VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

WEB PRO ZPRACOVÁNÍ HDR OBRAZU A VIDEA

WEB FOR HDR IMAGE AND VIDEO PROCESSING

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **MARTIN BERAN**
AUTHOR

VEDOUCÍ PRÁCE — **Doc. Ing. MARTIN ČADÍK, Ph.D.**
SUPERVISOR

BRNO 2017

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2016/2017

Zadání bakalářské práce

Řešitel: **Beran Martin**
Obor: Informační technologie
Téma: **Web pro zpracování HDR obrazu a videa**
Web for HDR Image and Video Processing
Kategorie: Uživatelská rozhraní

Pokyny:

1. Seznamte se s problematikou akvizice a zpracování HDR obrazu a videa.
2. Proveďte důkladnou rešerši existujících aplikací pro zpracování HDR, včetně webových služeb.
3. Navrhněte a implementujte webový systém pro zpracování HDR obrazu a videa. Systém bude umožňovat zejména složení HDR obrazu a videa z několika expozic (vč. registrace) a mapování tónů (tone mapping), včetně časově koherentního mapování (temporal tone mapping).
4. Se systémem experimentujte, posuďte jeho vlastnosti při převodu HDR obrazů a videa a diskutujte možnosti budoucího vývoje.
5. Dosažené výsledky prezentujte formou videa, plakátu, článku, apod.

Literatura:

- http://pfstools.sourceforge.net/
- http://cadik.posvete.cz/tmo/

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Čadík Martin, doc. Ing., Ph.D.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2016
Datum odevzdání: 17. května 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
612 66 Brno, Božetěchova 2

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

Hlavním cílem této bakalářské práce je vytvoření webového systému, pomocí kterého lze převádět fotografie a videozáznamy do formátu HDR a ty pak dále upravovat a zobrazit. Během zpracování a úpravy lze sledovat aktuální vzhled na miniatuře výsledné fotografie. Po dokončení úprav lze výsledný soubor zobrazit a stáhnout v plném rozlišení. Pro interakci s uživatelem je připraveno rozhraní pro nahrávání fotografií a sada operátorů a filtrů pro jejich úpravu.

Abstract

The main goal of this bachelor thesis is the creation of web system, by which it is possible to convert photos and videos into HDR format and then adjust and display them. During processing and adjusting it is possible to see the actual view in the miniature of subsequent photo. After completing the adjusments it is possible to display and download the final file in full resolution. For interaction with user is prepared interface for uploading photos and a set of operators and filters for their adjustment.

Klíčová slova

web, webová aplikace, HDR fotografie, HDR video, mapování tónů, pfstools

Keywords

web, web application, HDR imaging, HDR video, tone mapping, pfstools

Citace

BERAN, Martin. *Web pro zpracování HDR obrazu a videa.* Brno, 2017. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Čadík Martin.

Web pro zpracování HDR obrazu a videa

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Martina Čadíka, doc. Ing., Ph.D. Dále prohlašuji, že jsem uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Martin Beran
17. května 2017

Poděkování

Velmi rád bych poděkoval panu Martinu Čadíkovi, doc. Ing., Ph.D. za věnovaný čas, odborné vedení a cenné rady poskytnuté při řešení bakalářské práce.

Obsah

Kapitola 1

Úvod

HDR (High Dynamic Range) fotografie nabízejí mnohem vyšší dynamický rozsah, než běžně pořizované fotografie či video a umožňují tak zobrazit všechny detaily i tehdy, kdy to už není prostřednictvím běžných metod možné. Nejčastějším způsobem vytvoření HDR fotografie je zkombinováním několika různě exponovaných dílčích fotografií. Hlavní výhodou HDR snímků je především v rozsahu barev. Pro běžné formáty, jako je např JPEG nebo BMP, je pro barevný rozsah použito 8 bitů u každé barvy. Teprve dodatečně je počítáno osvětlení. Oproti tomu HDR systém definuje kromě barev také množství světla pro každý pixel snímku. Pro uložení celé informace jsou používána čísla s plovoucí desetinnou čárkou. Díky tomu je výsledný rozsah prakticky neomezený.

Cílem této bakalářské práce je vytvořit webovou aplikaci, ve které bude možné běžné videozáznamy a série různě exponovaných fotografií převést do HDR formátu

Druhá kapitola popisuje některé z existujících aplikací a webových služeb, které se zabývají problematikou vytváření HDR obrazu a jeho následnou úpravou, jejich možnosti a výhody, případně nevýhody oproti této práci.

Ve třetí kapitole jsou podrobně popsány všechny programovací jazyky a technologie, které byly při vytváření bakalářské práce využity.

Čtvrtá kapitola se zabývá návrhem a implementací celé aplikace. Jsou zde popsány jednotlivé části návrhu uživatelského rozhraní, struktura celého systému, a implementace všech jeho částí. Jsou zde vysvětleny a popsány důležité třídy a metody, způsob zpracování nahraných souborů, možnosti jejich úpravy a také různé způsoby uložení výsledků. Dále je tu popsáno rozhraní administrátorské části systému, sloužící pro jeho správu.

V poslední kapitole jsou popsány způsoby a výsledky testování vytvořeného systému.

Kapitola 2

Existující řešení

V této kapitole je čtenář seznámen s již existujícími řešeními této problematiky. Budou zde popsány možnosti a výhody i nevýhody jednotlivých řešení vzhledem k této bakalářské práci.

2.1 Software

2.1.1 easyHDR

Komerční software, který poskytuje široké možnosti nastavení, zarovnání a „odstranění duchů“ při vytváření HDR fotografie, nabízí širokou škálu operátorů tónování a podporuje velké množství HDR formátů.

2.1.2 Adobe PhotoShop

V tomto případě se jedná o software pro kompletní úpravu obrazu. Z toho důvodu je také jeho uživatelské rozhraní poněkud složitější na ovládání. Zejména možnost vytvoření HDR snímku ze série fotografií je poněkud těžké najít. V případě úspěchu jsou pak uživateli nabízeny čtyři metody tónování. Editování parametrů a tónovací křivky je však povoleno pouze u některých. Při následném uložení upraveného souboru je nabízeno několik různých formátů typu LDR, včetně RAW formátů.

2.1.3 Photomatix

V současnosti se jedná o zřejmě nejlepší software pro úpravu fotografií. Nabízí široké možnosti nastavení a opravdu velký výběr operétorů pro mapování tónů. Vzhledem k jeho vysoké ceně je však využíván především pro profesionální účely.

2.1.4 Luminance HDR

Tento software je volně šiřitelný a multiplatformní. Nabízí široké množství operátorů mapování tónů a editaci několika parametrů každého z nich. Provedené změny v rámci operátoru však nejsou vidět v reálném čase, ale až po stisku tlačítka tónování. Program také umožňuje uložení použité konfigurace a její následné načtení.

2.1.5 Picturenaut

Jedná se o volně dostupný software vytváření a zpracování HDR fotografií. Uživatelské rozhraní je velice jednoduché a přehledné, přestože není příliš přívětivé. Program umožňuje nahrát sekvenci snímků s nízkým dynamickým rozsahem, které jsou poté převedeny do jediného HDR souboru.Umožněno je také přímé nahrání HDR soubou.

V základní verzi program obsahuje čtyři metody mapování: bilaterální3.5.2, adaptivní logaritmickou3.5.1, metodu fotoreceptoru3.5.7. Poslední metodou úpravy je modifikace hodnoty expozice. Při úpravě lze také vybrat barevnou hloubku fotografie, zobrazit poslední použitou úpravu nebo uložit parametry aktuální úpravy, případně je načíst z připraveného souboru.

Upravený snímek lze exportovat do HTML stránky nebo uložit v několika různých HDR formátech.

2.1.6 FDR Tools

Tento software nemá příliš příjemné uživatelské rozhraní, za to ale nabízí nejširší možnosti nastavení a úprav. Po výběru série fotografií lze při jejich kombinování editovat celkový dynamický rozsah nebo minimální a maximální jas. Po vytvoření HDR snímku je nabízeno několik operátorů mapování tónů. U každého z nich lze editovat veškeré jeho parametry a lze přímo měnit také křivku tónování.

2.2 Webové služby

2.2.1 Fotor

Tato webová aplikace[1] má velice příjemné, přehledné a intuitivní uživatelské rozhraní s jednoduchým ovládáním. Pro neregistrované uživatele má však velké množství omezení. Výběr souborů pro vytvoření HDR souboru je omezen na dva nebo tři. Výběr více či méně snímků není povolen. Po vytvoření HDR souboru jsou k dispozici tři filtry pro jeho úpravu. U každého lze upravit kontrast, jas, saturaci, ostrost a intenzitu použitého filtru. Výsledný snímek však nelze bez registrace uložit či sdílet. Dalším omezením pro nepřihlášené uživatele je to, že přes střed vytvořeného snímku je přidaný vodoznak použité webové služby.

2.2.2 HDR converter

Uživatelské rozhraní a celkový design této aplikace[2] je také velice jednoduchý a intuitivní. Lze vybrat libovolný počet snímků nízkého dynamického rozsahu či videozáznamů. Úprava již předem vytvořených HDR souborů není povolena. Po nahrání je zpracováván každý snímek zvlášť. Je k dispozici množství předem definovaných filtrů pro úpravu každého z nich. Jakékoliv další parametry vybraného filtru nejsou zobrazeny. Mezi snímky lze pohodlně přepínat a na každý z nich lze aplikovat jiný filtr. Po dokončení úprav lze aktuálně zobrazený snímek stáhnout ve formátu JPG.

[1] http://www.fotor.com/app.html#!module/hdr/tool/Hdr
[2] https://hdrconverter.com/

2.2.3 Convertio

Jedná se o online nástroj[3] pro konverzi souborů mezi téměř jakýmikoliv formáty. Lze nahrát libovolný počet souborů určených ke konverzi a u každého z nich vybrat formát, do kterého bude zkonvertován. Žádné úpravy při konverzi do HDR formátu zde nelze provádět. Pro nepřihlášené uživatele je povolena konverze pouze dvou souborů najednou.

2.2.4 Photo-kako

Poslední nalezenou webovou aplikací[4], která se zabývá konverzí fotografií do HDR formátu je služba Photo-kako. Její uživatelské rozhraní je velice nepřehledné. Najednou lze nahrát pouze jeden snímek s nízkým dynamickým rozsahem. Nahrávání připravených HDR souborů opět není podporováno. Při zpracování či úpravě nahraného souboru není uživateli nijak patrné, zda aplikace pracuje nebo došlo k nějaké chybě, což je také dost matoucí. Po nahrání je k dispozici pět různých možností filtrů a široké možnosti dalších úprav. Nicméně vzhledem k velmi dlouhému načítání webu při každé provedené úpravě a nepřehlednému uživatelskému rozhraní je tato aplikace prakticky nepoužitelná.

[3] https://convertio.co/jpg-hdr/
[4] http://www.photo-kako.com/en/hdr.cgi

Kapitola 3

Použité technologie

V této kapitole je čtenář seznámen s technologiemi a prostředky, které byly pro vypracování této bakalářské práce použity. Budou zde vysvětleny důvody, proč jsou zmíněné jazyky, technologie a knihovny použity a jejich porovnání s různými dostupnými alternativami. Znalost těchto technologií je klíčová pro pochopení procesu vytváření této bakalářské práce. Jelikož se jedná o webovou aplikaci, hlavními použitými jazyky byly HTML, CSS, JavaScript a PHP. Jako vývojové prostředí jsem zvolil PhpStorm 2016.2.2. Aplikace byla vyvíjena a testována na systému Ubuntu 16.04 64-bit.

3.1 HTML5

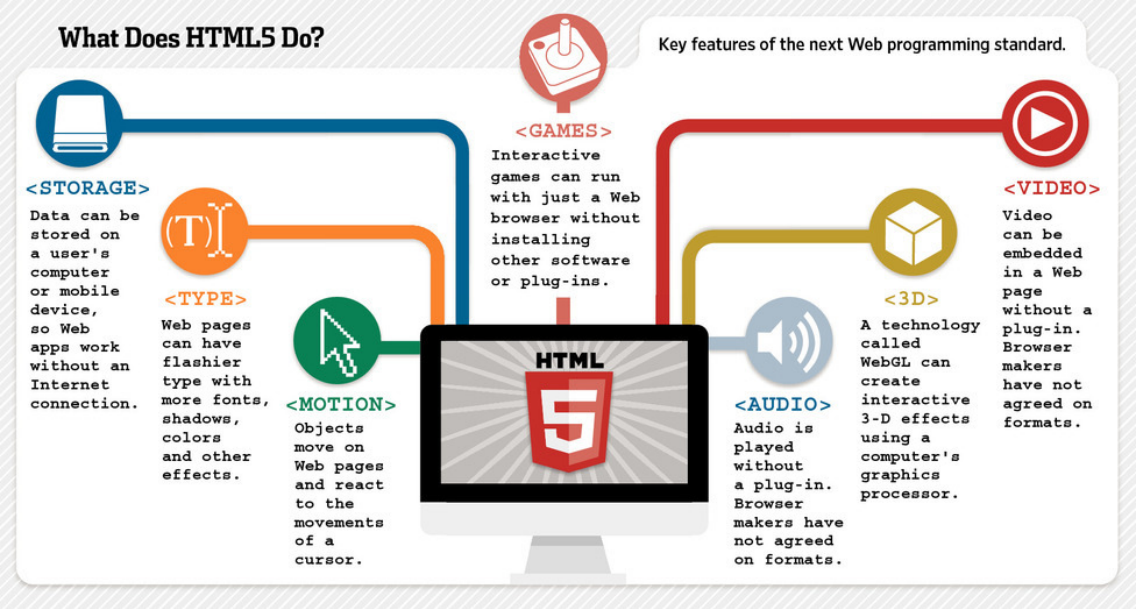
Jazyk HTML (HyperText Markup Language) je jeden z hlavních jazyků pro vytváření stránek v systému World Wide Web, který umožňuje publikaci dokumentů na internetu. Jazyk je aplikací již dříve vyvinutého univerzálního značkovacího jazyka SGML (Standard Generalized Markup Language) a je nejpoužívanějším jazykem pro tvorbu webových stránek a webových aplikací, které jsou mezi sebou propojeny hypertextovými odkazy. Vývoj jazyka HTML byl a stále je ovlivněn vývojem webových prohlížečů, zejména směrem a způsobem jejich vývoje.

Jazyk jsem do své práce zařadil proto, že se jedná o nejběžnější a také nejpoužívanější způsob vytváření dokumentů na internetu. Největší výhodou tohoto jazyka je, že každý element je charakterizován množinou značek. Těmto značkám lze přiřadit různé identifikátory a lze je zařadit do jednotlivých tříd. Díky tomu lze jednotlivé elementy uchopit, odkázat se na ně nebo jim přiřadit styl nebo animaci zobrazení definované jazyky CSS a javascript. Jazyk HTML je tedy jeden z nejvhodnějších pro definici významu obsahu webových stránek.

Nejnovější a aktuálně používanou verzí jazyka HTML je HTML5. Tato verze byla vydána v roce 2014 po patnácti letech vývoje od předchozí verze. Není již závislá na jazyku SGML, opravuje velké množství chyb předešlé verze, vyřazuje mnoho zastaralých a nepoužívaných prvků a přidává podporu nových a moderních technologií.

3.2 CSS

Jazyk CSS (Cascading Style Sheets) popisuje způsob zobrazení elementů na stránkách psaných v HTML, XHTML nebo XML. Definuje se zde například barva textu nebo pozadí, zarovnání odstavce nebo také různé animace elementů. Byly vydány verze CSS1, CSS2 a



Obrázek 3.1: Nové funkce jazyka HTML verze 5
[11]

CSS3. Hlavním smyslem jazyka je umožnit oddělení vzhledu dokumentu od jeho struktury a obsahu.

Existuje několik možností, jak lze CSS připojit k HTML elementům. Lze ho přidat přímo jako atribut HTML elementu `style="..."`, což při častém způsobuje, že HTML kód je značně nepřehledný, a proto se využívá jen velice zřídka pro stylování elementů, u kterých jsou vyžadovány specifické vlastnosti. Další možností zápisu CSS je vložení stylopisu do hlavičky HTML souboru. Zde jsou pravidla uzavřena mezi tagy `<style></style>`. Poslední a nejpoužívanější způsob je připojení externího stylopisu

```
<link rel="stylesheet" type="text/css" href="nazev_souboru.css">
```

Tento způsob je používán, pokud existuje více webových stránek s podobným vzhledem. Lze tedy pro ně vytvořit jeden společný externí stylopis, ve kterém budou zapsány všechny společné vlastnosti. V případě jakýchkoliv požadavků na změny stylu tudíž není nutné prohledávat všechny webové stránky a vyhledávat danou vlastnost.

Nejnovější verze CSS3 byla do jisté míry také reakcí na novou verzi jazyka HTML, konkrétně HTML5. Přináší podporu 2D a 3D animací, zaoblení rohů, stínování elementů i textu, nové modely barev (RGBA, HSL, HSLA) nebo také možnosti rozdělení obsahu do více sloupců.

3.3 JavaScript

Jedná se o multiplatformní, objektově orientovaný skriptovací jazyk vyvinutý Brendanem Eichem. V případě webových stránek je zpravidla používán jako interpretovaný programovací jazyk. Syntaxe tohoto jazyka patří do rodiny jazyků C/C++/Java. Ovšem JavaScript se od těchto jazyků výrazně liší, zejména sémanticky. Slovo Java je v názvu použito pouze z marketingových důvodů. V současné době je podporován naprostou většinou webových prohlížečů. Alternativami k JavaScriptu jsou například CoffeeScript, který je vyvíjený pod

MIT licencí a je inspirovaný jazyky Python a Ruby, TypeScript, který je prezentovaný jako nejintuitivnější alternativa k JavaScriptu a oproti němu nabízí například podporu tříd, modulů nebo typů, PythonJS a další. Bohužel pro tyto jazyky není tak rozšířená podpora jak ze strany prohlížečů, tak i ze strany komunity a lze k nim vyhledat pouze malé množství knih a studijních materiálů.

V prostředí webových stránek je obvykle používán k ovládání různých interaktivních prvků (např. tlačítka), validaci formulářových polí, vytváření animací, ale také k dynamické změně stylopisů nebo vytváření HTML elementů. Program JavaScriptu je spouštěn až po stažení webové stránky ze serveru, tedy na straně klienta. Z toho důvodu podléhá určitým bezpečnostním omezením. Například nemůže pracovat se soubory nebo s daty COOKIES, aby tím neohrozil bezpečnost klienta.

3.3.1 jQuery

Jedná se o nejrozšířenější knihovnu jazyka JavaScript s velice jednoduchou syntaxí, která klade důraz na interakci mezi JavaScriptem a HTML. Jejím autorem je John Resig a byla vydána na BarCampu v New Yorku. Software je v současné době pod licencí MIT a je dodáván na různých platformách několika společností, včetně Nokia nebo Microsoft pro použití v ASP.NET Frameworku.

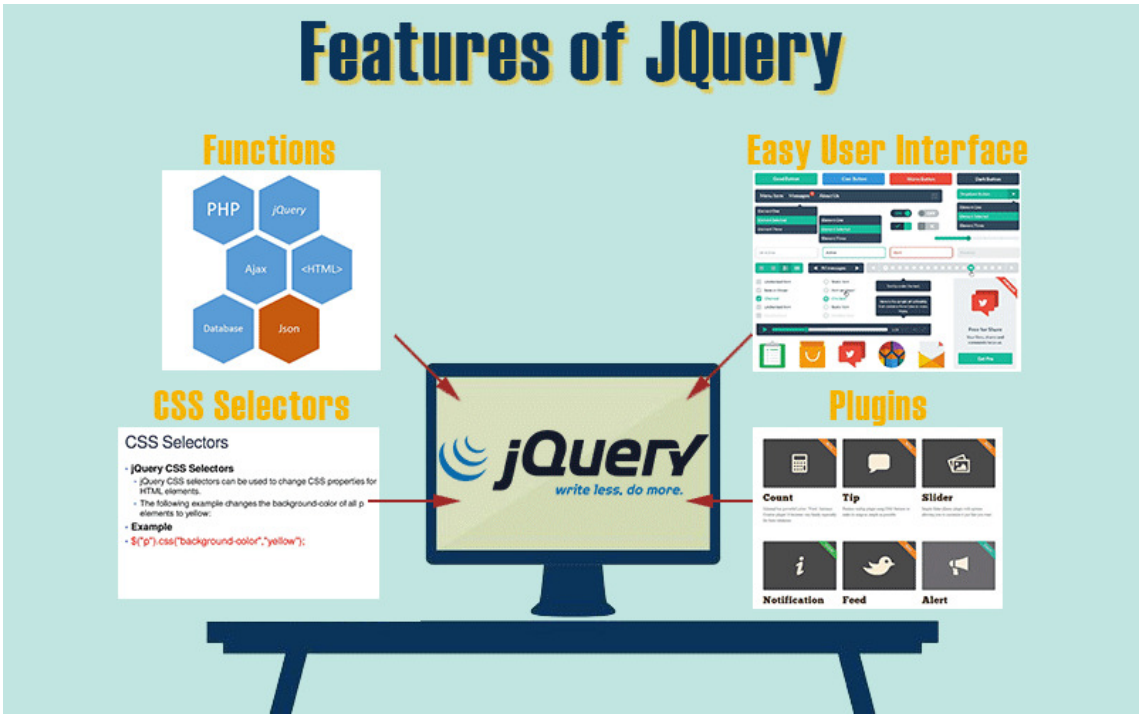
Snahou jQuery je oddělit chování od struktury HTML a umožňuje silnou interakci s DOM elementy díky použití CSS selektorů. Díky těmto vlastnostem výrazně usnadňuje práci a umožňuje psaní kratšího a přehlednějšího kódu v porovnání s obyčejným JavaScriptem. Dále také velice zjednodušuje práci s knihovnou AJAX, kde pro vytvoření požadavku stačí mnohdy pouze pár řádků kódu. Díky již zmíněné interakci s DOM elementy a použití CSS selektorů je možné jednoduše a efektivně k jednotlivým HTML elementům přistupovat, modifikovat jejich vlastnosti nebo vytvářet úplně nové prvky.

3.3.2 AJAX

AJAX (Asynchronous JavaScipt and XML) označuje technologie, které umožňují vyvíjet interaktivní webové aplikace tím, že mění obsah webových stránek bez nutnosti jejich kompletního znovunačítání. Děje se tak pomocí knihovny psané v JavaScriptu, která zajišťuje odesílání asynchronních požadavků na server a následné přijetí a zpracování odpovědi. To poskytuje uživateli příjemnější prostředí, ale vyžaduje použití moderních webových prohlížečů.

Mezi výhody patří samozřejmě změna obsahu stránek bez nutnosti jejich znovunačtení a téměř okamžitá odpověd serveru (v závislosti na rychlosti internetového připojení). Z tohoto důvodu byla také tato technologie použita v této bakalářské práci. Je jí ovlivněno hned několik operací, například změna operátoru tónování a zobrazení příslušných filtrů nebo také zobrazení náhledu zpracovávané fotografie či videa. Při odeslání požadavku je množství vyměňovaných dat výrazně nižší, než kdyby bylo nutné znovu sestavit celý HTML dokument. Z toho vyplývá potenciál snížení zátěže webového serveru.

Špatná implementace a nevhodné použití AJAXu však může mít za následek výrazně vyšší počet HTTP požadavků a tím nejen že zátěž serveru a sítě neklesne, naopak může ještě vzrůst. Další nevýhodou je, že do URL odkazu nelze předat data získaná pomocí AJAXu. Řešením je například použití neviditelných `<iframe></iframe>` prvků, což je ale výrazně ztěžuje návrh stránek a tím ztrácí efektivitu.



Obrázek 3.2: Možnosti knihovny jQuery
[1]

3.3.3 Dropzone.js

Jde o open source knihovnu JavaScriptu, která umožňuje a zjednodušuje uživateli nahrávání souborů na server. Tato knihovna nevyžaduje žádné speciální požadavky či externí knihovny a lze jí velice snadno a dobře modifikovat podle specifických požadavků uživatele.[2]

Zajišťuje nejen prostředí drag'n'drop pro nahrání souborů, ale i kompletní přenos nahrávaných souborů na server, možnost zobrazení náhledu a dodatečných informací o souborech nebo také limitovat typy souborů, které lze nahrát.

Knihovnu jsem využil především kvůli širokým možnostem modifikace a podpoře systémmu drag'n'drop, který je uživatelsky velice přívětivý. Její použití je velice snadné, instanci lze vytvořit dvěma způsoby.

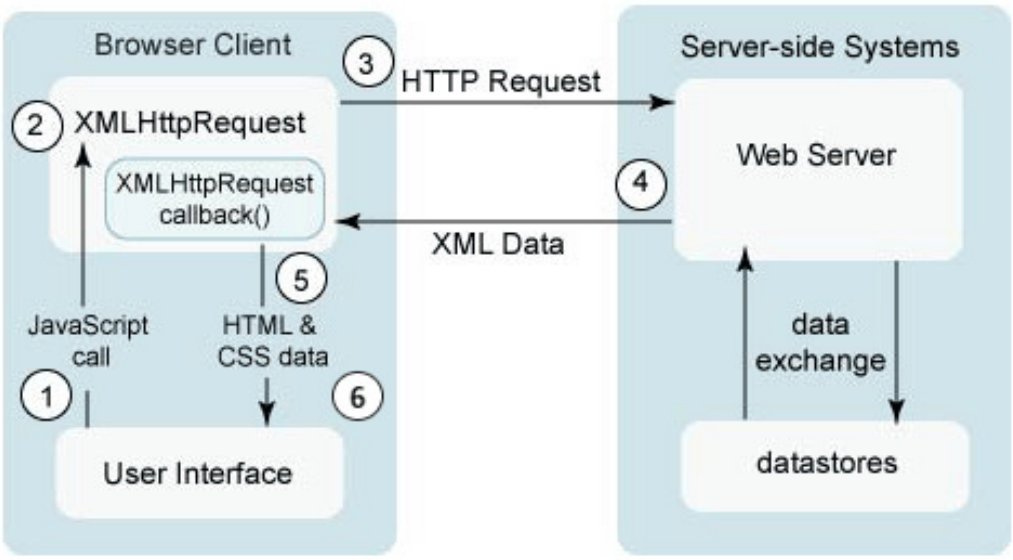
První variantou je umístění HTML elementu `<form></form>` do webové stránky a tomuto elementu přiřadit atribut `class="dropzone"`. Knihovna při svém načtení automaticky vyhledá v HTML dokumentu všechny elementy typu `<form></form>` obsahující třídu `dropzone` a k danému elementu se přiřadí. Umístění nahraných souborů je určeno formulářovým atributem `action`. Druhým způsobem je manuální vytvoření instance. Lze to udělat pouze s využitím JavaScriptu

```
var myDropzone = new Dropzone("div#myId", { url: "/file/post"});
```

nebo s využitím jQuery

```
$("div#myId").dropzone({ url: "/file/post" });
```

. U tohoto způsobu lze třídu přiřadit i jiným elementům, než je `form`, je však nutné vždy specifikovat atribut `url`, který udává, kam budou soubory nahrány.



Obrázek 3.3: Vzor asynchronního požadavku
[10]

3.3.4 ion RangeSlider

Další použitým pluginem je **ion RangeSlider**. Jak již název napovídá, tento jQuery plugin slouží k vytvoření sliderů. Jeho hlavními přednostmi jsou opět rozsáhlé možnosti modifikace vzhledu i funkcí.

3.4 PHP

Jedná se o dynamicky typovaný skriptovací jazyk[9]. Mezi jeho hlavní výhody patří jednoduchost, strmá křivka učení a nativní podpora několika databázových systémů. V současné době jde o nejpoužívanější skriptovací jazyk pro tvorbu webových stránek, je použit ve více než 82% webů.[4] U webových stránek je PHP skript prováděn na straně serveru, takže uživateli je prezentován až výsledek jeho činnosti.

3.4.1 Nette Framework

Jedná o jeden z nejrozšířenějších PHP frameworků. Je zaměřen především na rychlost a bezpečnost, dále pak klade důraz na přehlednost a úpravu celého kódu. Lze jej volně kombinovat s dalšími PHP frameworky, jakým je například Zend. Framework[7] je psán v čistě objektové formě a využívá také vlastností nejnovější verze PHP 7.1. Dále je pro něj vyvinuto velké množství různých volně dostupných doplňků, což významně urychluje vývoj aplikace.

3.5 pfstools

Jedná se rozsáhlou knihovnu obsahující množství příkazů pro čtení, vytváření a zpracování HDR obrazu a videa. Jednou z jejích částí poskytuje sadu příkazů pro spojení několika

snímků s nízkým dynamickým rozsahem (LDR) do jediného HDR snímku, rekonstrukci obrazu ze snímků RAW formátu a také fotometrickou kalibraci spojených LDR snímků.

Použité operátory `pfstmo_ferradans11`, `pfstmo_mai11` a `pfstmo_pattanaik00` umožňují časově koherentní mapování tónů, což je využíváno především při zpracování videozáznamů.

Další částí knihovny pfstools je sada operátorů pro mapování tónů vytvořených HDR snímků. V této bakalářské práci jsou použity následující metody mapování tónů[1]

3.5.1 Adaptivní logaritmické mapování

Tento operátor[14] provádí logaritmickou kompresi hodnot jasu, přičemž se snaží napodobit lidské vnímání světla. Základ pro logaritmus je počítaný pro každý pixel samostatně, což zajišťuje zachování kontrastu a jasu. Pro zvýšení či snížení kontrastu tmavších oblastí snímku je doporučeno změnit hodnotu gamma korekce snímku. Tento operátor je v knihovně pfstools nazýván `pfstmo_drago03`.

(a) Gamma korekce 2.2 (b) Gamma korekce 5

Obrázek 3.4: Ukázka zpracování HDR snímku operátorem drago03 při různých hodnotách gamma korekce

3.5.2 Rychlé bilaterální filtrování

Metoda[15] je založena na rozdělení snímku do dvou vrstev. Základní vrstvy a vrstvy detailů. Jedná se o nelineární filtr, ve kterém je váha každého pixelu vypočtena pomocí Gaussovy funkce v prostorové doméně. Tato váha je vynásobena hodnotou funkce ovlivnění oblasti intenzity, která snižuje váhu pixelů v oblastech s velkými rozdíly intenzity. V pfstools má tato metoda název `pfstmo_durand02`.

[1] Pokud není uvedeno jinak, je u všech náhledů aplikována hodnota gamma korekce 2.2

Obrázek 3.5: Ukázka zpracování HDR snímku operátorem durand02

3.5.3 Metoda komprese rozsahu jasu

vychází ze snižování rozsahu změn jasu. Obraz s nízkým dynamickým rozsahem je poté získán vyřešením Poissonovy rovnice při upraveném rozsahu změn jasu. Je využívána operátorem `pfstmo_fattal02`[16]

Obrázek 3.6: Ukázka zpracování HDR snímku operátorem fattal02

3.5.4 Operátor `pfstmo_mantiuk06`

provádí převod vstupního souboru z jasového prostoru na pyramidu snímků s nízkým kontrastem a poté do prostoru pro vizuální odezvu. Zde jsou zpracovány a pro získání upraveného souboru je proces převodu aplikován v obráceném pořadí[19]. Tento operátor má dva

způsoby spuštění. Podle specifikovaného parametru je využit buďto algoritmus vyrovnání kontrastu nebo je použito kontrastní měřítko.

(a) Použití algoritmu pro vyrovnání kontrastu (b) Použití kontrastního měřítka

Obrázek 3.7: Ukázka zpracování HDR snímku operátorem mantiuk06

3.5.5 Metoda časově závislé vizuální adaptace

Tato metoda[20] použitím parametrů `-t` a `--fps`. Při použití časové závislosti je zvolena globální metoda vizuální adaptace. Ta je závislá na hodnotě parametru `--fps` udávající počet snímků za sekundu. Z každého snímku je vytvořen jeden obrázek a jsou uložena data o jeho adaptaci. Tato data jsou poté použita pro adaptaci dalšího zpracovávaného snímku.

3.5.6 Metoda fotografické reprodukce tónů

využívána operátorem `pfstmo_reinhard02`[22] je založena na automatickém nastavení expozičního času snímku díky měření jasu v jednotlivých částech. Algoritmus automaticky zesvětluje příliš tmavé a ztmavuje příliš světlé části snímku.

Obrázek 3.8: Ukázka zpracování HDR snímku operátorem reinhard02

3.5.7 Operátor `pfstmo_reinhard05`

využívá metodu fotoreceptorů[21], jejíž průběh je podobný adaptačním procesům vnímání obrazu lidmi. Zde dochází k adaptaci již ve fotoreceptorech oka. Z toho důvodu operátor zpracovává každý barevný kanál zvlášť, čímž simuluje reakci fotoreceptorů na vstupní jas.

Obrázek 3.9: Ukázka zpracování HDR snímku operátorem reinhard05

3.5.8 Optimalizace tónovací křivky pro zpětně kompatibilní kompresi

je metoda[18] využívající model zkreslení vyplývající z kombinace procesů mapování tónů a komprese. S využitím tohoto modelu je možné nalezení optimální tónovací křivky pro minimalizaci tzv. „*Mean squared error*“ v rekonstruovaném HDR snímku. Tuto metodu využívá operátor tónování `pfstmo_mai11`.

Obrázek 3.10: Ukázka zpracování HDR snímku operátorem mai11, gamma korekce 1

3.5.9 Operátor `pfstmo_ferradans11`

využívá analýzy adaptace obrazu a vnímání kontrastu, aby při zpracování snímku byla zachována co největší podobnost vzhledem k reálnému vnímání[17]. Proces mapování tónů je rozdělen na dvě části. V první je provedena globální adaptace obrazu a ve druhé pak zvýšení kontrastu pro části snímku.

Obrázek 3.11: Ukázka zpracování HDR snímku operátorem ferradans11

Kapitola 4

Návrh a implementace

V této části je prezentován reálný postup návrhu a implementace celého webového systému. Je zde popsán význam a postup implementace jednotlivých tříd a metod, které jsou při řešení tohoto projektu využity.

4.1 Návrh

Celý systém je rozdělen na dvě části. Část administrátora a veřejnou část. V administrátorské části je implementována aplikace pro správu registrovaných uživatelů a také aplikace pro správu používaných operátorů tónování a jejich parametrů. Veřejná část obsahuje systém přihlášení a registrace a systém pro nahrání souborů, jejich úpravu pomocí operátorů tónování a zobrazení výsledků.

Před implementací klientské části každé webové aplikace je vždy potřeba provést návrh celého systému a také velmi podrobný návrh uživatelského rozhraní. Jeho ovládání musí být z pohledu uživatele jednoduché a intuitivní. Proto bylo jeho vytváření rozděleno na několik verzí. Každá verze byla prezentována několika testovacím uživatelům a případné připomínky nebo nedostatky byly zohledněny v další verzi. Tento postup byl opakován, až dokud nebyly všechny nedostatky opraveny.

Při návrhu administrátorské části byl kladen velký důraz na jednoduchost a přehlednost celého rozhraní.

4.2 Adresářová struktura

Jelikož se jedná o webovou aplikaci, musí být adresářová struktura velice striktní. Každý adresář na serveru má svou funkci a je důležité, aby nedošlo k jejich narušení. Z webového prohlížeče je povolen přístup pouze do adresáře `www`. Tím je zajištěno, že nikdo nebude moci z webového prohlížeče přistupovat k důležitým knihovnám frameworku, logovacím a konfiguračním souborům nebo k vytvořeným třídám této aplikace.

Výsledná adresářová struktura je následující:

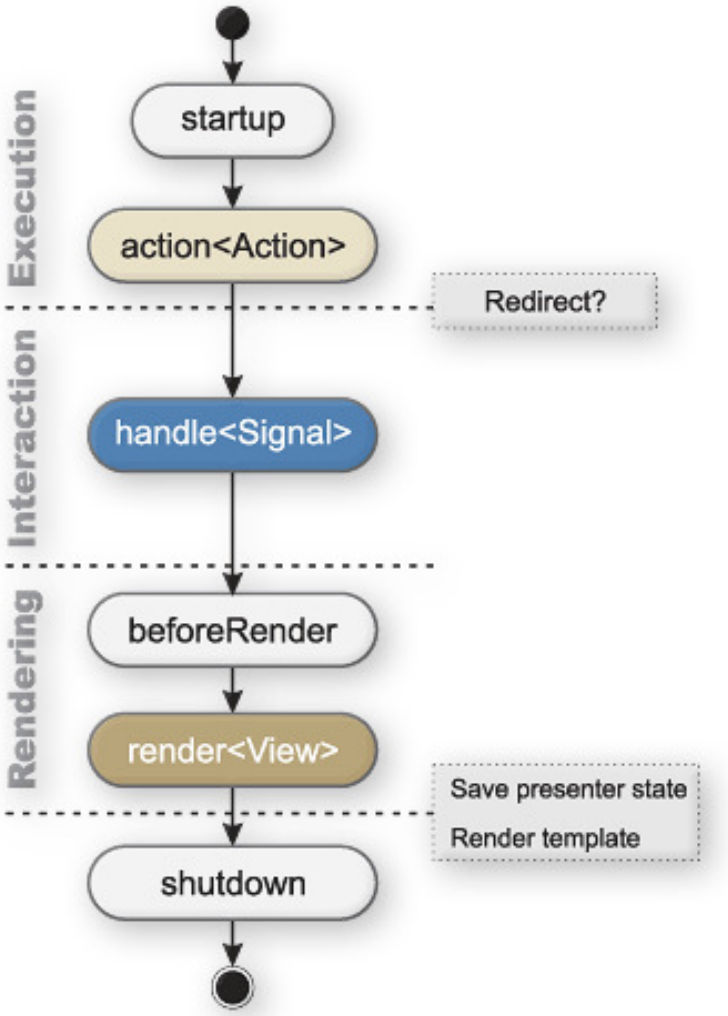
```
adresář projektu
├── app - všechny vytvořené třídy webového systému
│   ├── AdminModule - skripty pro režim admina
│   │   ├── forms -továrny pro formuláře administrátorského modulu
│   │   └── presenters - presentery pro každou stránku adminu
│   │       └── templates - šablony všech stránek v administrátorském modulu
│   ├── bin
│   ├── config - konfigurační soubory webu
│   ├── lang - slovníky pro jednotlivé použité jazyky
│   ├── model - třídy pro zpracování souborů, tónování, práci s uživateli
│   ├── PublicModule - skripty pro veřejnou část
│   │   ├── forms -továrny pro formuláře veřejné části
│   │   └── presenters - presentery pro každou stránku veřejné části
│   │       └── templates - šablony všech stránek v veřejné části
│   └── router - továrna pro routery
├── log - logovací soubory případných chyb systému
├── temp - soubory pro uložení webových sessions a cache soubory
├── vendor - knihovny frameworku Nette
└── www - jediný adresář, který je přístupný z webového prohlížeče
    ├── css - soubory s kaskádovými styly, které definují vzhled webu
    ├── images - obrázky, které jsou na webu použity
    ├── js - skripty jazyka JavaScript
    ├── temp - adresáře uživatelů s nahranými soubory a výsledkytónování
    ├── favicon.ico
    ├── index.php
    ├── robots.txt
    └── web.config
```

4.3 Vysvětlení pojmů a popis nejdůležitějších tříd

Důležitými třídami veřejné části systému jsou `BasePresenter` 4.3.3, `SignPresenter` 4.3.4, `RegisterPresenter` 4.3.5, `AppPresenter` 4.3.6 v adresáři `presenters` a `BaseProcess` 4.3.7, `FilesProcess` 4.3.8 a `TonemapProcess` 4.3.9 v adresáři `model`. Ty zajišťují mazání již nepoužívaných adresářů s výsledky zpracování souborů, registraci a přihlášení uživatelů, přijímání odesílaných dat z webu, nahrávání, kalibraci a stažení souborů, jejich tónování, generování náhledu, apod. Další důležité používané třídy se nacházejí v adresáři `forms`, konkrétně se jedná o třídy `FormFactory`, `ImagesFormFactory` a `SignFormFactory`.

4.3.1 Presenter

Jedná se o speciální typ komponenty Nette frameworku, tzv. komponentový kontejner. Je potomkem třídy `Control`, ale má speciální vlastnost, že uvnitř něj mohou být vytvářeny další komponenty, jako jsou například formuláře. Její základní vlastnost je vykreslení odpovídající šablony metodou `render<View>`, kde view je název šablony.



Obrázek 4.1: Životní cyklus presenteru
[13]

Metody `startup` a `shutdown` jsou volány na začátku a konci životního cyklu každého presenteru. Obě metody je možné modifikovat, v takovém případě je ale nutné vyvolat také stejnojmenné metody předka.

Alternativou k metodě `render` je metoda `action<Action>`. Slouží k vykonání požadované akce a následnému přesměrování. Tato metoda je volána před metodou `render`. Je tedy možné rozhodnout, která metoda vykreslení bude provedena.

Zpracování signálů je v presenteru prováděno metodou `handle<Signal>`. Je určena především pro komponenty presenteru, pro zpracování AJAX požadavků a formulářů. Data požadavků jsou zde získávána voláním metody komponenty `getParameter(nazev_parametru)`. Před návratem z této metody je obvykle odeslána odpověď serveru. To je prováděno metodou presenteru `sendResponse`, případně `sendJson` nebo voláním `redrawControl`. Jedná se o metodu třídy `Control` a provádí invalidaci neboli překreslení tzv. snippetu.

Metoda `beforeRender` je volána po provedení a zpracování všech akcí a signálů těsně před metodou `render`. Je zde možné zvolit šablonu, která bude vykreslena nebo definovat proměnné, které budou do šablony předány.

4.3.2 Snippet

Tento pojem je úzce spjatý s odesíláním AJAXových požadavků. Při prvním načtení stránky a jsou ze serveru přenesena a vykreslena všechna data a poté při každém odeslání AJAXového požadavku je presenter schopný si zapamatovat, zda došlo ke změnám, které vyžadují překreslení. V takových případech je volána metoda `redrawControl`, která překreslí všechny snippety dané komponenty, u kterých je to potřeba. Tuto metodu lze upřesnit, když je jí předán jako parametr název požadovaného snippetu. V takovém případě je překreslen pouze specifikovaný snippet a ostatní zůstávají beze změny.

4.3.3 BasePresenter

Jedná se abstraktní třídu, ve které jsou definované základní vlastnosti společné pro všechny presentery. Tyto vlastnosti jsou inicializace databázového připojení, načtení překladače a nastavení adresáře pro uložení `session` dat. V konstruktoru třídy je vždy prováděna kontrola a případné odstranění uživatelských adresářů.

V případě nepřihlášených uživatelů je název složky vytvářen z php session ID, ke kterému je připojeno náhodně vygenerované číslo. Toto číslo je uloženo v `sessionStorage`[12]. `sessionStorage` je prostor pro uložení dat ve webovém prohlížeči. Tento prostor je unikátní pro každou otevřenou záložku prohlížeče a při jejím zavření je smazán. Díky tomu je možné v různých webových záložkách zpracovávat naprosto odlišné soubory. Časový limit pro uchování dat na serveru je v tomto případě jeden den.

V případě přihlášených uživatelů je název složky složen z ID uživatele v databázi a opět z generovaného čísla webové záložky. V tomto případě jsou data na serveru uchována jeden týden.

Samotný proces mazání spočívá v tom, že pomocí iterátoru je kontrolován obsah adresáře `www/temp/`, ve kterém jsou dočasné adresáře uživatelů uloženy. V případě, že některý z nich nevyhovuje výše uvedeným podmínkám, je vyvolána metoda `deleteFolder`. Jako povinný argument tato metoda přijímá absolutní cestu k odstraňovanému adresáři. Metoda provádí iteraci nad veškerým jejím obsahem. V případě, že narazí na soubor, smaže jej. Pokud se jedná o podadresář, je tato metoda spuštěna rekurzivně s cestou k novému adresáři. Nakonec je odstraněn i aktuálně kontrolovaný adresář.

4.3.4 SignPresenter

Tento presenter je potomkem třídy `BasePresenter` 4.3.3. Zajišťuje přihlášení a odhlášení uživatelů. Obsahuje metodu `createComponentSignInForm`, ve které dochází k fyzickému vytvoření přihlašovacího formuláře definovaného ve třídě `SignFormFactory`. Dále je zde definována akce, která je provedena po úspěšném odeslání a zpracování formuláře. Jedná se o zobrazení odpovídající zprávy uživateli a přesměrování na domovskou stránku webu. Druhou metodou presenteru je `actionOut`. Tato metoda spouští odhlášení uživatele a následné vypsání zprávy a přesměrování v případě úspěchu.

4.3.5 RegisterPresenter

Pro registrační formulář není použita žádná továrna. Je tedy tvořen přímo zde, konkrétně v metodě `createComponentRegisterForm`. do proměnné `form` je přiřazen objekt třídy `Form` a následně jsou vytvořeny prvky celého formuláře metodami `add<nazev_prvku>`. Ke každému prvku formuláře jsou vázána různá validační pravidla pro kontrolu, zda je prvek vyplněn,

jestli má požadovaný minimální počet znaků nebo jestli odpovídá korektně zadané e-mailové adrese. Všechna validační pravidla jsou kontrolována přímo v prohlížeči. Teprve po splnění všech těchto pravidel jsou registrační data odeslána na server, kde je provedena kontrola, zda e-mailová adresa již není registrována. V případě úspěchu je uživatel uložen do databáze. V opačném případě je vyvolána příslušná výjimka.

4.3.6 AppPresenter

Zde je vytvořen formulář pro nahrání souborů na server a také zde jsou zachyceny a zpracovány odesílané signály.

Pro vytvoření a formuláře slouží metoda `createComponentFileUploadForm`. V ní je vyvolána továrna `ImagesFormFactory` a následně je fyzicky vytvořen formulář `fileUploadForm`. Akce, která je provedena po odeslání formuláře, je přiřazena k události formuláře `onSuccess`.

Nejprve jsou z formuláře získána data vyvoláním metody `getHttpData`. Ta je implementována ve třídě `Form`. Po získání dat je unikátní ID, vygenerované pro aktuální záložku, spojeno s php session ID, aby bylo zajištěno, že pro každou záložku v každém sezení bude vytvořen unikátní adresář uživatele.

Následně jsou vytvořeny objekty `files` a `tonemap` tříd `FilesProcess` 4.3.8 a `TonemapProcess` 4.3.9. Tyto objekty při svém vytvoření přijímají několik argumentů. Je to inicializované připojení k databázi, vytvořené session ID a informace o případném přihlášeném uživateli. Získaná data formuláře obsahují informace o nově nahraných souborech a také o souborech, které byly nahrány již dříve. Obě tyto informace jsou předány metodě `setFiles` 4.3.8 vyvolané z objektu `files`.

Po zpracování informací je volána metoda `makeSamples` 4.3.9, která na zpracované soubory aplikuje připravenou sadu operátorů tónování se základními hodnotami jejich parametrů. Metoda vrací asociativní pole obsahující ID a název použitého operátoru a cestu k vytvořenému souboru. Toto pole je uloženo do proměnné `samples`, která je pak dále využita v šabloně, konkrétně ve snippetu 4.3.2 stejného jména.

Ihned po vytvoření formuláře je odeslán požadavek na získání informací o souborech, které jsou již na serveru nahrány. Je zpracován metodou `handleGetUploadedFiles`. V požadavku je obsaženo ID záložky. Tím je identifikován adresář uživatele. V jeho podadresáři `upload` jsou poté iterátorem vyhledávány soubory. U každého souboru je uložen jeho název, velikost, typ a umístění. Dále jsou metodou `makeSamples` 4.3.9 a umístění upravených souborů na serveru. Všechny tyto informace jsou poté odeslány ve formátu JSON do JavaScriptu k dalšímu zpracování.

Po nahrání souborů na server je možné stáhnout vytvořený HDR soubor bez aplikování jakýchkoliv operátorů tónování. K zachycení tohoto požadavku slouží metoda `handleMakeFullHdr`. Je zde opět vytvořen objekt třídy `FilesProcess` a z něj je vyvolána metoda `makeFullHdr` 4.3.8. Její návratovou hodnotou je cesta k vytvořenému souboru. Před opuštěním metody je odeslána odpověď ve formě asociativního pole, které pod klíčem `path` obsahuje získanou cestu. Toto pole je zakódováno do formátu JSON.

Když je přijat požadavek na zobrazení parametrů operátoru, je zpracován metodou `handleGetAttributes`. V požadavku je opět odesláno ID záložky a ID zvoleného ope-

rátoru. Z databáze jsou podle ID operátoru získány informace o všech jeho parametrech. Dále je vytvořeno asociativní pole `gamma`, ve kterém jsou údaje o intenzitě nasvícení.

Všechny tyto získané údaje jsou metodami `setOperator`, `setGamma` a `setAttribute` 4.3.9 uloženy do připraveného objektu třídy `TonemapProcess`. Údaje o hodnotách parametrů operátoru a nasvícení jsou také uloženy do proměnných `gamma` a `attributes` využitých v šabloně ve snippetu `operator_attributes`.

Poté je vyvolána metoda `makePreview`, jejíž návratovou hodnotou je cesta k umístění náhledu aktuálního stavu úprav. Ta je uložena do proměnné `preview`, která je využita ve snippetu `preview`. Nakonec jsou oba snippety překresleny metodou `redrawControl`.

V případě změny některého z parametrů vybraného operátoru tónování je odesílán požadavek, který je zpracován metodou `handleGetPreview`. V požadavku jsou předány data o ID webové záložky, ID použitého operátoru a aktuálních hodnotách všech jeho parametrů. Je vytvořen objekt třídy `TonemapProcess` a všechna tato data jsou do něj uložena. Poté je vyvolána metoda `makePreview` a do proměnné `preview` je uložena její návratová hodnota. Následně je překreslen snippet `preview` pro zobrazení aktuálního stavu procesu tónování.

Po dokončení úprav je odeslán požadavek na zobrazení upravovaného souboru v plném rozlišení. Signál je zpracován metodou `handleMakeFinalImg`. Tato metoda je téměř totožná s metodou `handleGetPreview`. Požadavek odesílá opět ID záložky, ID operátoru a informace o hodnotách parametrů operátoru.

Je opět vytvořen objekt třídy `TonemapProcess` a do něj jsou uložena předávaná data. Rozdíl oproti předchozí metodě je v tom, že zde je vyvolána metoda `makeFullResolution` 4.3.9. Její návratovou hodnotou je cesta k umístění zpracovaného souboru v plném rozlišení a je předána do proměnné `final`. Před návratem je překreslen snippet `final`.

Předposlední dvě metody v tomto presenteru jsou volány v případě požadavku na stažení zpracovaného souboru po použití operátorů. Nazývají se `handleDownloadFinalJpg` a `handleDownloadFinalHdr`. Jak již jejich názvy napovídají, tyto metody slouží ke zprostředkování stažení zpracovávaného souboru se všemi provedenými úpravami.

Těla obou těchto metod jsou podobná metodám `handleGetPreview` a `handleMakeFinalImg`. V požadavku je opět předáváno ID záložky, ID parametru a aktuální hodnoty všech jeho atributů. Ty jsou uloženy do připraveného objektu třídy `TonemapProcess`. Následně je v obou vyvolána `makeFinal` 4.3.9, které je jako argument předá požadovaná přípona. Z této funkce je vrácena cesta k umístění vytvořeného souboru a ta je ve formě JSON řetězce odeslána zpět do webového prohlížeče.

Poslední metodou zde je metoda `renderImages`, která vykresluje šablonu `images.latte`.

4.3.7 BaseProcess

`BaseProcess` je abstraktní třída, která obsahuje pouze inicializaci základních společných vlastností tříd `FilesProcess` a `TonemapProcess`. Tyto vlastnosti jsou inicializovány přímo v konstruktoru a musí být předány ve formě argumentů při vytváření objektů. Dále třída obsahuje chráněnou metodu `clearDir`.

Tato metoda přijímá jako jediný povinný argument cestu k umístění adresáře určeného k vyprázdnění. Pomocí funkce `scandir` je získán obsah tohoto adresáře, který je následně kontrolován v cyklu. Speciální případy, jako jsou `.` a `..`, jsou vynechány. Všechny ostatní

objekty v adresáři jsou kontrolovány. Pokud se jedná o soubor, je vymazán. Pokud se jedná o podadresář, je tato metoda spuštěna rekurzivně.

4.3.8 FilesProcess

je potomkem abstraktní třídy `BaseProcess` a obsahuje metody pro nahrávání, kalibraci, čtení a vytváření souborů. Také obsahuje atributy `files`, ve které jsou uloženy cesty ke všem nahraným souborům pro aktuální záložku, `calibration`, která obsahuje cestu ke zmenšenému výsledku kalibrace, a `hdr_extensions`, což je pole, ve kterém jsou uloženy všechny možné přípony HDR souborů.

Metoda `setFiles` přijímá dva argumenty. Pole s informacemi souborech, které prozatím čekají na nahrání a pole s názvy souborů, které byly nahrány již dříve a uživatel s nimi chce dále pracovat. Nejprve jsou dříve nahrané soubory zkontrolovány metodou `readFiles` a podle její návratové hodnoty je nastaven příznak `recalibrate`. Poté je provedena kontrola, zda nějaké soubory čekají na nahrání. V takovém případě jsou nahrány na server metodou `uploadFiles` a příznak `recalibrate` je nastaven na hodnotu `TRUE`. Nakonec je podle hodnoty příznaku provedena kalibrace souborů metodou `recalibrate`.

Metoda `readFiles` nejprve kontroluje, zda byla vytvořena složka uživatele a v případě neúspěchu okamžitě končí s návratovou hodnotou `FALSE`. Následuje kontrola, jestli uživatel již někdy soubory nahrával a zda tedy existuje adresář `uploads`. Opět v případě neúspěchu metoda vrací hodnotu `FALSE`. Dále následuje kontrola nahraných souborů.

Každý s předávaného pole souborů je nejprve upraven tak, že je před jeho název přidána cesta. Je to z toho důvodu, funkce `glob`, která je použita pro získání nahraných souborů vrací pole, ve kterém jsou uloženy názvy včetně cest. Pole získaných souborů je iterováno cyklem `foreach` a každý ze souborů je vyhledáván v poli obsahující soubory, se kterými chce uživatel dále pracovat. Pokud se kontrolovaný soubor v tomto poli nachází, je jeho název uložen pro další použití. V opačném případě je soubor ze serveru smazán a příznak `recalibrate` nastaven na hodnotu `TRUE`.

Poslední částí metody `readFiles` je kontrola, zda existuje adresář `calibration` a zda obsahuje soubor `calibrate.jpg`. V případě, že adresář existuje a že tento soubor obsahuje, je cesta k němu uložena do atributu `calibration` a metoda končí s hodnotou uloženou v příznaku `recalibrate`. V případě, že jedna z podmínek není splněna, metoda končí s hodnotou `TRUE`.

V metodě `uploadFiles` dochází k nahrávání souborů na server. Které soubory budou nahrány, je specifikováno argumentem `uploadFiles`, což je pole objektů třídy `FileUpload`. Před vlastním nahráváním ještě probíhá kontrola, jestli existuje adresář uživatele a zda je v něm vytvořen podadresář `upload`. Oběma adresářům jsou následně přidělena práva plného přístupu. Poté jsou cyklem procházeny soubory k nahrání. Na server jsou uloženy pomocí vestavěné PHP funkce `move_uploaded_files`, jsou jim také přidělena práva plného přístupu a cesta k jejich umístění je uložena do atributu `files`.

`Recalibrate` je metoda, ve které se provádí zkalibrování a převedení souborů do HDR formátu. Na začátku opět probíhá kontrola, zda existuje adresář uživatele a podadresář `calibrate`. Oba jsou v případě potřeby vytvořeny a adresář `calibrate` je vyprázdněn využitím metody `clearDir`.

Následně je v cyklu iterován atribut `files`. V něm jsou uložena umístění všech souborů nahraných uživatelem. U každého souboru je extrahována jeho přípona pomocí vestavěné funkce `pathinfo` s použitím přepínače `PATHINFO_EXTENSION`. Tato přípona je poté porovnávána s příponami HDR souborů, které jsou uloženy v atributu `hdr_extensions`. V případě, že je nalezena shoda, je aktuální soubor z `files` atributu odstraněno.

Pokud po tomto procesu není pole `files` prázdné, je sestavena série příkazů programu pfstools[8], konkrétně se jedná o příkazy `pfsinme`, který přijímá jako volitelný počet argumentů názvy a umístění souborů pro načtení, `pfssize`, který hodnotami přepínačů `--maxx` a `--maxy` určuje šířku a výšku, na kterou jsou načtené soubory zmenšeny, v tomto případě je pevně definovaná šířka 250 pixelů, `pfshdrcalibrate`, který provede zkalibrování všech souborů do jediného a `pfsout`, který tento výsledek uloží a jako argument přijímá název a umístění výsledného souboru.

V případě, že po filtraci HDR souborů z původního pole `files` nezbývá žádný ke kalibraci, je příkazem `pfsin` načten HDR soubor, který je mezi nahranými první v abecedním pořadí, následně je zmenšen příkazem `pfssize` a opět uložen do nového umístění příkazem `pfsout`. Všechny tyto příkazy jsou spojeny znakem „pipeline“ a poté předány jako argument vestavěné PHP funkci `exec`. Ta umožňuje z prostředí PHP vyvolávat příkazy prostředí shell.

Poslední metodou třídy `FilesProcess` je metoda `makeFullHdr`. Její obsah je velmi podobný metodě `recalibrate`. Na jejím začátku je opět kontrola a případné vytvoření příslušných adresářů. Následuje kontrola, zda se mezi nahranými soubory nachází kromě HDR i jiné soubory. V takovém případě jsou tyto soubory zpracovány stejným způsobem jako v metodě `recalibrate`.

Název výsledného HDR souboru se však liší podle toho, kolik HDR fotografií je nahráno. Tento počet udává čítač `hdrCnt` a jeho hodnota je připojena k názvu kalibrovaného souboru. Poté jsou v adresáři `upload` procházeny všechny soubory s HDR příponou, jsou přejmenovány stejně jako kalibrovaný soubor, je jim přiřazeno pořadové číslo a jsou uloženy do cílového adresáře.

Nakonec je vytvořen ZIP archiv obsahující všechny HDR soubory v cílovém adresáři a umístění archivu je z metody vráceno. V případě, že čítač `hdrCnt` je nulový, je ke kalibrovanému souboru připojena tato hodnota a z metody je vráceno umístění tohoto souboru.

4.3.9 TonemapProcess

Třída `TonemapProcess` zajišťuje úpravu vybraných souborů. Obsahuje privátní atributy `gamma` pro uložení hodnoty parametru nasvícení, `operator`, kde jsou uloženy veškeré informace o použitém operátoru a `attributes`, což je pole, ve kterém jsou uloženy informace o jednotlivých parametrech zvoleného operátoru.

Jelikož jsou tyto atributy privátní, je nutné jejich hodnoty nastavovat veřejnými metodami `setGamma`, které je předána jako argument zvolená hodnota, `setOperator` a `setAttribute`.

Metoda `setOperator` přijímá jeden povinný parametr a tím je ID zvoleného operátoru. Následně je proveden dotaz do databázové tabulky `tonemapping_operators`, ze které jsou podle ID získány všechny další informace o operátoru. Ty jsou poté uloženy do atributu `operator`.

Metoda `setAttribute` slouží k uložení hodnoty jednoho parametru. Jako argumenty metoda přijímá název parametru používaný v programu `pfstools` a hodnotu tohoto parametru. Pro uložení všech parametrů daného operátoru je nutné tuto metodu spouštět cyklicky.

Další metodou třídy `TonemapProcess` je veřejná metoda `makeSamples`. Zde jsou vytvářeny náhledy zpracování nahraných souborů jednotlivými operátory při použití základních hodnot jejich parametrů. Tato metoda přijímá jeden parametr `read` ve formě příznaku udávajícího, jestli je prováděno pouze čtení existujících náhledů nebo jsou vytvářeny náhledy nové.

Nejprve je provedena kontrola, zda existuje cílový adresář pro uložení náhledů a v případě potřeby je vytvořen. Pokud je příznak `read` nastaven na hodnotu `FALSE` je tento adresář vyprázdněn. Poté jsou získány z databázové tabulky `tonemapping_operators` všechny aktuálně používané operátory a přes získaná data se iteruje. Je sestaven název souboru, který obsahuje název operátoru v `pfstools`, řetězec `_sample` a aktuální UNIX timestamp.

Pokud má `read` hodnotu `TRUE`, je využitím vestavěné funkce `glob` proveden pokus o vyhledání odpovídajícího souboru podle části jeho názvu, ve které je vynechán právě UNIX timestamp. V případě nálezu jsou cesta k souboru, název a ID operátoru uloženy a pokračuje se na další získaný operátor.

V případě, že `read` je nastaven na `FALSE`, je připraven řetězec obsahující příkaz pro načtení zkalibrovaného souboru `calibrate_sample`. K tomuto řetězci je připojen příkaz k provedení tónování odpovídajícím operátorem se základními hodnotami všech jeho parametrů. Tyto parametry jsou získány dotazem do databázové tabulky `tonemap_operators_attributes`. Na závěr je k řetězci připojen příkaz k uložení zpracovávaného souboru pod předem vytvořeným názvem. Poté je tato série příkazů spuštěna funkcí `exec`.

Po dokončení příkazu je uložena cesta k souboru, název a ID operátoru a pokračuje se na další operátor. Když jsou všechny operátory zpracovány, je z funkce vráceno pole obsahující informace o vytvořených nebo načtených náhledech.

`makePreview` je metoda pro vytváření náhledu při změně hodnoty některého z parametrů vybraného operátoru. Je v ní opět připravena série příkazů programu `pfstools` pro načtení zkalibrovaného a zmenšeného souboru, aplikování vybraného operátoru a jeho parametrů a uložení nově vytvořeného souboru do zvoleného umístění.

Před vykonáním této série příkazů je zkontrolováno, zda adresář pro náhled existuje a je vyprázdněn.Poté jsou příkazy funkcí `exec` vykonány a v případě úspěchu je z metody vrácena cesta k vytvořenému náhledu. V opačném případě jsou vráceny chyby, které při zpracování nastaly.

Po dokončení úprav je volána metoda `makeFullResolution` pro zobrazení zpracovávaného souboru v plném rozlišení. Nejprve je vyhledán a případně vytvořen adresář `HDRimages`, ve kterém se musí nacházet připravený HDR soubor `fullHDR0.hdr`.

Pokud takový soubor neexistuje, je volána metoda `makeFullHdr` 4.3.8 třídy `FilesProcess`. Následně je požadovaný soubor znovu vyhledán a jsou připraveny příkazy pro jeho zpracování – `pfsin` pro jeho načtení, `pfstmo_<pfstools-nazev-operatoru>` se všemi parametry operátoru, `pfsgamma` pro nasvícení a `pfsout` pro uložení upraveného souboru. Před provedením této série příkazů je ověřena existence adresáře `HDRfinal`. Po dokončení příkazů metoda vrací cestu k vytvořenému souboru.

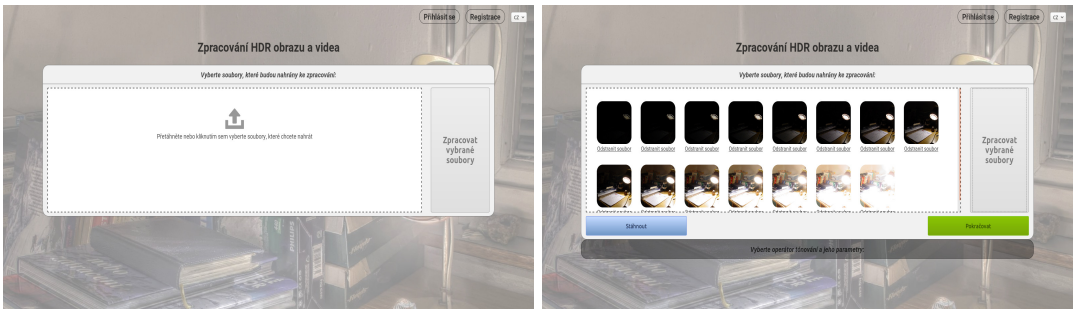
Poslední metoda třídy `TonemapProcess` se nazývá `makeFinal`. Přijímá jeden argument, který specifikuje příponu výstupních souborů. Metoda zpracovává všechny soubory, které se nachází v adresáři `HDRimages`. Tyto soubory jsou pro zpracování načteny příkazem `pfsin fullHDR%d.hdr`. Koncovka `%d` je zde umístěna pro případ, že by zvolený operátor byl časově koherentní. V takovém případě je adaptace prováděna v závislosti na již zpracovaných snímcích. K operátoru jsou následně přiřazeny odpovídající parametry a je aplikována gamma korekce příkazem `pfsgamma -g <hodnota>`. Nakonec jsou vytvořeny výstupní soubory příkazem `pfsout`. Názvy vytvořených souborů jsou skládány ze základu `fullHDR`, ke kterému je opět připojen čítač `%d`. V případě, že je výsledných souborů více, jsou všechny zpracovány do jediného ZIP archivu, jehož umístění je z metody vráceno. V opačném případě je vrácena cesta k vytvořenému souboru.

4.4 Nahrávání souborů

Po vytvoření a vykreslení formuláře pro nahrávání souborů při načítání stránky 4.3.6 je k tomuto formuláři připojen plugin DropzoneJS[2]. Při jeho inicializaci je odeslán požadavek zpracovaný metodou `handleGetUploadedFiles` 4.3.6 a vrácená data jsou zpracována a vyhodnocena. Pokud jsou na serveru v příslušném adresáři nějaké soubory nahrány, informace o nich jsou do pluginu přidány a tím jsou soubory zobrazeny i pro uživatele. Zároveň pokud jsou vytvořené náhledy zpracování jednotlivými operátory, jsou příslušné úseky webové stránky rovněž zobrazeny.

DropzoneJS rovněž podporuje nahrávání souborů systémem drag'n'drop. V případě, že používaný webový prohlížeč systém drag'n'drop nepodporuje, soubory pro nahrání lze vybrat kliknutím do nahrávacího pole.

Po případném výběru souborů určených ke zpracování jsou stiskem tlačítka „Zpracovat vybrané soubory“ k formuláři připojena dodatečná data o souborech, které byly nahrány již dříve a dojde k odeslání dat formuláře na server, kde jsou zpracována příslušnou metodou. Po zpracování formuláře a návratu z metody jsou zobrazena tlačítka pro možnost kalibrace nahraných souborů do HDR formátu 4.3.6 a stažení výsledného souboru a pro pokračování ve zpracování (obr. 4.2). Také je zobrazen blok obsahující náhledy zpracování jednotlivými operátory.

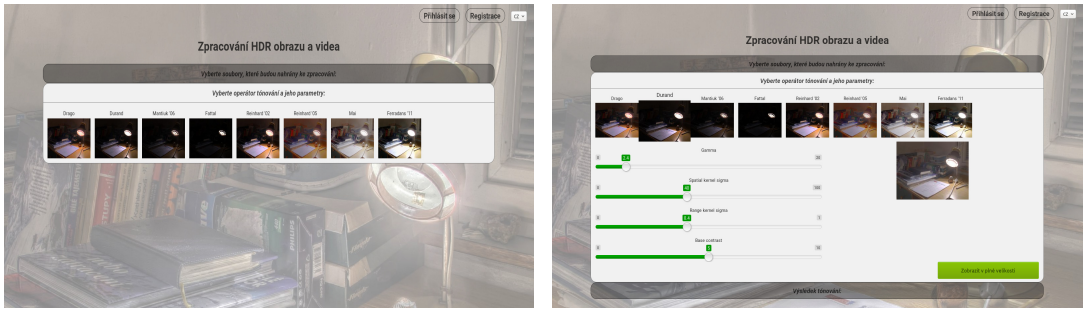


Obrázek 4.2: Ukázka nahrávání souborů

4.5 Použití a použití operátorů, zobrazení náhledu

Po stisku tlačítka pro pokračování je skryta oblast pro nahrané soubory a místo ní je zobrazen úsek pro výběr operátorů (obr. 4.3a). Po výběru některého z nich jsou zobrazeny

parametry daného operátoru společně s náhledem zpracovávaného souboru 4.3.6. Při změně hodnoty některého z parametrů náhled vždy téměř okamžitě změněn 4.3.6, aby odpovídal aktuálním hodnotám (obr. 4.3b).

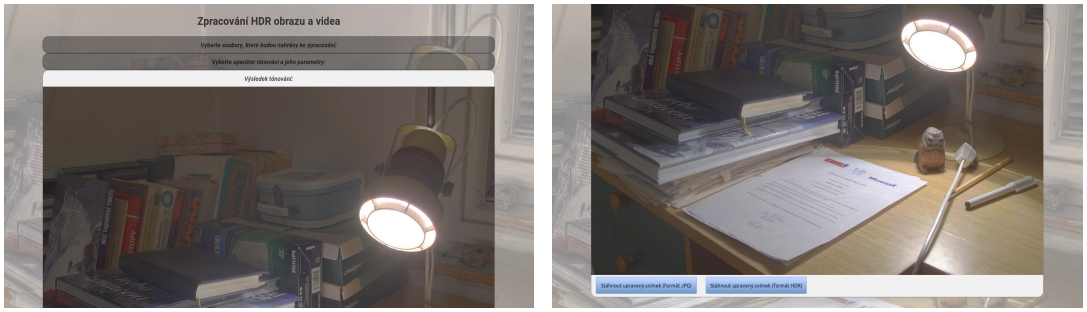


(a) Zobrazení (b) Změna parametru

Obrázek 4.3: Zobrazení operátorů a změna parametrů

4.6 Zobrazení a stažení výsledků

V případě, že je uživatel s provedenými úpravami spokojen, stiskem tlačítka „Zobrazit v plné velikosti“ spouští proces, kdy jsou veškeré provedené úpravy aplikovány na soubor v plném rozlišení 4.3.6. Po zpracování je snímek zobrazen 4.4a a jsou také zobrazena tlačítka pro jeho stažení v JPG nebo HDR formátu 4.4b.



(a) Zobrazení úprav v plném rozlišení (b) Zobrazení tlačítek pro stažení

Obrázek 4.4: Zobrazení úprav v plném rozlišení a tlačítek pro stažení

4.7 Administrátorská část

V této části se nacházení aplikace pro správu registrovaných uživatelů a také pro správu používaných operátorů a jejich parametrů. V obou aplikacích je kladen důraz na jednoduchost použití a s tímto požadavkem byly také navrhovány.

4.7.1 Továrna `appFormFactory`

Tato továrna slouží pro přípravu všech formulářů použitých v administrátorské části.

Metoda `createUsrMng` slouží k přípravě formuláře pro správu uživatelů. Jsou zde do formuláře přidány textové prvky pro jméno, příjmení a e-mail uživatele, prvek pro výběr

uživatelské skupiny, skryté prvky určující ID uživatele v případě úpravy nebo mazání a operace, kterou formulář provádí. Posledním prvkem přidaným do formuláře je odesílací tlačítko celého formuláře.

Pro správu používaných operátorů a jejich parametrů jsou v této továrně připraveny dva formuláře. Metoda `createTmoMng` vytváří formulář pro úpravu operátoru tónování a přidává do něj textové prvky pro zobrazovaný název operátoru a pro název korespondující v programu `pfstools`. Dále jsou do formuláře přidány skryté prvky pro uložení ID upravovaného, případně odstraňovaného operátoru a pro určení operace, která je prováděna a odesílající tlačítko formuláře.

Formulář pro úpravu parametru operátoru je vytvářen metodou `createTmoAttrMng`. Zde je do formuláře přidáno několik textových prvků pro definování zobrazovaného názvu parametru, korespondujícího názvu do programu `pfstools`, prvky pro zadání minimální, maximální, základní hodnoty parametru a kroku, po jakém bude posouván jezdec při úpravě souboru. Dalšími prvky, které jsou do formuláře přidány, jsou výběrové prvky pro určení typu upravovaného parametru a pro výběr parametrů, které nejsou v kombinaci s tímto povoleny.

4.7.2 Presenter `appPresenter`

Zde jsou vyvolány připravené formuláře a implementovány události po jejich odeslání.

V metodě `createComponentUsrMng` je vyvolán formulář pro aplikaci správy uživatelů. Po jeho odeslání jsou prvky formuláře získány metodou `getHttpData`. Pokud je hodnota prvku `_action` ědit, jsou hodnoty formuláře uloženy do asociativního pole `vals`. Jako klíče jsou použity odpovídající názvy sloupců do databázové tabulky `users`. Pokud je ve formuláři také odesíláno ID uživatele, je provedena operace update v tabulce uživatelů, v opačném případě je nový uživatel do této tabulky přidán.

V případě, že hodnota prvku `_action` je „delete“, je provedeno smazání upravovaného uživatele.

Po provedení těchto operací je připravena odpovídající zpráva, která při následném obnovení stránky zobrazena

Pro aplikaci správy operátorů a jejich parametrů jsou vyvolány dva formuláře metodami `createComponentTmoMng` a `createComponentTmoAttrMng`. Implementace událostí po odeslání je u obou vyvolaných formulářů prakticky totožná jako u předchozího případu, v případě hodnoty „delete“ prvku `_action` je odpovídající záznam smazán z příslušné databázové tabulky, v opačném případě jsou prvky opět uloženy do asociativního pole a jako klíče jsou použity názvy sloupců příslušné tabulky. Poté je provedena kontrola, zda jde o úpravu existujícího nebo přidání nového záznamu a jsou volány odpovídající databázové metody. Po jejich provedení je presenterové proměnné `flashMessage` předána zpráva o výsledku a stránka je obnovena.

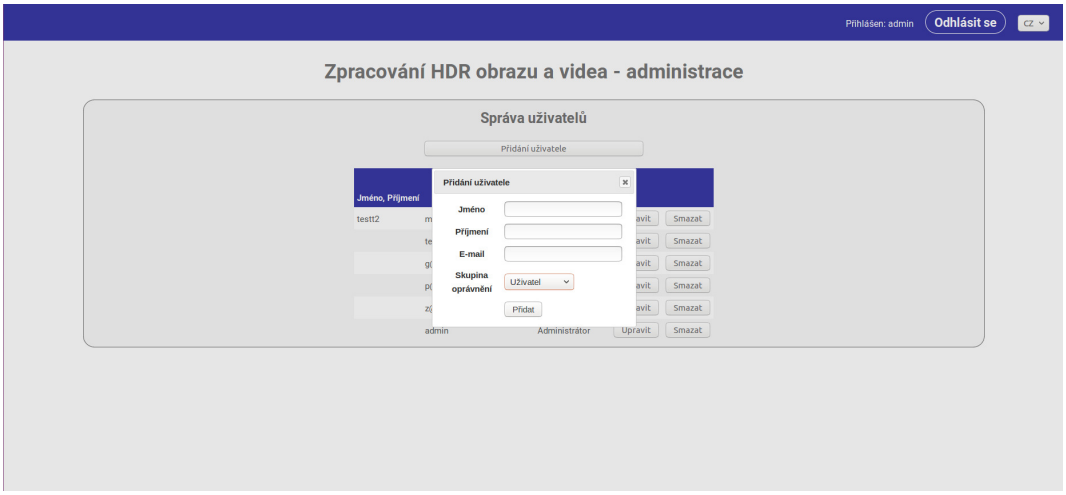
4.7.3 Správa uživatelů

Při načtení aplikace je zobrazeno tlačítko pro přidání uživatele a přímo v šabloně je proveden dotaz do databáze pro získání informací o všech uživatelích. Tyto informace jsou poté zobrazeny v tabulce existujících uživatelů. Na každém řádku jsou připravena také tlačítka

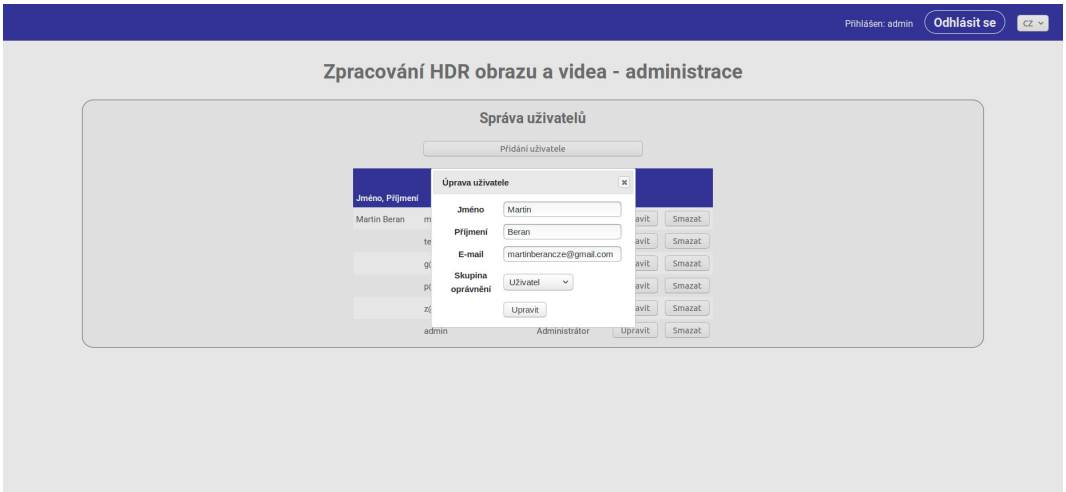
pro úpravu nebo odstranění konkrétního uživatele. Obě tato tlačítka obsahují jako hodnotu informace o uživateli zakódované do formátu JSON.

Při kliknutí na ně je příslušná událost odchycena v JavaScriptu, kde jsou informace dekódovány a předány do příslušných prvků editačního formuláře. V případě přidání (obr. 4.6) nebo úpravy (obr. 4.7) uživatele je tento formulář zobrazen jako modální okno. Při kliknutí na tlačítko mazání je zobrazeno modální okno s požadavkem na potvrzení smazání, ve kterém je na výběr z možností „Ano“ a „Ne“ (obr. 4.8). Po potvrzení smazání je formulář odeslán s polem `_action` nastaveným na hodnotu „delete“. V případě zrušení smazání jsou data z formuláře odstraněna všechna data a modální okno je zavřeno.

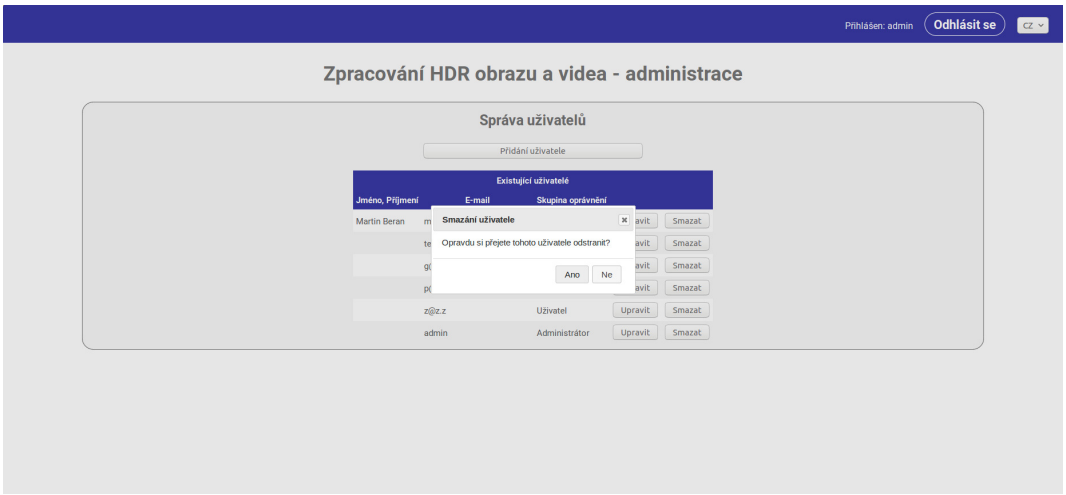
Obrázek 4.5: Uživatelské rozhraní aplikace „Správa uživatelů“



Obrázek 4.6: Přidání nového uživatele



Obrázek 4.7: Úprava uživatele



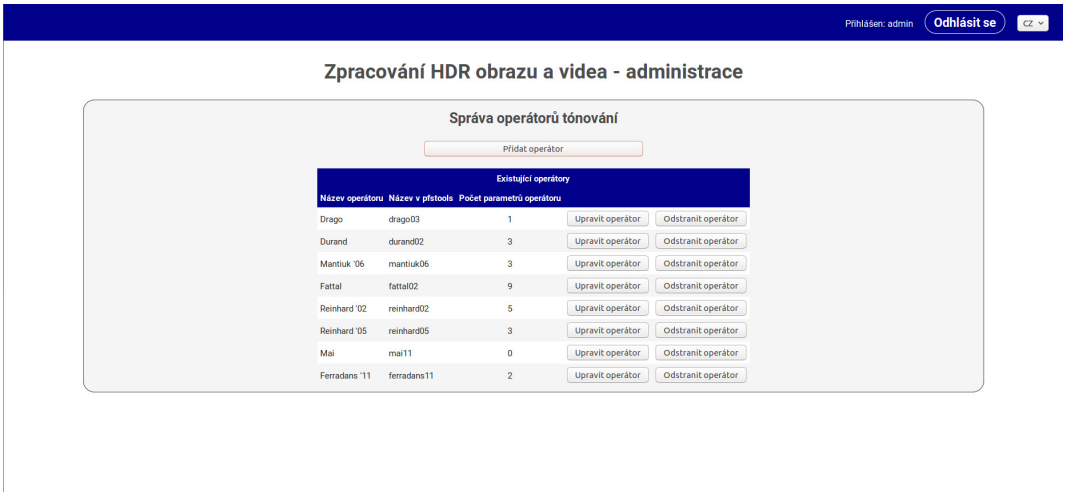
Obrázek 4.8: Potvrzení smazání uživatele

4.7.4 Správa operátorů a parametrů

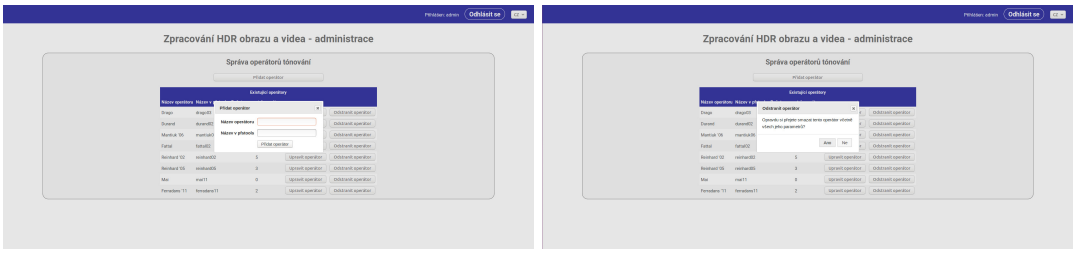
Tato aplikace je implementována obdobně jako „Správa uživatelů“. Při načtení aplikace je zobrazeno tlačítko pro přidání nového operátoru a z databáze jsou získány informace o všech používaných operátorech, které jsou pak zobrazeny v tabulce (obr. 4.9). Pro každý operátor je zobrazen jeho název zobrazovaný při tónování, název používaný v `pfstools` a počet parametrů operátoru. V každém řádku jsou pak zobrazeny tlačítka pro úpravu a smazání operátoru. Opět obsahují informace o operátoru a všech parametrech ve formě JSON řetězce.

Po kliknutí na tlačítko „Přidat operátor“ je zobrazeno modální okno s formulářem pro editaci dat operátoru.

Při mazání je opět zobrazeno modální okno pro potvrzení. Při smazání operátoru jsou odstraněny také všechny jeho parametry.



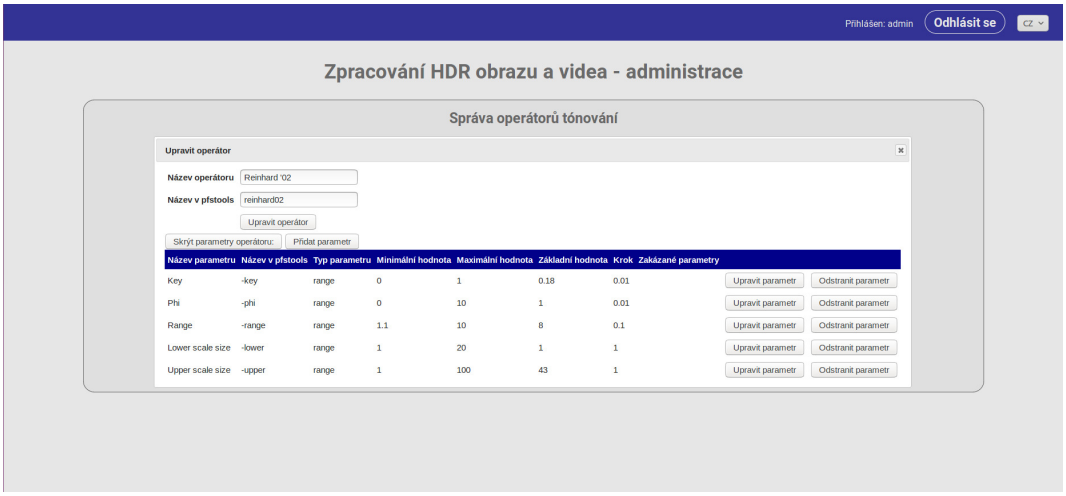
Obrázek 4.9: Zobrazení operátorů



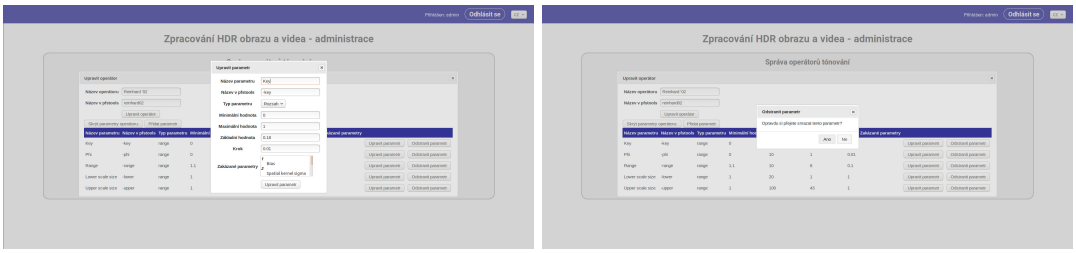
Obrázek 4.10: Vytvoření a mazání uživatele

Parametr lze přidat pouze k existujícímu operátoru. Z toho důvodu jsou tlačítka pro zobrazení existujících parametrů vybraného operátoru a pro přidání nového parametru zobrazena až po výběru úpravy operátoru (obr. 4.11).

Postup pro přidání, editaci a mazání jednotlivých parametrů je stejný jako u operátorů. Po kliknutí na tlačítko „Zobrazit parametry operátoru“ je zobrazena tabulka existujících parametrů, ve které jsou uvedeny všechny jejich vlastnosti. Na každém řádku se opět nacházejí tlačítka pro úpravu a smazání parametru.



Obrázek 4.11: Zobrazení parametrů operátoru



Obrázek 4.12: Úprava a mazání parametru

Kapitola 5

Testování systému

Aplikace procházela testováním již při jejím návrhu, kdy byl množině uživatelů různých věkových skupin předložen první návrh uživatelského rozhraní. Případné připomínky a nedostatky byly zaznamenány a v další verzi zohledněny a odstraněny. Celý vývoj uživatelského rozhraní byl zaměřen na jednoduchost a přehlednost. Jednotlivé prvky UI byly vytvářeny z přihlédnutím na intuitivnost ovládání celé aplikace. Fáze vývoje a testování návrhu probíhala až do odstranění všech nedostatků. V průběhu testování uživatelského rozhraní došlo k výrazné úpravě designu a všech ovládacích prvků aplikace.

Po dokončení „backendové“ části systému bylo provedeno testování aplikace znovu, tentokrát pro ověření funkčnosti všech jejích částí. Opět byl systém představen množině uživatelů, kteří měli za úkol bez předchozích znalostí systému vybrat a nahrát soubory z připraveného datasetu, zvolit operátory pro jejich úpravu a následně upravit soubory. Při této činnosti bylo sledováno, zda si uživatelé vždy ví rady. V průběhu testování této části byla funkce systému rozšířena o další operátory mapování tónů a dále o možnost stažení výsledků v různých formátech.

Kapitola 6

Závěr

Cílem této bakalářské práce bylo vytvořit webový systém, který bude umožňovat nahrání HDR či LDR fotografií nebo jednotlivých HDR snímků v případě zpracování videa a následná aplikace operátorů tónování na zpracované soubory včetně možnosti časově koherentního tónování. Jelikož se jedná o webovou aplikaci, byla nutná komunikace s uživateli při návrhu uživatelského rozhraní a následně také při testování jeho funkčnosti a funkčnosti celého systému. Ve fázi návrhu byly uživatelům představeny modely jednotlivých verzí UI a při testování měli uživatelé za úkol zpracovat připravené testovací soubory bez předchozích znalostí testovaného systému.

Pro doplnění informací o dané problematice bylo potřeba nastudovat literaturu týkající se zpracování a konverze obrazu do HDR formátu, dále pak literaturu zaměřenou na popis a možnosti použitých operátorů tónování. Pro implementaci celého systému bylo nutné rozšíření vědomostí o použitých programovacích a skriptovacích jazycích tak jako o využitých frameworcích. Pro úspěšný návrh uživatelského rozhraní celé aplikace bylo také nutné nastudovat potřebnou literaturu tak, aby byla tato část při použití jednoduchá a intuitivní.

Při vytváření „frontendu“ této bakalářské práce byl kladen důraz na jednoduchost a intuitivní ovládání. Při implementaci „backendu“ byly využity některé z možností frameworku Nette a skriptovacích jazyků PHP a JavaScript. Vlastní konverze souborů do HDR formátu a následná aplikace operátorů tónování je prováděna programem `pfstools`. Pro mapování tónů jsou použity operátory poskytované právě tímto programem, včetně operátorů pro umožňujících časově koherentní mapování tónů.

Aby bylo možné celý systém pohodlně a efektivně spravovat, bylo vytvořeno administrátorské rozhraní. Zde jsou implementovány aplikace pro přidání či modifikaci registrovaných uživatelů a také pro úpravy používaných operátorů tónování včetně jejich parametrů.

Byl proveden důkladný průzkum existujícího softwaru a webových služeb zabývajících se problematikou zpracování HDR obrazu a jeho tónování. Po zjištění, že žádná z volně dostupných webových služeb plně neodpovídá možnostem této aplikace věřím, že by vytvořený systém mohl být efektivně využíván v reálném provozu.

Jako možná rozšíření či vylepšení této aplikace se nabízí zvýšení efektivity při zpracování nahraných souborů a při procesu tónování, možnost nahrávání videozáznamů, zlepšení komunikace z uživatelem při zpracování a úpravě souborů nebo přidání podpory pro další jazyky.

Celý systém, stejně jako instalační skripty byly vyvíjeny a testovány na operačním systému Ubuntu 16.04 64-bit. Nelze tedy zaručit správné fungování při použití jiného operačního systému.

Literatura

[1] Best Features that JQuery Offer – Coder's Eye.
URL https://coderseye.com/tutorials/jquery/features/

[2] dropzone^{js}.
URL http://www.dropzonejs.com/

[3] EVALUATION OF TONE MAPPING OPERATORS. [Online; navštíveno 9.4.2017].
URL http://cadik.posvete.cz/tmo/

[4] Historical trends in the usage of server-side programming languages for websites.
URL https:
//w3techs.com/technologies/history_overview/programming_language/ms/y

[5] ion.rangeSlider. [Online; navštíveno 9.4.2017].
URL https://github.com/IonDen/ion.rangeSlider

[6] jQuery - write less, do more. [Online; navštíveno 9.4.2017].
URL http://jquery.com/

[7] Nette dokumentace. [Online; navštíveno 9.4.2017].
URL https://doc.nette.org/cs/2.3/

[8] pfstools man pages.
URL http://pfstools.sourceforge.net/man_pages.html

[9] PHP Documentation. [Online; navštíveno 9.4.2017].
URL http://php.net/docs.php

[10] Web Programming Step by Step, Lecture 18 AJAX.
URL http://courses.cs.washington.edu/courses/cse190m/11sp/lectures/
slides/lecture18-ajax.shtml#slide5

[11] Web Technology and Trend: HTML5.
URL http://www-scf.usc.edu/~chenemil/itp104/webtech.html

[12] Window.sessionStorage. [Online; navštíveno 9.4.2017].
URL
https://developer.mozilla.org/en-US/docs/Web/API/Window/sessionStorage

[13] Životní cyklus presenteru. [Online; navštíveno 9.4.2017].
URL
https://doc.nette.org/cs/2.3/presenters#toc-zivotni-cyklus-presenteru

[14] Drago, F.; Myszkowski, K.; Annen, T.; aj.: Adaptive Logarithmic Mapping For Displaying High Contrast Scenes. *Computer Graphics Forum*, ročník 22, č. 3, 2003: s. 419–426, ISSN 1467-8659, doi:10.1111/1467-8659.00689.
URL http://dx.doi.org/10.1111/1467-8659.00689

[15] Durand, F.; Dorsey, J.: Fast Bilateral Filtering for the Display of High-dynamic-range Images. *ACM Trans. Graph.*, ročník 21, č. 3, Červenec 2002: s. 257–266, ISSN 0730-0301, doi:10.1145/566654.566574.
URL http://doi.acm.org/10.1145/566654.566574

[16] Fattal, R.; Lischinski, D.; Werman, M.: Gradient Domain High Dynamic Range Compression. *ACM Trans. Graph.*, ročník 21, č. 3, Červenec 2002: s. 249–256, ISSN 0730-0301, doi:10.1145/566654.566573.
URL http://doi.acm.org/10.1145/566654.566573

[17] Ferradans, S.; Bertalmio, M.; Provenzi, E.; aj.: An Analysis of Visual Adaptation and Contrast Perception for Tone Mapping. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, ročník 33, č. 10, Oct 2011: s. 2002–2012, ISSN 0162-8828, doi:10.1109/TPAMI.2011.46.

[18] Mai, Z.; Mansour, H.; Mantiuk, R.; aj.: Optimizing a Tone Curve for Backward-Compatible High Dynamic Range Image and Video Compression. *IEEE Transactions on Image Processing*, ročník 20, č. 6, June 2011: s. 1558–1571, ISSN 1057-7149, doi:10.1109/TIP.2010.2095866.

[19] Mantiuk, R.; Myszkowski, K.; Seidel, H.-P.: A Perceptual Framework for Contrast Processing of High Dynamic Range Images. *ACM Trans. Appl. Percept.*, ročník 3, č. 3, Červenec 2006: s. 286–308, ISSN 1544-3558, doi:10.1145/1166087.1166095.
URL http://doi.acm.org/10.1145/1166087.1166095

[20] Pattanaik, S. N.; Tumblin, J.; Yee, H.; aj.: Time-dependent Visual Adaptation for Fast Realistic Image Display. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 2000, ISBN 1-58113-208-5, s. 47–54, doi:10.1145/344779.344810.
URL http://dx.doi.org/10.1145/344779.344810

[21] Reinhard, E.; Devlin, K.: Dynamic range reduction inspired by photoreceptor physiology. *IEEE Transactions on Visualization and Computer Graphics*, ročník 11, č. 1, Jan 2005: s. 13–24, ISSN 1077-2626, doi:10.1109/TVCG.2005.9.

[22] Reinhard, E.; Stark, M.; Shirley, P.; aj.: Photographic Tone Reproduction for Digital Images. *ACM Trans. Graph.*, ročník 21, č. 3, Červenec 2002: s. 267–276, ISSN 0730-0301, doi:10.1145/566654.566575.
URL http://doi.acm.org/10.1145/566654.566575

Přílohy

Seznam příloh

Příloha A

Obsah přiloženého paměťového média

- **src** - zdrojové soubory aplikace
- **doc** - bakalářská práce
  - **xberan34-zprac-hdr-obrazu.pdf** - text bakalářské práce
  - **doc-src** - zdrojové soubory textu
- **dataset** - přiložený dataset k otestování aplikace
- **install** - instalační a importní skripty
  - **install_all.sh** - hlavní instalační skript
  - **install_lamp.sh** - skript pro instalaci systému Apache2, jazyka MySQL a jazyka PHP včetně systému phpMyAdmin
  - **install_pfstools.sh** - skript pro instalaci programu pfstools a všech jeho závislostí
  - **import_db_web.sh** - skript importující adresář webu a databázi pro webový systém
  - **install_apache.sh** - dílčí skript první části zajišťující instalaci apache2
  - **install_mysql.sh** - dílčí skript první části zajišťující instalaci MySQL
  - **install_php.sh** - dílčí skript první části zajišťující instalaci php a phpMyAdmin
- db_script.sql - SQL skript obsahující připravenou databázi pro aplikaci .1
- **login.txt** - důležité přihlašovací údaje
- **public.ogv** - ukázka použití veřejné části aplikace
- **admin.ogv** - ukázka použití administrátorské části aplikace

Příloha B

Manuál

Pro funkčnost tohoto webového systému na nově nainstalovaném systému je nutné spustit instalační skript `install_all.sh`. Z něj jsou postupně spouštěny tři hlavní části instalace.

První je instalace a konfigurace systému apache2, jazyka MySQL a potřebných součástí jazyka PHP včetně phpMyAdmin.

Další částí je instalace potřebného softwaru pro správnou funkci programu pfstools.

V poslední části instalace je importována databáze webového systému. Zde je nutné zadat absolutní cestu k odpovídajícímu SQL skriptu. Dále je importován adresář webu. Zde je také potřeba zadat absolutní cestu pro přístup k tomuto adresáři. Import je proveden do základního umístění pro systém apache, a to `/var/www/html`. Po dokončení instalace je dále potřeba provést drobné úpravy několika souborů:

- soubor `/etc/apache2/sites-enabled/000-default.conf`
  - u klíče `DocumentRoot` je potřeba změnit umístění na `/var/www/html/www`
  - na následující řádek po ukončovacím znaku `</Directory>` vložit tento úsek kódu:

    ```
    <Directory /var/www/html/www/>
    Options Indexes FollowSymLinks MultiViews
    AllowOverride All
    Require all granted
    </Directory>
    ```

- soubor `/etc/apache2/sites-available/default-ssl.conf`
  - hodnotu klíče `DocumentRoot` změnit na `/var/www/html/www`

Po provedení těchto operací je ještě potřeba restartovat systém apache příkazem `service apache2 restart`.