VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

WEBOVÝ NÁSTROJ PRO PÁROVÝ TEST OBRÁZKŮ

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE VÍT ŠEVČÍK
AUTHOR

BRNO 2015

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

WEBOVÝ NÁSTROJ PRO PÁROVÝ TEST OBRÁZKŮ

THESIS TITLE

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE VÍT ŠEVČÍK

AUTHOR

VEDOUCÍ PRÁCE Ing. MARTIN ČADÍK, Ph.D.

SUPERVISOR

BRNO 2015

Abstrakt

Cílem této bakalářské práce je vytvoření internetové aplikace pro experimentální vyhodnocení multimediálních dat (obrazu a videa) pomocí párového testování. V práci jsou popsány další psychofyzikální metody pro experimenty, problémy zobrazování multimediálních dat v prohlížečích, parametry testů aplikace a její administrátorské rozhraní. V závěru práce je popsán reálný psychofyzikální experiment, který byl pomocí této aplikace zprostředkován.

Abstract

The goal of this thesis is to create an internet application for experimental evaluation of multimedia data (image and video) by pair testing. The thesis provides description of psychophysical methods for experiments, problems and solutions to displaying multimedia in web browsers, parameters of application tests, and its administration interface. The work is concluded by description of specific psychophysical experiment which was performed using presented application.

Klíčová slova

2AFC, Psychofyzikální metody, Experimentální testování, Párové testování

Keywords

2AFC, Psychophysical methods, Experimental testing, Pairwise testing

Citace

Vít Ševčík: Webový nástroj pro párový test obrázků, bakalářská práce, Brno, FIT VUT v Brně, 2015

Webový nástroj pro párový test obrázků

Prohlášení

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

. . . . . .
Vít Ševčík
20. května 2015

Poděkování

Rád bych poděkoval svému vedoucímu Ing. Martinu Čadíkovi, Ph.D., za ochotnou pomoc a cenné rady. Dále bych rád poděkoval panu Ing. Tomáši Kašpárkovi za zprostředkování hostingu a poskytnutí cenných rad. Tato práce vznikla v rámci projektu LOCATE 4SGA8694, který je financován z programu SoMoPro II, spolufinancovaného Evropskou unií a Jihomoravským krajem.

Obsah

1 Úvod **3**

2 Psychofyzikální experimenty **4**
2.1 Psychofyzika . . . 4
2.2 Experimentování . . . 4
2.3 Nastavovací metody . . . 5
2.4 Rozhodovací metody . . . 5
2.4.1 Metoda 2AFC . . . 6
2.5 Existující řešení . . . 7
2.5.1 Ranker . . . 7
2.5.2 Presentation . . . 7
2.5.3 PsychWithJavaM . . . 7
2.5.4 Psychtoolbox-3 . . . 7

3 Návrh systému **9**
3.1 Testy a sady obrázků . . . 9
3.2 Model případů užití . . . 9
3.3 Entitně-vztahový model . . . 10
3.4 Databáze . . . 13
3.5 Prohlížeče a mutlimédia . . . 13
3.6 Konvertování obrázků a videí . . . 14

4 Implementace **16**
4.1 Použité technologie . . . 16
4.2 Adresářová struktura . . . 17
4.3 Párový test . . . 17
4.4 Parametrizace testů . . . 18
4.4.1 Promíchání sad . . . 18
4.4.2 Sekce instrukcí a outro . . . 19
4.4.3 Aktivní sady obrázků pro testy . . . 19
4.4.4 Sekce dotazník . . . 19
4.5 Další nastavení . . . 21
4.5.1 Pořadí obrázků v sadě . . . 21
4.5.2 Sady s referencí . . . 21
4.5.3 Připojení databáze . . . 21
4.5.4 Ranker.ini . . . 21
4.6 Konvertování obrázků . . . 21
4.7 Logování chyb . . . 22

4.8 Administrátorské rozhraní 23
4.8.1 Zobrazování výsledků 23
4.8.2 Správa testů 23
4.8.3 Aktualizace databáze 24

5 Ukázkový experiment **26**
5.1 Experiment 26
5.2 Vyhodnocení výsledků 27

6 Závěr **30**

Kapitola 1

Úvod

Počítačová grafika je obor, který využívá počítače k tvorbě umělých grafických objektů. Jednou z nedílných součástí výzkumu počítačové grafiky je její experimentální vyhodnocení. Toto vyhodnocení je prováděno pomocí uživatelských experimentů, které se v praxi velice dobře osvědčují. Při experimentech jsou využívány psychofyzikální metody, které zkoumají vzájemné vztahy mezi fyzikálním a psychickým světem. Příkladem nám může být metoda *Yes-No*, kdy se ptáme uživatele, jestli zaregistroval podnět či nikoliv. Další metodou je například *Matching* (metoda průměrné chyby), kdy jsou uživateli prezentovány dva podněty a jeho úkolem je manipulovat s jedním z nich, aby odpovídal tomu druhému.

Cílem této bakalářské práce je vytvoření internetové aplikace pro experimentální vyhodnocení multimediálních dat (obrazu a vide) pomocí metody 2AFC (Two-alternative forced choice). Metoda 2AFC využívá párového testování a umožňuje jednoduše a robustně testovat podněty na uživatelích, kdy jsou jim prezentovány dva podněty a uživatel je vyzván, aby zvolil právě jeden z nich. Je používána k testování rozsahu volby u lidí nebo zvířat. Tato aplikace bude dále umožňovat správu obrázků, videí a jejich sad v různých testech, bude schopna zobrazovat výsledky ohodnocení obrázků nebo videí v různých formátech a bude kompatibilní s existujícím programem Ranker 2.5.1.

Ve druhé kapitole této práce jsou popsány různé psychofyzické metody, role psychofyziky při rozhodování mezi dvěma podněty.

Třetí kapitola se zabývá návrhem systému. Je zde popsána i problematika multimédií v internetových prohlížečích, úvod do testů aplikace nebo sad obrázků, jaký je datový model aplikace a různé nástroje pro převod multimédií na různé formáty.

Ve čtvrté kapitole je popsána implementace samotné aplikace. Jsou zde uvedeny konfigurační soubory, které řídí chod testů aplikace, jejich struktura a je zde popsáno administrativní rozhraní této aplikace, které slouží pro její správu.

Předposlední kapitola se věnuje testování aplikace a výsledkům vytvořených experimentů s obrázky a videi.

V závěrečné kapitole je celá práce shrnuta.

Kapitola 2

Psychofyzikální experimenty

2.1 Psychofyzika

Jedná se o exaktní vědu, která zkoumá vztah mezi fyzickými podněty, pocity a vjemy. U psychofyziky se testují lidské smysly — zrak, sluch, čich, hmat a chuť. V naší práci se budeme zabývat hlavně zrakem. V psychofyzice se nejčastěji využívá metody *threshold* (práh). Dále budeme pracovat s pojmem podnět, pod kterým si můžeme představit nějaký stimul, impuls, popud nebo motiv.

Psychofyzikální metody představují nástroje, pomocí kterých jsme schopni měřit vnímání a výkon uživatele. Používají se pro odhalení základního procesu vnímání k posouzení uživatelovi výkonnosti. Sledujeme tedy hodnotu, kdy uživatelé reagují na podněty. Například, při využití prahové metody se na začátku takového testu určí několik podnětů, které se pohybují kolem prahové hodnoty (hodnota, kdy uživatel reaguje na podnět). Podněty jsou pak předkládány uživateli v náhodném pořadí, kdy uživatel nám dává zpětnou vazbu, že zaznamenal podnět nebo ne. Každý podnět je předložen vícekrát a počet shodujících se odpovědí udává hodnotu podnětu.

2.2 Experimentování

Existují dva druhy experimentů — rozhodování (judgments) a nastavování (adjustments) [4]. Tyto typy jsou velice efektivní a jsou vzájemně obrácené – jeden je opakem toho druhého. Ovšem v praxi je jejich užití dost podobné. Při těchto experimentech rozlišujeme ještě podúkoly — detekce (uživatel rozhoduje mezi přítomností či nepřítomností podnětu), identifikace (poznávání podnětu, jestli se již v minulosti vyskytl a uchoval v paměti) a diskriminace (rozlišování podnětů a vybrání jednoho z nich na úkor toho druhého). U obou druhů experimentu chceme po uživateli, aby posuzoval kritérium, což může být barva, vzor, odstín apod.

Typ nastavování je orientován na vnímání a je velice subjektivní, protože záleží na tom, jak uživatel pochopí kritéria experimentu. Pokud bychom mu kritéria vysvětlili špatně nebo by uživatel správně nepochopil, co má dělat, budou výsledky experimentu zkreslené z důvodů odhadů a špatné manipulace uživatele s podněty.

Je důležité poznamenat, že existují metody, kde s podněty manipulují experimentátoři, ale jsou i takové metody, kde je nechána volnost uživateli a je mu umožněna manipulace s podněty. Pod tím si můžeme představit, že uživatel bude měnit jas nějakého obrázku nebo měnit hlasitost tónu. Mezi typické metody tohoto typu patří *threshold* (metoda schodů,

metoda limitů), *matching* (porovnávání podnětů a manipulace právě s jedním z nich) a takzvaná *nulling* (metoda úpravy nebo také metoda průměrné chyby).

Typ rozhodování je orientován na výkon uživatele. Měříme kolikrát a za jak dlouho se uživatel dokáže rozhodnout. Nevýhodou rozhodovací metody je, že se většinou tato metoda zdá uživatelům zdlouhavá a méně zábavná než metoda nastavovací [4]. Značnou výhodou rozhodování je jednodušší analýza, protože podněty jsou pod kontrolou experimentátora a uživatel nemá možnost manipulovat s podněty. Mezi nejznámější metody rozhodování patří Yes-No (ptáme se uživatele, jestli zaregistroval podnět či nikoliv) nebo 2AFC (uživateli jsou prezentovány dva podněty a je požádán, aby vybral právě jeden z nich).

2.3 Nastavovací metody

Jak již bylo zmíněno výše, tyto metody se uživatelům líbí více [4], protože mohou u některých metod manipulovat s podněty. Pomocí těchto metod jsme schopni rychle a dobře získat od uživatelů data, což je u experimentů žádoucí, ale ne všechny experimenty lze provádět těmito metodami.

Mezi nejznámější nastavovací metody bezpochyby patří metoda *práh* (threshhold). Tato metoda může být aplikována různými způsoby: metodou *limitů* a metodou *schodiště*. Metoda limitů má dva testovací způsoby — vzestupný a sestupný. Při vzestupném způsobu nastavíme výchozí hodnotu co možná nejvýše a postupným snižováním hodnoty testujeme, kdy uživatel zareaguje na podnět. U sestupné je to naopak. Výchozí hodnota je co možná nejmenší a postupným zvyšováním této hodnoty sledujeme, kdy uživatel zaregistruje změnu. Tyto hodnoty považujeme za prahy uživatele a získávání těchto hodnot se opakuje. Následně se vypočítá průměrná hodnota vyššího nebo nižšího prahu. Zjišťování hodnoty nižšího a vyššího prahu se střídá. Toto střídání může mít za následek, že si uživatel zvykne na úroveň, při které očekává změnu podnětu a může nám tak zkreslit výsledek. Proto se využívá metoda schodiště, při které se při reakci uživatele na podnět změní i směr hodnoty. Například, při reakci uživatele na zvyšování hlasitosti nějakého zvuku začneme hlasitost tónu snižovat. Uživatel na změnu podnětu zareaguje a my budeme hlasitost zvuku dále zvyšovat. Toto se může několikrát opakovat. Hodnoty experimentátor mění skokově.

Matching je metoda, kdy jsou uživateli prezentovány dva podněty. Uživatel je požádán, aby s podněty manipuloval tak, aby se co nejvíce podobaly. Existují dva typy této metody. U prvního typu jsou podněty velice podobné, takřka k nerozeznání. U druhého typu je uživatel pověřen, aby nastavoval jen určitý aspekt podnětu (jas, sytost apod.). Tato metoda je velice užitečná, ale výsledky, které jsou pomocí druhého typu této metody získány, mohou být zkreslené vlivem subjektivního vnímání daného aspektu rozdílnými jedinci.

Nulling je nejsilnější metoda. Zakládá se na zkreslení sledovaného podnětu a uživatel musí toto zkreslení pomocí manipulace s podnětem odstranit. Zkreslení podnětu je většinou jednoznačné a je tedy jednoduché pro uživatele poznat, kdy je toto zkreslení odstraněno.

2.4 Rozhodovací metody

U těchto metod má všechny podněty pod kontrolou experimentátor a jsou snadno analyzovatelné, což řadíme mezi výhody. Tyto metody se ovšem uživatelům zdají často zdlouhavé [4].

Yes-No (ano-ne) je nejjednodušší rozhodovací metoda, při níž je často využívána detekce. Otázkou experimentátora může být například: spatřili jste podnět, viděli jste ho? Jsou

tedy jen dvě alternativní odpovědi uživatele — ano, ne. Na základě získaných odpovědí je vypočítána pravděpodobnost, kdy uživatelé odpověděli ano a kdy ne.

2.4.1 Metoda 2AFC

Pokud by před nás dal někdo deset obrázků a chtěl po nás, abychom je seřadili podle toho, jak se nám nejvíce líbí, bude nám to trvat relativně dlouho, zvláště, když se obrázky liší malými detaily. Ovšem pokud by se naše volba měla vztahovat pouze na dva obrázky, bude pro nás jednodušší určit, který se nám více líbí.

2AFC (Two-alternative forced choice) je tedy objektivní metoda, kdy jsou uživateli prezentovány dva podněty v náhodném pořadí a uživatel má za úkol vybrat právě jeden z nich 2.1. Tato metoda se řadí mezi rozhodovací metody a naše aplikace této metody využívá při svých experimentech. Pomocí této metody se dosahuje přesnějších výsledků než u prahových metod. Podněty mohou být prezentovány vedle sebe nebo postupně. Dále hraje velkou roli čas. Jednotlivé podněty mohou být rozdílné, ale taky si mohou být velice podobné a tedy bude pro uživatele těžší se mezi dvěma podněty rozhodnout. Čas můžeme brát i jako kontrolní faktor. Pokud se uživatel rozhodl ve vteřině, znamená to pro nás, že podněty byly snadno k rozeznání nebo uživatel bez rozmyšlení vybral jeden z nich. Totéž platí u vysoké hodnoty rozhodování. Tato hodnota nám dává zpětnou vazbu v tom, že uživatel během testu zřejmě dělal další jiné věci a testu se moc nevěnoval. Toto měření nám může dávat zkreslené výsledky.

Metodu lze rozšířit i o referenční podnět. To znamená, že uživateli se budou zobrazovat tři podněty, kde dva budou podobné referenčnímu a uživatel bude vyzván, aby z těchto dvou podnětů vybral právě jeden, který nejvíce odpovídá referenčnímu podnětu. Příkladem nám může být experiment, kdy testujeme metody, pomocí kterých převádíme barevné obrázky na obrázky černobílé. Existuje jeden barevný obrázek a k němu jsou zobrazovány obrázky černobílé 2.2. Uživatel je vyzván, aby zvolil ten černobílý obrázek, který je nejpřirozenější tomu barevnému.

Metoda si zakládá na párovém testování a prezentaci vždy dvou podnětů. Existuje ovšem více verzí této metody, kde může být prezentováno více než dva podněty (Forced-Choice metody) [3].

Obrázek 2.1: Ukázka párového testu.

(a) Obrázek 1 (b) Referenční obrázek (c) Obrázek 2

Obrázek 2.2: Ukázka párového testu s referencí.

2.5 Existující řešení

Existuje mnoho open source projektů, různých rozhraní nebo rozšíření existujících programů pro aplikování psychofyzikálních metod a uskutečňování uživatelských experimentů pomocí těchto metod. Není mi ovšem známo řešení, které by nahlíželo na psychofyzikální experimentování pomocí internetu tak, jako v našem případě. Většinou se jednalo o rozšíření již existujících programů nebo o programy samotné.

2.5.1 Ranker

Ranker je program pro řazení obrázků. Jedná se o aplikaci v C++, která umožňuje uživateli nahrát a seřadit obrázky bud formou přímého řazení nebo pomocí párových testů (2AFC). Výsledky si pak můžeme exportovat ve formátech CSV nebo PDF. Tento program podporuje i video formát. Dále je možné v této aplikaci různě modifikovat testy (vertikální nebo horizontální zobrazení párů obrázků).

2.5.2 Presentation

Je světově nejpopulárnější program pro neurovědní experimenty. Lze ho spustit na operačním systému Windows XP, Vista7, 7, 8. Umožňuje nahrát podněty do programu a přesně definovat úkoly, které budou provádět daní uživatelé. Umožňuje prezentovat sluchové nebo vizuální (2D nebo 3D) podněty zvlášť nebo současně. Je programovatelný i rozšířitelný. Je to placený software, ale je možné využít 30-ti denní bezplatnou licenci pro vyzkoušení.

2.5.3 PsychWithJavaM

Je open source projekt, který nabízí nástroje pro vytváření psychofyzikálních experimentů pomocí Javy. Tento projekt může být použit spolu s programy Matlab nebo Mathematica. Tento projekt je šířen pod licencí GNU verze 2.

2.5.4 Psychtoolbox-3

Je to bezplatné rozšíření pro Matlab a GNU Octave programy, které se zaměřuje na vizuální výzkum. Toto rozhraní umožňuje na uživatelích testovat vizuální a sluchové podněty. Má dvanáct tisíc uživatelů a fórum, kde je dobře citováno. Psychtoolbox-3 je založen na jeho starší verzi Psychtoolbox-2, ale jeho MATLAB rozšíření (v jazyce C) bylo napsáno, aby bylo více modulární a využívalo OpenGL.

Psychtoolbox nabízí další možnosti pro programování psychofyzikálních a psychologických experimentů, jako je například PsychJava pro jazyk Java nebo PsychoPy pro python. Aktuální verze Psychtoolbox-3 porporuje Matlab 7.x a Octave 3.2.x na Mac OSX, Linux a Windows. Toto rozšíření je šířeno pod licencí GNU verze 3.

Kapitola 3

Návrh systému

V této kapitole je popsán návrh systému. Při návrhu systému musel být brán zřetel na to, aby aplikace byla kompatibilní s existujícím programem Ranker 2.5.1.

Webové stránky patří do množiny softwarových projektů a je potřeba na ně tak i nahlížet. Můžeme tedy využít různých metodik softwarového inženýrství.

U webových stránek je vhodné použít diagram případů užití (UML), pomocí kterého jasně a jednoduše lze stanovit, kdo bude se systémem zacházet a ER model, který znázorňuje model dat aplikace.

3.1 Testy a sady obrázků

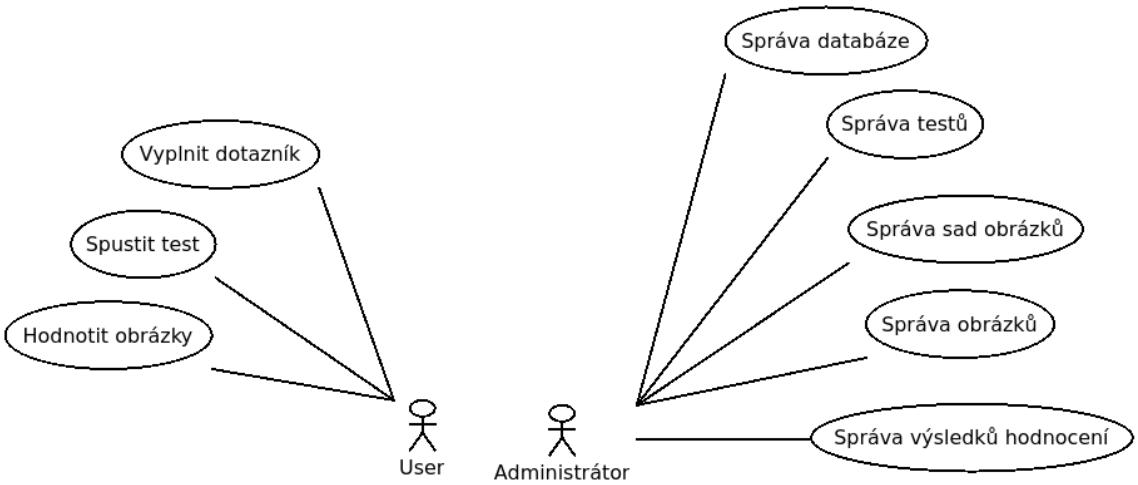
Ještě než začneme popisovat návrh naší internetové aplikace, musíme pochopit, co to vlastně test je a jak probíhá.

Každý test je adresář na serveru, může obsahovat obrázky nebo videa a sady (také adresáře) obrázků nebo videí. Uživateli je pak poslán test, ve kterém jsou zahrnuty všechny obrázky daného testu – všichni v rámci testu testují všechno. Při testu se uživateli následně zobrazuje pár obrázků nebo pár obrázků s referencí. Uživatel si vybírá (ohodnocuje) jeden z nich pomocí kliknutí myši na obrázek nebo kurzorových kláves (šipka vlevo, šipka vpravo). Obrázky obsažené v páru vždy patří do stejné sady v daném testu a v rámci sady jsou utvořeny páry každý s každým. Pokud je páru přidělen i referenční obrázek, je zobrazen mezi párovými obrázky a uživatel vybírá právě jedech z nich, který se mu zdá nejlepší. Počet párů v sadě se vypočítá pomocí rovnice, kde n je počet obrázků v sadě 3.1. Celkový počet párů obrázků v testu je součet hodnoty této rovnice, aplikované na každou sadu v testu.

$$x = \frac{n * (n - 1)}{2} \tag{3.1}$$

3.2 Model případů užití

Jazyk UML se používá jako univerzální jazyk při vývoji softwaru mezi klientem a programátorem. Pomocí něj znázorňujeme účastníky, kteří budou zahrnuti v systému, jaké hranice systém má a jak budou jednotlivý účastníci se systémem pracovat [1]. Tím vytvoříme model případů užití, který znázorníme pomocí diagramu případů užití 3.1.



Obrázek 3.1: Diagram případů užití

3.3 Entitně-vztahový model

Abychom mohli vytvořit rychlou internetovou aplikaci je v dnešní době absolutně nezbytné mít dobrý návrh datového modelu. A k tomu nám výborně poslouží ER model, v anglické literatuře definován jako Entity-relationship model, který je založen na chápání světa jako množiny základních objektů - entit (Entity) a vztahů (Relationship) mezi nimi. Tento model neznázorňuje operace nad daty, ale strukturu dat (tak, jakoby byly v klidu). Někdy ER model můžeme vidět pod zkratkou ERA, kde třetím prvkem jsou atributy (Attributes.)[8]. ER model znázorňujeme pomocí ER diagramu, který můžeme považovat za finální návrh naší databáze.

ER diagram naší aplikace můžeme vidět na obrázku 3.2, kde jsou zviditelněny jednotlivé entitní množiny a vztahy mezi nimi. Pro pochopení jejich významu si tyto entitní množiny následně popíšeme:

- **Users** – tato entitní množina uchovává záznamy o uživatelích, kterým byl test poslán a následně ho spustili. Jelikož ohodnocování obrázků probíhá anonymně přes internet, využívá jako jednoznačný identifikátor hodnotu, kterou vygenerujeme pomocí PHP metody session_id(). Ovšem v našem systému dále pracujeme s identifikátorem tohoto záznamu, který je v rámci systému jedinečný, a to hlavně proto, že zabírá méně místa než řetězová hodnota, kterou nám vrací výše zmíněná metoda session_id().

- **Tests** – je entitní množinou, která uchovává data o testech, které jsou v našem systému vytvořeny. V celém systému pracujeme s id hodnotou tohoto záznamu oproti jeho textové podobě na serveru, kterou bychom mohli také považovat za jedinečnou. Souborový systém operačních systému totiž nedovoluje vytvářet složky stejného jména v jednom adresáři (totéž platí pro soubory). Test může obsahovat obrázky, videa nebo další sady obrázků nebo videí.

- **Sets** – představuje jednotlivé sady obrázků nebo videí, které patří do jednotlivých testů. Je velice podobná entitní množině tests.

- **User_tests** – uchovává záznamy o tom, jaký test byl přidělen kterému uživateli. Dále si ukládáme časový údaj, kdy byl tento záznam vytvořen.

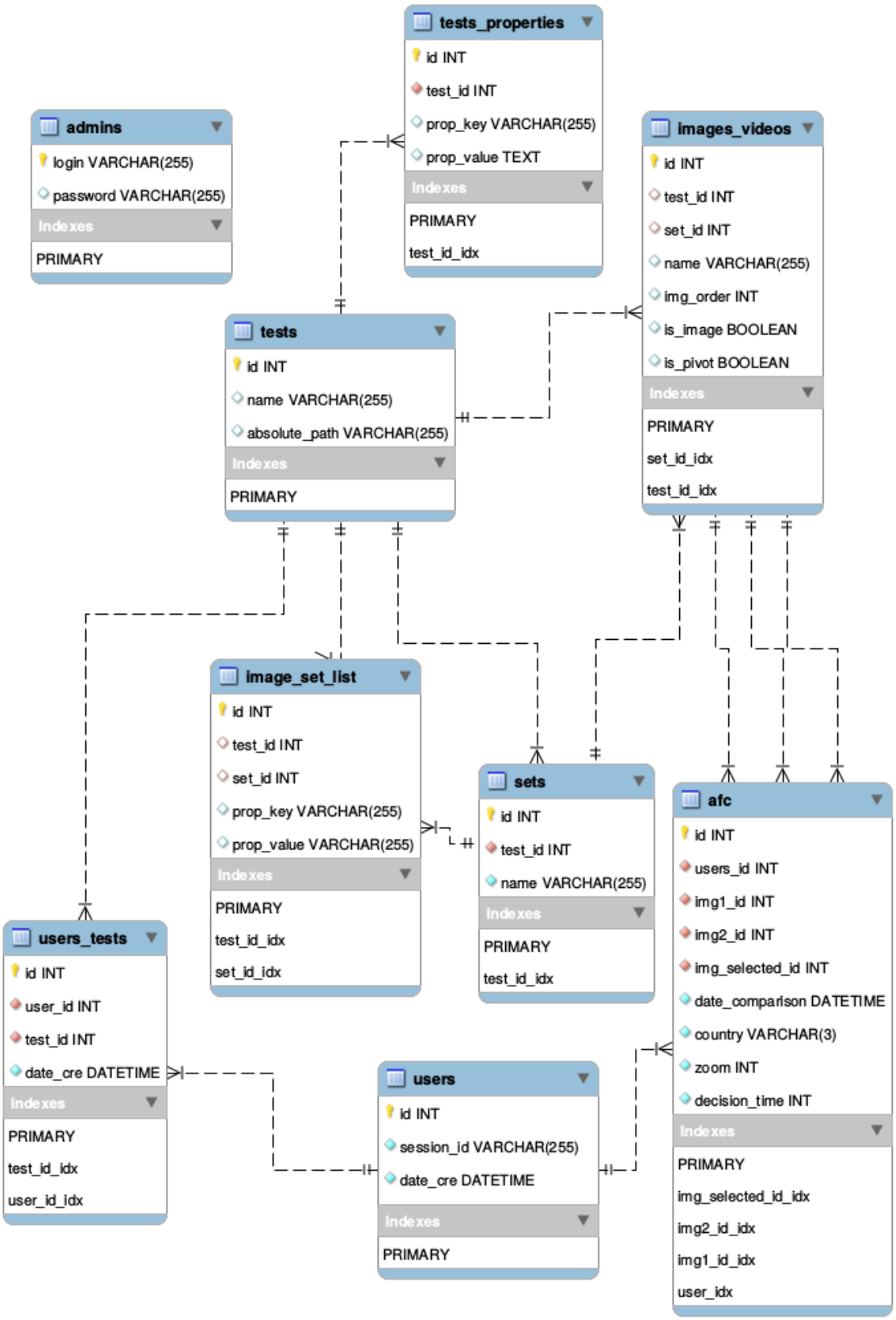
- **Tests_properties** – každý test má určité vlastnosti a na základě těchto vlastností přizpůsobuje svoje chování. Musíme si tedy ukládat informace, o jaký test se jedná. Vlastnost testu je řešena pomocí metody klíč hodnota, která nám zaručí určitou svobodu v přidávání nebo ubírání dalších vlastností.

- **Images_videos** – seznam obrázků nebo videí, které patří do určitého testu nebo sady. Jelikož obrázky mohou být rovnou v testu a nemusí být v sadě, může zde atribut set_id nabývat hodnoty NULL. Každý obrázek má definované pořadí, které může být měněno a díky atributu is_image poznáme, jestli se jedná o video nebo obrázek. Jednotlivé sady mohou obsahovat referenční obrázek. To je vždy právě jeden obrázek ze sady a v této entitní množině se značí atributem is_pivot.

- **Image_set_list** – definuje vlastnosti o sadách obrázků nebo videí, které mají být přidělovány uživatelům v rámci testu. Jako u množiny tests_properties, i zde jsou vlastnosti jednotlivých sad ukládány pomocí typu klíč hodnota, abychom docílili méně svazujícího způsobu případného budoucího přidávání vlastností.

- **Afc** – uchovává informace, které obrázky se uživateli zobrazili a který z nich vybral. Dále uchovává informace o tom, ve které zemi bylo ohodnocení pořízeno (pomocí http hlavičky zaslané prohlížečem) a jestli uživatel si přibližoval obrázek nebo ne.

- **Admins** – tato entitní množina se vztahuje k administrátorskému režimu internetové aplikace a proto nemá žádné vazby s předešlými entitními množinami. Zde je jednoznačným identifikátorem atribut login.



Obrázek 3.2: ER diagram

3.4 Databáze

Pro naši aplikaci využijeme služeb relační databáze MySQL, která má flexibilní licenční možnosti a je dostupná pod licencemi GNU General Public License (GPO) a komerční licencí. MySQL je velice výkonná relační databáze (možnost využití cachování dotazů) a platformově flexibilní. Například, mezi největší prominentní uživatele MySQL patří Yahoo!, které zpracovává miliardy zobrazování stránek měsíčně [6].

Struktura naší databáze odpovídá ER digramu (obr. 3.2), který můžeme vidět výše. Každá entitní množina představuje v databázi jednu tabulku a její atributy, sloupce tabulky. MySQL nám nabízí různé druhy úložiště a typy tabulek, z nichž využijeme typ InnoDB. Hlavními výhodami InnoDB je transakční zpracování (zamykání řádku nebo tabulek), referenční integrita (podpora cizích klíčů) a hodí se pro ukládání velkého počtu dat [5]. V naší práci hojně využíváme referenční integrity. Ta nám zajišťuje nemožnost vložení záznamu, jestliže neexistuje cizí klíč a jestliže by došlo k smazání cizího klíče v nadřazené tabulce, InnoDB nám dovoluje využít funkce ON DELETE CASCADE, která smaže všechny záznamy v podřadné tabulce.

MySQL disponuje velkou řadou datových typů sloupců tabulek. Můžeme do ní ukládat celá čísla, desetinná čísla, text nebo časové údaje. V naší práci jsme využili číselných typů hlavně jako primární klíč tabulek, který je v rámci tabulky jedinečný, nesmí nabývat hodnot NULL a automaticky se zvětšuje pomocí klauzule AUTO_INCREMENT. Je zde ovšem jeden číselný typ, který je svým způsobem zajímavý, a tím je BOOLEAN. MySQL interně žádný takový typ nemá a pokud ho použijeme, sloupec bude typu TINYINT(1), což je nejmenší možný číselný typ. Takového typu například využijeme u sloupce, kde potřebujeme zaznamenat, jestli hodnota je nebo není obrázkem.

Mezi textové typy byl nejčastěji použit typ VARCHAR, u kterého můžeme volit jeho velikost mezi 0-255 bajtů. K této hodnotě musíme připočítat ještě jeden bajt, který slouží pro zaznamenání délky. V naší aplikaci nastaly případy, kdy nám 255 bajtů nestačilo, a proto jsme museli využít typu TEXT. Tento typ má rozsah 0-65535 bajtů, což už je více než dostačující a jako u typu VARCHAR, musíme k této hodnotě připočítat 2 bajty pro zaznamenání délky. Jediná nevýhoda toho typu je, že nerozlišuje velikost písmem. Jako časový typ jsme použili DATETIME. Tento typ obsahuje celé datum i s časem, což je pro nás více než dostačující.

Data v tabulce jsou uložena neuspořádaně a pokud by tabulka obsahovala velké množství dat, stává se vyhledávání záznamů velice náročné. Proto je důležité dobře stanovit sloupce, podle kterých se budou vyhledávat záznamy a na tyto aplikovat databázové indexy. MySQL na základě toho vytvoří k tabulce další indexovou tabulku, kde budou odkazy na data do původní tabulky. Nevyplatí se ovšem indexy dávat na každý sloupec, protože tím bychom duplikovali data a k žádnému zrychlení by nedošlo. Příklad indexů v naší aplikaci můžeme vidět u každé tabulky v sekci indexes na obrázků 3.2.

3.5 Prohlížeče a mutlimédia

Výrobci webových prohlížečů se nedohodli na jednotném formátu pro přehrávání videa. S příchodem specifikace HTML5 byl doporučen jako standardní formát pro video Ogg Theora. Výrobci webových prohlížečů i tak zastávali různé názory na podporování tohoto formátu.

Firmy Apple a Nokia se například rozhodli nepodporovat kodeky Ogg a přijaly jako svůj standardní videoformát formát H.264. Tento formát má velkou hardwarovou i softwarovou podporu zejména v mobilních zařízeních, ale je zatížený patentem, a proto bylo nutné v

případě jeho používání platit poplatky. Ogg nemělo takovou podporu jako H.264. Společnosti Opera a Mozilla nechtěli tyto poplatky platit a proto H.264 vůbec nepodporovaly. Společnost Google začala podporovat jak H.264 tak i formáty Ogg v prohlížeči Chrome. Chromium ale H.264 nepodporuje. Microsoft byl dlouho neutrální, ale nakonec se rozhodl pro H.264. V roce 2009 vymizel ze specifikace HTML5 požadavek na kodek Ogg, stejně výrobci prohlížečů podporovali kodeky, jaké chtěli.

V současné době společnosti Mozilla a Opera podporují formát H.264, ale pouze v případě podpory hardwaru nebo softwaru třetích stran (např. operační systém Android).

Vznikl i nový formát WebM, který je sponzorovaný společností Google. Opera od verze 10.60 a Firefox od verze 4.0 tento formát také podporují. Prohlížeč Internet Explorer jej podporuje jen v případě, pokud je nainstalovaný software třetích stran. Safari (prohlížeč společnosti Apple) stále podporuje jen formát H.264 [2].

Abychom zajistili správné přehrávání videosouborů v dnešních prohlížečích musíme mít videa ve dvou formátech – H.264 a WebM. Musí však být splněna i podmínka dostačující verze dnešních prohlížečů, protože HTML5 podporuje mutlimédia v prohlížečích Chrome od verze 4.0, Internet Explorer od verze 9.0, Firefox od verze 3.5, Safari od verze 4.0 a Opera od verze 10.5.

Tohle omezení však pro nás není limitující, protože výrobci automaticky nabízí zdarma aktualizace svých prohlížečů. Z obrázku 3.3 můžeme vidět zastoupení prohlížečů k roku 2015 podle W3Schools [7]. Pokud bychom se podívali, jak na tom jsou starší verze jednotlivých prohlížečů (ty, které jsou výše zmíněné), zjistíme, že se jedná o minimální procenta a v dnešní době, kdy nové verze vychází častěji a častěji není třeba se razantně zaobírat staršími verzemi prohlížečů.

2015	Chrome	IE	Firefox	Safari	Opera
March	63.7 %	7.7 %	22.1 %	3.9 %	1.5 %
February	62.5 %	8.0 %	22.9 %	3.9 %	1.5 %
January	61.9 %	7.8 %	23.4 %	3.8 %	1.6 %

Obrázek 3.3: Zastoupení webových prohlížečů 2015 [7]

3.6 Konvertování obrázků a videí

V dnešní době je lehké uskutečnit základní převod formátů u multimédií a existuje k tomu celá řada volně dostupných nástrojů. Mezi takové patří například nástroj FFmpeg nebo program Miro Video Converter [2].

FFmpeg je nástroj pro příkazový řádek, který je více flexibilnější než zmíněný program Miro Video Converter. Tento nástroj je primárně určen pro operační systém Linux a disponuje otevřeným zdrojovým kódem. Existují ovšem i jeho spustitelné verze pro Mac OS X a Windows, které si můžeme stáhnout na jejich stránkách. FFmpeg podporuje celou řadu formátů a co je nejdůležitější, hlavně ty, které potřebujeme pro naši webovou aplikaci.

Miro Video Converter je program, který poskytuje grafické uživatelské rozhraní pro již zmíněný nástroj FFmpeg. Jeho hlavní výhodou je snadnější ovládání.

Jelikož budeme videa ukládat na server, měli bychom mít na paměti i velikost diskového prostoru, který videa v různých formátech zaplní. Nejlepším řešením je experimentovat a vybrat si střední cestu mezi výslednou kvalitou a velikostí souboru.

Pro převod obrázků je vhodné použít program ImageMagick. Tento program je volně dostupný pro operační systémy Linux, Windows a Mac Os X a disponuje širokou škálou možností pro převod obrázků. V operačním systému Linux existuje možnost spouštění tohoto programu pomocí příkazové řádky příkazem convert s příslušnými parametry.

Kapitola 4

Implementace

Tato kapitola se zabývá reálnou implementací aplikace. Jsou zde popsány použité technologie, konfigurační soubory párových testů, jejich význam a obsah, adresářová struktura a sekce administrátorského rozhraní aplikace.

4.1 Použité technologie

Jelikož se jedná o internetovou aplikaci, byly hlavně použity technologie PHP,CSS,HTML a JavaScript.

PHP (Hypertext Preprocessor) je skriptovací programovací jazyk. Pomocí něj můžeme programovat dynamické internetové stránky nebo webové aplikace ve formátech HTML, XHTML a další. Skripty jsou prováděny na straně serveru a k uživateli se dostává jen výsledná internetová stránka. Díky své jednoduchosti se stal nejrozšířenějším skriptovacím jazykem pro dynamické internetové projekty. Podporuje mnoho knihoven a nejčastěji je používán v kombinace s operačním systémem Linux, webovým serverem Apache a databázovým systémem (nejčastěji MySQL nebo PostrgeSQL). Nejvhodnější je pro menší a střední projekty, ale i tak se dá použít pro větší weby, kde nám může být příkladem otevřená encyklopedie Wikipedie, která je v tomto jazyce napsána.

HTML (HyperText Markup Language) je název pro textový značkovací jazyk pro tvorbu webových stránek s využitím hypertextových odkazů. Definují strukturu a obsah internetových stránek.

CSS (Cascading Style Sheets) v našem jazyce nazývané jako kaskádové styly. Tyto styly se používají pro popis jednotlivých elementů HTML dokumentu. Můžeme pomocí nich upravovat vzhled nebo i pozici jednotlivých elementů. Slouží převážně pro oddělení vzhledu dokumentu od jeho struktury a obsahu.

JavaScript je interpretovaný, objektově orientovaný programovací jazyk, který se vkládá přímo do internetových stránek. JavaScript se tedy vykonává na straně klienta a je nezbytnou technologií pro tvorbu dynamických webů. Pomocí něj můžeme spravovat jednotlivé elementy HTML dokumentu a měnit i jeho obsah.

jQuery je javascriptová knihovna, která nám ulehčuje interakci JavaScriptu s elementy HTML dokumentů a je podporovaná širokou škálou prohlížečů. Je to svobodný a otevřený software, který je šířen pod licencí MIT.

AJAX (Asynchronous JavaScript and XML) se používá pro vývoj interaktivních webových aplikací, kde není potřeba měnit obsah stránek pomocí jejího kompletního znovunačtení.

jqPlot je JavaScriptový framework, pomocí kterého jsme schopni vytvářet grafy na webových stránkách.

XML (Extensible Markup Language) je značkovací jazyk a je považován za standardní formát pro výměnu informací. Byl standardizován a vyvinut konsorciem W3C.

Aplikace byla vyvíjena ve vývojovém prostředí PhpStorm 8.0.3 na operačním systému Linux Mint 17. Před nasazení byla aplikace vyvíjena na technologiích PHP 5.3.3, Apache2 a MySQL 5.5.41.

4.2 Adresářová struktura

Adresářová struktura je v naší aplikaci velice striktní. Jelikož se jedná o internetovou aplikaci, musíme myslet na její zabezpečení. Každý adresář na serveru má svou funkci a leží v adresáři 2AFC. Uživatelé mají přístup jen do tohohle adresáře (je vhodné povolit jedno místo a vše ostatní zakázat). Tím si zajistíme, že nám nikdo nebude přistupovat pomocí prohlížeče k výsledkům, logovacím souborům, obrázkům nebo konfiguračním souborům. Velkou roli hrají také systémová práva k adresářům nebo souborům, které jsou uloženy na serveru.

- **Admin** – obsahuje skripty pro administrátorský režim.
- **Config** – v této složce jsou konfigurační soubory typu .ini a soubor s přihlašovacími údaji do databáze.
- **Css** – soubory s kaskádovými styly definující vzhled stránek.
- **Include** – tato složka obsahuje soubory s třídami a kontrolující skripty. Tvoří jádro systému.
- **Js** – jsou v ní uloženy JavaScriptové soubory.
- **Scripts** – obsahuje skripty, které provádí vždy jednu funkci například mazání testů, ukládání dotazníků apod.
- **Tests** – složka, která obsahuje všechny testy, které jsou v aplikaci k dispozici. Jednotlivé testy mohou obsahovat obrázky nebo videa a sady obrázků nebo videí, ale také soubory, pomocí kterých můžeme tyto testy konfigurovat. Sady již žádné další složky neobsahují a kdyby ano, jsou aplikací ignorovány.
- **Translations** – obsahuje xml soubory s českými a anglickými překlady pro řídící prvky testů.

4.3 Párový test

Ukázku párového testu můžeme vidět na obrázku 2.1. Uživateli je poslán URL odkaz na stránku, která provádí obsluhu testů. Jedná se o skript index.php, který je uložen v adresáři 2AFC, do kterého mají přístup uživatelé, kteří provádějí test. Tento skript je pro všechny testy stejný. Volbu testu provedeme pomocí GET parametru.

Všechny testy jsou uloženy v adresáři tests a na základě GET parametru je v tomto adresáři zvolen cílový test (složka s daty).

Jak ale tyto obrázky zobrazit v prohlížeči, když do složky tests, nemá nikdo zvenčí přístup? Na základě tohoto problému byl vytvořen skript, kterému se předá název testu, případné sady, název obrázku a test vypíše obsah obrázku. Tento skript je volán v atributu src, který vlastní HTML tag pro obrázek. Funkcionálně je to to stejné, jako kdybychom do atributu src vložili cestu k obrázku, ale díky skriptu jsme schopni určité kontroly vstupních dat a tedy i zamezit případné snahy někoho zvenčí zobrazovat jiná data, než bychom chtěli.

Jednotlivá data o testu jsou ukládány do superglobálního pole $_SESSION (dále jen session). Toto pole je asociativní, tedy umožňuje ukládat data pod nějakým textovým klíčem. Název klíče představuje jméno testu, který je uživateli poslán. Pod tímto klíčem se v session ukrývá další asociativní pole, které již obsahuje konkrétní data pro daný test.

Uživateli se tedy nejdříve zobrazí stránka s instrukcemi a tlačítkem "pokračovat". Po kliknutí uživatele na tlačítko jsou zobrazovány obrázky nebo videa vedle sebe, případně ještě s referenčním obrázkem nebo videem. Uživatel vybírá obrázky na základě předem stanovených kritérií pomocí kliknutí myši na jeden z nich nebo provádí výběr obrázku pomocí kurzorových kláves (šipka vlevo, šipka vpravo). Při kliknutí na obrázek se do databáze uloží jaké obrázky byly zobrazeny, jaký zvolil, jaké přiblížení nebo oddálení v prohlížeči měl, z jaké byl země (pomocí HTTP hlavičky) a jak dlouho se pro daný obrázek rozhodoval. Po uložení ohodnocení je odebrán daný pár obrázků ze session. Konec testu nastane tehdy, kdy nemáme již žádný pár obrázků pro ohodnocení v session.

Na konci testu se uživateli zobrazí dotazník (pokud je nadefinován). Počet položek o které se bude jednat, získáme z konfigurace testu, která je uložena v databázi.

Vzhled dotazníku je velice jednoduchý. Jednotlivé položky dotazníku jsou situovány na střed prohlížeče každá další položka je zobrazena pod tou předchozí. Položka se skládá z otázky, která je v hlavičce a možností, které jsou pod touto otázkou. Výsledky dotazníků do databáze již neukládáme, ale ukládáme je složky Questionnaires, kterou obsahuje každý test.

4.4 Parametrizace testů

Každý test (složka v adresáři tests na serveru) musí obsahovat konfigurační soubor pomocí kterého jsme schopni měnit parametry testu. Je textového formátu a musí být v kódování UTF-8. Struktura konfiguračního souboru je velice striktní a obsahuje povinné i nepovinné parametry. Aplikace přečte celý soubor a zpracuje dané parametry, všechny ostatní řádky jsou ignorovány. Pokud nastane chyba během čtení souboru, aplikace na to správce testu upozorní formou zápisu chyby do logovacího souboru pro chyby. Parametry jsou dvojího typu a to bud párové (začínají a končí určitou sekvencí znaků) nebo nepárové, které jsou stylu klíč hodnota. Existují zde i parametry, které jsou určeny pro různé jazyky a mění se tedy jen obsah sdělení, né funkčností. Příklad konfiguračního souboru pro test můžeme vidět na obrázku 4.1.

4.4.1 Promíchání sad

Tato funkčnost je implementována v konfiguračním souboru pomocí nepárového parametru randomizeSets. Jak už bylo zmíněno, uživateli se zobrazí všechny sady v rámci testu. Pokud je tento parametr nastaven na hodnotu true, jsou pak jednotlivé páry obrázku sad promíchány mezi sebou. V opačném případě jsou uživateli zobrazovány sekvenčně.

4.4.2 Sekce instrukcí a outro

Obsah této sekce je zobrazován uživateli na začátku testu. Existují dva typy instrukční sady, jedna anglická a druhá česká. Z pohledu programu se rozlišují pouze začínající a končící sekvencí znaků. Obě dvě mají stejný obsahový limit 65535 bajtů, kde maximum bylo stanoveno na základě typu atributu TEXT v databázi 3.4. Anglická instrukční sekce začíná řádkem se sekvencí znaků *instuctions:* a končí řádkem s řetězcem *endInstructions*. Pro českou instrukční sekci začíná sekce sekvencí *instuctionsCS:* a končí *endInstructionsCS*.

Obsah sekce outro je zobrazován uživateli po konci testu spolu s dotazníkem. Jako u výše uvedené sekce instrukcí, i tato obsahuje anglickou a českou verzi. Limit znaků mají stejný jako sekce instrukcí. Anglická sekce začíná řádkem se sekvencí znaků *outro:* a končí *endOutro*. Česká sekce začíná sekvencí *outroCS:* a končí *endOutroCS*.

4.4.3 Aktivní sady obrázků pro testy

V této sekci uvádíme, které sady obrázků jsou aktivní pro daný test (budou přidělovány uživatelům v testu). V adresáři daného testu můžeme mít několik sad obrázků, ale pokud je neuvedeme v této sekci, aplikace je bude při přidělování uživatelům ignorovat. Tato sekce začíná řádkem se sekvencí znaků *imageSetList:* a končí řádkem *endImageSetList*. Každá definovaná sada v této sekci je oddělena novým prázdným řádkem a obsahuje 4 parametry, které musí pro každou sadu být uvedeny a jsou to:

- **setName** – parametr je použit při pojmenování souboru s výsledky dané sady.
- **setDirectory** – obsahuje relativní cestu k sadě obrázků. Adresář musí existovat. Pokud bude tento parametr prázdný, jako sada obrázku se bere obsah adresáře testu.
- **setPreferencesFile** – hodnota tohoto parametru obsahuje jméno konfiguračního souboru, který je aplikován na danou sadu. Tento soubor je typu .ini a je uložen v adresáři config. Toto omezení je aplikováno z důvodu jasně dané striktní orientace souborů na serveru, aby uživatel nemohl zadávat adresy do jiných adresářů.
- **setRandomizeImages** – tento parametr nabývá hodnot true nebo false a udává, jestli obrázky dané sady budou uživateli zobrazovány náhodně nebo popořadě.

4.4.4 Sekce dotazník

Tato sekce znázorňuje dotazník, který se zobrazuje uživateli na konci testu. Aplikace rozeznává dvě formy dotazníků a to českou a anglickou verzi. Anglická verze je ohraničená řádkem se sekvencí znaků *questionnaire:* a končí řádkem *endQuestionnaire*. Česká verze je ohraničena řádkem *questionnaire:* a končí *endQuestionnaireCS*.

Jednotlivé otázky dotazníku jsou odděleny prázdným řádkem a každá otázka je znázorněna sérií parametrů, které si následně představíme:

- **type** – udává typ odpovědi na danou otázku dotazníku. Může být buď typu text, textarea, radio nebo checkbox. Typy jsou buď textové (text, textarea) nebo s možností výběru (radio, checkbox). Typ text má rozsah jednoho řádku a typ textarea má rozsah víceřádkový. U typů s možnostmi výběru se přidávají parametry, kde název parametru je jméno daného typu zakončený číslem pořadí možnosti. Maximum možností není dáno a číslování musí začínat číslovkou 0.
- **label** – obsahuje text otázky dané položky dotazníku.

```
#Promíchání sad
randomizeSets=true

#Instrukce pro uživatele
instructions:
Instrukce pro uživatele v angličtině.
endInstructions

instructionsCS:
Instrukce pro uživatele v češtině.
endInstructionsCS

outro:
Anglický text, který se zobrazí uživateli s dotazníkem.
endOutro

outroCS:
Text, který se zobrazí uživateli s dotazníkem.
endOutroCS

testHeadlineForUser:
Text v angličtině, který se zobrazuje během testu.
endTestHeadlineForUser

testHeadlineForUserCS:
Text v češtině, který se zobrazuje během testu.
endTestHeadlineForUserCS

#Seznam sad obrázků
imageSetList:
Obsahuje sady, které se budou uživateli zobrazovat.
endImageSetList

#Dotazník
questionnaire:
Obsahuje jednotlivé otázky dotazníku v angličtině.
endQuestionnaire

questionnaireCS:
Obsahuje jednotlivé otázky dotazníku v češtině.
endQuestionnaireCS
```

Obrázek 4.1: Ukázka konfiguračního souboru configTest.txt

4.5 Další nastavení

Mezi další nastavení patří soubor s daty pro úspěšné připojení databáze k aplikaci, určování pořadí obrázků v rámci sady a jestli sada používá referenčního obrázku.

4.5.1 Pořadí obrázků v sadě

Správce testů má možnost určovat pořadí obrázků v jednotlivých sadách pomocí souboru _customorder. Pokud tedy chceme řadit obrázky podle vlastního uvážení, musí tento soubor existovat v dané sadě. Pokud neexistuje, řadí se obrázky podle toho, jak jsou uspořádány v sadě (složce). Obsah tohoto souboru je striktní a na každém řádku obsahuje název obrázku dané sady. Pokud tento soubor neobsahuje názvy všech obrázků v sadě, jsou neuvedené obrázky řazeny za uvedené obrázky v souboru. Bílé znaky na začátku a konci řádku jsou ignorovány, musí se ovšem zachovat syntax, co řádek, to obrázek.

4.5.2 Sady s referencí

Sady s referenčním obrázek obsahují soubor s názvem _pivot. Obsahem tohoto souboru je název referenčního obrázku, který musí být obsažen v dané sadě, ale nezařazuje se mezi obrázky pro ohodnocování uživatelem. Všechny bílé znaky jsou v tomto souboru ignorovány. Tento obrázek bývá uprostřed mezi hodnocenými obrázky a slouží k rozhodování, který z ostatních dvou obrázků je pro nás, vzhledem k referenčnímu, přirozenější.

4.5.3 Připojení databáze

Aby administrátor měl možnost se přihlásit k databázi pod různými jmény a nemusel tyto údaje měnit v nějakém skriptu, byl vytvořen soubor s daty pro úspěšné připojení aplikace k databázi s názvem databaseConnection.txt. Tento soubor se nalézá ve složce config a musí existovat, jinak aplikace skončí s chybou. Syntaxe souboru je velice jednoduchá a obsahuje 4 řádky s parametry *hostName* (jméno serveru), *userName* (jméno uživatele, který se připojuje k databázi), *password* (heslo uživatele pro připojení) a *database* (jméno připojované databáze). Tyto parametry zapisujeme v podobě klíč=hodnota.

4.5.4 Ranker.ini

Pomocí tohoto souboru může mít každá sada v rámci testu jinou konfiguraci. Jedná se o vzhledové vlastnosti jako například barva pozadí testu.

4.6 Konvertování obrázků

Tato funkcionalita nemohla být implementována, protože většina webových hostingů zakazuje spouštění shellu na svých serverech. Není divu, protože shell je velice mocný nástroj a mohl by poskytovatelům hostingu přinášet spíše problémy. Proto byly vytvořeny 2 shellové skripty, pomocí kterých můžeme konvertování obrázků nebo videí uskutečnit, pokud již nevlastníme nějaký sobě známý program pro konvertování. Tyto skripty jsou dodatečné a nejsou na serveru s aplikací uloženy.

Skript pro konvertování obrázků se nazývá imageConverter.sh a pro videa videoConverter.sh. Jsou to spustitelné soubory, které spouštíme pomocí příkazové řádky. Funkcionalita

skriptů je velice podobná, ale každý skript pracuje s různým počtem parametrů. Parametry skriptů je možné psát dvěma způsoby, buď před název skriptu nebo za název skriptu.

Pokud voláme program imageConverter.sh s parametry za názvem programu, je přesně dána posloupnost a význam jednotlivých parametrů. Jsou 3, kde první parametr obsahuje relativní cestu ke vstupnímu adresáři, který bude celý přečten a dojde ke konvertování obrázků. Druhý parametr obsahuje na jaký typ obrázků chceme konvertovat a poslední třetí parametr obsahuje relativní cestu k výstupnímu adresáři, tedy kam budou konvertované obrázky ukládány.

Při konvertování obrázků musíme brát zřetel na to, aby typ obrázku byl zobrazitelný v daných prohlížečích, protože né každý prohlížeč podporuje všechny formáty obrázků a tedy je nezobrazuje. Je doporučeno, aby obrázky byly konvertovány na typ png, jpg nebo jpeg.

Musíme se také ujistit, je-li nainstalován program ImageMagick, který je použit v našich převodních programech. Pro správnou funkčnost programu se musíme ujistit, že vstupní adresář obsahuje jenom obrázky, protože program IMageMagick podporuje širokou škálu formátů pro převod na obrázky.

Možnost volání programu s parametry před názvem souboru nám poskytuje určitou svobodu v určování pořadí parametrů a ve stanovování vstupních a výstupních adresářů. Parametr input obsahuje cestu ke vstupnímu adresáři, output je pro výstupní adresář a parametr type značí typ obrázků, na který budou konvertovány. U této verze volání programu můžeme některé parametry vynechat. Při vynechání vstupního adresáře program automaticky prochází adresář, ve kterém se nachází. Při vynechání typu program automaticky konvertuje obrázky na typ png a při vynechání výstupního adresáře jsou nové obrázky ukládány do složky, kde se skript nachází.

Volání programu videoConverter.sh je velice podobné výše uvedenému skriptu. Voláme jej s parametry za nebo před názvem skriptu. Tento skript přijímá pouze 2 parametry. První parametr určuje relativní cestu ke vstupnímu adresáři a druhý parametr relativní cestu k výstupnímu adresáři. Program nalezená videa konvertuje do dvou formátů — webm a mp4. Pomocí těchto videoformátů jsme schopni video zobrazit na jakémkoli prohlížeči, kde jediným omezením je jejich verze (viz 3.5).

4.7 Logování chyb

V aplikaci byla implementována možnost ukládání vzniklých chyb do souborů na serveru při testech. Je to hlavně z toho důvodu, že chyby můžou představovat špatně nadefinovanou cestu k adresáři se sadou obrázků, různé syntaktické chyby v konfiguračních souborech testů apod. a správce testů nemá žádnou možnost jak tyto chyby odhalit před spuštěním testu. Jestliže nastane nějaká chyba během testu nebo před testem, test se nezobrazí a do příslušného logovacího souboru se přidá chyba. V těchto souborech je na každém řádku zapsána právě jedna chyba, která vznikla od zahájení používání aplikace. Formát je přehledný, na začátku řádku je datum a čas vytvoření chyby, následně jméno testu, ve kterém vznikla a pak samotná chyba. Odchytávání chyb v aplikaci je implementováno přes vyjímky, text samotné chyby může být kombinací konkrétní hlášky v češtině a případné chyby, kterou nám vrací PHP.

Logovací soubory jsou uloženy v adresáři logs (kvůli jasně dané adresářové struktuře na serveru). Vytváří se pouze soubory s názvem error.log nebo admin-error.log. Do těchto souborů aplikace pouze zapisuje a je tedy na administrátorovi, jak bude s těmito soubory pracovat (mazání jejich obsahu, nahlížení do souborů).

4.8 Administrátorské rozhraní

Do této doby mohl administrátor (správce testů) spravovat testy jen pomocí služeb (ftp klient, terminál...), které mu ovšem nemusí poskytovat komfort, které by potřeboval. Proto bylo implementováno administrátorské rozhraní, které poskytuje dostatečně širokou škálu služeb pro správu aplikace. Můžeme přidávat obrázky nebo videa, měnit obsahy konfiguračních souborů, získávat výsledky ohodnocení obrázků v různých formách. Jestliže ovšem chceme přistoupit do této sekce, musíme se přihlásit. Jména a hesla administrátorů jsou uložena v databázi a pokud bychom chtěli správce přidat, musíme se přihlásit k databázi a přidat administrátora ručně pomocí SQL dotazu.

Administrátorské rozhraní je webová stránka s jednoduchým a intuitivním uživatelským rozhraním. Byl kladen důraz na jednoznačnost a jednoduchost. Stránka je rozdělena na řídící (menu) a obsahovou část (místo pro data). Aby se zaručila správná funkcionalita administrátorského rozhraní, je zapotřebí využívat moderních prohlížečů, které podporují Ajax technologii, základní CSS3 a uživatel musí mít zapnutý javascript. Nemusíme se bát, že to jsou nějaké extrémní nároky, uvedené technologie v dnešní době vlastní drtivá většina prohlížečů.

4.8.1 Zobrazování výsledků

Tato sekce slouží jako spojení administrátora s výsledky testů. Výsledky jsou ukládány do adresáře Results, který se nachází v každém testu a pokud ne, je aplikací automaticky vytvořen. Výsledky je možné na server ukládat ve formě csv dokumentů. Tyto dokumenty obsahují názvy obrázků a představují matici výsledků.

Pro každého uživatele, který dokončil test, je v adresáři Results daného testu vytvořena složka, ve které je tolik csv souborů, kolik je sad v testu (pro každou sadu jeden csv soubor v rámci uživatele). Tento csv soubor je maticí, ve kterém diagonála (stejné obrázky) nabývá hodnot nula. Pokud byl vybrán některý obrázek před jiným, je hodnota na daném řádku a sloupci rovna jedné. Tyto hodnoty se čtou zleva, tj. obrázek na řádku byl vybrán před obrázkem ve sloupci.

Aplikace také umožňuje zobrazovat výsledky ve formě grafů. Administrátor si může zobrazit graf konkrétní sady testu nebo grafy všech sad testu zároveň. Pomocí těchto grafů byly vytvořeny obrázky, které jsou uvedeny v kapitole 5.

4.8.2 Správa testů

V této sekci administrátorského rozhraní má administrátor možnost spravovat testy a sady obrázků.

Nejdříve se mu zobrazí všechny testy (složky) na serveru. Existují zde jen dvě formy zanořování. Jinými slovy, máme na výběr buď editovat daný test (zobrazit si jeho obsah a ten měnit) nebo editovat sadu obrázků v rámci testu (měnit její obsah). Sady, které jsou zanořeny v dalších sadách jsou aplikací ignorovány.

Při editaci testů máme možnost tyto testy přidávat nebo mazat. Funkčnost u editace obsahu testů je daleko rozsáhlejší (obr. 4.2). Zde se již nezobrazují jen adresáře (v tomto případě sady), ale také obrázky, konfigurační soubory a videa. Byla implementována možnost vytvoření a mazání nové sady, přidávání a mazání obrázků nebo videí v testu, vytvoření nebo editace konfiguračních souborů.

Při editaci sad obrázků existuje možnost přidávání nebo mazání obrázků či videí a vytvoření konfiguračního souboru pro řazení pořadí obrázků. Pokud daná sada obsahuje

nějaký konfigurační soubor (_customorder, _pivot nebo configTest.txt), je zobrazeno na stránce tlačítko pro jejich editaci.

Editace probíhá následně – administrátor označí jeden soubor, který chce editovat, po stisknutí tlačítka se zobrazí textový blok, ve kterém je obsah zvoleného souboru. Po změně obsahu je třeba tyto změny uložit. Editace konfiguračních souborů je prováděna pomocí technologie Ajax a v javascriptu jsou implementovány kontroly, aby nedošlo k nežádoucímu zacházení s touto funkcionalitou. Při mazání jakéhokoli souboru nebo složky je administrátor upozorněn, jestli chce opravdu tyto položky smazat. V každém testu jsou složky Questionnaires a Results. Questionnaires obsahují dotazníky, které vyplnili jednotliví uživatele na konci testu. Tyto dotazníky jsou ukládány ve formě xml dokumentu a administrátor si může tyto dotazníky stáhnout pomocí kliknutí na jejich název. Výsledky jsou ukládány ve složce Results, která obsahuje další adresáře, pro každého uživatele jeden. V něm jsou uloženy výsledky jednotlivých sad ve formě csv dokumentů.

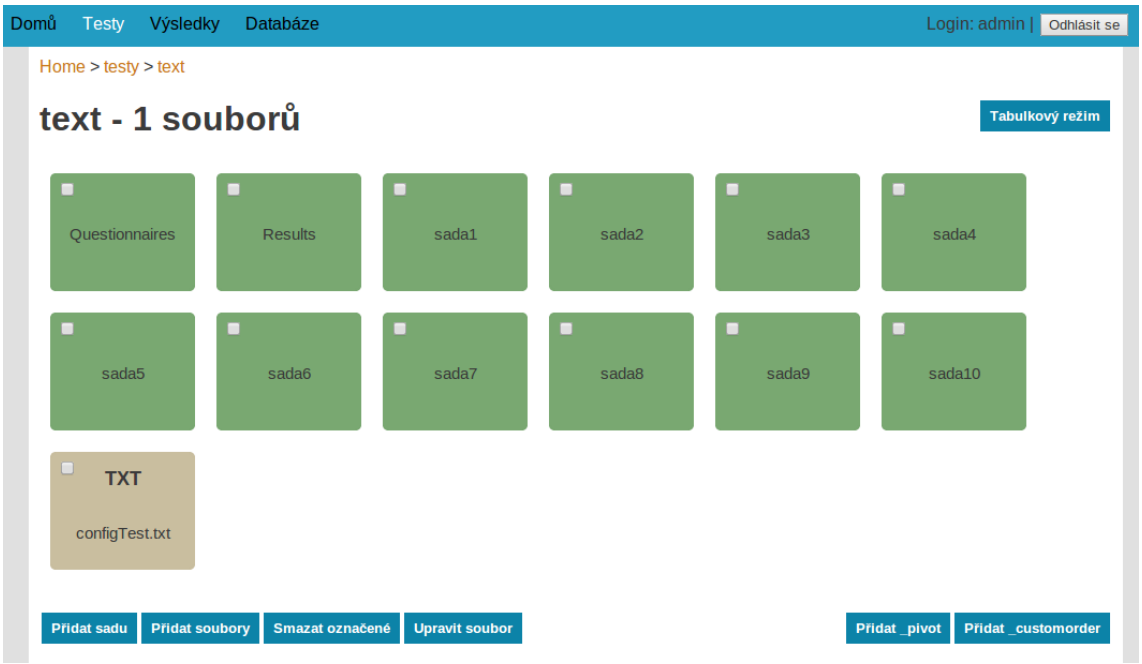
Administrátor má na výběr mezi dvěma módy zobrazování dat v této sekci — tabulkový nebo normální režim. Normální režim zobrazuje data v jednoduché formě. Každý soubor nebo adresář je zobrazen jako obdélník, který obsahuje název položky na disku. V případě souboru se zobrazuje v hlavičce obdélníku jeho přípona. Tento režim byl inspirován zobrazování souborů v operačním systému Linux Mint, kde máme zaškrtnutou možnost zobrazení s ikonami a byl implementován kvůli spotřebě místa na stránce. Tabulkový režim je detailnější, ale zabírá více prostoru na stránce. U testů zobrazuje počet položek v testu (adresáři) a poslední změnu v testu. U testu nám tento režim zobrazuje příponu souboru (pokud je to sada, vypisuje, že se jedná o adresář), velikost souboru a datum poslední změny.

4.8.3 Aktualizace databáze

Abychom dosáhli rychlé a spolehlivé aplikace, je dobré mít všechna data na jednom místě. Z naší adresářové struktury systému 4.2 můžeme vidět, že testy, jejich data a různé parametry testů jsou tvořeny soubory, ale výsledky ohodnocení si ukládáme do databáze. Bylo by značně neefektivní procházet adresáře na serveru, na základě toho hledat data v databázi a proto byla vytvořena funkce pro sesynchronizování dat v databázi s daty na serveru. Tato funkcionalita je velice důležitá, protože administrátoři mohou vytvářet testy a přidávat do nich obrázky i přes ftp klienta a databáze se o těchto změnách nemá jak dozvědět. K synchronizaci dat dojde vždy při přihlášení nebo odhlášení administrátora ze systému nebo při stisknutí tlačítka Uložit změny.

Všechny testy pracují s daty z databáze, a proto je nanejvýš důležité zachovat její konzistenci se serverem. K její synchronizaci dochází ze dvou hledisek — databáze a serveru. Z pohledu serveru se prochází celý adresář tests, a pokud v něm došlo k nějaké změně (vytvoření nového testu, změna konfiguračních souborů, přidání nebo smazání obrázků), je tato změna promítnuta do databáze.

Vkládání nových záznamů se provádí pomocí databázového příkazu insert. Tento příkaz velice zaměstnává databázový server a pokud jich je zpracováno programem více za sebou, je to časově náročné. Proto bylo potřeba tento proces urychlit. Toho bylo dosaženo pomocí víceřádkového vkládání [5]. Víceřádkové vkládání je v podstatě jeden příkaz insert, který obsahuje více vkládaných záznamů stejného formátů. Díky tomuto příkazu je možné zkrátit čas vkládání až třicetkrát. Při synchronizaci z pohledu databáze dochází pouze ke smazání neexistujících záznamů vůči serveru.



Obrázek 4.2: Ukázka administrátorského rozhraní pro tvorbu testů

Kapitola 5

Ukázkový experiment

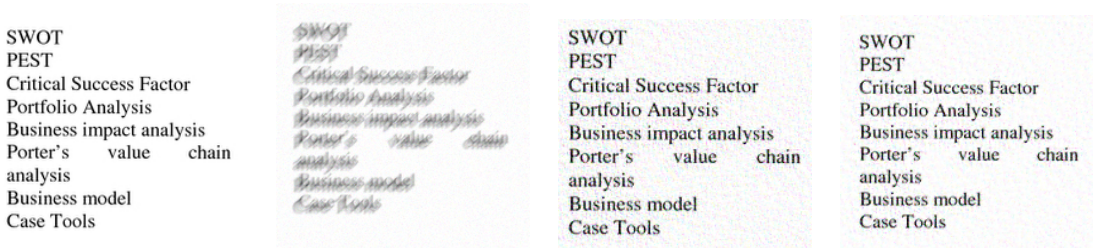
Pomocí námi navržené aplikace byl zprostředkován experiment, ve kterém se uživatelům zobrazovaly páry obrázků bez reference. Cílem experimentu bylo zjistit, jak které metody zpětně zaostřují rozmazaný text v obrázku, aby byl čitelný.

5.1 Experiment

Před začátkem experimentu byl pořízen originální a rozmazaný obrázek, které můžeme vidět na obrázku 5.1. Z rozmazaného obrázku bylo pomocí pěti metod vytvořeno dalších pět obrázků (ukázka dvou převedených obrázků 5.1). Každá sada tedy obsahovala sedm obrázků (pořízené obrázky s převedenými obrázky).

Úkol uživatele byl, zvolit takový obrázek, který se mu zdál čitelnější. Test dohromady obsahoval 210 párů obrázků, které byly zobrazeny každému uživateli. Experimentu se zúčastnilo 32 uživatelů, kde 3 uživatelé neudělali celý test a jejich výsledky byly smazány (pro lepší a přesnější analýzu výsledků experimentu). Průměrná délka testu činila 15,8 minut a průměrná rozhodovací doba uživatele 4,5 vteřin. Je důležité poznamenat, že zde byly uživatelé, kteří měli průměrnou rozhodovací dobu 0,9 vteřin nebo až 28,4 vteřiny.

Na konci testu byl uživatelům zobrazen dotazník, který vyplnilo 23 mužů a 6 žen. U mužů činil průměrný věk 23,6 let a u žen 24,6 let. U mužů mělo 12 oslovených oční vadu, která většinou byla krátkozrakost. 6 oslovených označilo svou zkušenost se zpracováním obrazu jako uživatelskou znalost.

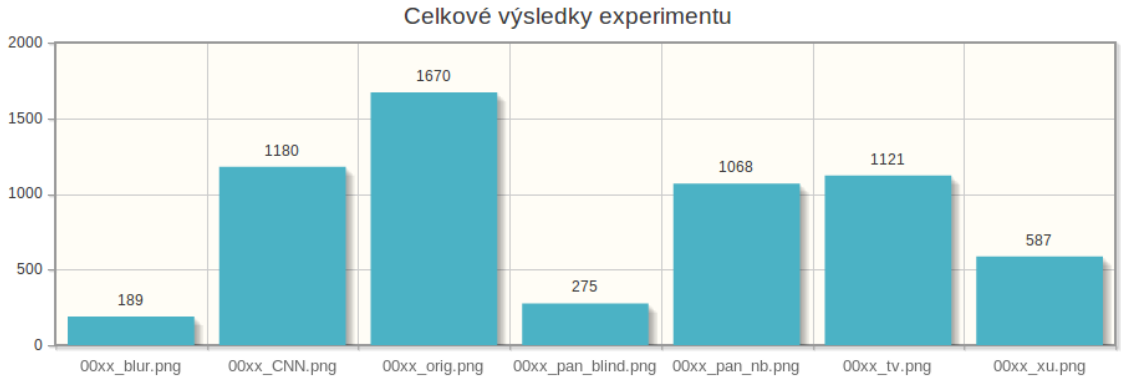


(a) Originální (b) Rozmazaný (c) Metoda tv (d) Metoda pan_nb

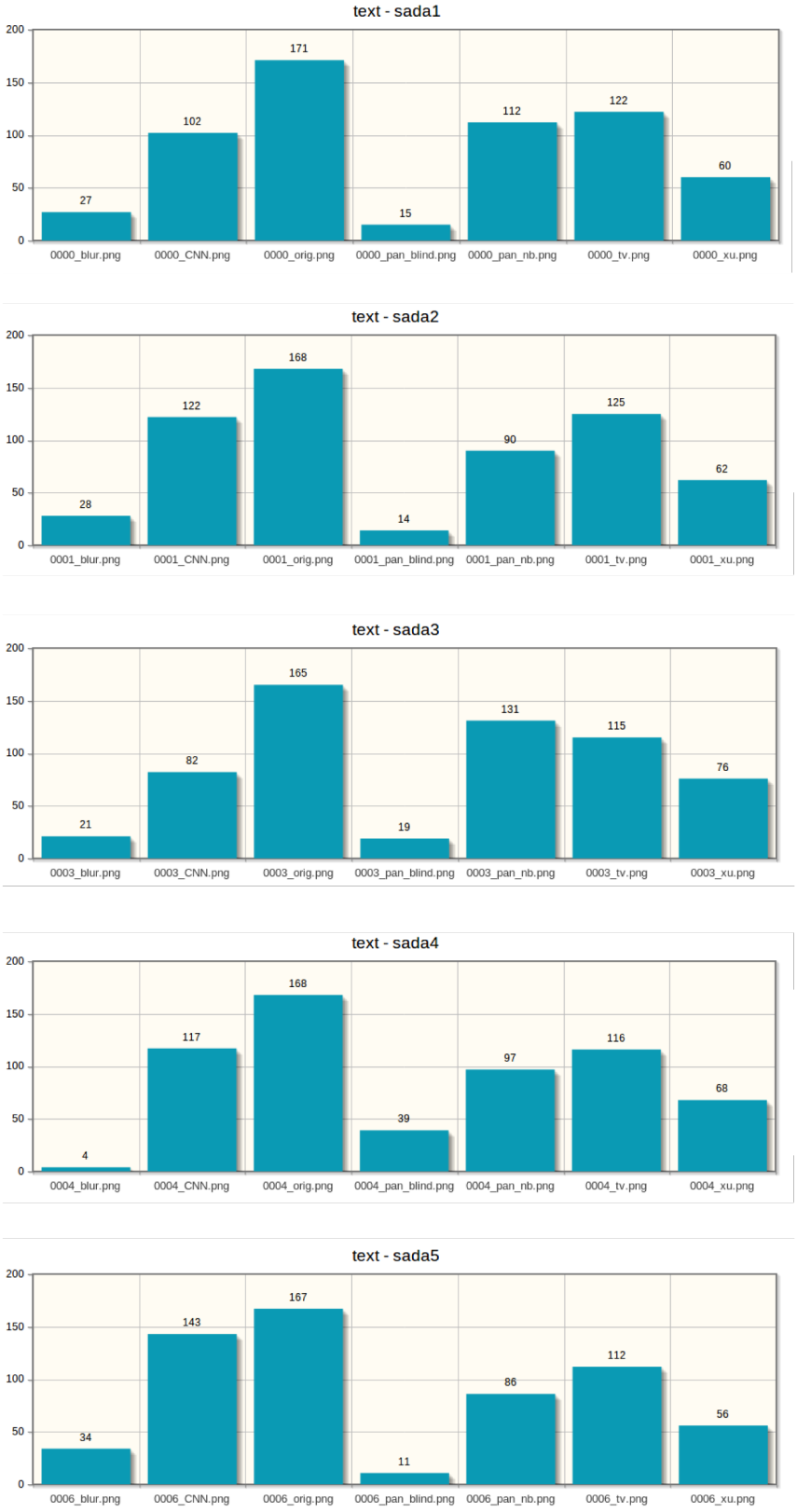
Obrázek 5.1: Ukázka použitých obrázků v experimentu.

5.2 Vyhodnocení výsledků

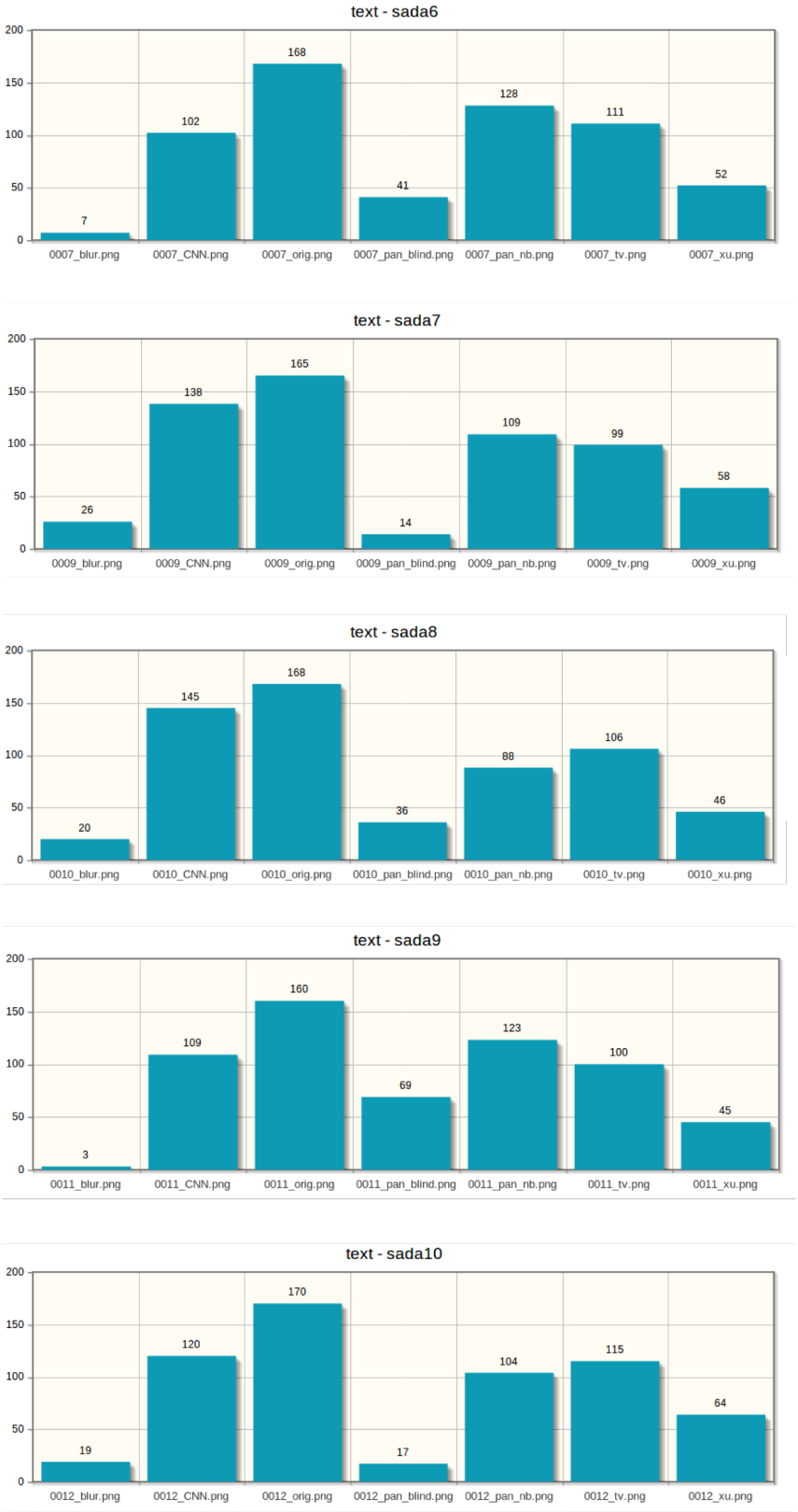
Z obrázku 5.2 můžeme vidět, jak si jednotlivé metody u uživatelů vedly. Každý sloupec představuje součet ohodnocení obrázků určité metody napříč testem. Když pomineme základní obrázky (orig a blur), uvidíme, že nejlépe si v našem experimentu vedly metody CNN a tv. Hodnocení obrázků jednotlivých sad můžeme vidět na obrázcích 5.3 a 5.4. Tyto grafy byly pořízeny administrátorským rozhraním naší aplikace.



Obrázek 5.2: Celkové ohodnocení obrázků experimentu



Obrázek 5.3: Hodnocení sad 1 - 5 testu



Obrázek 5.4: Hodnocení sad 6 - 10 testu

Kapitola 6

Závěr

Cílem této práce bylo vytvořit webový nástroj pro párový test obrázků. Uživatelům byly zaslány testy v podobě URL odkazu na určitou stránku webové aplikace, na které se jim zobrazovaly obrázky nebo videa k ohodnocení.

Bylo potřeba nastudovat odpovídající literaturu o formátech multimediálních dat, které jednotlivé prohlížeče podporují. Dále bylo třeba nastudovat, jak správně navrhnout internetovou aplikaci, aby její uživatelské rozhraní bylo přehledné a intuitivní. Dále bylo třeba promyslet a navrhnout datovou strukturu, do které budou ukládány informace o konfiguraci testů a výsledcích vyhodnocení testů jednotlivými uživateli. Rozhodující bylo rovněž zvážit, pomocí jakých metod je možné získat odpovědi na podněty, které jsou uživatelům při experimentech prezentovány.

Tato aplikace při testech využila psychofyzikální metody 2AFC. Základem této metody je párový test, při kterém jsou uživateli zobrazovány páry obrázků nebo videí a uživatel volí jednu z těchto položek na základě předem stanovených kritérií. K volbě položky využívá myši nebo kurzorové klávesy (šipka vlevo, šipka vpravo). Tato metoda může být rozšířena o referenční obrázek, přičemž dva prezentované obrázky uživatel vůči tomuto referenčnímu obrázku porovná 2.2.

Pro parametrizaci testů se využívá několika konfiguračních souborů, pomocí kterých je možné definovat instrukce pro uživatele, aktivní sady testu, dotazník, pořadí souborů v sadách, referenční soubor sady apod.

Aby bylo možné s aplikací uspokojivě pracovat, bylo vytvořeno administrátorské rozhraní, které umožňuje zpracovávat získané výsledky, přidávat nové testy, sady, modifikovat konfigurační soubory a aktualizovat databázi. Při tvorbě aplikace byl rovněž brán zřetel na kompatibilitu s existujícím programem Ranker 2.5.1.

Aplikace byla navržena tak, aby poskytovala datové výstupy pro další statistiky. Mezi případná rozšíření aplikace lze zařadit další sekce v konfiguračních souborech pro větší parametrizaci testů nebo různé typy testů.

Literatura

[1] Arlow, J.; Neustadt, I.: *UML a unifikovaný proces vývoje aplikací.* Computer Press, 2003, ISBN 80-7226-947-X.

[2] Brown, T. B.; Butter, K.; Panda, S.: *HTML5 okamžitě.* Computer Press, 2014, ISBN 978-80-251-4296-7, 69-76 s.

[3] Ehrenstein, W. H.; Ehrenstein, A.: Psychophysical Methods. [Online], [cit. 2015-05-19]. URL http://www.uni-leipzig.de/~isp/isp/history/texts/PSYPHY-M.PDF

[4] Pelli, D. G.; Farrel, B.: Psychophysical methods. 1995, [Online], [cit. 2015-05-19]. URL http://www.psych.nyu.edu/pelli/pubs/pelli1995methods.pdf

[5] Schneider, R. D.: *MySQL – Oficiální průvodce tvorbou, správou a laděním databází.* Pearson Education, 2005, ISBN 80-247-1516-3.

[6] Schneider, R. D.: *Velká kniha PHP 5 a MySQL.* Zoner Press, 2005, ISBN 80-86815-20-X.

[7] W3Schools: Browser Statistics. 2015, [Online], [cit. 2015-05-19]. URL http://www.w3schools.com/browsers/browsers_stats.asp

[8] Zendulka, J.: Konceptuální modelování a návrh databáze. [Online], [cit. 2015-05-19]. URL http://www.fit.vutbr.cz/study/courses/DSI/public/pdf/nove/2_2.pdf