ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA DOPRAVNÍ

Bc. Vojtěch Davidík

NÁVRH A REALIZACE KATALOGU DOPRAVNÍCH TECHNOLOGIÍ A ROZHRANÍ PRO JEHO SPRÁVU

Diplomová práce

2016

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta dopravní
d ě k a n
Konviktská 20, 110 00 Praha 1

K620..... Ústav dopravní telematiky

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení studenta (včetně titulů):

Bc. Vojtěch Davidík

Kód studijního programu a studijní obor studenta:

N 3710 – DS – Dopravní systémy a technika

Název tématu (česky): **Návrh a realizace katalogu dopravních technologií a rozhraní pro jeho správu**

Název tématu (anglicky): Creating a PHP Web Application Using a MySQL Database.

Zásady pro vypracování

Při zpracování diplomové práce se řiďte osnovou uvedenou v následujících bodech:

- Prověřte a aktualizujte předchozí návrhy struktury databáze dopravních technologií.
- Analyzujte a popište potřeby a požadavky budoucích uživatelů databáze.
- Realizujte databázi dopravních technologií.
- Navrhněte a realizujte webové rozhraní pro správu databáze (zápis, editace, prohlížení, odstraňování a vyhledávaní záznamů dle kritérií).
- V rámci přístupu k databázi zohledněte různá oprávnění jednotlivých uživatelských rolí.
- Finální návrh databáze pečlivě zdokumentujte.

Rozsah grafických prací: standardní

Rozsah průvodní zprávy: minimálně 55 stran textu (včetně obrázků, grafů a tabulek, které jsou součástí průvodní zprávy)

Seznam odborné literatury: PROCHÁZKA, David. PHP 6: začínáme programovat. 1. vyd. Praha: Grada, 2012. 183 s. Průvodce. ISBN 978-80-247-3899-4.

HOPKINS, Callum. PHP okamžitě. 1. vyd. Brno: Computer Press, 2014. 134 s. ISBN 978-80-251-4196-0.

Vedoucí diplomové práce: **Ing. Martin Langr**

Datum zadání diplomové práce: **30. července 2015**
(datum prvního zadání této práce, které musí být nejpozději 10 měsíců před datem prvního předpokládaného odevzdání této práce vyplývajícího ze standardní doby studia)

Datum odevzdání diplomové práce: **1. června 2016**
a) datum prvního předpokládaného odevzdání práce vyplývající ze standardní doby studia a z doporučeného časového plánu studia
b) v případě odkladu odevzdání práce následující datum odevzdání práce vyplývající z doporučeného časového plánu studia

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE Fakulta dopravní L. S.

doc. Ing. Pavel Hrubeš, Ph.D.
vedoucí
Ústavu dopravní telematiky

prof. Dr. Ing. Miroslav Svítek, dr. h. c.
děkan fakulty

Potvrzuji převzetí zadání diplomové práce.

Bc. Vojtěch Davidík
jméno a podpis studenta

V Praze dne 30. července 2015

Poděkování

Na tomto místě bych rád poděkoval mému vedoucímu Ing. Martinovi Langrovi Ph.D., za odborné vedení, cenné rady, trpělivost a ochotu při zpracovávání mé diplomové práce.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Nemám žádný závažný důvod proti užívání tohoto školního díla ve smyslu § 60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne 20. května 2016 _____ Bc. Vojtěch Davidík

Abstrakt

Úkolem této diplomové práce je navrhnout a vytvořit webové rozhraní pro správu databáze dopravních technologií, která bude sloužit pro ukládání dat a bude umožňovat přehledně zobrazovat obsah databáze, provádět změny v databázi, správu uživatelů a to podle požadavků a uživatelských potřeb. Práce obsahuje popis návrhu a implementace tohoto webového rozhraní.

Klíčová slova

Webová aplikace, databáze, detektor, dopravní aktor, dopravní senzor, dopravní technologie, katalog dopravních technologií, php.

Abstract

The goal of this diploma thesis is to design and create web administration interface for „Database of traffic technology“, where database data is stored and clearly and logically displayed. Enabling user to edit database data, manage users based on user requirements. The thesis describes the design and implementation process of this web interface.

Key words

Web application, databases, detektor, traffic actuator, traffic sensors, traffic technology, database design, php

Obsah

SEZNAM POUŽITÝCH ZKRATEK:

MS	Microsoft
DB	Database
LAMP	Linux, Apache, MySQL, PHP
WAMP	Windows, Apache, MySQL, PHP
ID	Identifikátor
MD5	Message Digest 5
SQL	Structured Query Language
DBMS	DataBase Management System
PDO	PHP Data Objects
URL	Uniform Resource Locators
AJAX	Asynchronous JavaScript and XML
CSS	Cascading Style Sheets
HTML	HyperText Markup Language
PHP	Hypertext Preprocessor
SW	Software
UML	Unified Modeling Language
NYC	New York City
ASP	Active Server Page
WebML	Web modeling
WWW	World Wide Web
TC	Test Case
HD	High-Definition
GIS	Graphic Information Systems

1. Úvod

Pro elektronizaci katalogů, kartoték a jiných dokumentů dopravních technologií bylo nutné vytvořit databázi, včetně struktury, která by odpovídala všem požadavkům návrhu. Práce v samotné databázi s využitím například pouhého databázového manažeru není příliš efektivní ani uživatelsky příjemná. Z tohoto důvodu je vhodné využít webovou aplikaci, kterou není nutno instalovat na zařízení uživatele (počítač, tablet, smartphone) a může být spuštěna z kteréhokoliv zařízení pomocí webového prohlížeče, protože je spuštěna na straně serveru. Vzhledem k tomu, že je třeba jen prohlížeč, se webová aplikace někdy nazývá též jako lehký klient. Diplomová práce se zabývá tedy návrhem webové aplikace včetně její samotné realizace. [1]

V prvním kroku je nezbytné vytvořit databázi s odpovídající strukturou a požadavky pro řešení digitalizace dokumentů dopravních technologií. To zahrnuje, pro svou technologickou náročnost a odlišnost dopravních technologii, rozdělení dopravních senzorů do podkategorií a to tak, aby vyhovovaly potřebám databáze. Kritéria pro dělení byla zvolena logicky dle fyzického principu, či technologie. Webová aplikace byla navržena primárně pro ukládání, editování a spravování záznamů v databázi z příslušného formuláře. Avšak pro bezproblémový chod byly vytvořeny i další mechanismy například pro ověření uživatelů vznikla přihlašovací stránka. Pro relevantnost uložených informací se zhotovil cyklus autorizace záznamů. Aby mohly být dohledávány zásahy do databáze, tak se zformovala kontrola odpovědnosti záznamu atd.

Z podstatné části návrh databáze vychází z bakalářské práce: „Návrh databáze dopravních technologií“, kde v její praktické části byly definovány požadavky a nároky a to podle typu uživatelů, kteří ji hodlají využívat. Návrh vycházel i pro požadavky GIS aplikace, která si nárokuje jedinečnost dat v rámci celé databáze. A proto vzniklo značení primárních klíčů pro jednotlivé kategorie dopravních senzorů. Dále se nadefinovali tabulky dopravních zařízení podle logického dělení. Tabulkám se přiřadily atributy včetně specifikací datového typu polí. Dále se určily vztahy mezi tabulkami pomocí primárních, cizích klíčů a pravidel integrity na úrovní vztahů popsané v teoretické části návrhu databáze. [2]

Výsledky z této práce se staly základy pro návrh aplikace. Aplikace je rozdělena do několika modulů: dopravní senzory, dopravní aktory, ostatní dopravní zařízení, ostatní senzory a firmy. Z technické náročnosti dopravních senzorů je patrné, že tento modul je nejnáročnější pro správný návrh a samotnou implementaci. Ostatní zbylé moduly mohou být analogicky vytvořeny. Webová aplikace je připravena pro další rozvoj a umožnuje realizaci v plánovaném rozsahu.

Pro samotnou realizaci a zpracování vývoje webové aplikace byly použity nástroje moderního jazyka WebML, které jsou odvozený z UML a adaptované pro specifické potřeby při tvorbě webových aplikací. WebML metodika do značné míry kopíruje klasické obecné schéma vývoje softwaru. Takzvaný WebML development proces se skládá z následujících fází:

- specifikace požadavků na webovou aplikaci,
- návrh datové struktury,
- tvorba hypertextového modelu,
- návrh architektury aplikace a implementace,
- testování,
- nasazení a údržba aplikace.

WebML klade důraz na úvodní fáze celého procesu, tedy fáze specifikace požadavků, analýzy a návrhu. [3]

Diplomová práce by se měla stát nejen dokumentací návrhu a posléze i realizací, ale měla by také sloužit i jako uživatelská příručka, která má poskytnout základní informace o mechanismech webové aplikace a jak se používají. Jednotlivé kapitoly diplomové práce se snaží uživatele seznámit s aplikací. Věnují se všem stěžejním částím, které jsou podrobně vysvětleny a popsány včetně jejich účelu. A především vysvětlují a rozebírají nejdůležitější mechanismy, jako jsou: vkládání, editování, mazání záznamů. Neopominulo se ani na autorizaci záznamu a multikriteriální vyhledávání. Diplomová práce by měla obsahovat i postup, který je zaměřen pro zdokumentování vnitřních funkcí aplikace. Obsahuje tedy i vývojářský manuál, který neslouží jako úvod do programování webových aplikací, ale spíše se zajímá o funkční specifika webové aplikace dopravních technologií.

2. Návrh databáze

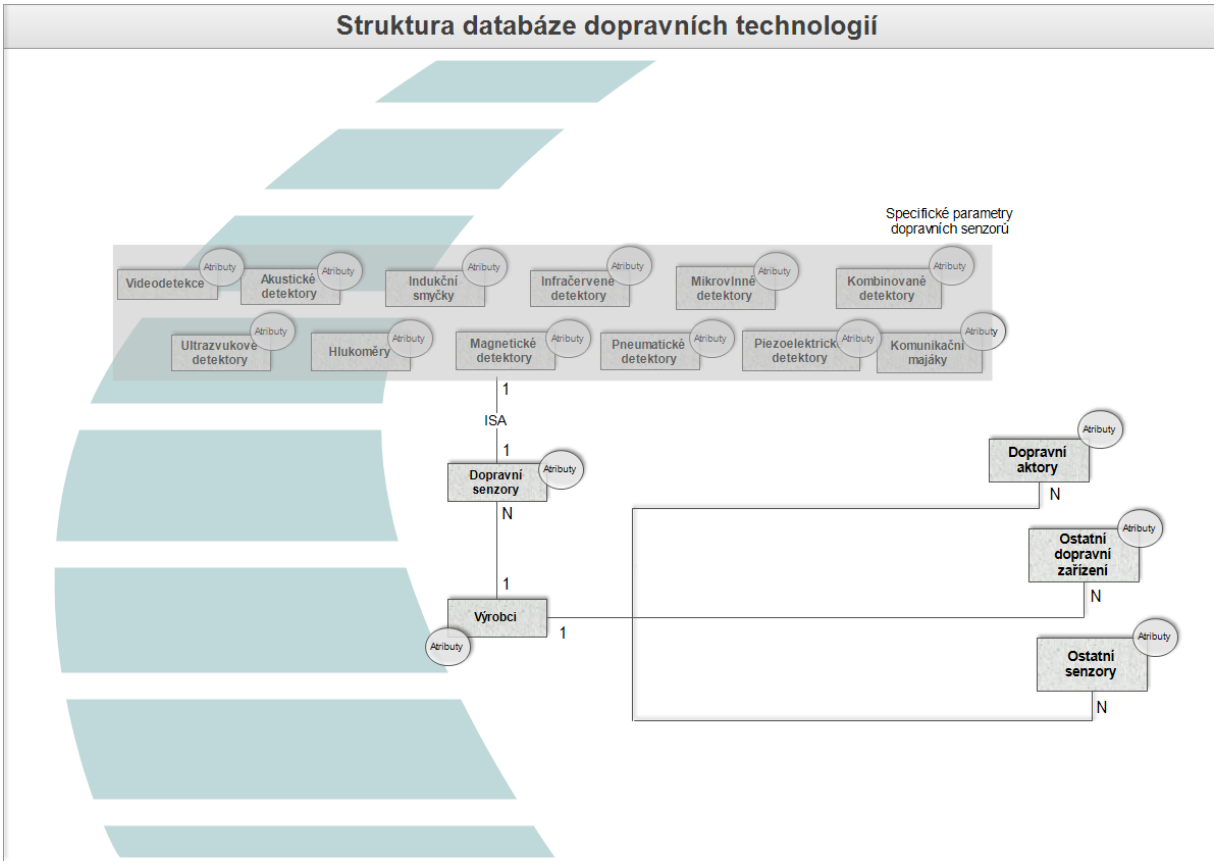
Pro ukládání databázových záznamů byla použita databáze MySQL, která obsahuje nyní přibližně čtyři desítky tabulek pro ukládání informací o dopravních technologiích, výrobcích a uživatelích. Při návrhu databáze bylo vycházeno z bakalářské práce, kde došlo k podrobné analýze celého procesu návrhu databáze od počátků až po samotnou realizaci a otestování v prostředí MS Access.

Databáze je navržena tak, aby poskytovala data o dopravních technologií a stala se součástí GIS aplikace pro efektivní analýzu dopravních zařízení na určitém území. Databáze po naplnění daty umožní porovnávání dopravních technologií, výběr vhodného zařízení nebo například jen poskytnutí informací o konkrétním senzoru, aktoru či firmě zabývající se dopravními technologiemi.

Návrh by měl umožnit, aby mohlo být dosaženo všech těchto informací v rámci jednoho katalogu dopravních technologií. Další požadavek na databázi je udržovat data dostupných dopravních technologií a dále využívat pro různé aplikace, která data vyžaduje pro analýzu konkrétních dopravních technologií, konkrétních výrobců podle různých kritérií zadaných na vstupu.

2.1. Datová struktura

Pro navržení vhodné datové struktury je nutné mít kvalitně provedenou fázi specifikace požadavků, protože závěry z této fáze jsou použity při návrhu datové struktury a přímo ovlivňují její výsledek. Analýza požadavků již proběhla v bakalářské práci, kde na jejím výsledku je datová struktura sestrojena. Návrh datové struktury byl lehce upraven, aby více vyhovoval potřebám webové aplikace. Vznikly nové entity a vztahy mezi nimi. Byly doplněny stávající tabulky a specifikovány jejich datové typy. Základním vstupem pro návrh datové struktury jsou tedy výsledky z fáze specifikace a analýzy uživatelských požadavků.



Obrázek 1:Schéma struktury DB.

2.1.1. Datové tabulky

Datové tabulky byly upraveny podle dalšího návrhu pro výstižné popsání všech důležitých aspektů a technických náležitostí jednotlivých dopravních zařízení. Největším doplněním si prošly datové tabulky, které jsou stěžejní pro modul dopravních senzorů.

Nejrozsáhlejší tabulka z tohoto modulu „senzory_obecne_parametry“ obsahuje všechny společné atributy dopravních senzoru. Ostatní specifické parametry se zaměřují na konkrétní tabulky dopravních senzorů. Tato kapitola je věnována důležitým tabulkám modulu „dopravní senzor“. Z rešeršních a ostatních prací zabývající se problematikou katalogu dopravní technologie, byly blíže popsány a určeny tyto tabulky modulu zabývající se dopravními senzory. Ostatní tabulky, které jsou pomocné jako výběry pro rolety nebo se podílejí na správném fungování aplikace, jsou k dohledání v příloze.

2.1.1.1. Tabulka senzory_obecne_parametry

Tabulka „senzory_obecne_parametry“ poskytuje informace, které jsou sdíleny všemi dopravními senzory. Jedná se o obecné parametry, které se vyskytují v naprosté většině

detektorů. Tabulka obsahuje nejen velké množství atributů pro obecný popis dopravních senzorů, ale i atributy, které přiřazují konkrétním dopravním senzorům jejich zařazení ve výsledném vyhledávacím seznamu popřípadě v jednotlivých speciálních seznamech. Výsledným vyhledávacím seznamem se myslí seznam dopravních technologií, kde jsou jednotlivé záznamy zkontrolovány a jsou připraveny být zařazeny do relevantních informací webové aplikace. Atributy, které se starají o správu záznamů, jsou: „potvrzeni“, „kontrola_potvrzeni“ a „nemazat“ a jsou vyhodnocovány zpravidla podle jednoho čísla, proto je zvolen datový typ „integer“. Většina atributů je intuitivních a jednoznačně napovídá kódérovi, jak textové pole vyplnit. Datové typy byly zvoleny na základě předpokládané informace plněné do textových polí formuláře. Primárním klíčem je sloupec „senzory_obecne_parametry.id“, který není zcela vyplňován automaticky náhodnými znaky, ale upřesňuje se podle vybraného dopravního senzoru. Cizím klíčem v této tabulce je pouze sloupec „id_vyrobce“ pro potřeby výběru z rolety výrobce resp. firmy. Ostatní atributy v tabulce vznikly na základě výsledků práce vyplívající z listů, brožur a jiných dokumentů o dopravních technologiích. V samotné webové aplikaci je nastavené našeptávání, které vysvětlí případné nejasnosti. Atributy, u kterých by mohlo docházet k rozporů sjednocení pochopitelnosti všech uživatelů aplikace, byla omezena nabídka roletou s jednou nebo více možností výběru. K tomuto účelu musela být vytvořena celá řada doplňkových tabulek s prefixem „roleta_“.

2.1.1.2. Tabulka vyrobci.

Tabulka vyrobci představuje centrální tabulku relační databáze. Skrze tuto tabulku jsou propojeny důležité datové tabulky dopravních zařízení. Tabulka obsahuje tyto atributy pro získání informace: „nazev“, „popis“, „web“, „stat“, „mesto“, „dokumenty“, „spoluprace“. Dále tabulka slouží pro zaznamenávání důležitých informací o výrobcích, popřípadě firmách, které se zabývají problematikou dopravních technologií. Je tvořena základními atributy, které popisují firmu, definují její oblast činnosti pomocí podmnožinové tabulky oblast_cinnosti a obsahuje i důležité atributy pro administraci záznamu.

2.1.1.3. Tabulka s prefixem roleta

Tabulky s prefixem roleta obsahují data o samotných dopravních senzorech. Rozšiřují stávající vložená data do obecné tabulky senzorů o další data z předpřipravených rolet. Jedná se o tyto tabulky:

„roleta_doba_mereni“, která slouží pro zaznamenávání doby měření dopravního senzoru,

„roleta_komunikace“ zaznamenává data o typu komunikace senzoru s jinou periferií nutné pro ovládání a softwarovou manipulaci. Například, zda je připojení možné přes fyzický prvek, či bezdrátově,

„roleta_moznosti“ zaznamenává data o možnostech komunikace senzoru s jinou periferií pro nastavení, spuštění, vypnutí, průběžnou kontrolu a vyčítání dat,

„roleta_pouziti“ zajišťuje data o softwaru dopravního senzoru a jeho možnostech pro správu naměřených dat: nastavení detektoru, průběžnou kontrolu měření, vyčítání dat, zpracování dat a export dat,

„roleta_typ_senzoru“ slouží, jak samotný název napovídá, pro volbu typu senzoru. Tato záložka je stěžejní pro přiřazení správné specifické tabulky a formuláře k formuláři obecných parametrů dopravních senzorů na stránkách určené pro náhledy, vkládání a úpravu záznamů,

„roleta_umisteni“ obsahuje data o umístění detektoru, nabízí možnosti výběru umístit detektor ve vozovce, na, nad nebo vedle,

„roleta_vlivy“ poukazuje na to, že při samotném měření, může docházet k jeho ovlivnění nepředvídatelnými jevy. Jeden z těchto jevů, je i počasí. Nabízí volbu vítr, teplotu, ale i jiné. Při volbě „jiné“ se zobrazí dodatečná kolonka pro ostatní okolnosti, které lze blíže upřesnit,

„roleta_zpusob_pouziti“ rozšiřuje možnost o přidání informace o režimu měření dopravního senzoru, zda je možné senzor použít ihned, nebo je vyžadováno předchozí nastavení parametrů,

S tabulkou „senzory_obecne_parametry“ jsou tyto tabulky s prefixem „roleta_“ spojeny vazbou n:m, to znamená, že bylo zapotřebí vytvořit další pomocné tabulky a to pro každý tabulkový vztah zvlášť. Tabulky, které tento vztah zajištují obsahují prefix „senzory_“ a jsou to tyto: „senzory_doba_mereni“, „senzory_komunikace“, „senzory_moznosti“, „senzory_pouziti“, „senzory_typ_senzoru“, „senzory_umisteni“, „senzory_veliciny“, „senzory_vlivy“ a „senzory_zpusob_pouziti“. Datová struktura tabulek obsahuje unikátní „ID“ a dva cizí klíče. Cizí klíč dopravního senzoru a cizí klíč konkrétní roletové volby.

2.1.1.4. Specifické tabulky

Specifické tabulky rozšiřují informaci o konkrétním dopravním senzoru. S tabulkou senzory_obecne_parametry mají vztah 1:1. Datová struktura se liší podle jednotlivých typů senzorů viz. příloha.

3. Webová aplikace

Za webovou aplikaci považujeme aplikaci běžící na serveru, s níž uživatel komunikuje pomocí klienta na svém počítači, kterým je nejčastěji webový prohlížeč. Ten sám o sobě nezná logiku aplikace, a proto se pro něj užívá označení tenký klient. V poslední době se webové aplikace těší stále větší popularitě. K jejich největším výhodám patří možnost aktualizace a spravování aplikací bez nutnosti stahování a instalace softwaru na osobní počítače uživatelů, existence pouze jedné verze aplikace, dostupnost odkudkoli, fungování bez ohledu na používaný operační systém, dostačující pro práci je webový prohlížeč.

Nevýhodou může být nedostupnost některých funkcí běžně známých z desktopových aplikací, jako je technika drag and drop. Webový prohlížeč totiž zobrazuje pouze webové stránky dynamicky generované webovou aplikací na serveru, které bez použití dalších technologií nemohou poskytnout stejné možnosti jako desktopové aplikace. K odstranění těchto omezení se využívá skriptování na straně klienta pomocí skriptovacího jazyka JavaScript. Ten umožňuje přidávání nejrůznějších dynamických prvků do statických stránek. Je také využíván technologií AJAX, díky které je možné načíst informace ze serveru bez nutnosti obnovování celé stránky. Další nevýhodou jsou odlišné implementace HTML a CSS v jednotlivých prohlížečích, které přidávají práci tvůrcům aplikací při snaze o stejný vzhled bez ohledu na prohlížeč. Tento problém se někdy řeší použitím technologií Adobe Flash či Java appletů, které jsou však třeba pomocí zásuvných modulů doinstalovat do webových prohlížečů. Výhody nicméně nad nevýhodami převažují, a proto se setkáváme s webovými aplikacemi na internetu čím dál častěji. [4]

3.1. Návrh webových aplikací

Na samotném počátku každé aplikace se nachází myšlenka či nějaký nápad. Ne vždy je ale přínosné snažit se takovou myšlenku ihned transformovat v aplikaci. Je docela možné, že podobný nápad už měl někdo dříve a taková aplikace již existuje. Pak je často mnohem výhodnější používat již existující aplikaci namísto vyvíjení zbrusu nové, která bude mít stejný účel. Další otázka, kterou je dobré si položit, se týká využití aplikace. Bude aplikace užitečná pouze pro mě nebo i pro další lidi? Usnadní skutečně to, k čemu bude vytvořena, nebo existuje jiný způsob, který je efektivnější a finančně méně náročný než vývoj a údržba aplikace? Jestliže se i po zodpovězení těchto otázek jeví nápad jako hodný realizace, je možné přistoupit k první

větší oblasti vývoje aplikace, kterou je specifikace požadavků a poté návrh samotné aplikace. [4]

3.2. Uživatelské potřeby

Fáze specifikace požadavků představuje kritický krok při vytváření webových aplikací, který si stále někteří programátoři neuvědomují. Její podcenění může mít velmi nepříjemné následky v pozdějších fázích, v nejhorších případech vede k přepisování zdrojového kódu aplikace. [3]

3.2.1. Identifikace uživatelských rolí a požadavky na personalizaci aplikace

V této fázi je nezbytné identifikovat typy uživatelů, kteří s aplikací budou pracovat a tyto skupiny rozeznat a určit, zda obsah a služby aplikace budou pro všechny uživatele totožné, nebo je požadována personalizace aplikace (přístupová práva a role). V aplikaci se rozlišují tyto uživatelské role s rozdílnými přístupovými právy.

Uživatel prohlížeč

Jedná se o příležitostného uživatele jen s právem pro prohlížení. Nezajímá se o fungování ovládání a filozofii aplikace. Do systému ničím nepřispívá, nemění ani nezakládá. Jeho role je pouze vyhledat informace v databázi.

Uživatel kódér

Uživatel kódér je pracovník s uživatelskými znalostmi webové aplikace. Role kódéra zahrnuje vytváření nových záznamů, popřípadě mazání a upravování svých vlastních. Pro svou práci je nutné, aby uživatel kódér mohl záznamy také vyhledávat, a to kvůli zvýšení relevantnosti a snížení možnosti duplicity záznamů.

Autorizovaný uživatel

Náplň autorizovaného uživatele je především v kontrole vytvořených záznamů, autorizace záznamů, přerozdělování potenciálně neaktuálních záznamů kodérům ke kontrole jejich správnosti, vytváření nových účtů pro kódéry, prohlížeče a udělování jim uživatelská práva. Autorizovaný uživatel má samozřejmě možnost i záznamy vytvářet, kontrolovat, upravovat a mazat, avšak neměla by to být jeho primární úloha. Jeho úkol lpí v dozoru nad ukládáním záznamů do databáze a na jejich správnosti. Pro vykonávání této funkce není potřeba znát programování webových aplikací, ale je záhodno být patřičně informován o všech důležitých mechanismech obsažené v aplikaci.

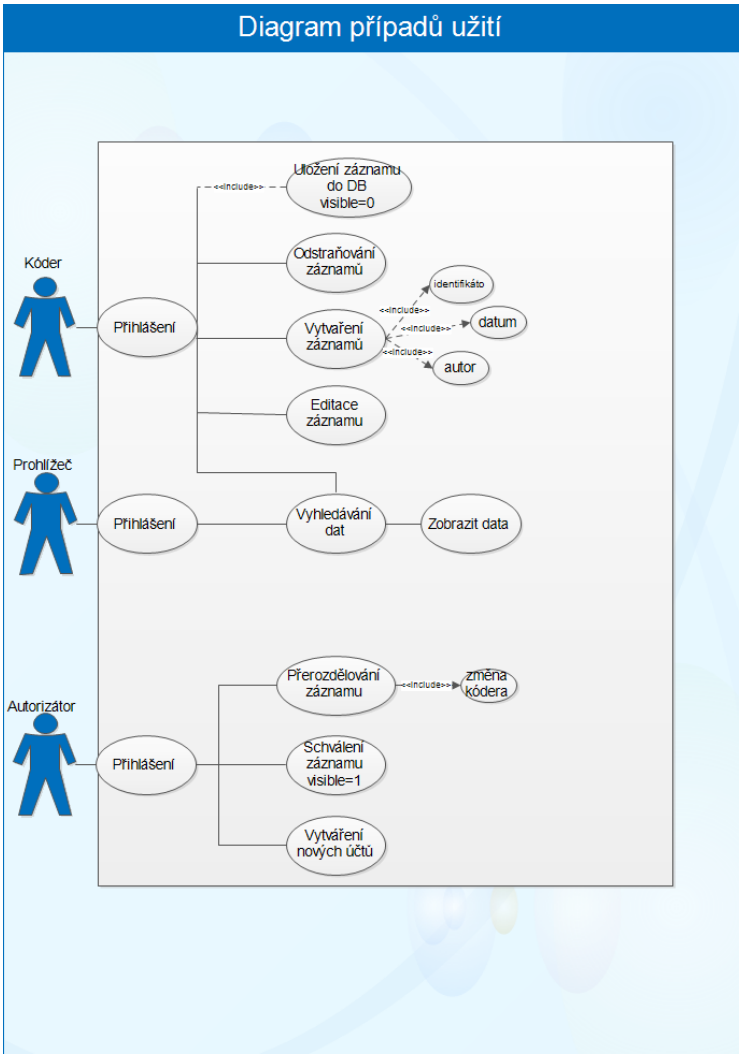
Administrátor

Administrátor má disponovat znalostmi v oboru programování webových aplikací. A to do takové míry, aby byl schopen kódu aplikace porozumět a zapracovat své případné změny a dále aplikaci rozvíjet, podle požadavků vlastníka aplikace. Měl by být schopný k záznamům přistupovat přímo v databázi, kde by je měl být také schopen spravovat.

3.2.2. Funkční požadavky

Funkční požadavky představují základní předmět systému a jsou měřeny konkrétními prostředky, jako jsou například hodnoty dat, logika a algoritmy rozhodování. [5]

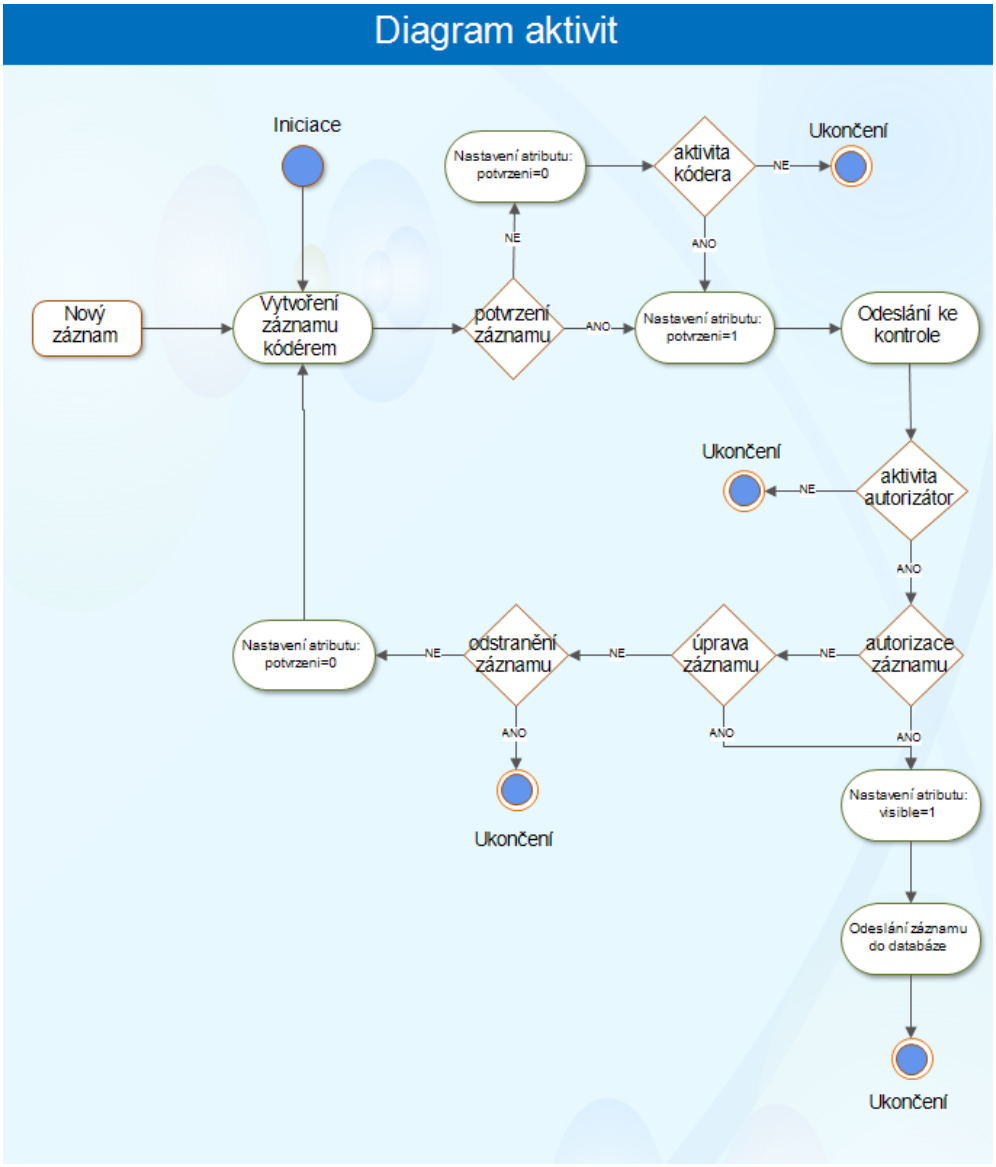
Funkční požadavky specifikují, co má webová aplikace dělat. Pro lepší přehled byla zvolena grafická podoba metody *„případů užití“* s doplňujícím textem.



Obrázek 2: Případy užití.

Autorizace záznamu

Důležitým krokem při určování požadavků je způsob autorizace (schválení) záznamu. Z funkčního hlediska si rozebereme metodu schválení záznamu autorizovaným uživatelem, která bude použita v mé aplikaci. Při přidání nového záznamu budou nastaveny defaultně hodnoty *„0“* u atributů, které jsou zodpovědné za viditelnost v konečném vyhledávacím seznamu. Pokud kódér usoudí, že je záznam v pořádku, pošle ho dále ke kontrole k autorizovanému uživateli, který na základě svých zkušeností rozhodne, zda záznam bude schválen, poslán zpět k autorovi, popřípadě k jinému kódérovi, nebo jestli bude odstraněn. Pro názorné zobrazení procesů je použit diagram aktivit, který modeluje procedurální logiku v procesu schválení záznamu.



Obrázek 3:Diagram aktivit.

3.3. Hypertextový model

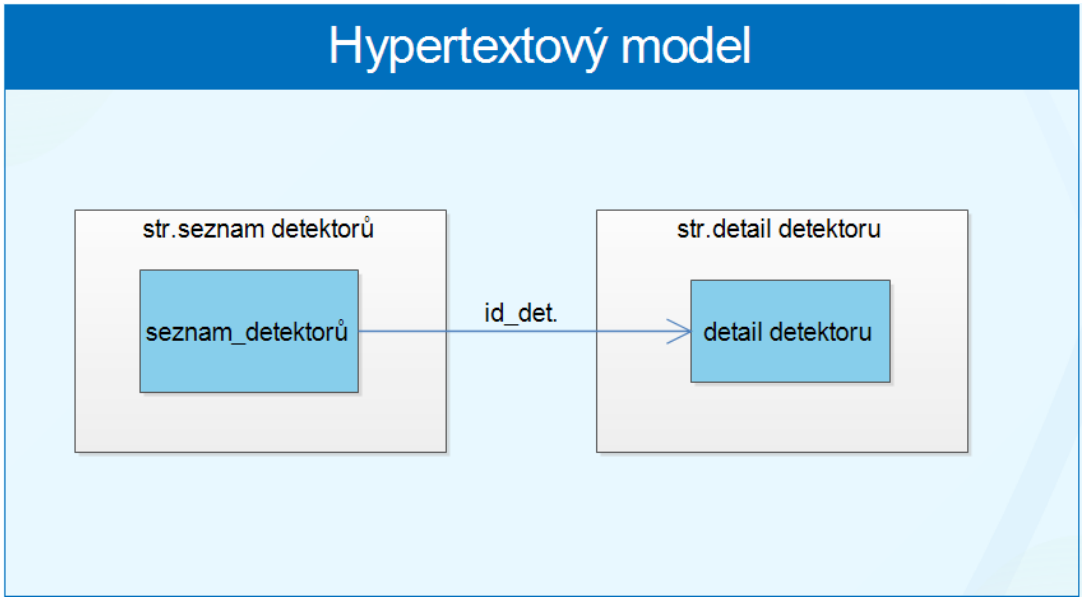
Hypertextový model staví na datovém modelu. Modeluje navigaci uživatele po webové aplikaci. Výsledkem modelu kompozice je sada webových stránek a definice jejich struktury. Samotná struktura ale samozřejmě nemůže vytvářet hypertext. Ten vzniká až po přidání hypertextových odkazů. Cílem navigačního modelu je propojit informační obsah jednotlivých stránek pomocí odkazů a navrhnout způsob navigace na webových stránkách.

Odkazy, které propojují jednotlivé webové stránky hypertextového modelu, se nazývají nekontextové odkazy. Klasickým nekontextovým odkazem v našem elektronickém obchodě může být propojení stránek seznamu zboží příslušné kategorie a detail nákupního košíku. Mezi těmito stránkami nemusíme přenášet žádné parametry. [6]

Naopak odkazy, propojující dvě „units“ v rámci jedné nebo dvou webových stránek, jsou označovány jako kontextové odkazy, protože slouží pro přenos parametrů mezi stránkami, které jsou využívány v takzvaných parametrizovaných selektorech příslušných „units“.

Hypertextový model se skládá ze dvou modelů, zakreslených do jednoho diagramu [3]:

- Kompoziční model – zachycuje logické elementy, z kterých je stránka složena. Definuje význam jednotlivých elementů.
- Navigační model – modeluje, jakým způsobem se uživatel pohybuje po webové aplikaci. Popisuje jak webové stránky, tak i odkazy mezi stránkami.



Obrázek 4: Hypertextový model.

3.4. Struktura webové aplikace

Po úspěšném dokončení předchozího kroku specifikace požadavků a identifikaci uživatelských rolí je možné přistoupit k samotné realizaci webové aplikace.

Tato kapitola se věnuje především vlastní implementací návrhu. Jsou zde uvedeny použité technologie. Krátce je zde popsáno fyzické rozložení souborů a adresářů aplikace, které částečně vychází z volby technologií. Nakonec je zde uvedeno několik vybraných zajímavých problémů, které při samotné implementaci musely být řešeny. Princip jejich řešení je popsán, případně odůvodněn.

3.4.1. Použité nástroje v tvorbě webové aplikace

Skriptovací jazyk PHP

Stále nejvíce používaným programovacím jazykem webových aplikací zůstává PHP. Tento jazyk má dobře propracovanou dokumentaci a často se vyučuje již na středních školách. Firmy poskytující hosting PHP aplikací nejčastěji nabízejí variantu tzv. LAMP serveru. Jedná se o množinu obsahující operační systém Linux, apache v roli webového aplikačního serveru. [7]

MySQL jako databázový systém a PHP jako programovací jazyk. Jelikož se jedná o svobodný software, pořizovací náklady jsou nulové. Některé prvky této množiny lze libovolně nahrazovat podobným řešením dle přání zákazníka. Místo databáze MySQL lze využít například PostgreSQL. Místo programovacího jazyka PHP využít jiné skriptovací jazyky, jako jsou například Perl nebo Python. Alternativou k serveru LAMP je WAMP server. Toto řešení využívá místo open source operačního systému Linux komerční systém Windows od společnosti Microsoft. Nevýhodou PHP je, že se jedná o skriptovací jazyk. Nedochází k překladu do binární podoby, ale jednotlivé příkazy jsou interpretované řádek po řádku. Tímto je jazyk připraven o možnost optimalizace a je tedy i pomalejší oproti kompilovaným jazykům. [8]

PDO

Rozšíření PHP Data Objects (PDO) definuje lehké a konzistentní rozhraní pro přístup k databázím v PHP. Každý databázový driver, který implementuje rozhraní PDO může rozšířit specifické databázové vlastnosti například regulární rozšiřujících funkcí. Všimněte si, že nelze

sámo o sobě provádět žádné databázové funkce pomocí rozšíření PDO; je nutné použít specifický databázový driver PDO databáze pro přístup k databázovému serveru.

PDO poskytuje přístupy k abstraktní vrstvě, což znamená, že bez ohledu na to, které databáze používáte, můžete použít stejné funkce pro dotazy a načítání dat z databáze. PDO neposkytuje databáze abstrakce; nepřepisuje SQL nebo nenapodobuje chybějící funkce.

PDO ships se dodávají s PHP 5.1 a jsou k dispozici jako PECL rozšíření pro PHP 5.0; PDO vyžaduje nové OO vlastnosti v jádru PHP 5, a proto je není možné spustit v dřívějších verzích PHP. [9]

MySQL

Je relační databázový systém typu DBMS (database management system). Každá databáze v MySQL je tvořena z jedné nebo více tabulek, které mají řádky a sloupce. V řádcích rozeznáváme jednotlivé záznamy (řádek=záznam). Sloupce mají jméno a uvozují datový typ jednotlivých polí záznamu (sloupec=pole). Práce s databázemi, tabulkami a daty se provádí pomocí příkazů, respektive dotazů, vycházejí z deklarativního programovacího jazyka SQL (Structured Query Language). Systém MySQL je využitelný v C, C++, Java, Perl, PHP, Python, Tcl, Visual Basic, .NET. [10]

HTML

HTML (HyperText Markup Language) je značkovací jazyk určený pro tvorbu webových stránek či aplikací zobrazitelných ve webovém prohlížeči. [11]

Slouží pro prezentaci informací ze serveru uživateli. V této aplikaci je jeho prostřednictvím realizováno uživatelské rozhraní a veškerá interakce s uživatelem.

CSS

CSS (Cascading Style Sheets) je strukturovaný jazyk popisující vzhled dokumentu napsaného ve značkovacím jazyce. Díky využití CSS je možné oddělit definici vzhledu stránky od jejího obsahu, a také použít jednu definici vzhledu pro více stránek, což pomáhá zpřehlednit a usnadnit manipulaci s designem stránek. [12]

JavaScript

Je skriptovací jazyk, který je interpretován webovým prohlížečem na straně klienta. Vychází syntaxí z jazyka C, je dynamicky typovaný a lze v něm pracovat s objekty, které zde však nevznikají instanci čí tříd, ale jako takzvané prototypy. Umožňuje interakci s uživatelem

a manipulaci s obsahem stránky bez toho, aby se stránka musela znovu celá načítat ze serveru. Vhodným použitím JavaScriptu lze učinit používání stránky komfortnější a uživatelsky přívětivější. Jeho nevýhodou však je, že některá zařízení jej nemusí plně podporovat, jeho podpora se také mírně liší napříč různými prohlížeči a v neposlední řadě také výpočetně zatěžuje cílové zařízení. V této aplikaci se stará především o interaktivitu generovaných grafů, animování některých přechodů pro větší názornost a pro odesílání AJAX požadavků. [13]

jQuery

Jedná se o mezi-platformní JavaScriptovou knihovnu, jejíž hlavním účelem je propojení HTML a JavaScriptu. Knihovna jQuery je vytvořena tak, aby ulehčovala práci s DOM3 prvky, usnadňovala vytváření animací, obsluhu událostí a tvorbu AJAX4 aplikací. Dále podporuje manipulaci s CSS a její funkčnost lze jednoduše rozšířit pomocí zásuvných modulů. Knihovnu poprvé představil John Resig na BarCamp NYC roku 2006 a od té doby se stala nejpoužívanější knihovnou vůbec. Tento balíček je rovněž součástí ASP. NET MVC Frameworku. [14]

3.4.2. Funkční mechanismy aplikace

Účelem aplikace je evidovat informace dopravních technologií v uspořádaném katalogu. Vytvářet záznamy obsahující informace o dopravních technologiích a jejich výrobcích. Ale i uchovávat informace o autorech, odpovědnosti jednotlivých kóderů atd. Tyto funkce jsou zajištěny pomocí mechanismů, které jsou hlavní součástí webové aplikace a pro které byla vůbec aplikace navržena a vytvořena. Dále aplikace zpřístupňuje již uložené záznamy všem uživatelům a umožňuje v nich vyhledávat.

Přihlášení do databáze

Jelikož se jedná o webovou aplikaci a její síla právě spočívá v užívání databáze, je nutné se nejdříve do databáze připojit. K databázi se přistupuje pomocí PHP Data Objects (PDO).

```
try {
$db = new PDO('mysql:host='.$server.';port='.$port.';dbname='.$dbname.';charset=utf8', $user, $pass);
} catch (PDOException $e) {
    die("Error!: " . $e->getMessage() . "<br/>");
  }
```

Obrázek 5:Připojení k databázi.

Níže jsou v jednoduchosti popsány proměnné ve skriptu:

port

MySQL naslouchá standardně na portu 3306, pokud není při konfiguraci určeno jinak. Jestliže není vyplněn, je číslo portu 3306 doplněno automaticky.

dbname

Zde by mohlo dojít k určitému matení pojmů: Na databázovém serveru MySQL existuje většinou několik vzájemně nesouvisejících databází. Každá databáze neobsahuje žádnou nebo obsahuje více tabulek a každá může mít vlastní nastavení práv. V praxi to většinou bývá tak, že součástí hostingových služeb je možnost používat jednu databázi. V ní je možné mít teoreticky neomezený počet tabulek a ostatní uživatelé serveru nemají k vaší databázi přístup. Vy zase nemáte přístup k jejich databázím.

user

Je potřeba zdůraznit, že uživatelské jméno k databázovému serveru nijak nesouvisí s přihlašovacím jménem k serveru. Ve skutečnosti většinou není na serveru, na němž databáze poběží, zřízen systémový účet. Uživatelské jméno je k dostání od správce databáze.

pass

Stejně tomu tak i heslo je k dostání od správce databáze. Většinou bude splňovat základní bezpečnostní pravidla, tzn., bude dostatečně dlouhé a bude se skládat z písmen, číslic, případně dalších znaků. [15]

Relace a přihlášení uživatele do aplikace

Relace začíná session start() funkcí. Relační proměnné jsou nastaveny globální proměnnou $_SESSION. Proměnné v relaci se nepřenášejí mezi serverem a počítačem klienta, skript na serveru je dostává přímo z interpretu PHP. [16]

Relace je jedna z možností, jak uchovávat informace v proměnných a to přes více stránek v aplikaci, aniž by bylo nutné využívat cookies. Informace totiž nejsou uloženy v uživatelském počítači. Jednoduše řečeno aplikace ví, že je přihlášená autorizovaná osoba a umožnuje jí přistupovat k dalším a dalším stránkám aplikace až do odhlášení nebo určité prodlevě v aktivitě. Avšak v mé webové aplikaci automatické odhlášení není nastaveno. [17]

Přihlášení probíhá přes formulář na stránce index.php, kde jsou načteny proměnné myusername a mypassword do paměti. Za zmínku stojí, že proměnná „$mypassword“ je pro zvýšení bezpečnosti „hašována“ MD5 algoritmem.

```php
session_start();
require_once("mysql_db.php");

// username and password poslan z formuláře
$myusername = htmlspecialchars($_POST['myusername']);
$mypassword = MD5 (htmlspecialchars($_POST['mypassword']));

$sql = "SELECT * FROM members WHERE username=? and password=?";

try {
$select = $db->prepare($sql);
$select->execute(array($myusername, $mypassword));
$result = $select->fetchAll(PDO::FETCH_ASSOC);
} catch (PDOException $e) {
    die("Error!: " . $e->getMessage() . "<br/>");
  }

if(count($result) == 1) {
    $_SESSION['logged'] = true;
    $_SESSION['username'] = $myusername;
    $_SESSION['password'] = $mypassword;
    header('location:main.php');
}
else {
  $_SESSION['logged']=false;
  header('location:index.php');
```

Obrázek 6: Založení relace s proměnnými.

Na obr. 6 je vidět založení relace s proměnnými „$myusername“ a „$mypassword“, které jsou odeslány z formuláře úvodní přihlašovací stránky. NA proměnnou „$mypassword“ je použita „hašovací“ funkce MD5. Dále webová aplikace komunikuje s databází (připojení již proběhlo pomocí stránky mysql_db.php) SQL dotazem. Následně porovnává uložené proměnné z formuláře s daty v databázi. Jednoduchou logikou je vyhodnoceno, zda vyhovět, či nikoliv požadavku uživatele.

Vytvoření záznamu

Celý příkaz je založen tak, aby na místa, kde je nutné dosadit nějaké proměnné z PHP, jsou dosazeny jen otazníky. Vlastní data se potom předají jako pole metodě „execute“, která dotaz daty naplní a provede. Nahrazování otazníků daty probíhá v pořadí dat v poli, které se předávají do metody „execute“. Je to rychlá a efektivní metoda, která byla v implementaci návrhu využita.

Další možností, jak vkládat záznamy do databáze, je kromě otazníků možné používat i „PDO proměnné“ s dvojtečkou na začátku. U komplikovanějších dotazů to může zvýšit čitelnost a přehlednost kódu. Ostatní příkazy pro odstraňování a editaci jsou v podobné formě, jak je vidět na obr. 7.

```
$sqle = "INSERT INTO odpovednost(id_ds, odpovedny_koder) VALUES (?,?)";

    try {
        $stmt = $db->prepare($sqle);
        $stmt->execute(array($identifikator,$myusername));
    } catch (PDOException $e) {
        die("Error!: " . $e->getMessage() . "<br/>");
    }
```

Obrázek 7:Vytvoření záznamu.

Vícekriteriální vyhledávání

Vyhledávání je multikriteriální, to znamená, že je možné vyhledávat napříč všemi hodnotami. Pro lepší použitelnost filtru bylo rozhodnuto, že vyhledávání bude zpřístupněno pouze pro kritéria, která jsou předefinovaná v roletách. Dalším důvodem pro vyhledávání pouze ucelených výrazů je i to, že různé překlepy a nepřesnosti ve vyhledávaných slovech, by měly za následek nesprávný nebo nulový seznam výsledných záznamů po filtraci.

Sestrojit filtr jednotlivých atributů s roletovým výběrem jedné možnosti není nikterak velký problém. Avšak rolety, kde bylo možné zvolit více možností anglicky tzv. „multi select dropdown list“, si vyžadovaly větší pozornost.

Níže budou objasněny všechny kroky v mechanismu filtrace záznamů při samotné implementaci návrhu. Nejprve se vytvořil vyhledávací formulář, který je v podstatě stejný jako formulář pro vkládání nebo editaci záznamů s tím rozdílem, že obsahuje jen předefinované rolety z důvodů, které jsou vysvětleny výše. Proměnné s jednou volbou výběru jsou klasicky odeslány a uloženy.

```
$vyrobce_nazev = htmlspecialchars($_POST['vyrobce_nazev']);
$senzory_typ = htmlspecialchars ($_POST['senzory_typ']);
$pocet_j_p = htmlspecialchars ($_POST['pocet_j_p']);
$smerovost = htmlspecialchars($_POST['smerovost']);
$pamet = htmlspecialchars($_POST['pamet']);
$zpusob_umisteni = htmlspecialchars($_POST['zpusob_umisteni']);
$narus_voz = htmlspecialchars($_POST['narus_voz']);
$prubez_kontrola_nas = htmlspecialchars($_POST['prubez_kontrola_nas']);
$prubez_kontrola_zar = htmlspecialchars($_POST['prubez_kontrola_zar']);
$nap = htmlspecialchars($_POST['nap']);
$druh_el_proud = htmlspecialchars($_POST['druh_el_proud']);
$mobility = htmlspecialchars($_POST['mobilita']);
$ucelenost = htmlspecialchars($_POST['ucelenost']);
$material = htmlspecialchars($_POST['material']);
$rozmery = htmlspecialchars($_POST['rozmery']);
```

Obrázek 8: Odeslané proměnné z formuláře.

Rolety s více možnostmi výběru jsou uloženy do pole. Jednotlivá pole byla předpřipravena a pomocí funkce „count“ byly spočteny „itemy“, neboli množství vybraných možností z rolety.

```
if (isset($_POST['dopravni_veliciny'])) {
$dop_vel = count ($_POST['dopravni_veliciny']);
    }
if (isset($_POST['komunikace'])) {
$komu = count ($_POST['komunikace']);
    }
if (isset($_POST['moznosti'])) {
$moz = count ($_POST['moznosti']);
    }
if (isset($_POST['doba_mereni'])) {
$dobm = count ($_POST['doba_mereni']);
    }

if (isset($_POST['umisteni'])) {
$umi = count ($_POST['umisteni']);
    }

if (isset($_POST['vlivy'])) {
$vliv = count ($_POST['vlivy']);
    }

if (isset($_POST['pouziti'])) {
$pou = count ($_POST['pouziti']);
    }

if (isset($_POST['zpusob_pouziti'])) {
$zpupou = count ($_POST['zpusob_pouziti']);
    }
```

Obrázek 9: Spočteny „itemy“ v poli.

Dále bylo zapotřebí přiradit jednotlivým možnostem v „multi select dropdown list“ jejich identifikátory. Pro představu, jestliže uživatel vybere z rolety „dopravni_veliciny“ veličiny: intenzita, klasifikace a rychlost, do pole budou připsaný jejich identifikátory tedy 1, 3 a 5. A to právě pomocí cyklu „foreach“.

```
foreach ($_POST['dopravni_veliciny'] AS $key => $val)    {
    if (is_numeric($val)) {
        $_POST['dopravni_veliciny'][$key] = intval($val);
    } else {
        unset($_POST['dopravni_veliciny'][$key]);
    }
}
```

Obrázek 10:Cyklus přes všechny „itemy“ v poli.

Pro potřeby filtrace je nutné, aby výstup byl v SQL jazyce. Jelikož se bude právě pomocí php jazyka dotazovat na databázi, kde na základě definovaného vstupu, bude vrácena náležitá odpověď. Proto byla vybrána funkce „implode“, která je definována parametrem pole a oddělovačem. Pokud to bude aplikováno na předchozí příklad, kdy pole bude „dopravni_veliciny“ a čárka jako oddělovač, dostane se tento výraz ve formě řetězce: „1, 3, 5“.

```
$dopravni_veliciny = implode (',', $_POST['dopravni_veliciny']);
```

Obrázek 11:Fce „implode“.

Nyní když byly předpřipraveny řetězce hodnot z pole, bylo se možné věnovat samotnému SQL dotazu. SQL dotaz, který je doprovázen komentářem, má výchozí stav v tomto tvaru:

„**SELECT DISTINCT**“

Zajímá nás výběr bez duplicit, který je už zajištěn při vkládání, ale přesto jsme se raději „pojistili“ i při načítání hodnot z databáze.

„**column.table1,column.table2,column.table(n)**“

Necháme si zobrazit sloupce v tabulkách, které nás zajímají.

„**FROM table1 LEFT OUTER JOIN table2 ON table1.id = table2.id LEFT OUTER JOIN table(n) ON table(n).id = table(1).id** „

Spojujeme pouze ty tabulky, které potřebujeme pro filtr. Zvolili jsme formu vnějšího spojení „left outer join“. Tedy zobrazujeme všechny řádky z první tabulky a z ostatních jen ty kde jsou shodné jejich cizí klíče s primárním klíčem první tabulky v ostatních případech je hodnota „null“.

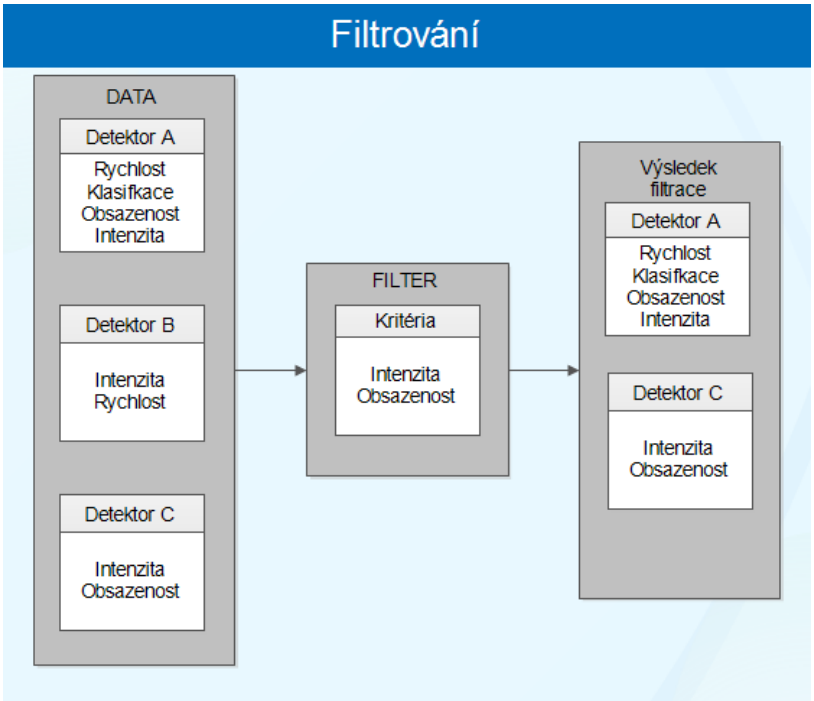
„**WHERE "1"="1"**“

Jelikož může dojít k případu, kdy nebudou vybrány žádné kritéria, museli jsme si použít pro zachování funkčnosti dotazu pomocnou klauzuli.

„**AND table2.id_roleta IN (1,2,n) AND table2.id_roleta IN (1,2,n) AND table(n).id_roleta IN (1,2,n) GROUP BY senzory_obecne_parametry.id having count(*) = 1 * (zvolený počet možnosti v roletě1) * (zvolený počet možnosti v roletě2) * (zvolený počet možnosti v roletě(n))**“

Výčet id hodnot zvolených možností dané rolety, jako specifikace „IN“ operátoru. Záznamy jsou seskupeny podle senzorů a správná kombinace je zajištěna počtem kombinací v „HAVING“ klauzuli.

Tento návrh filtru umožňuje vyhledávat senzory podle více kritérií. Velkou výhodou je, že neúplná shodnost kombinací v množině je vyhodnocena jako vyhovující a je přiřazena do výsledku seznamu filtrovaných záznamů. Pokud se vrátíme k našemu příkladu, kde byla roleta „dopravni_veliciny“, z které byly vybrány 3 hodnoty rychlost, klasifikace a intenzita. Tak tedy hledaný dopravní detektor s těmito atributy bude do výsledného seznamu přiřazen, pokud hledaný výraz v dopravních veličinách bude celá množina nebo jen část z tohoto výběru možnosti rolety „dopravni_veliciny“. A to je přesně ono, co bylo v požadavcích na filtr vyžadováno. Z logiky věci vyplývá, že není žádoucí být při výběru senzoru omezován pouze na konkrétní přesnou kombinaci možností. Ale je potřebné a nutné, aby hledaný detektor umožňoval vše, co bylo v kritériích filtru a pokud nabízí i další funkce, tím lépe.



Obrázek 12:Filter.

Načítaní hodnot z databáze

Mechanismus vracení proměnných zpět do jednotlivých polí ve formuláři je nezbytný pro editaci a náhled záznamu. Načítání hodnot je použito u všech formulářů webové aplikace. Data dostaneme použitím SQL příkazu „SELECT“, kde s použitím klauzule vyhledáme konkrétní záznam pomocí jeho unikátního „ID“ klíče. Klíč záznamu je odeslán metodou „GET“, který je vidět v „URL“ adrese (znaky, které následují za otazníkem), maximální délka je 1024 znaků. Výhoda této metody je převážně v jednoduchosti. Data se odešlou do proměnné. Na obr. 13 je vidět odkaz s proměnnou „ID“. Po kliknutí na tento odkaz je uživatel v tomto případě přesměrován do editačního formuláře konkrétního výrobce.

```
echo" <td><a href='vyrobci_edit.php?id=".$row["id"].
```

Obrázek 13: Proměnná "ID" v odkazu.

Adresa přijímací stránky pak vypadá takto:

https://detektory.lss.fd.cvut.cz/vyrobci_edit.php?id=769451

Obrázek 14: „URL“ adresa.

Samotný dotaz je k vidění na obr. 15, kde dochází k selektování všech záznamů konkrétního výrobce a to právě na základě jeho identifikačního čísla. Zde je patrné, že právě tato proměnná byla získána metodou „GET“.

```php
$UID = htmlspecialchars($_GET['id']);

$sql = "SELECT * FROM vyrobci WHERE id = ?";

try {
        $stmt = $db->prepare($sql);
        $stmt->execute(array($UID));
        $result = $stmt->fetchAll(PDO::FETCH_ASSOC);
      } catch (PDOException $e) {
        die("Error!: " . $e->getMessage() . "<br/>");
        }

if(count($result) > 0){
      $id = $result[0]['id'];
      $nazev = $result[0]['nazev'];
      $popis = $result[0]['popis'];
      $web = $result[0]['web'];
      $stat = $result[0]['stat'];
      $mesto = $result[0]['mesto'];
      $dokumenty = $result[0]['dokumenty'];
      $spoluprace = $result[0]['spoluprace'];


}else{
    echo 'No entry found. <a href="javascript:history.back()">Go back</a>';
}
```

Obrázek 15:Načtení hodnot z databáze.

Další krok, který byl nutný vykonat, aby bylo možné zobrazovat data ve formuláři, bylo přirazení proměnné každému poli v editovacím popřípadě náhledovém formuláři. Jak je vidět na formuláři obr. 16, kde na místo hodnoty pole je přiřazena proměnná z předchozího kroku. Pro ukázku byl vybrán formulář „vyrobci“, který neobsahuje tolik atributů a proto je přehlednější.

```
<table width="625" border="1" align="left" cellspacing="15" class="vyrobci_table">
  <caption class="pismo">
    <strong>Informace o firmě</strong>
  </caption>
  <tbody>
    <tr align="left" valign="top" class="pismo">
      <td width="333" height="332" align="left" valign="top" class="pismo"><p>
        <label for="textfield">Jméno firmy:</label>
        <input type="text" name="jmeno_firmy" id="textfield" value="<?=$nazev?>" >
      </p>
        <p>Popis firmy:<br>
          <textarea name="popis_firmy" id="textarea" title="Zde nastínit velikost,
          působení firmy, možná i její strukturu či historii" ><?=$popis?></textarea>
        </p>
        <p>
          <label for="textfield3">WWW:</label>
          <input type="text" name="www" id="textfield3"value="<?=$web?>" >
        </p>
        <p>
          <label for="textfield4">Dokumenty:</label>
          <input name="dokumenty" type="text" id="textfield4" title="letáky na k620 např" value="<?=$dokumenty?>" >
        </p>
        <p>
          <label for="textfield5">Spolupráce s FD:</label>
          <br>
          <textarea name="spoluprace" id="textarea2"><?=$spoluprace?></textarea>
        </p>
        <p><span >Stát:</span>
```

Obrázek 16:Formulář s proměnnýma.

Ve formuláři se nenacházejí pouze textová pole ale i „select dropdown“ tzv. vyjíždějí rolety.Ve formuláři „vyrobci“ jsou to například státy. Státy jsou předem dány a časem jich mnoho nepřibývá ani nezaniká, proto je logické, že jsou přímo předefinované pro výběr. Na obr. 17 je SQL dotaz pro zobrazení názvů států a jejich vzestupné seřazení. Dále se samotný dotaz vykoná a je vrácen výsledek.

```
        $sql="SELECT country_name FROM countries ORDER BY country_name";

try {
  $select = $db->query($sql);
  $result = $select->fetchAll(PDO::FETCH_ASSOC);
} catch (PDOException $e) {
    die("Error!: " . $e->getMessage() . "<br/>");
  }
```

Obrázek 17:Načtená roleta.

Zatím je to tedy shodné s jednoduchými textovými poli. Rozdíl nastává až v následujícím kroku, kdy se pomocí funkce „foreach“ hledá přes všechny hodnoty v roletě tu hodnotu, která odpovídá proměnné „$stat“ respektive je shodná. Nakonec se pouze vybere jako výchozí hodnota v roletě tak, jak je tomu na obr. 18

```php
foreach($result as $row){
  echo '<option value="'.$row['country_name'].'"'.($row['country_name'] == $stat ? ' selected="selected"' : '').'>'.$row['country_name'].'</option>';
}
```

Obrázek 18: Výběr hodnoty v roletě.

A nakonec bylo zapotřebí vyřešit poslední problém, a to jak se vypořádat s roletami s více možnostmi výběru („multi dropdown list“). I tato situace má své řešení. Nejdříve je nutné načíst všechny možnosti z tabulky a na jejím základě označit hodnoty v roletě. Pokud se nahlídne do datové struktury tabulek týkající se všech tabulek „vyrobci“, je patrné, že obsahuje i tabulku „oblast_cinnosti“. Ta je spojena s tabulkou „vyrobci“ n:m vazbou. Proto se nevystačí se stávajícím provedením php zápisu. Avšak ještě předtím, než se dostane k řešení samotné problematiky, je nutné pro pochopení si říci něco o polích:

Pole je datový typ a obecně slouží k uložení více hodnot stejného typu, které spolu nějak logicky souvisí. Pomocí pole je možné mít v proměnné např. obsah 30ti textových polí, aniž by bylo nutné je zakládat samostatně: „$text1, $text2, $text3“ atd. až „$text30“. Založí se pouze jedno pole. To je možné vnímat jako přihrádku v paměti, kde je v každé přihrádce jeden prvek pole. Pole se založí pomocí klíčového slova „array“.

```php
<?php
$pole = array();
```

Obrázek 19:Založení pole.

Pole je prázdné. Pro naplnění pole hodnotami stačí vložit hodnoty do závorky za „array“ a oddělit čárkou. Například takto:

```php
<?php
$barvy = array('červená', 'zelená', 'žlutá', 'fialová');
```

Obrázek 20:Naplněné pole.

K prvkům pole je přistupováno přes hranatou závorku tzv. „indexer“. Takto založené pole je tzv. číselně indexované. Prvky jsou číslovány od nuly a odkazuje se na ně pomocí indexů:

```php
<?php
$barvy = array('červená', 'zelená', 'žlutá', 'fialová');
echo $barvy[2];
```

Obrázek 21:Načtení hodnoty z pole.

Výstup z volání příkazu „echo“ bude hodnota „žlutá“, jelikož se indexuje od nuly.

PHP je interpretovaný jazyk a proto je možné si dovolit poskytnout programátorovi vysoký komfort. Pro začátek si řekněme, že pole v PHP jsou "nafukovací". Nemají pevnou velikost, je možné do nich prvky libovolně přidávat. Prázdné závorky odkazují na další index do pole, který v něm ještě není. Takto je do pole možné zapsat další prvek: „$pole[]“ ukazuje vždy na index o jednu větší než všechny indexy v poli:

```php
<?php
$barvy = Array('červená', 'zelená', 'žlutá', 'fialová');
$barvy[] = 'bílá';
```

Obrázek 22:Možnost vkládání prvku do pole.

Nejjednodušší možnost jak vypsat všechny prvky z pole, je použití foreach cyklu, který projde všechny prvky v poli:

```php
<?php
$barvy = Array('červená', 'zelená', 'žlutá', 'fialová');
foreach ($barvy as $barva)
{
        echo $barva . ' ';
}
```

Obrázek 23:Výpis pole.

Výstup z volání příkazu „echo“ bude řetězec znaků „červená zelená žlutá fialová“. Tečka spojuje znaky do jednoho řetězce a dva apostrofy vloží mezi hodnoty mezeru.

Pole se dají i vnořovat jedno do druhého:

```php
<?php
$pole[2] = "Text";
$pole[3] = 12;
$pole[8] = Array("Vnořené pole", 8);
print_r($pole);
```

Obrázek 24: Vnořené pole.

Výpis $pole potom vypadá následovně:

```
Array
(
    [2] => Text
    [3] => 12
    [8] => Array
        (
            [0] => Vnořené pole
            [1] => 8
        )

)
```

Obrázek 25:Výpis vnořeného pole.

Základní informace o polích byla nastíněna. Nyní je možné se vrátit k řešení problému webové aplikace a načítaní hodnot do elementu „select“ s atributem „multiple“ tzv. „multi dropdown list“. Nejprve je nutné opět načíst data z databáze. V tomto případě je dotaz o něco složitější, jelikož muselo dojít k propojení celkem tři tabulek. Tabulka „vyrobci“ s tabulkou „oblast_cinnosti“ přes pomocnou tabulku „vyrobci_cinnosti“, protože se jedná o n:m vazbu. Dotaz ještě byl samozřejmě doplněn o „WHERE“ klauzuli, aby se vracel jen výsledek konkrétního výrobce:

```
$sql2 = "SELECT oblast_cinnosti.id FROM vyrobci INNER JOIN vyrobci_cinnosti ON vyrobci.id = vyrobci_cinnosti.id_vyr
                                INNER JOIN oblast_cinnosti ON oblast_cinnosti.id = vyrobci_cinnosti.id_cin
                                  WHERE vyrobci.id  = ?";
try {
          $stmt = $db->prepare($sql2);
          $stmt->execute(array($UID));
          $result = $stmt->fetchAll(PDO::FETCH_ASSOC);
        } catch (PDOException $e) {
          die("Error!: " . $e->getMessage() . "<br/>");
          }
```

Obrázek 26:Načítaní dat z databáze.

Po vrácení výsledku v poli, je nutné si uvědomit, že se záznamů vrátí více a to podle toho, kolik bylo vybráno možností v roletě. Například výrobce „Jindřišské dráhy“ je po provedení „selectu“ na obr. 27 zobrazen hned několikrát a to právě z důvodů činností, které je schopen vykonávat.

```
SELECT * FROM vyrobci INNER JOIN vyrobci_cinnosti ON vyrobci.id = vyrobci_cinnosti.id_vyr INNER JOIN oblast_cinnosti ON oblast_cinnosti.id = vyrobci_cinnosti.id_cin WHERE vyrobci.id = 153297
```

Prof

Zobrazit vše | **Obnovit pořadí sloupců** | Počet řádků: 25 | Filtrovat řádky: Vyhledávání v této tabulce

+ Nastavení

id	nazev	id	nazev	popis	web	stat	mesto	dokumenty	spoluprace
153297	Jindřišské dráhy	5	Správce infrastruktury	Především se zabývá žel. prov. ale častečně i siln...	www.jd.cz	Czech Republic	Jindřichův Hradec	K432-K324	Nespolupracuje
153297	Jindřišské dráhy	7	Technologie dopravních systémů	Především se zabývá žel. prov. ale častečně i siln...	www.jd.cz	Czech Republic	Jindřichův Hradec	K432-K324	Nespolupracuje
153297	Jindřišské dráhy	8	Dopravní značení	Především se zabývá žel. prov. ale častečně i siln...	www.jd.cz	Czech Republic	Jindřichův Hradec	K432-K324	Nespolupracuje
153297	Jindřišské dráhy	9	Stavební činnost	Především se zabývá žel. prov. ale častečně i siln...	www.jd.cz	Czech Republic	Jindřichův Hradec	K432-K324	Nespolupracuje

Obrázek 27: Výsledek dotazu „phpmyadmin“.

Pokud se podíváme na pole, které bylo tímto „selectem“ naplněno uvidíme (pro lepší přehlednost jsou vypsána jen „ID“ oblastí činnosti):

„array(4) {
[0]=> array(1) {
["id"]=> string(1) "5"
}
[1]=> array(1) {
["id"]=> string(1) "7"
}
[2]=> array(1) {
["id"]=> string(1) "8"
}
[3]=> array(1) {
["id"]=> string(1) "9"
}
}“

Stejně jako v předchozí kapitole věnované stručnému seznámení s polem se jedná o vnořené pole. Jen s tím rozdílem, že vnořené pole v tomto případě má vždy jen jeden prvek o jednom znaku. Je to logické, jelikož se jedná právě o identifikátor oblasti činnosti jednotlivého výrobce. Společnost Jindřišské dráhy s příkladem oblasti činností: správce infrastruktury, technologie dopravních systémů, dopravní značení a stavební činnost. Nejprve bylo nutné vyřešit problém a vybrat pouze všechny identifikátory výrobců a vložit je do pole. Jelikož se jedná o vnořené pole, byly použity dva cykly. První cyklus, který operuje ve vnějším poli a druhý vnitřní cyklus, který získává „ID“ výrobce z vnořeného pole. Před začátkem cyklu bylo založeno prázdné pole, do kterého se průběžně ukládala data přesně, jak je tomu na obr. 28.

```
$checksub = array();

    foreach ($result as $item){
        foreach ($item as $item2)
        {
            }

$checksub[] = ($item2);
                }
```

Obrázek 28: Cyklus pro vytvoření pole s identifikátorem výrobce.

Podobně, jak tomu bylo u rolet s jedním možným výběrem, i zde je nutné načíst hodnoty z databáze. Aby bylo možné přiřazení správných hodnot výběru z více možností, je zapotřebí načíst hodnoty do pole. Proto, jak je vidět na obr 29, je název rolety „oblast_cin[]“.

```php
<select name="oblast_cin[]" size="13" required multiple id="select14">
  <?php
  $sql="SELECT * FROM oblast_cinnosti ORDER BY nazev ";
  try {
      $select = $db->query($sql);
      $result = $select->fetchAll(PDO::FETCH_ASSOC);
  } catch (PDOException $e) {
      die("Error!: " . $e->getMessage() . "<br/>");
  }
```

Obrázek 29:Načtená roleta do pole.

Opět cyklem je zkontrolována shodnost. Jen se v tomto případě nehledá přes všechny hodnoty v poli a nejsou porovnávány s proměnnou, nýbrž se využívá funkce „in_array“, která porovnává, zda se hledaná hodnota tedy identifikátor činnosti výrobce nenachází v poli. V tomto případě v poli „checksub“. Pokud se v poli nachází, je tato možnost v roletě zvýrazněna. Touto cestou se zkontrolují všechny možnosti v roletě.

```php
foreach($result as $row){
echo '<option value="'.$row['id'].'"'.(in_array($row['id'], $checksub)  ? ' selected="selected"' : '').'>'.$row['nazev'].'</option>';
}
```

Obrázek 30: Cyklus s funkcí „in_array“.

Vytváření seznamu záznamů z databáze.

Tato podkapitola se věnuje vytváření seznamu záznamů z databáze. Stejně jako v ostatních případech, kde byly načítány data z databáze, tak zde je potřeba načíst všechna data výrobců. Je vhodné si již v SQL dotazu předpřipravit sloupce pro samostatnou tabulku.

```php
$sql = "SELECT vyrobci.id AS id,vyrobci.nazev AS nazev,popis,web,stat,mesto,dokumenty,spoluprace,autor,datum,oblast_cinnosti.nazev AS cinnost
FROM vyrobci, oblast_cinnosti, vyrobci_cinnosti
WHERE vyrobci.id = vyrobci_cinnosti.id_vyr
    AND oblast_cinnosti.id=vyrobci_cinnosti.id_cin
    GROUP BY vyrobci.nazev
    ORDER BY vyrobci.nazev";


try {
$select = $db->query($sql);
$result = $select->fetchAll(PDO::FETCH_ASSOC);
} catch (PDOException $e) {
    die("Error!: " . $e->getMessage() . "<br/>");
}
```

Obrázek 31: Načítání dat do seznamu.

Po spuštění funkce se vykoná databázový dotaz. A vrátí se všechny řádky, které dotaz vybral. Samotný SQL dotaz obsahuje 3 tabulky. Tabulku „vyrobci“, „vyrobci_cinnosti“ a „oblast

cinnosti“, které jsou spojeny přes cizí klíče. Je vidět, že jsou v příkazu použity i aliasy, které zjednoduší v dalším kroku vypisování jednotlivých sloupců. Zde nepoužíváme klauzuli „WHERE“ pro „SELECT“ konkrétního záznamu, ale pro propojování tabulek právě přes cizí klíče. V dotazu je použita i agregační funkce „GROUP BY“, kvůli duplicitám výsledků dotazu. Jak je uvedeno v předchozí kapitole, v dotazu se vyskytuje n:m vazba mezi tabulkami „vyrobci“ a „oblast_cinnosti“. To znamená, že by nebylo bez agregační funkce dosaženo podobné odpovědi, jak je tomu na obr. 27. Ale v seznamu výrobců nechceme mít vícekrát téhož samého výrobce či firmu, a proto je využito této funkce. Dále se jen záznamy seřadí vzestupně podle jména výrobce

Výsledkem dotazu je pole řádků, které je uloženo do proměnné „$result“. V hlavičce tabulky jsou názvy sloupců, které musejí v cyklu odpovídat počtu buněk, které byly do tabulky vloženy. Následně je pole pomocí „foreach“ cyklu řádek po řádku procházeno a funkcí „echo“ je vygenerován seznam záznamů do HTML tabulky. Ve skriptu si ještě nachází IF podmínka, která podle oprávnění umožnuje záznam upravovat, či mazat.

```
echo '<table id="myTable" class="vyber_vyrobci"><thead><tr><th>Jméno firmy</th><th>Popis firmy</th><th>Web</th><th>Stát</th><th>Město</th><th>Dokumenty</th><th>Oblast činnosti</th>
        <th>Spolupráce s FD</th><th>Autor</th><th>Vytvoření záznamu</th><th></th><th></th><th></th><th></th><th></th></tr></thead>';
echo '<tbody>';

foreach($result as $row){
    echo "<tr><td>" . $row["nazev"]. "</td><td>" . $row["popis"]. "</td><td>" . $row["web"]. "</td><td>". $row["stat"]. " </td><td>". $row["mesto"]. "</td><td>". $row["dokumenty"].
        "</td><td>". $row["cinnost"]."</td><td>". $row["spoluprace"]. "</td><td>". $row["autor"]. "</td><td>". $row["datum"]. "</td><td><a href='vyrobci_nahled.php?id=".$row['id'].
        "'><img src='./images/eye.png'></a></td>";
          if ($prava == 5) {
            echo" <td><a href='vyrobci_edit.php?id=".$row["id"].
        "'><img src='./images/edit.png'></a></td></td><td><a href='delete_vyrobci.php?id=".$row["id"].
        "'class='jelitko'><img src='./images/delete.png'</a></td>";
          } else {
            echo "<td></td><td></td>";
          }
    echo "</tr>";
}
echo "</table>";
```

Obrázek 32:Kód pro seznam výrobců v tabulce.

4. Uživatelské prostředí

Prostředí aplikace bylo navrženo tak, aby bylo přehledné pro všechny uživatele různých uživatelských rolí, interakce s ním byla intuitivní, a aby se dokázalo přizpůsobit koncovým zařízením klienta.

4.1. Požadavky na klienta

Pro klientské zařízení by mělo stačit, aby obsahovalo internetový prohlížeč, který dokáže spouštět JavaScript. Pro zvýšení uživatelského komfortu je doporučeno, aby prohlížeč podporoval i standardy HTML5 a CSS3, pro zobrazení stránky to však není nutné.

Webová aplikace byla úspěšně testována s prohlížeči Mozilla Firefox verze 45.0.1, Google Chrome verze 49.

4.2. Popis aplikace

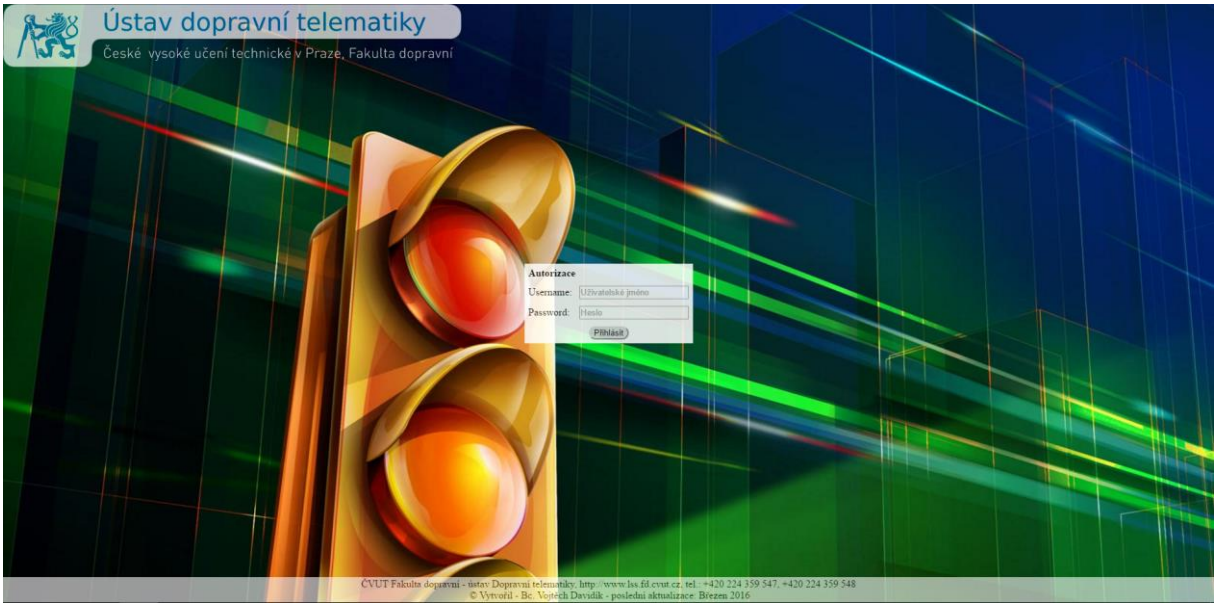
Všechny stránky aplikace mají společné rozložení. To zahrnuje horní pravý pruh, kde je zobrazeno uživatelské jméno přihlášené osoby a možnost odhlásit se. Všem uživatelům je pak zobrazen dolní pruh obsahující informaci o autorství aplikace.

Obrázek 33: Přihlašovací lišta.

4.2.1. Úvodní stránka

Úvodní stránka je stručná a obsahuje pouze název fakulty a ústavu. V centru stránky je umístěn přihlašovací formulář a v patičce je umístěno autorství aplikace. Pro přístup do aplikace je nutné vyplnit správné uživatelské jméno a heslo.

Není žádná jiná možnost, jak se standardně dostat dále do aplikace. Na stránce se nenacházejí jiné prvky.



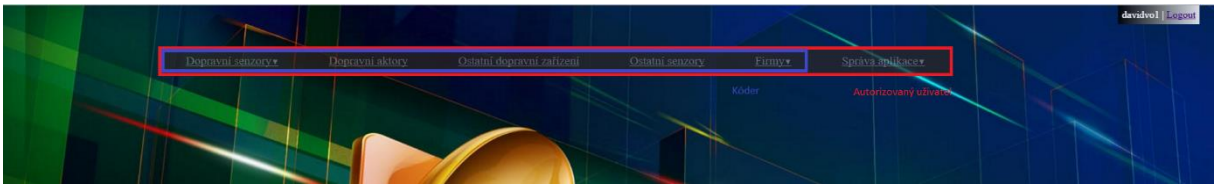
Obrázek 34: Úvodní stránka.

4.2.2. Menu stránka

Informace zobrazené na základní stránce po přihlášení je opět v podobě přihlašovací lišty, patičky a menu. Podle práva jsou uživatelům změněny jednotlivé možnosti výběru v lištovém menu. Autorizovaný uživatel má přístup pro správu aplikace. Tedy práva pro vytváření nových uživatelských přístupů, včetně přiřazení uživatelských rolí a právo pro sledování přidělených záznamů kódérům. Toto sledování není automatické a spouští se, až na základě požadavku autorizovaného uživatele.

Menu je zachyceno na obr. 35, kde je patrné, že menu obsahuje několik základní možnosti na výběr: dopravní senzory, dopravní aktory, ostatní dopravní zařízení, ostatní senzory a firmy. Z návrhu datové struktury vyplívá, že aplikace je rozdělená do modulů. A to podobně jako je tomu ve výběru v menu. Modul dopravní senzory je nejrozsáhlejší a nejnáročnější pro realizaci. Struktura tohoto modulu vzešla na základě rešeršní práce, z které bylo čerpáno při vytváření aplikace. Ostatní zbylé moduly ještě nebyly blíže popsány a jejich samotná realizace a implementace do webové aplikace proběhne po analýze později. Tedy je nutné vytvořit požadavky na zařízení jednotlivých modulů, nejlépe podle výsledků z rešeršních prací dopravních technologií. Podobně, jak tomu bylo u dopravních senzorů. Tento modul je plně funkční a na jeho základě dojde ke zkonstruování i zbylých částí aplikace. V nabídce se nacházejí i firmy. Firmy působí v menu jako další samostatný modul, ale neplatí to zcela.

Firmy ostatní moduly sjednocují, jak je patrné ze struktury databáze. Avšak v kategorii firem jsou umožněný identické operace jako v jiných modulech.



Obrázek 35: Menu.

4.2.3. Stránky seznamu záznamů

Aplikace nejenže, musí umět vytvářet a editovat záznamy, ale také je musí umět zobrazit v přehledné formě. K tomuto účelu slouží seznamy, do kterých je umístěna podle nějakého klíče záznamy z databáze. Klíčem se myslí interní atributy, podle kterých se rozhoduje, zda daný záznam je relevantní a jeho umístění v konečném seznamu je správné. Jedná se o atributy v administrátorském řádku. Tyto atributy nejsou nikde v seznamu viditelné a o jejich změny se zasluhují vyšší autority webové aplikace.

Stránka seznamu dopravních senzorů

Na stránce modulu dopravní senzory se nachází hned v úvodu seznam záznamů dopravních senzorů. V seznamu jsou pouze ty zařízení, které prošly řádným schválením. Na obr. 36 je tento seznam patrný. Obsahuje základní informace o dopravním senzoru jako je: název, typ technologií, typ detektoru, jméno autora, datum vytvoření a informace o editování. Dále se na konci v jednotlivých řádcích seznamu zobrazují tři ikony:

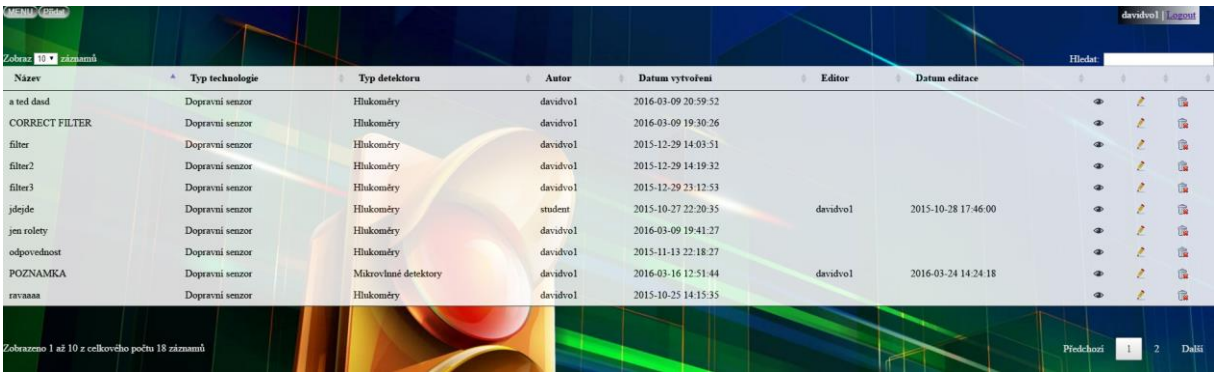

Nástroj slouží k zobrazení detailu konkrétního zařízení.


Nástroj slouží k editaci záznamu.


Nástroj pro smazání záznamu.

Tyto nástroje jsou dostupné v závislosti na pravomocích uživatelů. Dostupné všechny možnosti a neomezenou správu nad záznamy má autorizovaný uživatel. Kodér a uživatel prohlížeč mají nabídku nástrojů omezenou. Samotný seznam nabízí volbu počtu zobrazených záznamů, přepínání jednotlivých stran katalogu v pravém dolním rohu a dynamické jedno kriteriální vyhledávaní.

Na stránce existuje klasická lišta s přihlášením a opět patičku. V levém horním rohu přibyly dvě tlačítka. Tlačítko „MENU“ odkazuje uživatele zpět na základní stánku po přihlášení a tlačítko „Přidat“ slouží pro přidání nového záznamu.



Obrázek 36: Seznam dopravních technologií.

Stránka seznamu výrobců

Na stránce je k dispozici seznam, který obsahuje nejdůležitější informace o firmách, které se zabývají dopravními technologiemi. Firmy spojují ostatní dopravní zařízení, a proto by zde mohla existovat vyšší kontrola vkládaných záznamů firem. Avšak důvod, proč není i zde autorizace záznamů, je v tom, že další kontrolní cyklus, by blokoval kontrolu vkládání záznamů dopravních zařízení.

Kódér při vkládání nové technologie je nucen určit firmu, která technologii vyrábí, popřípadě distribuuje atd. Ale problém by nastal, pokud by tato firma nebyla v možnostech výběru rolety ve formuláři pro vložení nové dopravní technologie. Jediné východisko by z této situace existovalo v uložení rozpracovaného formuláře, tedy uložení bez možnosti „závazného odeslání“. Tento krok by zaručil, že čas investovaný do vyplnění formuláře by nebyl ztracený. Další krok by následoval v otevření formuláře pro vkládání nových firem,

vyplnění a odeslání požadavku k vyšší autoritě pro uložení nové dopravní firmy do databáze. Nyní by došlo k delegovanému problému, jelikož kodér by musel čekat na reakci autorizovaného uživatele ohledně potvrzení nové firmy, nemohl by se tedy vrátit k dodělání formuláře a celý proces by byl pozastaven, až do doby, než by došlo k validaci nové firmy ze strany autorizovaného uživatele.

Tyto dva cykly kontroly záznamů, cyklus záznamu kontroly dopravní technologie a cyklus kontroly záznamu dopravní firmy, který je podmíněn, by vedly k velkým prodlevám a tím pádem i ke snížení efektivitě vkládaných záznamů. Z toho důvodu nedošlo k navázání podobného mechanismu pro vkládání nových záznamů firem jako je tomu u dopravních technologií.

Obrázek 37: Seznam firem.

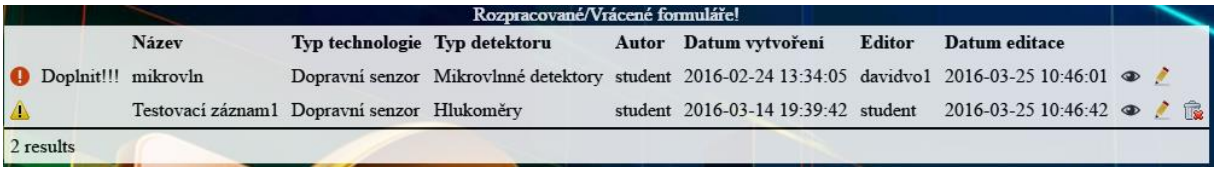
Speciální seznamy záznamů

Pro přehlednost byly v aplikaci vytvořeny kromě hlavního konečného seznamu, z kterého lze vyhledávat včetně multikriteriálního filtru, i speciální seznamy. Tyto seznamy slouží pro orientaci v záznamech., jelikož v aplikaci existuje mechanismus pro jejich správu a validaci. Jak je vidět na obrázku 38, kde se nachází dva záznamy. Každý záznam se liší hned v hlavičce řádku ikonou. Nutno doplnit, že každý kodér má unikátní seznam podle množství nezávazně odeslaných záznamů, popřípadě vrácených nebo přeposlaných autorizovaným uživatelem. V aktuální verzi webové aplikace jsou speciální seznamy umístěny spolu s hlavním seznamem. Po zpřístupnění i ostatních modulů, by bylo vhodné, aby speciální seznamy byly umístěny na své vlastní stránce. Popřípadě umístěny na stránce, která nemá bližší přimknutí na konkrétní modul. Typy jednotlivých záznamů, obsažených ve speciálním seznamu odlišených ikonou:

Označuje záznam ve speciálním seznamu, který nebyl závazně odeslán. Všechna data jsou ve formuláři uložena. Předpokládá se, že uživatel se k záznamu vrátí a po řádném doplnění záznam odešle.

Označuje záznam ve speciálním seznamu, který byl vrácen nebo přeposlán autorizovanou osobou zpět kódérovi na doplnění, zpřesnění informací ve formuláři. Standardně se nesrovnalosti popíší v poznámce, která se zobrazuje ihned v náhledových parametrech seznamu.

Ikona „new“ označuje nově závazně odeslaný záznam od kódera. Autorizované osobě se po přihlášení zobrazují tyto záznamy a čekají na schválení, popřípadě na úpravu samotným autorizovaným uživatelem nebo odeslání zpět na přepracování kóderovi.



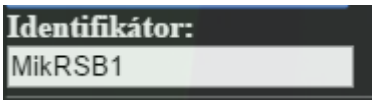
Obrázek 38: Speciální seznam.

4.2.4. Stránka pro vkládání nových záznamů dopravní technologie

Formulář pro vkládání nových záznamů obsahuje řadu textových polí, rolet s jedním výběrem a rolet s více možností výběru. Formulář se skládá z jedné základní tabulky obecných parametrů a potom také s dynamicky se měnícími tabulkami specifických parametrů podle výběru v roletě s názvem typ detektoru.

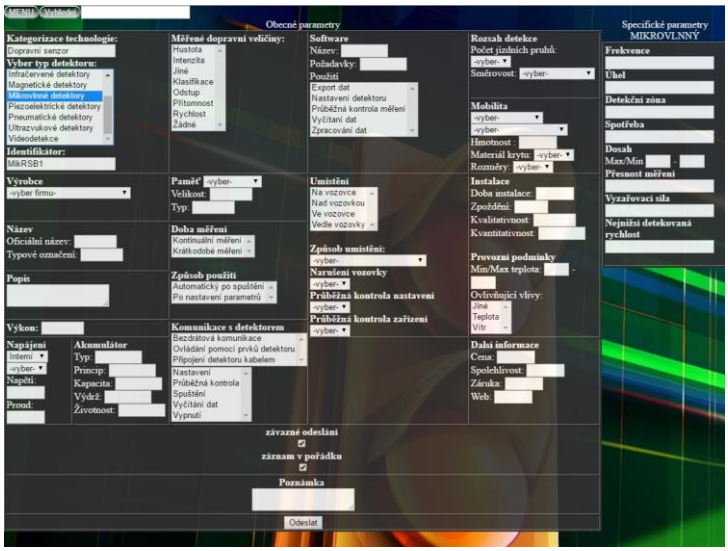
Každému záznamu je automatický přiřazeno identifikační číslo. Identifikace jednotlivých záznamů je unikátní v celé databázi. Na obr. 39 je vidět identifikátor, který nejen identifikuje jednoznačně záznam, ale i nese krátkou informaci. První tři písmena označují skupinu senzorů.

Mik – mikrovlnný, Mag – magnetický, Ind – indukční atd. Další zbylé čtyři znaky tvoří generátor náhodných čísel a písmen pro unikátnost záznamu v celé databázi.



Obrázek 39:Identifikátor.

Rolety s více možností výběru se dají částečně uživatelsky „customizovat“. V databázovém manažeru „phpmyadmin“, lze přidávat další možnosti k výběru. Změna se projeví ve všech formulářích a stránkách. Rozšiřovat nebo upravovat rolety je možné u všech tabulek v prefixem „roleta_“. Změna se projeví až po obnovení stránky.



Obrázek 40: Vkládání nových záznamů.

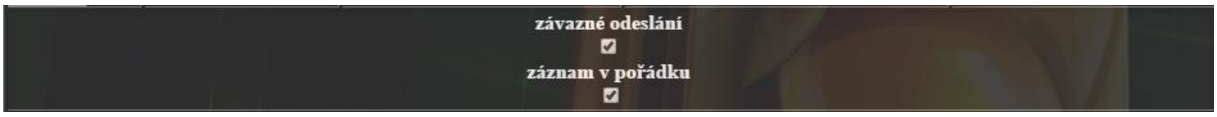
Řádek administrátorských nástrojů

Formulář pro vkládání záznamů obsahuje speciální řádek pro administraci záznamu. Opět podle práva se řádek upravuje. V režimu kódéra řádek nabízí možnost tzv. závazného odeslání. To znamená, že po zaškrtnutí tohoto kontrolního boxu bude záznam odeslán výše

k autorizovanému uživateli. Kódér i přesto, že je autorem záznamu, ztrácí nad ním kontrolu, až do doby, kdy záznam zkontroluje autorizovaný uživatel a použije možnost vrácení záznamu zpět na dopracování kódérovi.

Další možnost, jak se kódér může dostat k editaci záznamu, který byl již závazně odeslán, je výběr konkrétního kódéra při kontrole záznamu. I v tomto případě podnět musí zadat autorizovaný uživatel. Jestliže kodér nezaškrtne závazné odeslání a formulář odešle, nedojde k možnosti kontrole tohoto záznamu autorizovaným uživatelem, ale záznam bude zobrazen ve speciálním seznamu na dopracování. Identickou operaci se záznamem, může samozřejmě učinit i autorizovaný uživatel.

Pokud uživatel bude mít dostatečná práva, zobrazí se mu v administrátorském řádku i přímá možnost vložení záznamu do databáze. Tedy po zaškrtnutí políčka „závazné odeslání“ a „záznam v pořádku“ se přesměruje ihned do databáze do konečného seznamu připraven k vyhledávání.



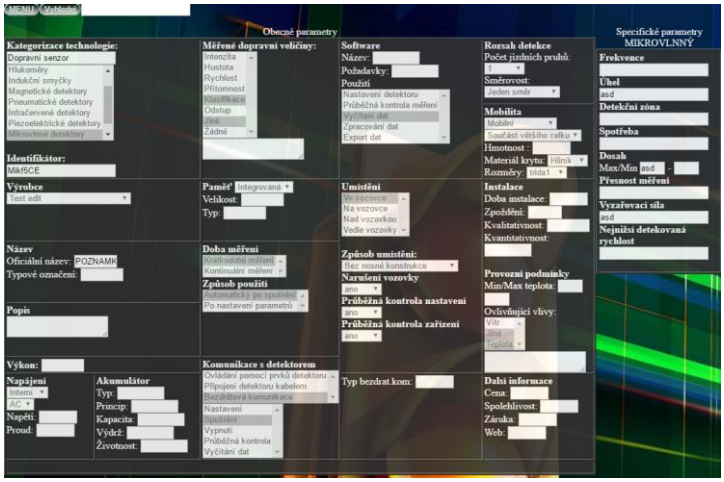
Obrázek 41:Administrátorský řádek ve formuláři vkládání záznamů.

Administrátorský řádek ve formuláři pro kontrolu záznamu, který má ve své kompetenci uživatelská role autorizovaného uživatele, se drobně liší. V řádku je možnost odeslat záznam v předefinované volbě autorovi (kódérovi) záznamu, popřípadě je možnost vybrat i jiného kódéra pomocí rolety. Pokud autorizovaný uživatel posoudí záznam a uváží, že je v pořádku, tak potvrdí políčko „Poslat do katalogu“. Z logiky vyplívá, že není možné záznam poslat do katalogu a zároveň odeslat zpět kódérovi, proto je kombinace možností omezena jen na východisko „odeslat zpět“ nebo „poslat do katalogu“. Poslední zaškrtávací kontrolní box nabízený v administrátorském řádku je volba: „nepovolit mazání“. Po potvrzení tohoto pole dojde k evidenci záznamu a spárování s odpovědným kódérem vybraný v roletě „odpovědný kodér“. Přístup k těmto údajům o odpovědnosti je v menu při dostatečných právech uživatele.

Obrázek 42: Administrátorský řádek ve formuláři kontrola záznamů.

4.2.5. Stránka pro detailní zobrazení záznamů dopravní technologie

Přistup na stránku je zajištěn pomocí náhledové ikony. Po kliknutí kurzorem myši na ikonu je na ni uživatel aplikace odkázán. Na stránce se nachází vyplněný formulář dopravních technologií včetně specifických atributů konkrétních zařízení. Do formuláře není možné zasahovat a upravovat ho. Stránka je přístupná pro všechny uživatelské role.



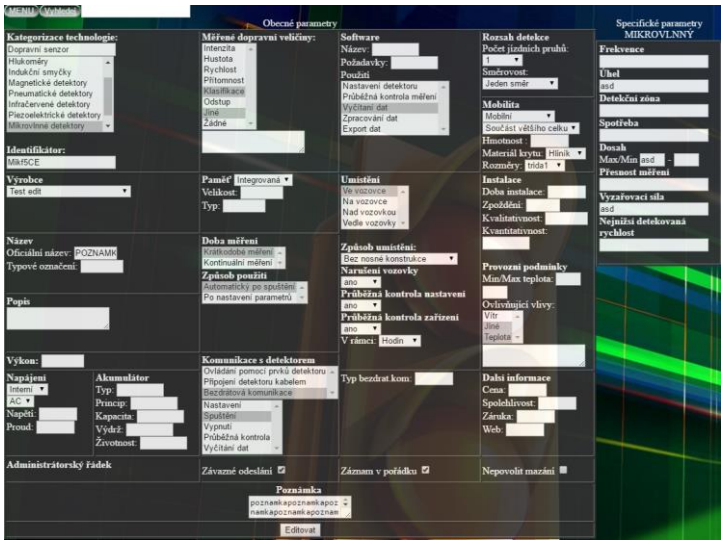
Obrázek 43: Stránka pro náhled záznamu.

4.2.6. Stránka pro editaci záznamů dopravní technologie

Stránka pro editaci záznamu umožnuje upravovat záznam. Zasahováno může být do všech textových polí, rolet s jedním výběrem i rolet s více možností výběru. Pouze v textovém poli identifikátoru a roletě kategorizace technologie nelze upravovat.

Opět podle pravomocí uživatele je k dispozici, popřípadě není dostupný administrátorský řádek, který umožnuje daný záznam z výsledného seznamu vyřadit na úroveň kontroly autorizovaného uživatele, nebo dokonce vrátit na přepracování kódérovi.

K tomu dojde po odškrtnutí pole „záznam v pořádku“, „odeslat zpět“. Poznámka slouží k další připojené informaci, která se přímo netýká samotného zařízení na technické úrovni, nýbrž specifikuje jasné požadavky uživatelů pro manipulaci se záznamem.



Obrázek 44: Stránka pro editaci záznamu.

4.2.7. Stránka filtrace dopravní technologie

Stránka slouží k vyhledávání v konečném seznamu dopravních detektorů. Tedy v těch záznamech, které byly schváleny autorizovaným uživatelem a bylo u nich potvrzené pole „záznam v pořádku“. Formulář obsahuje pouze ucelené výrazy a rolety s předefinovanou možností výběru. Tímto krokem se předejde nesrovnalostem v textovém poli a tím vyhledávaní nesprávných řetězců znaků.

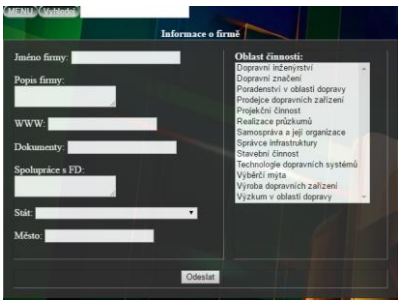
Filtrovací skript umožňuje vyhledávat přes libovolné množství rolet. Pokud nebude vybrána žádná možnost z rolet, tak ve výsledném seznamu vyhledávaní se kritéria neprojeví. Jestliže při vyhledávání dojde jen k částečnému prolnutí kritérií filtru s hledanými zařízeními, jsou tyto zařízení vyhodnoceny jako kladné a jsou vypsány z konečného vyhledávacího seznamu.

Například pokud existuje v konečném seznamu pro vyhledávání detektor, který měří tyto dopravní parametry: rychlost, přítomnost, intenzita. V kritériích filtru bude, že jsou hledána všechna zařízení, která měří rychlost a přítomnost, tak i když zde není přímý průnik, dopravní senzor bude zařazen do výsledku vyhledávání.

Obrázek 45: Stránka filtrace dopravních senzorů.

4.2.8. Stránka pro vkládání nových záznamů firem

Stránka obsahuje formulář pro specifikaci dopravních firem. Data po odeslání jsou ihned připravena pro samotné vyhledávání a pro zobrazení v roletě výrobci ve formuláři dopravních technologiích. Formulář obsahuje několik textových polí a jednu roletu s více možností výběru. Identifikátor záznamu je automaticky vygenerován a uložen se zbytkem informace o firmě do databáze.



Obrázek 46: Stránka pro vkládání nových záznamů o firmě.

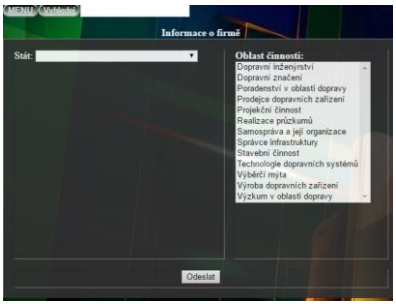
4.2.9. Stránka pro editaci záznamů firem

Formulář je opět jako u dopravních technologií předvyplněný informacemi z databáze. Přístup je povolen do všech polí a rolet jak pro uživatelskou roli kodér, tak i pro autorizovaného uživatele. Identifikátor, jelikož nenese žádnou další informaci, je skryt.

Obrázek 47: Stránka pro editaci firem.

4.2.10. Stránka filtrace firem

Formulář pro filtraci je strohý, jelikož se filtrují pouze ustálené výrazy.



Obrázek 48:Stránka filtrace firem.

4.2.11. Stránky pro správu webové aplikace

Tyto stránky mají přístupné pouze vyšší autority z hlavní nabídky. Slouží pro vytváření nových přístupů do webové aplikace, popřípadě pro jejich editaci. Stránka pro zobrazení odpovědnosti záznamů se používá tehdy, pokud je potřeba dohledat kódéra a odeslaný záznam, pokud byla zaškrtnuta možnost „nepovolit mazání“ v administrátorském řádků při kontrole záznamu. Autorizovaný uživatel pak jednoduše může zjistit, komu záznam byl odeslán.

Stránka správy uživatelských účtů

Formulář obsahuje základní atributy pro zřízení nového přístupu do aplikace.

Stránka pro hlídání odpovědnosti záznamů

Stránka zobrazuje uživatelské jméno a záznam, za který uživatel odpovídá.

5. Testování webové aplikace

Aplikace jako celek byla po samotné realizaci testována studenty na ústavu Dopravní telematiky. Volba spočívala ve výběru vhodných testerů, kteří již mají odbornou znalost v dopravní telematice, ale zároveň nebyly úzce spjati s webovou aplikací. A tak mohli na případné nedostatky a závady narazit díky obecnějšímu pohledu.

Testovací prostředí běží na serveru ústavu Dopravní telematiky ve verzi databázového serveru 5.5.45 – MySQL a verzi webového serveru 5.4.45 – PHP.

5.1. Návrh testování

Návrh testování byl sestrojen tak, aby byla prověřována funkčnost a chování jednotlivých nástrojů pro jednotlivé uživatelské přístupy. Testování probíhalo v několika běžně používaných prohlížečích, kterými jsou zejména Google Chrome, Mozilla Firefox a Opera. V rozlišení full HD 1920*1080.

Důležitější částí testování, na které se v návrhu zaměřilo, bylo ověření bezpečnosti jednotlivých nástrojů, zejména možnost neoprávněného přístupu a využívání funkcí uživatelem bez patřičných práv. Bylo ověřováno, aby nepřihlášený uživatel nemohl využívat funkce přihlášeného, a aby přihlášený uživatel bez administrátorských oprávnění nemohl využívat a přistupovat k nástrojům určeným administrátorovi. V neposlední řadě byly prověřeny formuláře pro odeslání, editaci a vyhledávaní záznamů podle kritérií. Všechny nalezené chyby byly reportovány a po jejich opravě došlo k opětovnému testování.

5.2. Realizace testovacích případů

Testovací případ, často se využívá i anglický výraz „test case“, popisuje konkrétní akce prováděné s určitou softwarovou komponentou a jejich očekávané výsledky [18]. Softwarovou komponentou v tomto případě může být například část aplikačního rozhraní nebo také softwarový systém běžící na několika strojích souběžně. Testovací případy se vytvářejí jak pro manuální, tak i automatizované testy. Pro účely manuálního testování jsou tvořeny seznamem prováděných kroků a očekávaných výsledků. Automatizované testovací případy se také někdy označují jako testovací skript. Tvoří je sada programových instrukcí a na rozdíl od manuálních by měly být schopny sami rozpoznat, zda uspěly, či selhaly. U větších projektů se testovacích případů vytváří běžně desítky až stovky.

Testovací případ je tedy dokument, popisující určitou činnost, kterou je potřeba otestovat. Obecně lze říci, že obsahuje kroky se skutečnými vstupními hodnotami spolu s očekávanými výsledky.

V kapitole Testovací formuláře jsou zobrazeny jednotlivé kroky pro správu testovacího případu. Dále je popsáno a vysvětleno několik podstatných vlastností, které přispívají ke kvalitě testovacího případu.

Identifikátor – jedinečný identifikátor, na který se budu odkazovat z ostatních dokumentů

Účel – vysvětlení k čemu daný testovací případ slouží

Podmínky – seznam potřebných dat či vlivů prostředí podstatných pro provedení testovacího případu

Specifikace kroků a vstupů – seznam všech kroků a souvisejících vstupních dat, nutných pro úspěšné a opakovatelné provedení testovacího případu

Očekávané výsledky – souhrn všech informací, potřebných k určení, zda případ proběhl úspěšně či nikoliv [19]

5.2.1. Testovací formuláře

Testování aplikace proběhlo podle tohoto testovacích scénáře. Poslední verze aplikace, která byla testována, vedla na tyto výsledky:

ID 1: Formulář přihlašování

Účel testovacího případu: „formulář přihlašování“ je pro ověření, zda proběhne přihlášení uživatele, a zda jsou přístupné příslušné mechanismy uživatelských práv aplikace dle uživatelských rolí (mechanismy pro správu uživatelských účtů, odpovědnosti kóderů za přiřazené záznamy a zobrazení administrátorského řádku pouze v případě přístupu uživatelské role „autorizovaný uživatel“).

Typ testu je verifikační pro uživatelské role: autorizovaný uživatel, kodér.

Podmínka pro testování je vytvoření přístupů pro uživatele do aplikace.

Jednotlivé kroky testování:

1. Zobrazit přihlašovací stránku webové aplikace.

2. Vyplnit přihlašovací formulář (uživatelské jméno, heslo).

    2.1. Správné uživatelské jméno; správné heslo.

    2.2. Správné uživatelské jméno; nesprávné heslo. (Záměrně se vyplňuje nesprávné heslo, aby se zjistilo, zda se zamezí přístupu nepovoleným osobám. Volím náhodný řetězec znaků a prázdné pole.).

    2.3. Nesprávné uživatelské jméno; správné heslo. (Záměrně se vyplňuje nesprávné přihlašovací jméno, aby se zjistilo, zda se zamezí přístupu nepovoleným osobám. Volím náhodný řetězec znaků a prázdné pole).

    2.4. Nesprávné uživatelské jméno; nesprávné heslo. (Záměrně se vyplňuje nesprávné heslo a zároveň i nesprávné přihlašovací jméno, aby se zjistilo, zda se zamezí přístupu nepovoleným osobám. Volí se náhodné řetězce znaků a prázdná pole).

Výsledky testování:

1. Proběhlo zobrazení stránky s formulářem (bez diakritických a jiných vad).

2. Podle jednotlivých kroků došlo na tyto varianty:

    2.1 Úspěšné přihlášení do aplikace.

    2.2 Nepřihlášení do aplikace a zobrazení textu: „Wrong username or password.“

    2.3 Nepřihlášení do aplikace a zobrazení textu: „Wrong username or password.“

2.4 Nepřihlášení do aplikace a zobrazení textu: „Wrong username or password.“

ID 2: Formulář pro ukládání nových záznamů dopravních technologií

Účel testovacího případu: „formulář pro ukládání nových záznamů dopravních technologií“ je pro ověření, zda se ukládají všechny dostupné informace o dopravních technologiích na úrovní uživatelské role kodér/autorizovaný uživatel.

Typ testu je verifikační pro uživatelské role: autorizovaný uživatel, kodér.

Podmínka pro testování je vytvoření přístupů pro uživatele do aplikace. Role kodér/autorizovaný uživatel.

Jednotlivé kroky testování:

1. Přihlášení do webové aplikace.
2. Přejít na stránku s formulářem pro ukládání nových záznamů dopravních technologií.
    2.1. Vybrat konkrétní možnost resp. možnosti, pokud je roleta s více možnostmi výběru, v nabídce rolety.
    2.2. Vyplnění všech textových polí formuláře pro ukládání nových záznamů ve správném datovém formátu.
3. Odeslat formulář.
    3.1 Odeslat formulář bez zaškrtnuté možnosti: závazné odeslání a záznam v pořádku (jen pro autorizovaného uživatele) v roli kódéra i autorizovaného uživatele.
    3.2 Odeslat formulář se zaškrtnutou možnosti závazného odeslání v roli kódéra
    3.3 Odeslat formulář se zaškrtnutými možnostmi: „závazné odeslání“ i „záznam v pořádku“ v roli autorizovaného uživatele.
4. Zkontrolovat v seznamu dopravních technologií, zda se nachází vytvořený záznam, který byl odeslán v uživatelské roli autorizovaný uživatel s atributy: „závazné odeslání“ a současně „záznam v pořádku“.
    4.1 Podívat se do detailního náhledu, zda se ke všem testovaným atributům připsaly zadané hodnoty, které byly zadány na vstupu ve formuláři při zakládání nového záznamu.
5. V případě nezaškrtnutí kontrolního pole pro závazné odeslání (platí jak pro uživatelskou roli kodér, tak i pro autorizovaného uživatele), zkontrolovat, zda se vytvořil nový záznam ve speciálním seznamu
    5.1 Podívat se do detailního náhledu respektive editace záznamu, zda se ke všem testovaným atributům připsaly zadané hodnoty, které byly zadány na vstupu ve formuláři při zakládání nového záznamu
6. Pro případ uživatelské role „kódér“, kdy se potvrdí kontrolní box „závazně odeslat“ dojde k odeslání záznamu, který je k dohledání ve speciálním seznamu v uživatelské roli „autorizovaný uživatel“.
    6.1 Přihlásit se jako „autorizovaný uživatel“ a zkontrolovat, zda opravdu existuje daný záznam, včetně všech uložených informací, připravených ke schválení.

Výsledky testování:

1. Proběhlo přihlášení do aplikace bez jakýchkoliv problémů.
2. Proběhlo zobrazení prázdného formuláře včetně administrátorského řádku, či nikoliv. A to podle uživatelských práv - administrátorský řádek zobrazen pouze pro uživatelskou roli: „autorizovaný uživatel“.
   2.1. V roletě byla možnost vybrat jednu nebo více možností, a to podle toho, zda se jedná o roletu s jedním nebo s více možnostmi výběru. Po výběru se konkrétní možnost zvýraznila.
   2.2. Na základě našeptávání, které se spustilo při podržení myši na daném textovém poli, bylo možné vyplnit informaci ve správném formátu.
3. Formulář se úspěšně odeslal bez chyb.
   3.1. Záznam byl k dohledání při přihlášení daného kódéra popřípadě autorizovaného uživatele v jeho speciálním seznamu.
   3.2. Záznam byl k dohledání ve speciální tabulce přidaných nových záznamů pro autorizované uživatele.
   3.3. Záznam se odeslal do konečného vyhledávacího seznamu s kompletními daty zadanými do formuláře při ukládání.
4. V konečném seznamu pro vyhledávání se vytvořil záznam.
   4.1. V bližším náhledu byly k dohledání všechny uložené informace o dopravní technologii z formuláře, který byl vyplněn včetně správného formátu a odeslán do databáze k uložení.
5. Záznam se vytvořil ve speciální tabulce konkrétního uživatele.
   5.1. Záznam v bližším detailu obsahoval všechny zadané informace včetně správného formátu.
6. Odeslaný záznam se objevil ve speciální tabulce nových záznamů k vidění pro všechny autorizovaný uživatele.
   6.1. Při přihlášení do aplikace v uživatelské roli: „autorizovaný uživatel“ se v náhledu nově přidaného záznamu zobrazily všechny informace zadané na vstupu kódérem včetně možností vložené poznámky.

ID 3: Formulář pro editování záznamů dopravních technologií

Účel testovacího případu „formulář pro editování záznamů dopravních technologií" je pro ověření, zda se ukládají všechny editované informace o dopravních technologiích na úrovní uživatelské role kodér/autorizovaný uživatel.

Typ testu je verifikační pro uživatelské role: autorizovaný uživatel, kodér.

Podmínky pro test je vytvoření přístupu pro uživatelské role autorizovaný uživatel a kodér.

Jednotlivé kroky testování:

1. Přihlášení do webové aplikace
2. Přejít na stránku se seznamem uložených záznamů dopravních technologií v roli kódéra (Uživatelská role „kódér" nemá práva pro editaci záznamů v konečném vyhledávacím seznamu. Pro otestování tohoto editovacího formuláře, je nutné nový záznam vytvořit a posléze uložit jako rozpracovaný, tedy uložit bez zaškrtnuté možnosti: "záznam v pořádku").
    2.1. Vybrat záznam pro editaci tak, že se kurzorem myši klikne na ikonu tužky, která odkáže uživatele do editaci záznamu.
    2.2. Vybrat náhodná textová pole a výběrové rolety s jednou nebo s více možnostmi výběru.
3. Odeslat formulář.
    3.1 Pro autorizovaného uživatele odeslat formulář bez zaškrtnuté možnosti: „závazné odeslání" a „záznam v pořádku", v roli kódéra bez zaškrtnuté možnosti „záznam v pořádku".
    3.2 Odeslat formulář se zaškrtnutou možností „závazného odeslání" v roli kódéra.
    3.3 Odeslat formulář se zaškrtnutými možnostmi: „závazné odeslání", „záznam v pořádku" v roli autorizovaného uživatele.
4. Zkontrolovat v seznamu dopravních technologií, zda se nachází editovaný záznam, který byl odeslán v uživatelské roli autorizovaný uživatel s atributy: „závazné odeslání" a současně „záznam v pořádku".
    4.1 Přejit do detailního náhledu a zkontrolovat, zda se ke všem testovaným atributům připsaly zadané hodnoty, které byly zadány na vstupu ve formuláři při editování záznamu.

5. V případě nezaškrtnutí možnosti pro „závazné odeslání“ (platí jak pro uživatelskou roli kodér, tak i pro autorizovaného uživatele), zkontrolovat, zda proběhla editace ve speciálním seznamu.
   5.1 Přejít do detailního náhledu respektive editace záznamu, zda se ke všem testovaným atributům připsaly zadané hodnoty, které byly zadány na vstupu ve formuláři při editaci záznamu.
6. Pro případ uživatelské role „kódér“, kdy se potvrdí možnost „závazně odeslat“ dojde k odeslání záznamu, který je k dohledání ve speciálním seznamu v uživatelské roli autorizovaného uživatele.
   6.1 Přihlásit se jako „autorizovaný uživatel“ a zkontrolovat, zda opravdu existuje daný záznam, včetně všech uložených a editovaných informací, připravených ke schválení.

Výsledky testování:

1. Proběhlo přihlášení do aplikace bez jakýchkoliv problémů.
2. Zobrazil se seznam uložených záznamů v konečném vyhledávání a seznam záznamů ve speciálních seznamech (rozpracované záznamy resp. vrácené).
   2.1. Proběhlo zobrazení editovaného formuláře včetně administrátorského řádku, či nikoliv a to podle uživatelských práv. Administrátorský řádek zobrazen pouze pro uživatelskou roli autorizovaného uživatele.
   2.2. V roletě se zvýraznila jedna nebo více možností, a to podle toho, zda se jedná o roletu s jedním nebo s více možnostmi výběru a podle daného předchozího výběru odeslaného formulářem. V textových polí se nachází vyplněné informace, které lze editovat, až na výjimku pole, které určuje identifikátor a typ dopravního senzoru.
3. Formulář se úspěšně editoval bez chyb.
   3.1. Záznam byl k dohledání při přihlášení daného kódéra popřípadě autorizovaného uživatele v jeho speciálním seznamu rozpracovaných záznamů.
   3.2. Záznam byl k dohledání ve speciální tabulce, která zobrazuje přidané nové záznamů pro autorizované uživatele.
   3.3. Záznam byl odeslán do konečného vyhledávacího seznamu s kompletními daty zadanými do formuláře při ukládání.
4. V konečném seznamu pro vyhledávání se editoval záznam.

4.1. V bližším náhledu byly k dohledání všechny uložené a editované informace o dopravních technologiích z formuláře, který byl vyplněn včetně správného formátu a odeslán do databáze k uložení.

5. Záznam se vytvořil ve speciální tabulce konkrétního uživatele.
   5.1. Záznam v bližším detailu obsahoval všechny zadané, popřípadě editované informace včetně správného formátu.
6. Odeslaný záznam se objevil ve speciální tabulce nových záznamů k vidění pro uživatelskou roli: „autorizovaný uživatel“.
   6.1. Při přihlášení do aplikace v uživatelské roli autorizovaný uživatel se v náhledu resp. editovacím formuláři nově přidaného záznamu zobrazí všechny změny zadané na vstupu kódérem s možností vložené poznámky.

ID 4: Formulář pro autorizaci záznamů

Účel testovacího případu „formulář pro autorizaci záznamů“ slouží pro ověření, zda funguje správně mechanismus pro ověřování záznamů.

Typ testu je verifikační pro uživatelské role: autorizovaný uživatel, kodér.

Podmínky pro testování je vytvoření přístupů pro uživatele do aplikace. Role kodér/autorizovaný uživatel.

Kroky testování:

1. Přihlášení do webové aplikace.
2. Vytvořit záznam v roli kódéra se zaškrtnutou možností „závazné odeslání“.
3. Přihlásit se v uživatelské roli „autorizovaný uživatel“.
   3.1. Přejit na stránku seznamu záznamů dopravních technologií.
   3.2. Kurzorem myši kliknout na název konkrétního záznamu v seznamu.
4. Odeslat formulář s editovanou změnou a vloženou poznámkou.
   4.1 Odeslat formulář bez žádné zaškrtnuté možnosti v administrátorském řádku: „odeslat zpět“, „poslat do katalogu“, „nepovolit mazání“.
   4.2 Odeslat formulář jen se zaškrtnutou možností „odeslat zpět“. A ve výběru zvolit kódéra, kterému bude záznam odeslán.

4.3 Odeslat formulář se zaškrtnutou možností: „odeslat zpět“ a „nepovolit mazání“. Ve výběru se definuje kódér, kterému bude záznam odeslán.

4.4 Odeslat formulář jen se zaškrtnutou možností „poslat do katalogu“.

5. Zkontrolovat v konečném vyhledávacím seznamu dopravních technologií, zda se nachází editovaný záznam, který byl odeslán v uživatelské roli autorizovaný uživatel s atributy „poslat do katalogu“.

5.1 Nahlídnout do detailního náhledu, zda se ke všem testovaným atributům připsaly zadané hodnoty, které byly zadány na vstupu ve formuláři při editování záznamu.

6. V případě odeslání záznamu kontrolnímu kódérovi, tedy s atributem „odeslat zpět“ a výběrem kódéra, zkontrolovat úspěšné přijetí odeslaného záznamu.

6.1 Zkontrolovat v detailním náhledu respektive editace záznamu, zda se ke všem testovaným atributům připsaly zadané hodnoty, které byly zadány na vstupu ve formuláři při editaci záznamu před samotným odesláním.

7. V případě odeslání záznamu kontrolnímu kódérovi, tedy s atributem: „Odeslat zpět“ a výběrem kódéra a atributem nepovolit mazání, zkontrolovat úspěšné přijetí odeslaného záznamu autorizovaným uživatelem a dále, zda není možné záznam smazat v uživatelské roli kódér, a zda je záznam k dohledání v seznamu „Záznamy –Odpovědnost“ ve správě aplikace přístupný pro autorizované uživatele.

7.1 Zkontrolovat v detailního náhledu respektive editace záznamu, zda se ke všem testovaným atributům připsaly zadané hodnoty, které byly zadány na vstupu ve formuláři při editaci záznamu před samotným odesláním.

Výsledky testování

1. Proběhlo přihlášení do aplikace bez jakýchkoliv problémů.
2. Záznam se vytvořil a odeslal do speciální tabulky dopravních technologií, kde vyčkal na kontrolu autorizovaného uživatele.
3. Proběhlo úspěšné přihlášení do aplikace jako autorizovaný uživatel.

3.1. Zobrazil se seznam speciální tabulky nově přidaných záznamů.

3.2. Proběhlo detailní zobrazení určitého záznamu včetně všech předvyplněných polí a rolet.

4. Došlo k odeslání záznamu.

4.1. Záznam zůstal viset ve speciální tabulce autorizovaného uživatele

4.2. Záznam byl přesměrován do speciální tabulky vybraného kódéra včetně všech náležitostí.

4.3. Záznam byl přesměrován do speciální tabulky vybraného kódéra včetně všech náležitostí bez možnosti smazání.

4.4. Záznam se odeslal do konečného vyhledávacího seznamu s kompletními daty zadanými do formuláře při ukládání.

5. V konečném seznamu pro vyhledávání se editoval záznam.

5.1. V bližším náhledu byly k dohledání všechny uložené a editované informace o dopravních technologií z formuláře, který byl vyplněn včetně správného datového typu a odeslán do databáze k uložení.

6. Záznam se vytvořil ve speciální tabulce konkrétního kódéra.

6.1. Záznam v bližším detailu obsahoval všechny zadané, popřípadě editované informace včetně datových typů.

7. Záznam se vytvořil ve speciální tabulce konkrétního kódéra, kde nebylo povoleno záznam smazat a dále se záznam propojil s daným kódérem. Tato vazba je k dohledání ve správě aplikace „Záznamy-Odpovědnost“.

7.1. Záznam v bližším detailu obsahoval všechny zadané, popřípadě editované informace včetně správného formátu.

7.2. Závěr testování

Podle testovacího scénáře, který se skládá z testovacích případů, se otestovala funkčnost webové aplikace. Webová aplikace si prošla během svého vývoje celou řadou testů. Výsledky z těchto testů se staly základem pro vyřešení mnoha problémů. Verze, která je součástí přílohy, byla otestována přesně podle testovacích případů výše. Všechny testované mechanismy byly schopné dostát své funkcionality.

7. Závěr

Hlavním cílem diplomové práce bylo vytvořit webovou aplikaci přístupnou pouze autentizovaným uživatelům, která jim umožní dle uživatelské role vkládat, editovat, nahlížet, odstraňovat a vyhledávat záznamy. V prvním kroku se přistoupilo ke sběru, specifikaci a analýze požadavků, na jejichž výsledcích se aktualizovala datová struktura databáze, aby vyhovovala požadavkům a splňovala kritéria pro ukládání a spravování dat o dopravních technologiích. Vytvořili se tedy nové entity a vztahy mezi nimi, včetně určení datových typů. Před samotnou realizací proběhla důkladná analýza uživatelských potřeb, které vzešly ze specifikace požadavků. Na kvalitní specifikaci požadavků přímo závisí úspěch celého projektu. Proto tato fáze byla detailně analyzována, jelikož další změna a to i nepatrná může vest ke změně datové struktury, včetně přepisování značné části zdrojového kódu. Situace byla o to náročnější, že návrh musel předpokládat i další rozšíření webové aplikace. A proto je bezesporu velkou výhodou se v problematice, která webová aplikace obsahuje a které se věnuje, vyznat nejenom jako vývojář, ale i jako dopravní inženýr.

Sběr požadavků jako podklad pro budoucí návrh probíhal v dlouhém období. Začal již během zpracovávání letáků, brožur a dokumentace věnované dopravní technologii. Z velké míry se zformoval během tvorby bakalářské práce a vývoj postupoval dále při zpracování návrhu diplomové práce. Po diskuzích s vedoucím práce se dospělo i k nadefinování funkčních požadavků. Tedy té funkce, které je systém povinen v rámci aplikace realizovat. A dále se identifikovali jednotlivé typy uživatelů s uživatelskými rolemi a k ním se přiřadili pravomoci nutné nejen pro přihlášení, ale hlavně pro správu záznamů. Po skončení fáze specifikace požadavků se začalo s návrhem webové aplikace. Navrhl se datový model, který se skládal z množiny entit později nejčastěji realizovaných jako tabulka relační databáze, které jsou provázány vazbami s příslušnou násobností (kardinalitou). Jako webové rozhraní byla použita PHP aplikace s prvky jQuery pro příjemnější a snazší uživatelskou interakci.

Během vytváření a implementace mechanismů pro autorizaci uživatelů, správu záznamů a jiné jsem si osvojil základy PHP jazyka, včetně SQL, HTML a CSS. Lehce jsem narazil i na jQuery. Myslím si, že jsem dostatečné pochopil problematiku, aby mohly být splněny všechny cíle v zadání práce. Vznikla přihlašovací stránka s formulářem pro autorizaci uživatelů. Dále bylo vytvořeno několik stránek pro vytváření, editaci a odstraňování záznamů. Vyvinuly se stránky s formuláři i pro zobrazování seznamu záznamů dopravních senzorů, výrobců a

speciálních seznamů pro autorizaci záznamů. Navrhlo se a realizovalo více kriteriální vyhledávání. A vznikl mechanismus pro autorizaci nových záznamů.

Testování webové aplikace probíhalo pomocí testovacích případů. Během testování, kdy se testovali především funkční mechanismy autorizace záznamu, ale i vkládání a editace na všech stránkách a formulářích aplikace, se objevilo několik problémů více, či méně závažných, které se podařilo odladit. Poslední verze aplikace již žádné závady nevykazovala.

Jak již bylo zmíněno webová aplikace je rozdělaná do několika virtuálních sekcí, kde je plně zprovozněna nejkomplikovanější z nich dopravní senzory a rozpracován modul zabývající se firmami. Jednotlivé kroky při vývoji aplikace byly zdokumentovány a na jejich základech se mohou zkonstruovat i zbylé moduly dopravních technologií. Samozřejmě po návrhu, který bude vyplývat ze závěru analýzy v technické rovině.

V praktické části práce se podařilo vytvořit aplikaci, která dokáže skladovat a spravovat informace o dopravních senzorech a firmách. Plně připravená aplikace dokáže poskytovat data o všech typech dopravní technologie a v kombinaci s GIS aplikací se stane součástí mocného systému pro efektivní analýzu dopravních zařízení. Bude umožňovat sdílet informace týkajících se nejen popisných, ale i polohopisných charakteristik objektů. Dokáže lokalizovat dopravní technologie na konkrétním území s následnou vizualizaci. A tím se stane výborným pomocným nástrojem pro vytváření studií, analýz a modelů zabývající se nejen oboru dopravního inženýrství. Další využitelnost aplikace je v možnosti porovnávat dopravní technologie, vybrat vhodné zařízení na bázi více kriteriálního výběru nebo například jen poskytnout informace o konkrétním senzoru, aktoru či firmě.

Součástí přílohy je zdrojový kód webové aplikace a struktura databáze.

8. Citovaná literatura

[1] Management mania, „*Webová aplikace (Web Application)*,“ 2016. [Online], [cit. 12. 3. 2016] Dostupné z WWW: https://managementmania.com/cs/webova-aplikace-web-application.

[2] Davidík V., *Návrh databáze dopravních technologií*, Praha, 2014, Bakalářská práce. ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE Fakulta Dopravní.

[3] Novák J., *Webová aplikace pro evidenci hydrologických událostí v krajině*, Ostrava. Bakalářská práce. VYSOKÁ ŠKOLA BÁŇSKÁ - TECHNICKÁ UNIVERZITA OSTRAVA, Institut geoinformatiky, 2010.

[4] Faltýnek J., *Návrh a tvorba webových aplikací*, Brno, 2011, Diplomová práce. MASARYKOVA UNIVERZITA Fakulta informatiky.

[5] J. &. S. Robertson, „*Volere- Šablona pro specifikaci požadavků*,“ Atlantic Systems Guild, 2003. [Online], [cit. 14. 4. 2016] Dostupné z WWW: http://www.volere.co.uk/pdf%20files/template_cz.pdf.

[6] Zelenka P., „*WebML – navigační model*“ Interval.cz, 2004 [Online], [cit. 25. 3. 2016]. Dostupné z WWW: https://www.interval.cz/clanky/webml-navigacni-model/

[7] T. P. Group, „*PHP Manual*,“ 2013. [Online], [cit. 28. 3. 2016]. Dostupné z WWW: http://cz2.php.net/manual/en/index.php.

[8] ZELENKA P., *Webová aplikace na platformě java pro odevzdávání dat a jejich vyhodnocení*, Brno , 2013, Diplomová práce. VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ Fakulta elektrotechniky a komunikačních technologií.

[9] The_php_group, „*php.net*,“ 2016. [Online], [cit. 22. 3. 2016] Dostupné z WWW: http://php.net/manual/en/intro.pdo.php.

[10] Oracle Corporation, „*MySQL*“ 2016. [Online], [cit. 22. 3. 2016] Dostupné z WWW: http://www.mysql.com/.

[11] W3Schools, „*Introduction to HTML*“ 2016. [Online], [cit. 24. 3. 2016]. Dostupné z WWW: http://www.w3schools.com/html/html_intro.asp.

[12] W3Schools, „*CSS Tutorial*,“ [Online], [cit. 30. 3. 2016]. Dostupné z WWW: http://www.w3schools.com/css/default.asp.

[13] A. JavaScript, „*Mozilla Developer Network*“ [Online], [cit. 24. 3. 2016]. Dostupné z WWW: https://developer.mozilla.org/en-US/docs/.

[14] R. Havlíček, *Webová aplikace pro astronomické pozorovací deníky*, Brno, Masarykova Univerzita, fakulta informatiky, 2015.

[15] Linuxsoft, „*PHP (36) - Připojujeme se k MySQL*“ 2004. [Online], [cit. 6. 2. 2016]. Dostupné z WWW: http://www.linuxsoft.cz/article.php?id_article=336.

[16] W3school, „*PHP 5 Sessions*“ 2016. [Online], [cit. 12. 2. 2016]. Dostupné z WWW: http://www.w3schools.com/php/php_sessions.asp.

[17] Interval, „*Začínáme používat sessions v PHP*“ 2002. [Online], [cit. 6. 3. 2016]. Dostupné z WWW: https://www.interval.cz/clanky/zaciname-pouzivat-sessions-v-php/.

[18] PAGE, A. J. K., *Jak testuje software Microsoft*, Brno: Computer Press. 2009. 384 s. ISBN 978-80-251-2869-5

[19] Creative Commons, „*Test Case – Testovací případ*,“ [Online], [cit. 10. 5. 2016]. Dostupné z WWW: http://testovanisoftwaru.cz/dokumentace-v-testovani/test-case/

9. Seznam obrázků

10. Seznam příloh.

Příloha č.1: E-R Diagram

Příloha č.2: SQL skript pro vytvoření datové struktury.

Příloha č.3: Zdrojové kódy webové aplikace.