaní obsahových sekcí apod.) (6).

Nejčastěji se pomocí teplotních map zobrazují tyto interakce:

- **Kliky na webu (click tracking)** – mapa zobrazuje, kam návštěvnici na webu klikají. Je tedy možné testovat účinnost aktivní prvků apod. (7)
- **Pohyb myši (mouse-eye tracking)** – mapa zobrazuje pohyb kurzoru myší návštěvníků. Dle publikované studie je prokázán vzájemný vztah mezi pohybem kurzoru po stránce a pohybem očí (pohledu) návštěvníka. Lze tedy s určitou přesností měřit, kam se návštěvník na webové stránce dívá (7, 8).
- **Posun okna (scroll-reach)** – mapa zobrazuje, kam se dostane návštěvník v rámci posunu stránky. Lze tedy zjistit, zda se návštěvníci dostanou k prvků, které jsou pod ohybem stránky (výřez stránky, který návštěvník vidí aniž by musel posunovat obsah) (7).

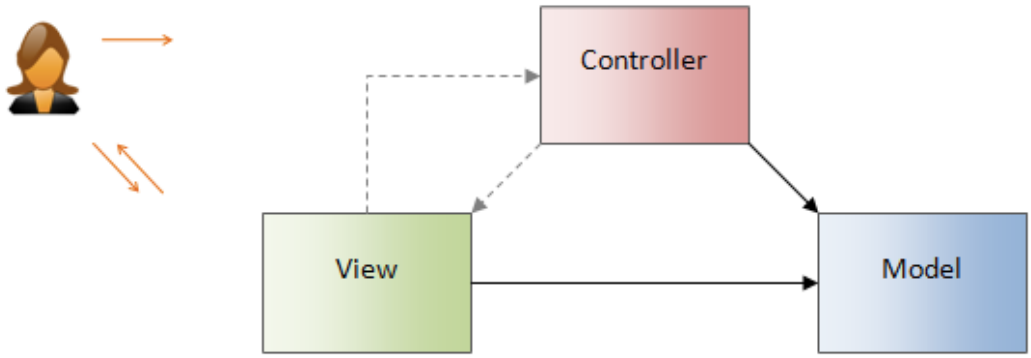
Obrázek 2: Příklad teplotní mapy (6)

1.4 Architektura MVC

Cílem architektury MVC je oddělit aplikační a prezenční logiku webové aplikace. Aplikace je rozdělená do tří samostatných logických celků. To přináší výhody ve formě lehčí udržitelnosti a správy i v případě, že na aplikaci se podílí tým více lidí (9).

Aplikace se z pohledu MVC dělí do těchto logických celků:

- Model
- Viev
- Controller (9)



Obrázek 3: Schéma MVC aplikace (9)

1.4.1 Model

Model obsahuje aplikační a business logiku aplikace. Zajišťuje přístup k datům a práci s nimi. Model je naprosto nezávislý na ostatních dvou částí aplikace (9).

1.4.2 View

Zobrazuje data přijatá z modelu. Data nemusí být zobrazena pouze ve formátu HTML, ale je možné je zobrazovat například ve formátech XML nebo JSON (9).

1.4.3 Controller

Controller zajišťuje o chod celé aplikace. Má na starosti uživatelské akce a zajišťuje změny v pohledech a modelech (9).

1.5 Použité technologie

1.5.1 HTML

HTML (HyperText Markup Language) je značkovací jazyk, který slouží k vytváření dokumentů určených pro zobrazování na internetu. Podstatou dokumentu HTML je textový souboru s příponou .html. Jako interpreter jazyka HTML slouží prohlížeč, který se stará také o jeho zobrazení. Základními stavebními prvky jazyka HTML jsou tzv. Tagy (10).

Aktuální verzi jazyka HTML je verze číslo 5. První pracovní verze jazyka HTML5 vznikla v roce 2008. Specifikace jazyka, kterou mají na starosti organizace WHATWG(Web Hypertext Application Technology Working Group), W3C(WorldWide Web Consortium) a IETF(Internet EngineeringTaskForce), ještě není ve své finální podobě. Díky tomu, že HTML5 řeší velmi praktické problémy při vývoji webů, tvůrci prohlížečů aktivně implementují jeho nové funkce (10).

Mezi nové rozhraní API, které přináší HTML5, patří mimo jiné tyto:
Canvas – poskytuje rozhraní pro snadné malování 2D a 3D objektů pomocí tagu <canvas> 10).

WebSocket API – poskytuje rozhraní a protokol pro obousměrnou komunikaci využivající pouze jeden socket (10).

Webové uložiště – rozhraní pomocí něhož lze ukládat data mezi webovými požadavky. Oproti cookies nemá tak přísné omezení velikosti a nepřenáší se při každém požadavku na server a zpět (10).

Web Workers – umožňuje webovým aplikacím provádět operace na pozadí s využitím samostatných vláken (10).

1.5.2 CSS

CSS (Cascading Style Sheets, česky Kaskádové styly) je jazyk, který slouží k popisování vzhledu elementů jazyků HTML a XHTML. Byl vytvořen za účelem oddělení významu stránky (HTML) od jejího vzhledu (CSS). Jazyk CSS byl navržen organizací W3C (WorldWide Web Consortium). Aktuální dokončenou verzí je verze 2.1. Aktivně se pracuje na verzi 3 (11).

Šablony stylů se skládají z tzv. pravidel stylů. Jedno pravidlo se skládá ze selektoru (určuje element, který bude vybrán k přiřazení stylu) a deklarace jedné nebo více vlastností (11).

1.5.3 LESS

LESS (Lean CSS) je preprocesor jazyka CSS, který vznikl původně v jazyce Ruby. LESS kód se kompiluje do jazyka CSS rovnou při vývoji, při každém požadavku na serveru, případně na straně klienta pomoci jazyka JavaScript. Jak se stal jazyk LESS populárním, vznikly jak porty do ostatních jazyků, tak GUI programy, které LESS umí kompilovat (12).

LESS přináší mimo jiné tyto vlastnosti:
Proměnné – Je možné deklarovat proměnné ve tvaru `@blue-color: #5B83AD`. Hodnotou proměnné může být i jednoduchý výraz (12).

Mixins – Jsou to makra, která umožňují zapsat sadu pravidel a tu poté používat napříč kódem voláním ve tvaru `.rounded-border(10px).` Mixin může přijmout jeden nebo více parametrů (12).

Vnoření pravidel – Zjednodušuje zápisy dlouhých selektorů (12).

Výrazy – LESS umí počítat jednoduché matematické výrazy. Při počítání rozlišuje jednotky. Ve výrazech lze používat čísla, barvy a proměnné. (12)

1.5.4 PHP a Nette

PHP (Personal Home Page, později akronym proHypertext Preprocessor) je PERL skript vyvinutý v roce 1995. Původně sloužil pouze pro zobrazování počtu návštěvníků na webové stránce. Stal se velice oblíbeným a tak autor (Rasmus Lerdorf) přepsal kód do jazyka C a začal přidávat nové funkce. Mnozí považovali za oficiální debut jazyka až vydání PHP verze 4 v roce 2002. Ten přinesl

zdokonalené zacházení s prostředky, základní podporu objektového programování, podporu sessions, podporu šifrování a další funkce (13).

Aktuální verzi jazyka je verze číslo 5. Ta přináší nesmírné zdokonalení objektového programování, zavedla zpracování výjimek try/catch, zdokonalila zpracování řetězců a podporu XML a webových služeb (13).

Nette je open-source framework napsaný v jazyce PHP 5 využívající architekturu MVP (namísto Controlleru se používá pojem Presenter). Je vyvíjen organizací Nette Foundation od roku 2009. Nette přináší nástroje a postupy ke zkvalitnění, zrychlení a usnadnění vývoje webových aplikací v jazyku PHP (14).

Mezi výhody Nette framewroku mimo jiné patří:
Ladící nástroje – Nette obsahuje knihovnu s názvem Tracy. Tato knihovna dokáže přehledně a rychle detekovat chyby v kódu, dokáže vytvářet logy s chybami, měřit čas a další pomůcky k debuggování kódu (14).
Zabezpečení – Nette velmi dobře zvládá zabezpečení aplikace. Automaticky ošetřuje vstupy na webu a tím brání útoků Cross-Site Scripting. Dále chrání formuláře proti útokům CSRF a dalším typům útoků (14).
Komunita – Framework je vyvíjen skupinou českých vývojářů. Je tedy kolem něho velká komunita lidí, kteří dokážou poradit v češtině. Výhodou je také u nás velká poptávka po programátorech, kteří s tímto frameworkem pracují (14).
Moderní technologie – Framework využíva a usnaďnuje využití moderních technologií a postupů. Například AJAX a Dependency Injection (14).

1.5.5 SQL a MySQL

SQL (Structured Query Language) je dotazovací jazyk. Vznikl na základě projektu firmy IBM, jehož cílem bylo vytvořit jazyk pro práci s údaji v databázi, který by byl blízký angličtině. Postupně se k tomuto standardu přidávaly další firmy a vznikl tak „nepsaný standard“ databázového jazyka SQL (15).

Příkazy standardní množiny jazyka SQL lze rozdělit na:

- **Data Definition Language** – pomocí tohoto typu příkazů lze definovat, vytvářet, měnit a mazat objekty a struktury v relačních databázích. Je taktéž možné přidávat a odebírat uživatelská oprávnění uživatelům a skupinám. Do této skupiny patří například příkazy: CREATE DATABASE, CREATE TABLE, DROP TABLE, ALTER TABLE a další (15).
- **Data Manipulation Language** – do této skupiny příkazů patří příkazy pro manipulaci s údaji. Mezi tyto příkazy patří: SELECT, INSERT, UPDATE a DELETE (15).
- **Data Control Language** – do této skupiny patří speciální příkazy prořízení provozu a údržby databáze (15).

MySQL je relační databázový systém, který je založen na jazyku SQL. Je však rozšířen o další funkce a možnosti (16).

V MySQL je možné využít tří typů tabulek:

- **MyISAM** – představuje základní typ tabulek v MySQL. Jedná se o stabilní, jednoduše spravovatelný a vyspěly typ (16).
- **InnoDB** – novější typ, který plně podporuje všechny vlastnosti typu MyISAM, ale navíc má dvě vlastnosti. Databázové transakce se dají spouštět jako transakce a je zde podpora pro používání pravidel integrity. Avšak chybí zde podpora pro fulltextový index a tento typ ještě nedosáhl vysoké stability tabulek typu MyISAM (16).
- **HEAP** – tyto tabulky využívají jako úložiště pro data pouze paměti RAM. Jsou tedy vhodné pro malé objemy dat, ke kterým je potřeba rychlý přístup (16).

1.5.6 JavaScript a jQuery

JavaScript je programovací jazyk původně určený k programování stránek na straně klienta (například v prohlížeči). Syntaxe vychází z jazyka C. Začlenění

skriptu do zdrojového kódu HTML umožňuje tag <SCRIPT>, který umožňuje jak vložit JavaScriptový kód přímo do HTML, tak ho nahrát z externích souborů (17).

jQuery je knihovna funkcí napsaná v jazyku JavaScript, která, po nalinkování souboru s knihovnou do webové stránky, zpřístupňuje objekt jQuery (nebo jeho alias „$“). Tento objekt obsahuje řadu funkcí pro zjednodušení práce zejména v oblastech: (18)

- Vyhledávaní elementů na stránce a jejich práce s nimi (změna vlastností, změna tříd a identifikátoru, ...)
- Práci s elementy (vkládaní, mázání a přesouvání elementů v rámci stránky)
- Práci s událostmi (spouštění funkcí, po nastání určité události, například kliknutí, pohyb myši atd.)
- Animování elementů
- Zjednodušení AJAX požadavků

1.5.7 CoffeeScript

CoffeeScript je programovací jazyk, který se kompiluje do jazyka JavaScript. Vše co lze napsat v JavaScriptu lze napsat také v CoffeeScriptu. Vznikl v roce 2009 a je to tzv. syntaktický cukr. Tedy zjednodušuje práci programátorovi odstraněním zbytečných znaků (složené závorky, středníky) a dostupností nových funkcí pro práci s poli, objekty apod. Samotný kompilátor je dostupný v monha jazycích na straně serveru a dokonce v jazyku JavaScript v prohlížeči, kde se CoffeScript kompiluje během načítání stránky (19).

1.5.8 AJAX

AJAX (Asynchronous JavaScript and XML) je označení pro technologii sloužící pro vývoj webových aplikací, které mění obsah aniž by potřebovaly znovu načítat kompletní stránku. Není to konkrétní technologie, ale pojem pro společné využití několika již existujících technologií(HTML, CSS, JavaScript, XML) (20).

Hlavními výhodami jsou podstatné zrychlení uživatelského rozhraní, kdy uživatel nemusí čekat na nové načtení celé stránky a nižší nároky na webový server. Ten nemusí plýtvat výkonem na generování kompletní stránky, ale generuje pouze tu část, kterou je potřeba aktualizovat. Z toho vyplývá i úspora přenosové kapacity (20).

AJAX stále využívá protokol http, takže je stále omezen jeho vlastnostmi. Při velké intenzitě komunikace pomocí AJAXu se tedy může stát, že se komunikace zahltí a přestane fungovat. Další nevýhodou AJAXu se týká změny způsobu používání webové stránky. Web již nefunguje jako série stránek, mezi kterými se dá přepínat, ale spíše jako aplikace s vnitřní logikou. To znamená, že tlačítka Zpět a Další v prohlížeči přestanou fungovat. To se dá však napravit užitím například hashtagu (#) v URL adrese, případně pomocí HTML5 API (20).

1.5.9 Node.js

JavaScript byl jazyk původně určený pro psaní skriptů na straně klienta, většinou v prohlížečích. V roce 2009 byl představen projekt s názvem Node.js, který umožňuje využívat JavaScript na straně serveru. To je umožněno využitím V8 JavaScript engine. Je to prostředí určené pro kompilaci a spouštění JavaScriptových zdrojových kódů vyvinuté společnosti Google původně pro prohlížeč Google Chrome. Node.js je tedy skládá z V8 enginu a několika knihoven (výstup do konzole, práce se soubory a další) (21).

Node.js využívá stylu programování řízeného událostmi (event-driven). Tím je umožněno asynchronního zpracování příkazů. Program při spuštění funkce, například dotaz na databázi, nečeká na výsledek, ale pokračuje dál ve vykonávaní další příkazů. Taková funkce tedy nevrací žádný výsledek (jelikož program na ní nečeká), ale přijímá jako parametr funkci (callback), která bude zavolána po dokončení dotazu (21).

NPM, Node Package Manager, způsob pomocí kterého lze jednoduše přidávat novou funkcionalitu pro Node.js ve formě balíčku, které jsou volně dostupné v open-source licencích. NPM využívá souboru package.json, ve kterém se

definují, názvy modulů a jejich verze a jejich závislosti na ostatních modulech. NPM se poté již automaticky postará o stažení všech potřebných modulů (21).

2 Analýza současné situace

Doplněním teoretických východisek o analytickou část získám veškeré podklady pro vytvoření konkrétního návrhu modulu.

Po představení společnosti a samotného produktu zpracuji analýzy daného produktu z uživatelského a technického hlediska. Tyto dvě analýzy budou sloužit jako základ pro vytvoření prototypu modulu a návrhu uživatelského rozhraní. Společně s analýzou konkurence je výsledkem těchto analýz seznam funkcí, který zaručí, že modul bude konkurenceschopný na reálném trhu.

2.1 Představení společnosti

Obchodní firma: Smartsupp.com, s.r.o
Sídlo: Milady Horákové 1957/13,Černá Pole, 602 00 Brno
IČO: 03668681
Právní forma: Společnost s ručení omezeným

Společnost Smartsupp, s.r.o. je poměrně mladá firma, která byla založena 23. Prosince 2014. Jednatelem společnosti je pan Petr Janošík. Společníky jsou Dušan Kmeť a Vladimír Šandera.

Společnost vyvíjí a provozuje produkt Smartsupp. Tento produkt je vyvíjen již od roku 2012. Před tím, než byla vytvořena společnost Smartsupp, s.r.o., byl produkt provozován na živnostenský list jednoho ze zakladatelů.

2.2 Představení produktu

Smartsupp je služba, která poskytuje majitelům webových stránek, e-shopům a aplikacím možnost komunikovat s návštěvníky v reálném čase. To je umožněno umístěním chatovacího okénka na webovou stránku majitele. Na jedné straně tedy komunikuje návštěvník webu pomocí chatovacího okénka umístěném na

webových stránkách majitele a na druhé straně komunikuje majitel webové stránky v prostředí aplikace Smartsupp.

Produkt funguje na principu SaaS, tedy Software poskytovaný jako služba. Zákazník (majitel webové stránky) si nekupuje hotové, krabicové řešení za vysokou částku, ale pronajímá si produkt za měsíční paušál. Výše měsíčního paušálu se odvíjí podle zvoleného balíčku a podle uživatelů, kteří budou Smartsupp obsluhovat.

Společnost tento produkt cílí na globální trh. Webová stránka je přeložena do 14 jazyků včetně například čínštiny nebo hindštiny. Hlavní prostředí produktu, Dashboard, je přeložen do 12 jazyků. Podle interních statistik společnosti je zhruba 50% uživatelů z České a Slovenské republiky a 50 % z ostatních zemí.

Výhodou pro návštěvníky webu je možnost okamžité možnosti komunikace s podporou webové služby nebo obchodu. Zákazník nemusí dohledávat telefonní čísla nebo emaily. Stačí na jakékoliv stránce webu kliknout na banner chatu a začít komunikovat.

Výhodou pro majitele webových stránek je zejména zkvalitnění podpory svých produktů a služeb. Smartsupp dokáže odesílat speciální události do služby Google Analytics. Pomocí toho je možné změřit chování návštěvníků, pokud chat používají nebo ne. Je dokázáno, že pokud chatovací okénko na webu je, tak návštěvníci stráví delší čas na webu a projdou více stránek. Další výhoda pramení ze způsobu distribuce produktu. Majitel webu nemusí instalovat žádný software na svůj server. Ke zprovoznění produktu stačí vložit na webovou stránku JavaScriptový kód. Tento kód je univerzální a funguje na všech platformách a ve všech prohlížečích.

Možnost komunikace zákazníků s podporou v reálném čase je opravdu důležitá. To si začínají uvědomovat i velcí hráči na tuzemském internetovém trhu a začínají

využívat této služby. Mezi zákazníky služby Smartsupp například patří Alza.cz, Kasa.cz nebo Škoda Auto.

Smartsupp původně začínal jako produkt pro komunikaci s návštěvníky v reálném čase. Stále jsou však vyvíjeny nové funkce. Jednou z těchto funkcí je možnost přehrávaní chování návštěvníku na webových stránkách. Aniž by majitel webové stránky musel instalovat nový software, nebo přidávat další kód do zdrojového kódu, je mu umožněno, po aktivaci této funkce, přehrávat seance jednotlivých návštěvníku na webu. V nahrávce lze vidět odkud a kam návštěvník přechází v rámci webu, kam posouvá ukazatelem, kam kliká a další interakce s webovou stránkou.

Produkt Smartsupp je nabízen v několika balíčcích, které se liší v ceně a funkčnosti. Každý balíček je možné objednat na 1, 3, 12 nebo 24 měsíců. Čím delší je objednané období, tím je vyšší procentuální sleva na celkovou částku.

Mezi nabízené balíčky patří (uvedená cena je platná k datu psaní této práce a je uvedena měsíční platba při objednání na 12 měsíců za jednoho operátora):

- FREE – základní balíček, který je poskytován zdarma. Návštěvníky může obsluhovat pouze jeden operátor, který může vést pouze jednu souběžnou konverzaci. Historie konverzací je omezená na 14 dní.
- MINI – 99 Kč/měsíc. Tento balíček nabízí neomezený počet souběžných konverzací, umožňuje vlastni nastavení vzhledu chatovacího okénka a umožňuje prohlížet video nahrávky návštěvníků.
- STANDARD – 199 Kč/měsíc. Tento balíček navíc umožňuje využití JavaScript API pro pokročilé nastavení chatovacího okénka, využití REST API pro snadnou správu účtů a omezený počet automatických zpráv, pomocí kterých si mohou operátoři zjednodušit používání produktu.
- PROFI – 499 Kč /měsíc. Tento balíček navíc umožňuje využití většího množství automatických zpráv, možnost zobrazení statistik jednotlivých operátorů a možnost využít skupiny operátorů. Pomocí této funkce je

možné nastavit více skupin operátorů, kteří obsluhují specifický segment návštěvníků například v závislosti na jazyku, obchodní sekci apod.

Každé objednání služby začíná 30-denní zkušební dobou. Tato zkušební doba je co se týče funkčnosti velmi podobná balíčku PROFI. Tato zkušební doba slouží k tomu, aby si zákazník vyzkoušel veškeré nabízené funkce produktu. Účtování za použití produktu začíná až po uplynutí zkušební doby.

Na začátku roku 2015 spustila společnost Smartsupp, s.r.o. privátní testování nové funkce produktu pro vybrané zákazníky. Tato funkce se jmenuje VisitorTrace a umožňuje majitelům webů sledovat nahrávky akcí, které návštěvník na jejich webu udělá. Tím se Smartsupp posouvá do další kategorie webových služeb. Kromě chatování v reálném čase nabízí i analytiku webových stránek. Tím se značně odlišuje od konkurence.

2.3 Popis produktu z uživatelského hlediska

2.3.1 Nasazení na web

Aby Smartsupp vůbec mohl fungovat, musí uživatel vložit na své stránky JavaScriptový kód. To je možné dvěma způsoby:

Manuálně – uživatel vloží na svůj web (nejčastěji na konec HTML stránky, před tag </body>) kód i se svým Smartsupp Chat ID manuálně.

Pomocí pluginů – uživatel může využít připravené moduly pro systémy Wordpress (i s podporou WooCommerce), Prestashop nebo OpenCart. Poté stačí jen modul nainstalovat a vložit své Smartsupp Chat ID, případně nastavit rozšířené funkce.

Pokud uživatel na svém webu správně nasadil Smartsupp kód, začne se jeho návštěvníkům zobrazovat Smartsupp Chat okénko. Uživatel zároveň v Dashboardu uvidí návštěvníky svého webu.

2.3.2 Dashboard

Dashboard je prostředí, pomocí kterého mohou operátoři obsluhovat návštěvníky na webu a komunikaci s nimi. Dashboard je rozdělen do těchto částí:

- **Návštěvníci** – hlavní obrazovka dashboardu. Jsou zde zobrazeni všichni návštěvníci webu, kteří jsou online. U každého návštěvníka jsou viditelné užitečné informace jako jméno, jak dlouho prochází web, aktuální stránka na které se nachází, počet jeho návštěv apod. Pro lepší přehlednost lze výpis těchto návštěvníků filtrovat pomocí několika kritérií.

  Zde operátoři vidí zahájené konverzace ze strany návštěvníků, případně mohou konverzaci sami iniciovat. Po rozkliknuti detailu návštěvníka může operátor vidět další informace. Mezi tyto informace patří historie procházení webu, soukromé poznámky operátorů, technické informace o návštěvníkovi (poloha, prohlížeč, operační systém), informace o firmě ve které se návštěvník pravděpodobně nachází a informace, které lze nastavit pomocí JavaScript API.
- **Historie** – Zde je možné dohledat veškeré konverzace mezi zákazníkem a operátorem a případně všechny zmeškané pokusy o konverzaci. Historie zpráv dle filtrovat pomocí textu, data proběhlé konverzace, obsluhujícího operátora, hodnocení konverzace a dalších atributů.
- **Statistiky** – V této části lze dohledat statistiky používání chatu. Jak celkové statistiky jako počet konverzací, průměrné hodnocení, průměrná doba první odpovědi, tak přehled těchto hodnot u jednotlivých operátorů. Tím ej možné sledovat jejich výkonnost.
- **Video záznamy** – toto je sekce nové funkce video nahrávek Zde může operátor, či majitel webu sledovat jednotlivé seance návštěvníků webu. Tyto záznamy je možné filtrovat pomocí velkého množství filtrů jako: URL adresy počátečních nebo koncových stránek seance, datum seance, délka seance, IP adresa návštěvníka, email návštěvníka a další.

Po rozkliknuti vybrané seance se zobrazí samotný přehrávač, který zobrazuje chování návštěvníka na webu. Přehrávač umožňuje pozastavit přehrávaní, zrychlit přehrávaní a přepínání mezi jednotlivými navštívenými stránkami během seance.

- **Správa** – V této sekci může vlastník účtu spravovat další operátory, vytvářet skupiny, automatické zprávy a offline e-maily.
- **Nastavení** – Zde je možné nastavit vzhled chatovacího okénka a jeho texty. Dále se zde nachází nastavení profilu v dashboardu, nastavení zvukových a obrazových notifikací a nastavení zkratek.

2.3.3 Chat okénko

Chat okénko je vztyčným bodem pro komunikaci s podporou z pohledu návštěvníka na webu. Kromě své primární funkce, tedy psaní zpráv v reálném čase, nabízí ještě tyto další funkce:

- Hodnocení operátorů – pokud je tato funkce aktivována, návštěvník má možnost ohodnotit po skončení konverzace s operátorem tuto konverzaci ohodnotit. Hodnocení probíhá pomocí grafické stupnice s hodnotami negativní, neutrální a spokojený. Tato hodnocení a jejich průměry se poté zobrazí ve statistikách operátorů v Dashboardu.
- Formulář před začátkem chatu – Pokud je tato funkce aktivována, je návštěvník nucen před začátkem konverzace vyplnit jednoduchý formulář. Políčka formuláře lze měnit z Dashboardu. Toto slouží například k tomu, aby návštěvník vyplnil své jméno a e-mail tak, aby konverzace s operátorem nebyla úplně anonymní. Případně se může jednat o vyplnění předmětu dotazu apod.
- Přepis konverzace – Po skončení konverzace si může návštěvník odeslat kompletní konverzaci na e-mail.

Vzhled a chování chatovacího okénka lze výrazně měnit pomocí Javascript API. Pomocí tohoto API lze například vkládat informace o návštěvníkoví z webové stránky do Dashboardu. V případě e-shopů se může jednat o informace týkající se obratu návštěvníka, počtu objednávek apod. Operátor poté vše vidí v průběhu

konverzace. Dále je pomocí API možné nastavit vlastni vzhled, obrázky a případně pozměnit chování okénka a reagovat na něj pomocí napojení na události.

2.3.4 Správa účtu

Správa Smartsupp účtu probíhá v modulu Můj účet aplikace Smartsupp. Zde je možné měnit, případně prodloužit zvolený balíček. Zde se také nachází vystavené faktury a online platební brána pro platbu paušálu za užívání produktu.

K dispozici je také rozsáhlé REST API pomocí kterého lze celý učet, nastavení a správu ovládat programově.

2.4 Popis produktu z technického hlediska

Celý produkt Smartsupp je rozdělený na několik modulů. Každý tento modul je spravován pomocí vlastních repositářů. Mezi hlavní moduly patří:

- Mini-chat
- Web-socketový server
- Dashboard
- Webová aplikace

Mini-chat se stará o zobrazování chatovacího okénka v prohlížečích návštěvníků. Je napsána v Javascriptu a integruje se do stránky umístěním Smartsupp kódu na libovolné místo na stránce. Uživatelské rozhraní mini-chatu je generované opět přes Javascript, což umožňuje velkou míru flexibility při manipulaci s chatem. Jelikož je mini-chat nahráván při každém načtení webové stránky, je nutné aby jeho velikost byla co nejmenší. Toho je docíleno minifikací zdrojového kódu a využitím asynchronního načtení skriptu.

Mini-chat se po načtení připojuje na **web-socketový server**. Tento server je napsán pomocí technologie Node.js. Server spravuje připojené návštěvníky (mini-chat) a přeposílá zprávy mezi operáty v dashboardu a mini-chatem. Mimo přeposílání samotných správ má server přeposílá další informace spojené s chatováním. Například „typing“ notifikace (notifikuje pokud někdo

v komunikaci zrovna píše zprávu), změna stavu operátorů (online, away, offline), změna údajů o návštěvníkovi (většinou pokud přejde na novou stránku). Server má taktéž na starosti sledování aktivity návštěvníků a spouštění automatický událostí (například automatické zprávy).

VisitorTrace, funkce pro nahrávaní návštěvníků na webu, je implementovaný jako modul do aplikace Smartsupp. Pokud je tento modul aktivní, mini-chat při načítání aktivuje funkci záznamu. Tato funkce poté sleduje akce návštěvníka a změny, které se dějí v HTML kódu stránky. VisitorTrace server umožňuje spravovat data pomocí rozšíření. Základním rozšířením je funkce ukládaní dat na disk.

Webová aplikace je aplikace napsaná v PHP za pomocí frameworku Nette. Tato aplikace má na starost samotnou webovou prezentaci produktu (smartsupp.com, nápověda, dokumentace) a také správu uživatelů aplikace a jejich balíčků (fakturace, prodloužení, zrušení). Webová aplikace také ukládá do databáze data, která operátor může nastavit v dashboardu aplikace (automatické zprávy, nastavení skupin atd.)

2.5 Zadání a požadavky společnosti

Během testování funkce VisitorTrace dostala společnost zpětnou vazbu ve formě požadavků rozšíření funkcionality o další způsoby analytiky webových stránek. Jednou z nejčastěji požadovaných funkcí byla možnost zobrazovat interakce návštěvníků na webových stránkách pomocí teplotních map.

Zadáním společnosti je tedy vytvoření prototypu modulu pro vytváření teplotních map z dat získaných z interakcí návštěvníka na webu. Funkcionalita má být vybrána na základě analýzy poskytovaných funkcí konkurence. Prototyp by měl mít rozhraní pro otestování funkčnosti a měla by být navrhnut grafický návrh rozhraní modulu v dashboardu aplikace.

2.6 Analýza konkurence

Účelem této analýzy je zjistit rozsah poskytovaných služeb a nabízených funkcí konkurenčních služeb v oblasti poskytování teplotních map pro analýzu webové stránky. Na základě této analýzy poté vyberu rozsah funkčnosti pro vytvářený prototyp modulu. Služby jsem vybral na základě výsledku vyhledávání po zadání klíčového slova.

2.6.1 heatmap.me

Tato služba se zaměřuje pouze na poskytování analýzy interakcí formou teplotních map. Služba je poskytována formou SaaS na bázi měsíčního paušálu. Nabízí 3 balíčky (Free, Premium a Enterprise) lišící se cenou, omezením počtu návštěvnosti webu a množstvím podstránek na webu.

Služba nabízí zobrazení jednoho typu interakcí a to kliknutí myši. Dále podporuje stránky tvořené pomocí responzivního webdesignu a možnost sledování map v reálném čase.

2.6.2 crayzegg.com

Tento produkt se kromě zobrazování teplotních map také zabývá dalšími způsoby zobrazení interakcí návštěvníků. Pomocí teplotních map zobrazuje dvě interakce, kliknutí myši a posun na webové stránce (scroll). Dále nabízí funkci Overlay, která zobrazuje počty kliknutí podle HTML elementů na stránce a funkci Confetti, podle niž lze identifikovat jednotlivé kliky pomocí zdrojů návštěvnosti apod.

Produkt je opět poskytován formou služby za měsíční předplatné. Je rozdělen do 4 balíčků lišících se podle ceny, návštěvnosti a velikosti webu.

2.6.3 Inspectlet.com

Primární funkcí této služby je poskytování nahrávek akcí návštěvníků na webu. Doplňující funkci jsou teplotní mapy. Teplotní mapy poskytuje pro tři druhy interakcí: pohyb kurzoru, kliknutí a posun stránky. Služba neumí zpracovat responzivní webové stránky. Tento produkt také nabízí filtrování webových adres pomocí „náhrady části řetězce“ (wildcards).

Produkt je opět provozován formou SaaS za měsíční poplatek. Je poskytována v 6 různých balíčcích lišících se cenou, počtem webových stránek a počtem nahraných nahrávek.

2.7 Závěr analýzy

Z analýzy funkčnosti konkurence vyplynulo to, jaký rozsah funkcí by měl modul poskytovat tak, aby byl konkurenceschopný.

Funkce modulu jsem vybral tyto:

- zaznamenávaní 2 typů interakcí (kliknutí, posun stránky) s možností rozšíření
- podpora responzivního webdesignu pro tři typy zařízení (mobil, tablet, desktop)
- možnost pokročilé filtrace URL adres pomocí tzv. „wildcards“, které zastupují vybranou část adresy

Všechny tyto části by podle výše uvedené analýzy měly být konkurenceschopné a zároveň uspokojí požadavky zákazníků.

3 Návrh řešení

Tato kapitola se zabývá návrhem prototypu modulu aplikace Smartsupp pro zobrazování teplotních map. Je rozdělena do čtyř částí:

- návrh způsobu pro sběr a ukládaní dat potřebných pro zobrazování teplotních map
- návrh uživatelského rozhraní prototypu pro snadné otestování funkčnosti
- grafický návrh uživatelského rozhraní zasazeného do aplikace Smartsupp
- návrh rozšíření funkcionality modulu nad rámec požadavků společnosti

3.1 Sběr a ukládaní dat

Sběr a ukládaní dat probíhá v aplikaci Visitortrace. Tato aplikace je napsaná v Node.js za pomocí jazyka CoffeeScript. Od začátku byla navrhnuta tak, aby bylo možné její funkcionalitu rozšířit pomocí modulů. To je docíleno vysíláním *eventů* (událostí), které modul může odposlouchávat a zpracovávat.

Jsou to tyto události:

- record.start(record)
- record.write(record, html, events)
- record.close(record)
- record.persist(record, success)

3.1.1 Inicializace modulu

Vytvářený modul, nazvaný jako **HeatmapExtension,** se nachází v adresáři *visitortrace-ext-heatmaps* v základním adresáři aplikace. V adresáři se nachází dva soubory:

- **index.coffee** – tento soubor obsahuje a zpřístupňuje metodu *inject,* která definuje službu *visitortrace.ext.heatmap*. Tato služba vrací instanci třídy *HeatmapExtension*, které je zároveň předána služba *visitortrace.application*

- **HeatmapsExtension.coffee** – obsahuje třídu *HeatmapExtensions*, která obsahuje veškerou logiku modulu.

Aby byla inicializace modulu kompletní, je potřeba upravit soubor VisitortraceServer.js. Toto je základní skript celé aplikace. V tomto souboru je potřeba napřed zavést novou službu a poté ji spustit.

3.1.2 Události a jejich parametry

Modul pracuje s událostmi vysílanými aplikací. Zde je výčet těchto událostí a jejich parametrů.

Události:

- **Record.start** – tato událost nastává, když se návštěvníkovi na webu zobrazí nová stránka. Návštěvník tedy může přijít na stránku z externího odkazu, může zadat její adresu do adresního řádku nebo pouze aktualizuje stránku, na které se zrovna nachází.
- **Record.write** – tato událost nastává pokud návštěvník inicializuje nějakou akci. Touto akcí může být pohyb myší, kliknutí apod. Událost nevzniká ihned po vykonání akce, ale vzniká po dávkách v určitém intervalu. To znamená, že může předávat více událostí zároveň.
- **Record.close** – tato událost nastává, když návštěvník opustí aktuální stránku. To znamená, že například přejde na jinou stránku téhož webu, odejde z webu úplně, případně zavře okno prohlížeče. Po této události nenásledují žádné události *record.write*.
- **Record.persist** – tato událost nastává, když jsou všechna data uložena do databáze.

Parametry:

- **record** – [OBJECT] – je to objekt popisující základní parametry během nahrávaní návštěvníka na webu. Pro tento modul jsou využity parametry *domain*, *pageUrl* a *screenWidth*
- **html** – [OBEJCT] – parametr html obsahuje objekt reprezentující HTML stránku a změny, které nastaly v rámci interakce návštěvníka se stránkou.
- **events** – [ARRAY] – pole objektů reprezentující události, které návštěvník na stránce vykonal. Události se liší podle parametru *name* a každá událost je objekt obsahující další informace.
- **success** – [BOOL] – tento parametr má hodnotu TRUE, pokud veškerá data byla správě uložena do databáze a na disk. V opačném případě má hodnotu FALSE.

3.1.3 Schéma databáze

Modul využívá dvě tabulky v databázi. Tyto dvě tabulky jsou nezávislé na ostatních tabulkách aplikace, jelikož prototyp modulu nevyžaduje napojení na tabulky aplikace. Těmito tabulkami jsou: *Heatmap* a *HeatmapScreenshots***.**

Tabulka Heatmap

Tabulka *Heatmap* je hlavni tabulka modulu. Budou v ní uložena veškerá data potřebná pro vykreslení teplotní mapy.

Struktura tabulky je následující:

- **id** – [INTEGER] – identifikátor, primární klíč tabulky
- **type** – [ENUM] – typ uložených událostí. Možné hodnoty jsou `click` a `scroll`.
- **values** – [TEXT] – hodnoty události. Uložené jako JSON objekt převedený na řetězec (serialize).
- **domain** – [VARCHAR] – doména webové stránky, ze které byla data uložena.

- **relativePath** – [VARCHAR] – relativní adresa stránky. URL adresa bez hodnoty `domain`.
- **device** – [ENUM] - typ zařízení. Možné hodnoty jsou mobile, tablet a desktop.
- **createdAt** – [TIMESTAMP] – čas vytvoření dat

Tabulka HeatmapScreenshots

Tabulka *HeatmapScreenshots* obsahuje informace o uložených obrázcích (screenshotech) webových stránek.

Struktura tabulky je následující:

- **id** – [INTEGER] – identifikátor, primární klíč tabulky
- **domain** – [VARCHAR] – doména webové stránky, ze které byla data uložena.
- **relativePath** – [VARCHAR] – relativní adresa stránky. URL adresa bez hodnoty `domain`.
- **createdAt** – [TIMESTAMP] - – časová hodnota posledního vytvoření nebo aktualizování obrázku.

3.1.4 Sběr dat

Pro sběr dat jsou využity dvě události. *Record.write* a *record.close*. Tyto události jsou obsluhovány pomocí metod *onRecordWrite()* a *onRecordClose()* ve třídě *HeatmapsExtension*.

onRecordWrite()

Hlavním cílem teto metody je projít všechny přijaté události z pole *events* a vyfiltrovat pouze události, které chceme zaznamenat. U každé této vyfiltrované události poté uložit potřebné informace. Událostmi, které potřebujeme vyfiltrovat a potřebnými informacemi jsou:

- **„click“** – události s tímto názvem označují událost kliknutí myši. Zde jsou potřeba uložit hodnoty *options.pageX* a *options.pageY*, které uchovávají souřadnice, kde na stránce bylo kliknuto.
- **„scroll“** – události s tímto názvem označují událost posunu obsahu v okně prohlížeče (pomocí kolečka, klávesových šipek, posuvníku atd.). Ukládanou hodnotou v tomto případě bude maximální hodnota v pixelech, kterou návštěvník na stránce posunul počítanou od hodnoty 0, což je začátek stránky.

Data, která potřebujeme, se uloží do lokální proměnných c*lick* a *scrolls* podle typu událostí.

Po dokončení smyčky **for** jsou volány metody *addClicks()* a *addScroll()*. Tyto metody přijímají parametry *recordId* a *data*. Účelem těchto metod je uložit data z lokálních proměnných (*clicks*) do proměnných instance (*@clicks*). To je potřeba, protože po každém zavoláním metody *onRecordWrite*() se lokální proměnné *click* a *scrolls* vymažou.

onRecordClose()

Tato metoda napřed volá metodu *@checkScreenshot*, která kontroluje a případně vytváří screenshoty webové stránky. Vytváření screenshotů je pospáno v kapitole 3.6.

Poté kontroluje, zda návštěvník má požadované rozlišení. Pokud ano, nastaví příznak *saveData* na TRUE a příznak *device* na hodnotu v závislosti na rozlišení (mobile, tablet, desktop).

Pokud návštěvník nemá požadované rozlišení, data se z proměnných *@clicks* a *@scrolls* vymažou, jelikož nejsou potřebná. V případě opaku, se data připraví na vložení do databáze. A uloží jako objekt.

Objekt který se zde vytváří vypadá následovně:

```
{
     type: 'click',
     values: JSON.stringify(@clicks[record.id]),
     domain: record.domain,
     relativePath:record.pageUrl.slice(record.domain.le
     ngth),
     screenWidth: record.screenWidth
}
```

Jak lze vidět, struktura objektu souhlasí se strukturou tabulky *Heatmaps*. Takto vytvořený objekt, reprezentující jeden řádek tabulky, se přidá do pole s názvem *@cache* pomocí metody *push*();

3.1.5 Ukládaní dat do databáze

Zatím se data přesouvala pouze z lokálních proměnných *clicks* a *scrolls* a přes proměnné instance až po připravená data uložena v poli *@cache*. Zatím se žádná data neukládala do databáze.

Pokud by data byla ukládána v metodách *onRecordWrite()* nebo *onRecordClose()*, znamenalo by to velkou zátěž pro databázi způsobenou vysokým počtem požadavků na zápis dat. Zápisy dat by byly přímo úměrně závislé na počtu návštěvníků a počtu vykonaných akcí.

Proměnná *@cache* funguje jako jednoduchá *cache*. Díky této proměnné je možné ukládat data v konstantních intervalech nezávisle na počtu uživatelů.

Toho je docíleno voláním funkce *setInterval()* v kontruktoru třídy *HeatmapExtension*. Funkce je tedy poprvé spuštěna ihned po inicializaci modulu a poté po uplynutí určitého intervalu (nyní nastaveného na 10 sekund),

zkontroluje, zda v se proměnné *@cache* nachází nějaká data. Pokud ano, tak je dávkově uloží do databáze a obsah proměnné vymaže.

3.1.6 Vytvoření screenshotů

Aby byla teplotní mapa kompletní, jsou potřeba nejen samotná data, ale také podklad, nad kterém budou data vykreslena. V tomto případě se jedná o screenshot (obrázek) webové stránky. K tomu slouží metoda *@checkScreenshot()*.

Napřed se v tabulce *HeatmapsScreenshots* vyhledá záznam podle sloupců *domain* a *relativePath*, které vytvářejí URL adresu aktuální webové stránky. Pokud záznam existuje, screenshot pro aktuální webovou stránku je již vytvořen a uložen. Aby se screenshot nevytvářel při každém zavoláním metody *onRecordClose()* a tím se zbytečně nevyužíval výkon serveru na zbytečné operace, je použita časová podmínka. Tato podmínka kontroluje hodnotu sloupce `createdAt`, která označuje čas pořízení screenshotu. Pokud je screenshot starší než, v tomto případě, 7 dnů nebo pokud záznam v tabulce vůbec neexistuje, pokračuje program dále ve vytváření screenshotu.

Pokud se program nachází v tomto kroku, znamená to, že buď screenshot pro aktuální webovou stránku vůbec neexistuje, nebo je ho potřeba aktualizovat. Pro obojí se využije baliček **webshot**, který je stažen pomocí Node Package Manageru (npm). Tento modul využívá knihovny phantomJS, což je bez-hlavičkový prohlížeč s jádrem webkit, který je možné spouštět z terminálu a je možné ho skriptovat pomocí JavaScriptu. Modul webshot zpřístupňuje aplikaci metodu s názvem *webshot(url, filePath, options, callback)*. Tato metoda přijímá 4 parametry:

- **url** – URL adresa webové stránky, z které chceme vytvořit screenshot. Alternativně je možné použit místo URL adresy HTML kód stránky.
- **filePath** – relativní cesta souboru včetně jeho názvu, do kterého se screenshot uloží.

- **options** – objekt, který obsahuje konfiguraci pro vytváření screenshotů. Mezi důležité hodnoty konfigurace patří:
    - *screenSize* – šířka a výška vytvořeného okna prohlížeče phantomJs
    - *shotSize* – šířka a výška pořízeného screenshotu. Pokud se výška nastaví jako ‚all', je pořízen obrázek webu v celé jeho výšce.
    - *renderDelay* – spoždění v milisekundách, po kterém se screenshot vytvoří.
    - *onLoadFinished* – funkce, která se vykoná po načtení webové stránky.
- **callback** – metoda webshot funguje asynchronně. Program tedy nečeká až se screenshot vytvoří. Callback je funkce, která je zavolána právě potom, co je screenshot vytvořen.

Funkce, která slouží jako callback, poté uloží, případně aktualizuje záznam v tabulce *HeatmapScreenshots*. Jeden řádek v tabulce ve skutečnosti reprezentuje tři uložené obrázky pro tři různá zařízení (mobil, tablet, desktop).

Screenshoty se ukládají na pevný disk v adresářích nazvaných podle domény webové stránky ve tvaru [nazev_souboru]-[zařízení].jpg. Tyto adresáře poté budou umístěny v adresáři static/screenshots. Kompletní cesta bude vypadat následovně: /static/screenshots/[doména]/[nazev_souboru]-[zařízení].jpg. Aby byl název souboru v rámci adresářů jednotlivých webových stránek unikátní, je stejný jako hodnota sloupce id řádku reprezentujícího konkrétní sadu screenshotů v tabulce *HeatmapScreenshots*.

3.1.7 Responzivní webové stránky

Moderní weby jsou tvořeny pomocí responzivní techniky. To znamená, že na různých rozlišeních (důležitá hodnota v rozlišení je pouze ta, která určuje horizontální velikost rozlišení) se web zobrazuje v jiné podobě. V případě zobrazování teplotních map nastává problém. Není možné sbírat data všech návštěvníků, jelikož návštěvníci mohou mít různá rozlišení. Problém by nastal

v tom, že by naměřená data (souřadnice kliků) nekorespondovala s podkladem teplotní mapy, tedy se screenshotem webové stránky.

Řešením je rozdělit teplotní mapy do nezávislých kategorií. V tomto případě:

- Mobil
- Tablet
- Desktop

Pro každou kategorii se budou zvlášť vytvářet screenshoty a zvlášť sbírat data. Každé zařízení může mít více různých rozlišení. Je tedy potřeba vybrat pro každé zařízení rozlišení tak, aby pokrývalo co největší počet návštěvníků.

Dle interní databáze, které nashromáždil Smartsupp o návštěvnicích, jsem vybral tato rozlišení:

- Mobil – 360 px
- Tablet – 920 px
- Desktop – 1366 px

Hodnoty rozlišení jsou uloženy v polích *@desktopResolutions*, *@tabletResolutions* a *@mobileResolutions*. Pomocí těchto polí je poté určeno podmínkami určeno, zda se budou data konkrétního uživatele ukládat a s jakou hodnotou ‚device' se budou ukládat do databáze.

3.2 Uživatelské rozhraní prototypu

Aby se nasbíraná data dala jednoduše testovat a funkčně zkoušet, je v rámci prototypu vytvořené uživatelské rozhraní. Toto rozhraní prototypu se nachází v aplikaci **Visitortrace-App**. Jedná se o aplikaci napsanou v PHP a frameworku Nette. Uživatelská část modulu se bude skládat ze tří souborů:

- HeatmapPresenter.php – presenter modulu. Zpracovává formuláře a vykreslování šablony, které předává data načtená z databáze.

- Heatmap/default.latte – šablona modulu. Obsahuje HTML, CSS a Javscriptový kód, který se stará o vykreslení všech potřebných prvků. Přijímá data od presenteru.
- Heatmaps.php – v tomto souboru se nachází třída, která obsahuje metody pro skládání SQL dotazů a provádění těchto dotazů nad tabulkami Heatmap a HeatmapScreenshots

Uživatelská část prototypu modulu bude zpracovávat požadavky na server pomocí AJAXu. Tím se nebude vždy přenášet celá stránka, ale pouze obrázky a data. Jelikož toto rozhraní je vytvářeno pouze pro účely otestování modulu, není potřeba řešit autentizaci uživatelů.

3.2.1 Presenter HeatmapPresenter.php

Presenter HeatmapPresener obsahuje 3 metody:

- renderDefault() – zpracuje požadavek na vykreslení stránky a vykreslí šablonu *default.latte*. Nepřenáší žádná data. Při prvním načtení stránky se nezobrazí mapa, ale pouze filtrovací formulář, pomocí kterého lze data načíst.
- createComponentHeatmapControls() – tato metoda vytváří formulář pro filtraci dat, který se poté vykreslí v šabloně *default.latte*.
- actionLoadMapData() – metoda, která je volána, když je formulář odeslán. Tato metoda je volána asynchronně pomocí technologie AJAX. Zpracuje data z formuláře a zavolá metodu *heatmaps->getData(),* která vrátí veškerá data potřebná pro vykreslení mapy. Ty jsou poté odeslány pomocí metody *sendResponse()* ve formátu JSON.

3.2.2 Třída Heatmaps

Tato třída slouží k dotazování se databáze. Obsahuje dvě metody:

getData()

Tato metoda shromažďuje data z databáze a vrací pole, které poté presenter přeposílá šabloně. Napřed se v metodě nastaví výchozí hodnoty pro některé parametry. Pokud není zadán parametr *$domain*, metoda rovnou vrací prázdné pole. Pokud parametry *$relativePath* a *$device* nebyly ve formuláři zadány, nastaví se zde na hodnoty „/“ respektive „desktop“.

Poté se zavolá první dotaz na databázi, konkrétně na tabulku *HeatmapScreenshots*. Zde se na základě sloupečků ‚domain‘ a a ‚relativePath‘ pokusí najít záznam. Pokud záznam neexistuje, znamená to že screenshot webu neexistuje a metoda opět vrací prázdné pole. Pokud záznam existuje, jeho id (spolu s typem zařízení) se rovná názvu souboru s obrázkem webové stránky. Vytvořená cesta k souboru se uloží do pole *$data* s klíčem „imageUrl“.

Poté se volá metoda *buildQuery()*. Tato metoda vrací pole, které slouží jako dotaz pro funkci *query(),* která vytvoří dotaz na tabulku *Heatmaps*, konkrétně na sloupeček ‚values‘. Pokud výsledek dotazu není prázdný, projde se pomocí smyčky *foreach* a pomocí funkce *json_decode()* se převede s řetězce na proměnnou a uloží do pole *$data* s klíčem „values“. Metoda poté už jen vrátí pole $data.

buildQuery()

Tato metoda přijímá potřebné parametry filtrů a určuje jejich výchozí hodnoty. Poté na základě těchto hodnot vytvoří SQL dotaz, který vrací jako pole. Zde dochází ke zpracování hodnoty *relativePath*, tedy relativní cesty URL adresy. Pomocí funkce *str_replace*()i se zamění znak hvězdička (*) za znaky tečka a hvězdička (.*). To je potřeba pro správné fungování SQL dotazu REGEXP. Tím je zajištěna pokročilá možnost filtrování URL adres.

3.2.3 Šablona default.latte

Základní šablona stránky. Vykresluje filtrovací formulář a teplotní mapu. Dále obsahuje jQuery kód, který zpracuje odeslání formuláře a pomocí funkce *post(),* zavolá asynchronně metodu *actionLoadMapData()* v presenteru *HeatmapPresenter*.

3.2.4 Formulář s filtry

Druhou části uživatelské části prototypu je formulář, který umožňuje filtrovat a zobrazovat konkrétní teplotní mapy. Formulář obsahuje tyto prvky:

- Domain – doména webové stránky, kterou chce uživatel zobrazit. Toto políčko je zde pouze z testovacích důvodů. V ostrém provozu se doména zjistí automaticky na základě nastavení účtu zákazníka služby Smartsupp.
- Relative path – relativní adresa webové stránky, kterou chce uživatel zobrazit. Vždy začíná lomítkem. V případě uvedení pouze lomítka, se zobrazí hlavní stránka webu. Jednotlivé části relativní adresy lze nahradit znakem hvězdička (*). Tento znak nahrazuje část URL adresy. To je vhodné například pro zobrazení stránky kategorie elektronického obchodu. Obchod může mít mnoho kategorií s vlastní adresou na které se liší pouze produkty, nikoliv vzhled. Nahrazením konkrétního názvu kategorie hvězdičkou, lze vyfiltrovat právě všechny stránky všech kategorií.
- Platform – výběr ze tří možnosti zařízení: Mobil, tablet, Desktop
- From – datum, kterým začíná výběr
- To – datum, kterým končí výběr
- Show – tlačítko, které odesílá formulář na server jako AJAX požadavek

3.2.5 Vykreslení teplotní mapy

Vykreslení samotné mapy zajišťuje funkce (callback), která je zavolána, když nastane událost odeslání formuláře. Ta napřed pomocí funkce *preventDefault()*

zabrání formuláři v přesměrování či nového načtení stránky. Poté se do proměnných uloží hodnoty zadané ve formuláři.

Dále se zavolá jQuery funkce *post()*. Ta zajišťuje asynchronní zavolání URL adresy a předání parametrů. Jako první parametr je nastavena URL adresa, která na požadavek odpoví. Druhým parametrem je objekt se všemi proměnnými, ve kterých jsou uloženy hodnoty z formuláře. Posledním parametrem je funkce (callback), která je zavolána po úspěšně vykonaném požadavku na server.

V této funkci se napřed vytvoří objekt třídy *Image()*. Pomocí callbacku *onload()* zajistíme, aby se mapa načetla až po celkovém načtení screenshotu webové stránky. V tomto callbacku se napřed nastaví HTML elementu *#heatmap* vyška a šířka, která se rovná rozměrům screenshotu. Poté se nastaví CSS vlastnost *backgroundImage* na hodnotu odpovídající URL adrese screenshotu.

Samotná teplotní mapa se vykreslí pomocí skriptu simpleheat.js, který je dostupný na adrese https://github.com/mourner/simpleheat distribuovaného s open-source licencí. Tento skript vykreslí mapu pomocí HTML tagu <*canvas*>. Tento element je přímým potomkem elementu *#heatmap*. Skript zpřístupňuje funkci *simpleheat()*, která původně přijímala data ve formátu [[x, y, value], [x, y, value], ...], avšak pro potřeby prototypu jsem skript upravil a zjednodušil a akceptuje data ve formátu [[x, y], [x, y], ...]. Díky tomu, že v tomto formátu se data ukládají do databáze, není potřeba je transformovat.

Element *#heatmap* má dále nastavenou CSS vlastnost *zoom* na hodnotu 0,5. Tím je docíleno zmenšení obrázku a celkové mapy na 50%. Pokud by tato vlastnost nebyla použita, mapa by zakrývala celou plochu prohlížeče.

3.3 Návrh uživatelského rozhraní

Ačkoliv je aplikace VisitorTrace, která obstarává nahrávky návštěvníku a které je součástí tento modul, nezávislá na aplikaci Smartsupp, běžný uživatel toto nepozná. Nahrávky uživatelů se nyní nachází v hlavním prostředí aplikace v levém menu pod položkou Video záznamy.

Po položkou Video záznamy se nachází jednotlivé záznamy a možnost jejich filtrace. Po kliknutí na konkrétní záznam se otevře nové okno se samotným přehrávačem nahrávky. Jelikož teplotní mapy nepotřebují žádné složité ovládací prvky, otevíraní nového okna v tomto případě nebude nutné.

Aby byl zachována podobnost rozhraní a uživatelský zážitek jsem se rozhodl vytvořit rozhraní pro teplotní mapy se stejným rozložením, jako nyní mají Video nahrávky. V levé části okna se nachází možnost filtrace a v pravé časti (namísto seznamu nahrávek) se nachází samotné zobrazené teplotní mapy.

Aby uživatel nebyl zmaten velkým počtem položek v hlavním levém menu aplikace, rozhodl jsem se vytvořit novou položku s názvem VisitorTrace. Pod tuto položku se přesune tlačítko Video záznamy a rovněž se zde vytvoří tlačítko Teplotní mapy.

Grafický návrh uživatelského rozhraní se nachází v příloze číslo 1.

3.4 Návrhy na zlepšení

3.4.1 Zaznamenávání dalších událostí

Prototyp modulu momentálně zaznamenává dva typy událostí: kliky myší na stránce a rozsah posunu (skrolování) webové stránky. Lze tedy měřit účinnost aktivních prvků na stránce a rozsah toho, kam až se návštěvník dostane v rámci stránky.

Modul je na ukládání další typů událostí připraven. Postačí pouze v metodě *onRecordWrite()* v příkazu *switch* přidat další podmínku. Podmínka by kontrolovala název události s názvem ‚mousemove'. Poté by bylo potřeba vyřešit logiku spočívající především v tom, v jakém intervalu události vyhodnocovat a ukládat.

3.4.2 Responzivní weby

Momentálně se data ukládají pouze pokud návštěvník má nastavené konkrétní rozlišení prohlížeče. Tato rozlišení jsou tři (mobil, tablet, desktop) a jsou vybrána tak, aby pokryla co největší počet návštěvníků.

Responzivní weby se realizují pomocí určitého počtu *breakpointů*. Tedy velikosti šířky rozlišení, ve který se vzhled změní tak, aby více odpovídal velikosti zařízení. Většinou tyto *breakpointy* používají tři. Mezi těmito *breakpointy* se seskupení obsahu nemění a zvětšuje se pouze bílé místo na okrajích webové stránky.

Je tedy možné použít namísto konkrétních hodnot rozlišení intervaly pro každou skupinu zařízení. Bylo by potřeba změnit v modulu systém ukládaní souřadnic. Momentálně se souřadnice jednotlivých kliků počítají s počátkem v levém horním rohu prohlížeče a pokračují ve směru vpravo a dolů. Nově by se souřadnice počítaly s počátkem uprostřed nahoře v okně prohlížeče.

Nynější systém souřadnic:

- levý horní roh – [0px, 0px]
- pravý spodní roh [1366px, 2500px]

Nový systém souřadnic:

- levý horní roh – [-683px, 0px]
- pravý spodní roh [683px, 2500px]

Touto úpravou a úpravou skriptu vykreslujícího teplotní mapy by bylo umožněno zavést intervaly velikosti rozlišení pro jednotlivá zařízení, které by eliminovalo bílé místo na okrajích webových stránek a tím zvětšit počet měřených návštěvníků.

4 Závěr

Cílem této bakalářské práce bylo navrhnout prototyp modulu aplikace Smartsupp. Tento modul slouží k analýze webových stránek pomocí teplotních map na základě akcí návštěvníků webových stránek. Podnětem pro tuto práci byla zpětná vazba zákazníků společnosti Smartsupp, s.r.o., kteří tuto funkcionalitu vyžadovali.

Analýza se zaměřuje na popis současného stavu společnosti a samotného produktu. Produkt je dále zanalyzován jak z uživatelského, tak technického hlediska. Poslední část analýzy se zaměřuje na analýzu konkurence nabízející podobnou funkcionalitu. Na základě teto výsledků byl vytvořen návrh konkrétních funkcí modulu. Mezi tyto funkce patří například zaznamenávání více typů interakci, podpora responzivních webových stránek, možnost pokročilé filtrace dat a další.

Praktická část se zaměřuje na konkrétní návrh prototypu modulu. Tento návrh se skládá ze čtyř částí. První část se představuje návrh sběru a ukládaní konkrétních dat uživatelů. Druhá část se zaměřuje na vytvoření uživatelského rozhraní pro potřeby testování modulu. Třetí část popisuje návrh uživatelského rozhraní, tzn. jak by mohlo vpadat v praxi. Poslední částí je několik návrhů, jak rozšířit funkcionalitu modulu v budoucnu.

Seznam použitých zdrojů

(1) VELTE, Anthony T, Toby J VELTE a Robert C ELSENPETER. *Cloud Computing: praktický průvodce*. Vyd. 1. Brno: Computer Press, 2011, 344 s. ISBN 978-80-251-3333-0.

(2) WIKIMEDIA.ORG. Znázornění cloud computingu. Wikimedia.org [online]. [cit. 2015-05-09]. Dostupné z: http://commons.wikimedia.org/wiki/File:Cloud_computing.svg

(3) GoodyGoody. Responzivní webdesign – k čemu to je dobré? Goodygoody.cz [online]. [cit. 2015-05-09]. Dostupné z: <http://www.goodygoody.cz/2013/02/responzivniwebdesign-k-cemu-to-je-dobre/>

(4) STATCOUNTER.COM. Gs.statcounter.com [online]. [cit. 2015-05-09]. Dostupné z: http://gs.statcounter.com/#desktop+mobile+tablet-comparison-eu-monthly-201501-201503-bar

(5) MARCOTTE, Ethan. Responsive web design. New York: A Book Apart, 2011. 214 s. ISBN 978-0-9844425-9-1.

(6) BOHÁČKOVÁ, Klára. Teplotní mapy pro web: porovnání nástrojů a k čemu slouží. Lupa [online]. [cit. 2015-05-09]. Dostupné z: http://www.lupa.cz/clanky/teplotni-mapy-porovnani-nastroju-a-studie/

(7) FORGÁČ, Ján. Teplotní mapy (heat mapy). ARTweby.cz [online]. [cit. 2015-05-09]. Dostupné z: http://www.artweby.cz/blog/tepelne-mapy-heat-mapy

(8) CHEN, Mon Chu, John R. ANDERSON a Myeong Ho SOHN. What can a mouse cursor tell us more? *CHI '01 extended abstracts on Human factors in computing systems - CHI '01* [online]. New York, New York, USA: ACM Press,

2001, : 281- [cit. 2015-05-26]. DOI: 10.1145/634067.634234. ISBN 1581133405. Dostupné z: http://portal.acm.org/citation.cfm?doid=634067.634234

(9) BERNARD, Borek. Úvod do architektury MVC. Zdroják [online]. [cit. 2015-05-09]. Dostupné z: http://zdrojak.root.cz/clanky/uvod-doarchitektury-mvc/

(10) LUBBERS, Peter, Brian ALBERS a Frank SALIM. *HTML5: programujeme moderní webové aplikace*. Vyd. 1. Brno: Computer Press, 2011, 304 s. ISBN 978-80-251-3539-6.

(11) DOMES, Martin. *333 tipů a triků pro CSS*. 2., aktualiz. vyd. Brno: Computer Press, 2011, 272 s. ISBN 978-80-251-3366-8.

(12) MALÝ, Martin. LESS: stejné CSS za méně peněz. Zdroják [online]. [cit. 2015-05-09]. Dostupné z: http://www.zdrojak.cz/clanky/less-stejne-css-za-mene-penez/

(13) KOFLER, M. a B. ÖGGL. *PHP 5 a MySQL 5: průvodce webového programátora*. 1. vyd. Brno: Computer Press, 2007. 608 s. ISBN 978-80-251-1813-9.

(14) ČAPKA, David. Úvod do Nette frameworku pro PHP. ITnetwork.cz [online]. [cit. 2015-05-09]. Dostupné z: http://www.itnetwork.cz/uvod-do-php-frameworku-nette

(15) CONOLLY, T., C. E. BEGG a R. HOLOWCZAK. *Mistrovství - databáze: profesionální průvodce tvorbou efektivních databází*. 1. vyd. Brno: Computer Press, 2009. 584 s. ISBN 978-80-251-2328-7.

(16) STEPHENS, Ryan K, Ronald R PLEW a Arie JONES. *Naučte se SQL za 28 dní: [stačí hodina denně]*. Vyd. 1. Brno: Computer Press, 2010, 728 s. ISBN 978-80-251-2700-1.

(17) KOUT, Pavel. *Praktický JavaScript*. Vyd. 1. Brno: Zoner Press, 2004, 325 s. ISBN 80-86815-00-5.

(18) ŠERÝ, Richard. JavaScript s jQuery – lehký úvod. Interval [online]. [cit. 2015-05-09]. Dostupné z: https://www.interval.cz/clanky/javascript-s-jquery-lehky-uvod/

(19) YAHKEM. Úvod do CoffeeScriptu. ITnetwork.cz [online]. [cit. 2015-05-09]. Dostupné z: http://www.itnetwork.cz/coffeescript-uvod

(20) LACKO, Ľuboslav. *Ajax: hotová řešení*. Vyd. 1. Brno: Computer Press, 2008, 269 s. K okamžitému použití (Computer Press). ISBN 978-80-251-2108-5.

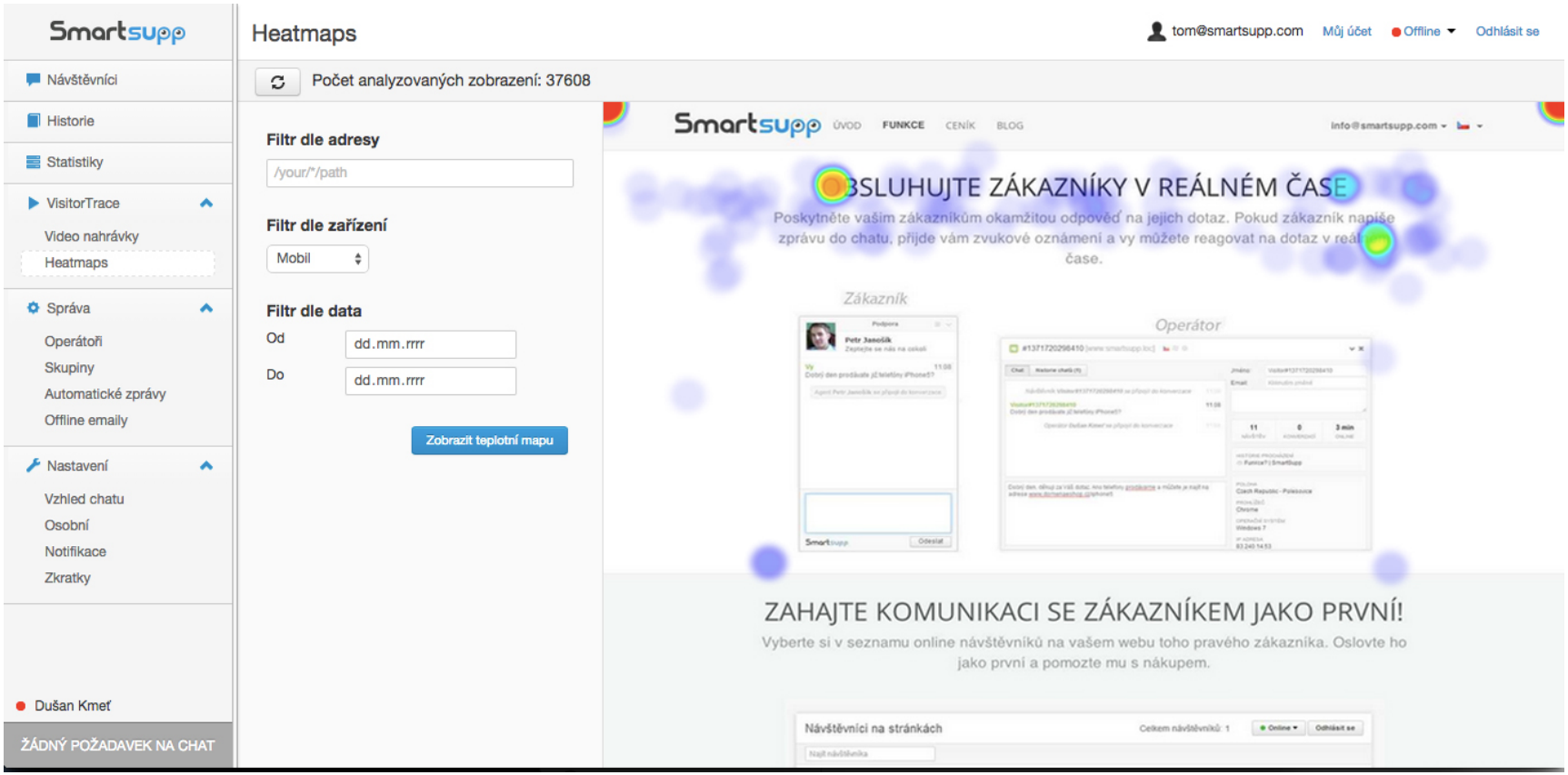
(21) TEIXEIRA, Pedro Dennis. *Professional node.js: building javascript based scalable software*. 1st ed. Indianapolis, IN: Wiley Pub., Inc., 2012, p. cm. ISBN 1118185463.

Seznam tabulek

Seznam obrázků

Seznam příloh

Příloha 1: Grafický návrh uživatelského rozhraní



Příloha 2: HeatmapExtension.coffee

```
"webshot": "0.15.4"

webshot = require "webshot"

class HeatmapExtension

     app: null
     database: null

     desktopResolutions: [1366]
     tabletResolutions: [920]
     mobileResolutions: [360]

     webshotMobileOptions:
          screenSize:
               width: 360
               height: 400
          shotSize:
               width: 360
               height: 'all'
          streamType: 'jpeg'
          quality: '50'
          renderDelay: 2000
          onLoadFinished:
               fn: (status) =>

     window.scrollTo(0,document.body.scrollHeight)

     webshotTabletOptions:
          screenSize:
```

```
                width: 920
                height: 1000
            shotSize:
                width: 920
                height: 'all'
            streamType: 'jpeg'
            quality: '50'
            renderDelay: 2000
            onLoadFinished:
                fn: (status) =>

        window.scrollTo(0,document.body.scrollHeight)

        webshotDesktopOptions:
            screenSize:
                width: 1366
                height: 1000
            shotSize:
                width: 1366
                height: 'all'
            streamType: 'jpeg'
            quality: '50'
            renderDelay: 2000
            onLoadFinished:
                fn: (status) =>

        window.scrollTo(0,document.body.scrollHeight)

        constructor: (@app, @database) ->
            @app.on 'record.start', (data) =>
    @onRecordStart(data.record)
```

```
          @app.on 'record.write', (data) =>
@onRecordWrite(data.record, data.html, data.events)
          @app.on 'record.close', (data) =>
@onRecordClose(data.record)
          @app.on 'record.persist', (data) =>
@onRecordPersist(data.record, data.success)

          # interval to insert to db
          setInterval(()=>
               if @cache.length > 0

                    @database.query 'INSERT INTO
Heatmap SET ?', @cache, (err, result) =>
                         if err then throw err
                         @cache = []

          , 10000)

          return

     # occours when recorder starts recording (called
before all write messages)
     onRecordStart: (record) ->

          return

     # occours when record data received from recorder
     onRecordWrite: (record, html, events) ->

          if   record.screenWidth in
@desktopResolutions or
```

```
                record.screenWidth in @tabletResolutions
or
                record.screenWidth in @mobileResolutions

                clicks = []
                scroll = []
                maxScroll = 0

                for e in JSON.parse(events)
                     switch e.name
                          when 'click'

     clicks.push([e.options.pageX, e.options.pageY])
                          when 'scroll'
                               if
e.target.body.scrollTop > maxScroll then maxScroll =
e.target.body.scrollTop


                @addClicks(record.id, clicks)
                @addScroll(record.id, maxScroll)

          return

     # occours when recorder stop recording (called
after all write messages, there will by not further
message)
     onRecordClose: (record) ->

          saveData = false
          device = ''
```

```
            domain = record.domain
            relativePath =
record.pageUrl.slice(record.domain.length)

            @checkScreenshots(record)

            if record.screenWidth in @desktopResolutions
                 saveData = true
                 device = 'desktop'

            if record.screenWidth in @tabletResolutions
                 saveData = true
                 device = 'tablet'

            if record.screenWidth in @mobileResolutions
                 saveData = true
                 device = 'mobile'

            if saveData

                 if @clicks[record.id].length > 0

                      cacheClicks = {
                           type: 'click',
                           values:
JSON.stringify(@clicks[record.id]),
                           domain: domain,
                           relativePath: relativePath,
                           device: device,
                           createdAt: Date.now
```

```
                    }

                    @cache.push cacheClicks


               if @scrolls[record.id].length > 0

                    cacheScroll = {
                         type: 'scroll',
                         values: @scrolls[record.id],
                         domain: domain,
                         relativePath: relativePath,
                         device: device,
                         createdAt: Date.now
                    }
                    @cache.push cacheScroll

          @clicks[record.id] = []
          @scrolls[record.id] = []

          return


     # occours when record was stored into database.
record should has property insertId as database id
     onRecordPersist: (record, isSuccess) ->

          return

     #putting all click together to one array of
objects
```

```
    addClicks: (recordId, data) ->
        clicks = @clicks[recordId] || []
        @clicks[recordId] = clicks.concat data
        return

    # check for website screenshots. If one doesn't
exist or is outdated, new screenshot is created and
saved both on disk and to database
    checkScreenshots: (record) ->
        @database.query 'SELECT * FROM
`HeatmapScreenshots` WHERE `domain` = ? AND
`relativePath` = ? LIMIT 1', [domain, relativePath ]
(err, result) =>
            if err then throw err

            if not result
                @database.query 'INSERT INTO
`HeatmapScreenshots` SET ?', [domain, relativePath,
Date.now] (err, result) =>
                    @createScreenshots(record,
result.recordId)

            else if Date.now - result.createdAt >
1000*60*60*24*7
                @database.query 'UPDATE INTO
`HeatmapScreenshots` SET ?', [domain, relativePath,
Date.now] (err, result) =>
                    @createScreenshots(record,
result.recordId)

        return
```

```
    #creates the actual screenshot
    createScreenshots = (record, id) =>
        filePathMobile = '/static/screenshots/' +
record.domain + '/' + id + '-mobile.jpeg'
        filePathTablet = '/static/screenshots/' +
record.domain + '/' + id + '-tablet.jpeg'
        filePathDesktop = '/static/screenshots/' +
record.domain + '/' + id + '-desktop.jpeg'

        webshot record.pageUrl, filePathMobile,
@webshotMobileOptions, (err) =>
            if err console.log err

        webshot record.pageUrl, filePathTablet,
@webshotTabletOptions, (err) =>
            if err console.log err

        webshot record.pageUrl, filePathDesktop,
@webshotDesktopOptions, (err) =>
            if err console.log err


module.exports = HeatmapExtension
```

Příloha 2: HeatmapPresenter.php

```
<?php

namespace App\AppModule;

use Tracy\Debugger;
use App\Heatmaps;
use Nette\Application\Responses\JsonResponse;

/**
 * HeatmapPresnter
 */
final class HeatmapPresenter extends BasePresenter
{

     /** @var \App\Heatmaps @inject */
     protected $heatmaps;

     protected function startup() {
          parent::startup();
     }


     public function renderDefault($sessionId = null,
$debug = null, $lang = 'en') {
     }

     public function actionLoadMapData() {

          $domain = $this->getHttpRequest()-
>getPost('domain');
```

```
        $relativePath = $this->getHttpRequest()-
>getPost('relativePath');
        $device = $this->getHttpRequest()-
>getPost('device');
        $from = $this->getHttpRequest()-
>getPost('from');
        $to = $this->getHttpRequest()->getPost('to');

        $data = $this->heatmaps->getData($domain,
$relativePath, $device, $from, $to);

        $this->sendResponse(new JsonResponse($data));
    }


    protected function
createComponentHeatmapControls() {

        $form = new \Nette\Application\UI\Form;

        $form->addText('domain', 'Domain')
            ->setAttribute('class', 'form-control')
            -
>setDefaultValue('rec.visitortrace.loc');

        $form->addText('relativePath', 'Relative
path')
            ->setAttribute('class', 'form-control')
            ->setDefaultValue('/tests/page.html');
```

```
          $form->addRadioList('device', 'Platform:',
array('mobile' => 'Mobil', 'tablet' => 'Tablet',
'desktop' => 'Desktop'));

          $form->addText('from', 'Od')
               ->setType('date')
               ->setAttribute('class', 'form-control');

          $form->addText('to', 'Do')
               ->setType('date')
               ->setAttribute('class', 'form-control');

          $form->addSubmit('submit', 'Show')
               ->setAttribute('class', 'btn btn-
default');

          $form->onSuccess[] = $this-
>heatmapFormSucceed;


          return $form;
     }

     public function
heatmapFormSucceed(\Nette\Application\UI\Form $form,
$values) {
     }

}
```

Příloha 3: Heatmaps.php

```
<?php
namespace App;
use Tracy\Debugger;

class Heatmaps
{
     private $connection;

     public function __construct(\DibiConnection
$connection) {
          $this->connection = $connection;
     }

     public function getData($domain, $relativePath,
$device, $from, $to) {
          $data = array();
          $values = array();

          if ( !$domain )
               return array();

          if ( !$relativePath )
               $relativePath = '/';

          if ( !$device )
               $device = 'desktop';
```

```
          $image = $this->connection->query("SELECT *
FROM [HeatmapScreenshots] WHERE [domain]=%s AND
[relativePath]=%s", $domain, $relativePath)->fetch();

          if ($image) {
               $filePath =
'//visitortrace.com/static/screenshots/' . $domain .
'/' . $image->id . '-' . $device . '.jpg';
               $data["imageUrl"] = $filePath;
          } else {
               return array();
          }

          $query = $this->buildQuery($domain,
$relativePath, $device, $from, $to);

          $values = $this->connection->query($query)-
>fetchAll();

          if ($values) {
               foreach ($values as $v) {
                    $json = json_decode($v->values);
                    foreach ($json as $j) {
                         $data["values"][] = $j;
                    }
               }
          } else {
               $data["values"] = array();
          }

          return $data;
```

```
    }

    public function buildQuery($domain, $relativePath,
$device, $from, $to) {

        $query = array("SELECT * FROM [Heatmap]
WHERE");

        array_push($query, "[domain]=%s", $domain);

        $relativePath = str_replace('*', '.*',
$relativePath);

        array_push($query, "AND [relativePath] REGEXP
%s", $relativePath);

        array_push($query, "AND [device]=%s",
$device);

        if ( $from ) {
            array_push($query, "AND [createdAt]=%s",
$relativePath);
        }

        if ( $to ) {
            array_push($query, "AND [createdAt]=%s",
$relativePath);
        }

        return $query;
}}
```

Příloha 4: default.latte – skript vykreslující teplotní mapu

```
$(function(){
     $('form').on('submit', function(e) {
          e.preventDefault();
          var domain = $('input[name="domain"]').val();
          var relativePath =
$('input[name="relativePath"]').val();
          var device = $('input[name="device"]').val();
          var from = $('input[name="from"]').val();
          var to = $('input[name="to"]').val();

          $.post('/heatmap/load-map-
     data?accountKey=CE66Z0bDusVHv66d85IoUNFb6ZxbXM1n',
          { domain: domain, relativePath: relativePath,
device: device, form: from, to: to },
          function(result){
               var heatmapImg = new Image();
               heatmapImg.onload = function() {
                    var heatmap = $('#heatmap');
                    heatmap.css('width', this.width);
                    heatmap.css('height', this.height);
                    heatmap.css('backgroundImage',
'url("'+result.imageUrl+'")');
                    var canvas =
               document.getElementById('canvas');
                    canvas.style.width ='100%';
                    canvas.style.height='100%';
                    canvas.width  = canvas.offsetWidth;
                    canvas.height =
canvas.offsetHeight;
```

```
                    var heat =
simpleheat('canvas').data(result).max(10);
                    heat.draw();
                }
                heatmapImg.src = result.imageUrl;
            });
        });
});
```