Vizualizace výsledků vyšetření sluchu

Visualization of results of hearing tests

Jiří Ančinec

Bakalářská práce
2010

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2009/2010

ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Jiří ANČINEC**
Osobní číslo:	**A06135**
Studijní program:	**B 3902 Inženýrská informatika**
Studijní obor:	**Informační a řídicí technologie**
Téma práce:	**Vizualizace výsledků vyšetření sluchu**

Zásady pro vypracování:

1. **Seznamte se s druhy vyšetření v ORL lékařství, způsoby jejich provedení, vizualizace a archivace.**
2. **Zpracujte návrh systému pro vizualizaci výsledků vybraných vyšetření na webu.**
3. **Realizujte softwarové řešení zajišťující vhodnou implementaci navrženého systému.**
4. **Ověřte řešení na rozsáhlejší databázi vyšetření a definujte možnosti jeho praktického nasazení.**

Rozsah bakalářské práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **Bingham, B. Hawke, M. Atlas of clinical otolaryngology. Mosby-Year Book; illustrated edition edition, 1991. ISBN 978-1556643156.**
2. **Lejska, M. & kolektiv autorů. Základy praktické audiologie a audiometrie. Idvpz. Brno, 1994. ISBN 80-7013-178-0.**
3. **Uchytil, B., Smilek, P., Kostřica, R. , Novotný, M. Vyšetřovací metody a základní diagnostika v otorinolaryngologii. 1. vyd. Praha : Triton, Edice Levou zadní, 2002. ISBN 80-7254-190-0.**
4. **Kosek, J. PHP a XML. 1. vyd. : Grada publishing, a.s., 2009. ISBN 978-80-247-1116-4.**
5. **Nette Foundation. Dokumentace [online]. 2010 [cit. 2010-02-03]. Dostupný z URL: <http://nettephp.com/cs/dokumentace>.**
6. **Achour, M. & kolektiv autorů. PHP Manual [online]. 2010 [cit. 2010-02-04]. Dostupný z URL: <http://www.php.net/manual/en/>.**

Vedoucí bakalářské práce: **Ing. Viliam Dolinay**
Ústav automatizace a řídicí techniky

Datum zadání bakalářské práce: **5. března 2010**

Termín odevzdání bakalářské práce: **1. června 2010**

Ve Zlíně dne 5. března 2010

L.S.

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky

prof. Ing. Vladimír Vašek, CSc.
děkan

doc. Ing. Ivan Zelinka, Ph.D.
ředitel ústavu

ABSTRAKT

Název práce je „Vizualizace výsledků vyšetření sluchu“. Cílem této bakalářské práce bylo vytvoření webové aplikace, pomocí které může lékař zveřejnit ostatním lékařům výsledky ORL vyšetření, bez nutnosti aby tito lékaři museli vlastnit speciální software. Aplikace je napsána ve skriptovacím jazyce PHP 5, používá databázi MySQL 5.1 a testována na softwarovém serveru Apache. Zobrazit ji lze v jakémkoli prohlížeči podporující standart XHTML 1.0 Strict. V teoretické části jsou představeny jednotlivé technologie, které byly použity k vytvoření aplikace. V praktické části je uveden návrh aplikace a detailní seznámení s vytvořenou aplikací.

Klíčová slova: Vyšetření sluchu, webová aplikace, php, mysql, xml, Nette Framework

ABSTRACT

The title of this thesis is „Visualization of results of hearing tests „. The main aim of this thesis was the creation of the web application, which allows the doctor to disclose the results of hearing tests to other doctors without need to having some special software in their computer. The application is written in scripting language PHP 5, it uses a MySQL 5.1 database and is tested on an Apache server. Application can be viewed in any browser that supports standard XHTML 1.0 Strict. The theoretical part presents various technologies that were used to create this applications. The practical part includes design applications and detailed introduction to this applications.

Keywords: Hearing tests, web application, php, mysql, xml, Nette Framework.

Poděkování

Rád bych poděkoval vedoucímu bakalářské práce Ing. Viliamu Dolinayovi za jeho odborné rady a připomínky týkající se problematiky zpracování dat a fungování programu Fowler. Také bych rád poděkoval Miroslavu Zajíčkovi za pomoc s vytvořením grafického návrhu.

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně …….…….

podpis diplomanta

OBSAH

ÚVOD

Internetové stránky již dávno nejsou pouze statickými dokumenty, které se mohou pomocí celosvětové sítě prohlížet prakticky bez možnosti jakékoli interakce, pouze s použitím odkazů. Dnešní webové technologie přímo vybízí k tomu, aby se na internetu objevovaly webové aplikace, dynamické stránky, které reagují na chování uživatelů a které mohou uživatelům zobrazit jen ty informace, které si přejí vidět. Mezi takové aplikace patří nejrůznější vyhledávací portály, eshopy, chaty, seznamky, internetová rádia a televize. Cílem této bakalářské práce je navrhnout a realizovat webovou aplikaci, která bude sloužit lékařům pro zobrazování výsledků vyšetření ORL. Tento systém byl navržen tak, aby byl uživatelsky co nejvíce přívětivý, jednoduchý na používání. Od nasazení aplikace do provozu se o ni tedy bude moci starat pouze administrátor, který nemusí být žádný odborník na webové technologie a jehož hlavním úkolem bude správa uživatelů a aktualizace dat pacientů a vyšetření pomocí importu XML souborů.

S rozvojem možností dynamických webů se bohužel rozvíjí i internetová kriminalita. Nejrůznější útočníci se snaží získat pomocí rozličných postupů osobní data uživatelů. Využívají k tomu bezpečnostní díry v internetových aplikacích, programovacích jazycích, ale i v samotných softwarových serverech a databázových strojích. Proto se klade velký důraz na zabezpečení aplikací, které pracují s těmito citlivými daty. Dvojnásob to ale platí u aplikací, které obsahují informace o zdraví pacientů, mezi které patří i aplikace, která je výsledkem této bakalářské práce. Samotný programátor má možnost ovlivnit pouze zabezpečení své aplikace a typ jazyka, ve kterém tuto aplikaci napíše. Protože byl tento projekt vytvořen v jazyce PHP, byl vybrán jako pomůcka pro programování NETTE Framework. NETTE totiž klade důraz na dva důležité směry, a to jsou bezpečnost a efektivita. Aplikace je vytvářena a testována na softwarovém serveru Apache s podporou SSL a databází MySQL5.1.

I. TEORETICKÁ ČÁST

1 AUDIOLOGICKÁ VYŠETŘENÍ

Jednou ze základních charakteristik audiometrie je skutečnost, že musí k vyšetření používat speciální přístroje. Současný rozvoj audiologie je dán kromě poznatků ve fyziologii sluchového vnímání především novými moderními technickými možnostmi. [5]

1.1 Audiometr

Audiometr je přístroj na vyšetření sluchu, který je schopen produkovat různé akustické signály.

Podle specifikace měřeného signálu lze rozlišovat:

- Tónový audiometr – pomocí kterého se stanovuje sluchový práh pro přesně definované tóny
- Audiometr slovní – pro stanovení srozumitelnosti slov, vět a jiných složek řeči.
- Audiometr pro objektivní audiometrii – nevyžaduje spolupráci pacienta – tympanometr, audiometr pro vyšetření evokovaných potenciálů. Používá se k vyšetření sluchu u speciálních stavů – malé děti, simulanti apod.

Audiometry jsou elektronické přístroje, které se skládají z několika vzájemně propojených součástí. Hlavní součástí přístroje je tónový generátor. Tóny je možno frekvenčně volit: 250, 500, 1000, 2000, 3000, 4000, 6000, 8000 Hz. Doporučeny jsou také kmitočty 125, 750 a 1500 Hz. Kmitočty pro kostní vedení zvuku jsou 250, 500, 1000, 2000, 3000, 4000 Hz. [5]

1.2 Audiogram

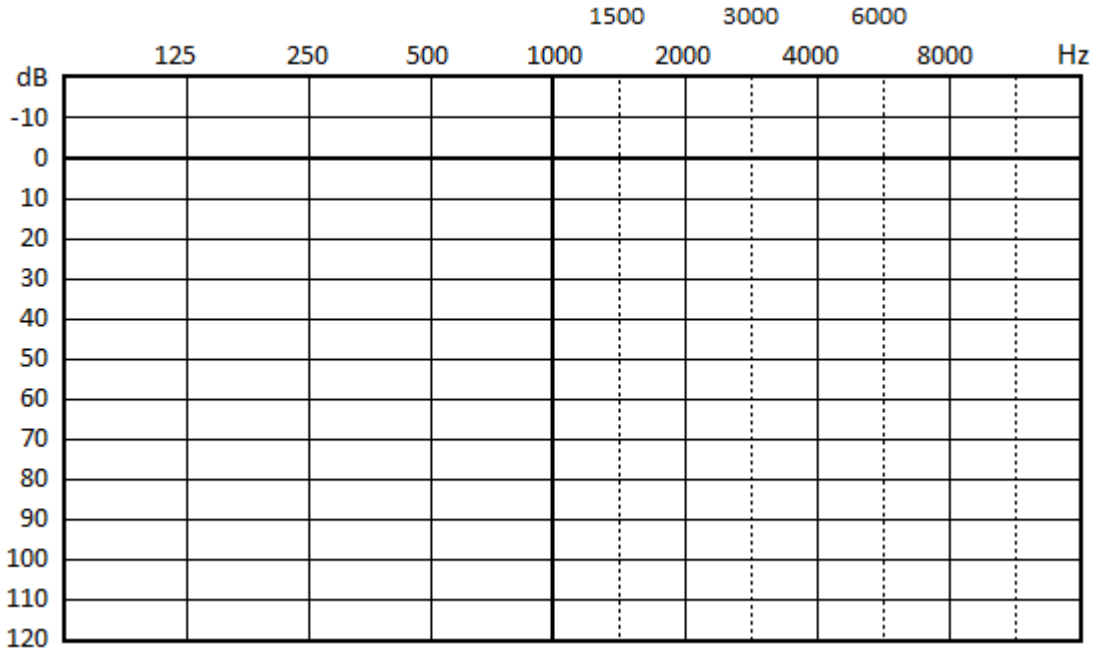
Audiogram je formulář s grafickým označením sluchového prahu vzdušného a kostního vedení pravého i levého ucha.

Podle typu zobrazení a způsobu měření známe audiogramy:

- **Absolutní** – na audiogramu jsou zapsány hodnoty prahu sluchu pro jednotlivé frekvence v absolutních hodnotách akustického tlaku.
- **Relativní** - vzniká tak, že na okrajových frekvencích je prahový akustický tlak zesilován právě o tolik, aby normální prahové hodnoty v celém vyšetřovaném poli odpovídaly přímce

- **Ztrátový** – je každý audiogram, který vyjadřuje sluchovou ztrátu ať absolutně, nebo relativně

Osnova audiogramu je tvořena předtištěnou sítí vodorovných a svislých čar – úseček. Úsečky vodorovné určují hladiny intenzity tónu v decibelech. Hladina 0 dB neznamená nulovou intenzitu, ale označuje takový akustický tlak, který odpovídá ideálnímu prahu sluchu. Úsečky pod touto úrovní, které jsou značeny 10 – 110 dB v deseti krocích, určují intenzitu zesílení nad ideálním sluchem. Označují nárůst intenzity, která je nutná, aby vyšetřovaný udal svůj individuální práh sluchu. Rozdíl mezi hladinou intenzity 0 dB a individuální prahovou hladinou vyjadřuje sluchovou ztrátu v decibelech. Svislé úsečky udávají frekvence vyšetřovaných tónů v hertzích (Hz). [5]



Obrázek 1. Formulář audiogramu

Formulář audiogramu je doplněn některými údaji [5]:

A. Administrativní – jméno pacienta, typ audiometru, datum, podpis vyšetřující osoby.

B. Výsledky vyšetření

1. Základní – zobrazení prahových hodnot čistých tónů pro jednotlivé frekvence pravého a levého ucha, zvlášť pro vzdušné a kostní vedení. Hodnoty pravého ucha značíme červenou a levého ucha barvou. Značení, které je uvedeno, je nejobvyklejší a nejrozšířenější.

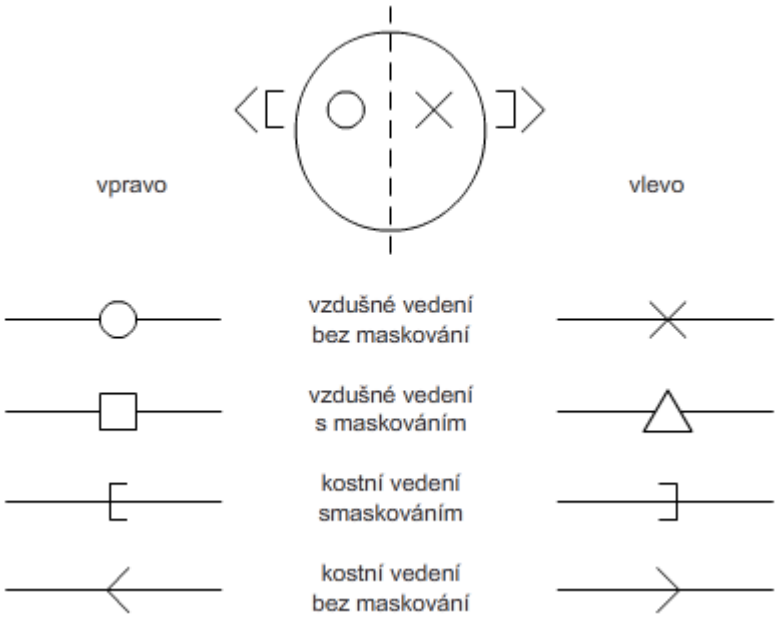
   a) Vzdušné vedení – jednotlivé frekvenční prahy sluchu se označují kroužkem pro pravé a křížkem pro levé ucho a tyto body vzájemně spojujeme plnou čárou.

   b) Kostní vedení – prahové body jsou graficky zobrazeny hranatými závorkami pro maskované nebo ostrými pro nemaskované ucho. Orientace závorek je opačná pro pravé a levé ucho. Jednotlivé body spojujeme přerušovanou čarou.

2. Rozvíjející – obsahují údaje, které rozvíjejí, případně doplňují výsledky vyšetření.

   a) Audiometrické zkoušky – výsledek je zapisován buď v grafice (např. šumová audiometrie, SAL test apod.), nebo jako symbol (SISI, Lü, KW). Je nutné vždy připojit vysvětlující legendu.

   b) Matematické výpočty – především určení velikosti ztrát sluchu v procentech.



Obrázek 2. Značení na audiogramu (převzato z [5])

2 XML

XML, neboli Extensible Markup Language je obecný značkovací jazyk, který byl vyvinut a standardizován konsorciem W3C. Je zjednodušenou podobou staršího jazyka SGML. Umožňuje snadné vytváření konkrétních značkovacích jazyků (tzv. aplikací) pro různé účely a různé typy dat.

Mezi hlavní výhody patří mezinárodní podpora. V XML dokumentu můžeme bez problémů přepínat mezi různými jazyky. Jako znaková sada se implicitně využívá ISO 10646 (Unicode). Současně je přípustné i libovolné jiné kódování (např. v ČR často používané windows-1250, iso-8859-2), musí však být v každém dokumentu přesně určeno. Další výhodou je vysoký informační obsah. Pomocí značek (tagů) můžeme v dokumentu vyznačit význam jednotlivých částí textu. Díky tomu dokumenty obsahují více informací, než kdyby se používalo značkování zaměřené pouze na vzhled.[12]

2.1 Využití XML

Na webu syntaxi XML využívají mnohé prezentační formáty, mezi které patří jazyk XHTML, vektorový grafický formát SVG, jazyky pro definici uživatelského rozhraní v moderních RIA (Rich Internet Application) prostředích, jako je XAML v Silverlightu a MXML ve Flashi.

XML se využívá k publikování metainformací. Jedná se například o formáty pro publikování přehledů nových článků, jako jsou RSS nebo Atom, pro informace o zboží v internetových obchodech pro zbozi.cz, nebo pro pomoc vyhledávacím robotům využitím Sitemap, kde se nachází informace o všech stránkách www aplikace.

Další využití XML je pro komunikaci a předávání dat. Využívá se pro výměnu dat mezi backendy jednotlivých aplikací, nebo v AJAXových aplikacích pro aktualizaci dat v prohlížeči. [13]

2.2 Syntaxe

XML je textový dokument většinou kódovaný v Unicode, v Česku obvykle jako UTF-8. Efektivita XML je závislá na struktuře, obsahu a integritě. Aby byl dokument považován za správně strukturovaný, musí mít následující vlastnosti [13]:

- Musí mít právě jeden kořenový (root) element

- Neprázdné elementy musí být ohraničeny startovací a ukončovací značkou. Prázdné elementy mohou být označeny tagem „prázdný element“.
- Všechny hodnoty atributů musí být uzavřeny v uvozovkách – jednoduchých (‘), nebo dvojitých (“), ale jednoduchá uvozovka musí být uzavřena jednoduchou a dvojitá dvojitou. Opačný pár uvozovek může být použit uvnitř hodnot.
- Elementy mohou být vnořeny, ale nemohou se překrývat. Každý element s výjimkou kořenového musí být tedy obsažen v jiném elementu.
- Jména elementů v XML rozlišují velká a malá písmena.

3 APACHE

Apache HTTP Server je softwarový webový server s otevřeným kódem pro většinu operačních systémů jako je GNU/Linux, BSD, Solaris, Mac OS X, MS Windows a další. V současné době je nejpoužívanějším webovým serverem na světě.

3.1 Historie

Vývoj Apache začal v roce 1993 v NCSA (National Center for Supercomputing Aplication) na Illinoiské univerzitě. V roce 1995 byla vydaná první veřejná verze s označením 0.6.2. Následovalo kompletní přepracování kódu a založení Apache Group, která je dnes základem vývojářského týmu. V roce 1996 byl Apache nejpopulárnějším serverem na internetu. V roce 1999 běžel na 57% všech serverů a v roce 2005 jeho používanost dosáhla 69%.[1]

3.2 XAMPP

XAMPP je softwarové řešení pro použití webového serveru na počítači bez nutnosti složité instalace Apache a dalších součástí, nutných pro běh webových aplikací, jako jsou PHP, MySQL apod.

3.2.1 Instalace

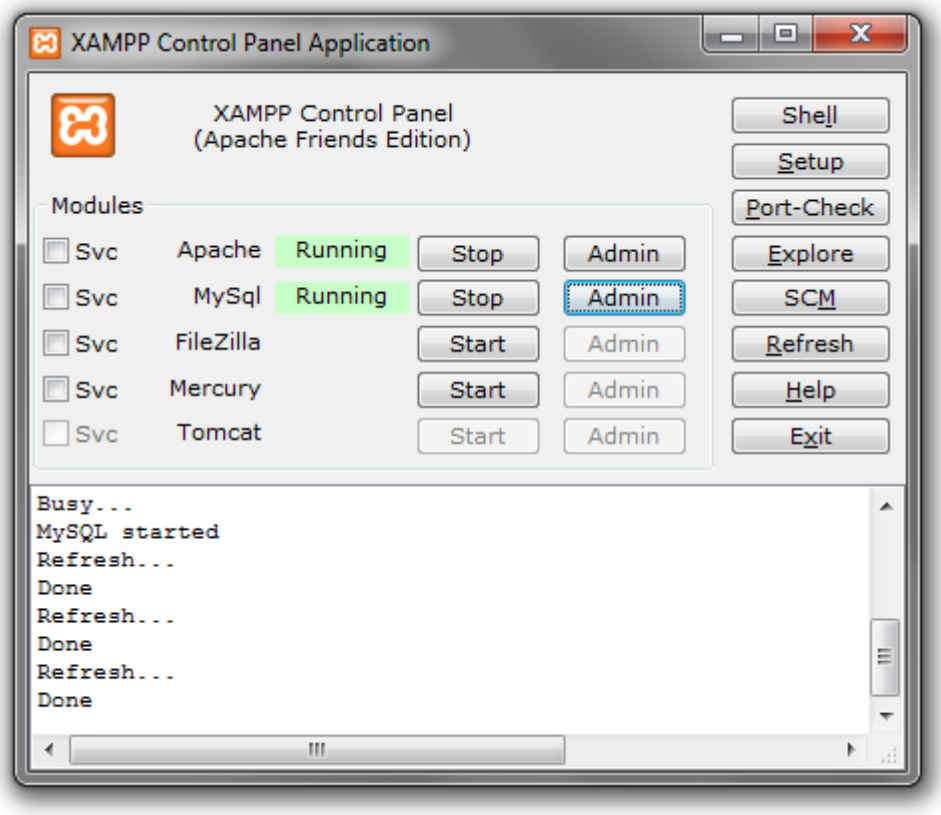
Instalace tohoto produktu je velmi jednoduchá, stačí rozbalit ZIP soubor, který lze stáhnout z webu http://www.apachefriends.org/en/xampp.html. Po rozbalení je potřeba XAMPP ještě nastavit. K tomu slouží v základním adresáři soubor setup_xampp.bat.

Po spuštění setupu je potřeba zodpovědět na několik dotazů, jako je, zda vytvořit zástupce na ploše apod. Skript také provede nastavení jednotlivých komponent na nové umístění v počítači. Je vhodné složku s XAMPPem umístit v počítači tak, aby se v adrese nevyskytovala diakritika. Toto totiž způsobuje problémy.

3.2.2 Spuštění

Po dokončení nastavení se XAMPP automaticky spustí. Případně lze XAMPP spustit pomocí souboru xampp-control.exe.

Po spuštění se zobrazí okno kontrolního panelu (Obrázek 3), ve kterém je možné spustit Apache a databázový server MySQL kliknutím na příslušné tlačítko „Start“. Že jsou tyto dvě aplikace spuštěny, indikuje nápis „Running“ v zeleném poli.



Obrázek 3. Kontrolní panel aplikace XAMPP

4 SSL

SSL je protokol, resp. Vrstva vložená mezi vrstvu transportní transportní (např. TCP/IP) a aplikační (např. HTTP), která poskytuje zabezpečení komunikace šifrováním a autentizaci komunikujících stran.

Ustanovení SSL spojení funguje na principu asymetrické šifry, kdy každá z komunikujících stran má dvojici šifrovacích klíčů – veřejný a soukromý. Veřejný klíč je možné zveřejnit, a pokud tímto klíčem kdokoliv zašifruje nějakou zprávu, je zajištěno, že ji bude moci rozšifrovat jen majitel použitého veřejného klíče svým soukromým klíčem.

Ustavení SSL spojení (SSL handshake, tedy „potřásání rukou“) pak probíhá následovně [11]:

- Klient pošle serveru požadavek na SSL spojení, spolu s různými doplňujícími informacemi (verze SSL, nastavení šifrování atd.).
- Server pošle klientovi odpověď na jeho požadavek, která obsahuje stejný typ informací a hlavně certifikát serveru.
- Podle přijatého certifikátu si klient ověří autentičnost serveru. Certifikát také obsahuje veřejný klíč serveru.
- Na základě dosud obdržených informací vygeneruje klient základ šifrovacího klíče, kterým se bude šifrovat následná komunikace. Ten zašifruje veřejným klíčem serveru a pošle mu ho.
- Server použije svůj soukromý klíč k rozšifrování základu šifrovacího klíče. Z tohoto základu vygenerují jak server, tak klient hlavní šifrovací klíč.
- Klient a server si navzájem potvrdí, že od teď bude jejich komunikace šifrovaná tímto klíčem. Fáze handshake tímto končí.
- Je ustaveno zabezpečené spojení šifrované vygenerovaným šifrovacím klíčem.
- Aplikace od teď dál komunikují přes šifrované spojení. Například POST požadavek na server se do této doby neodešle.

4.1 HTTPS

Protokol SSL se nejčastěji využívá pro bezpečnou komunikaci s internetovými servery pomocí HTTPS, což je zabezpečená verze protokolu HTTP. Po vytvoření SSL spojení (session) je komunikace mezi serverem a klientem šifrovaná a tedy zabezpečená.

Protokol HTTPS využívá asymetrické šifrování. Obě strany si před zahájením komunikace vygenerují pár klíčů (privátní a veřejný). Při zahájení komunikace si vymění veřejné klíče, které by obě strany měly ověřit pomocí jiného komunikačního kanálu. Ověření může proběhnout kontrolou výtahu (otisk, miniatura, hash) veřejného klíče u protistrany například pomocí telefonu nebo lze použít princip přenosu důvěry, kdy nám protistrana předá veřejný klíč, který je digitálně podepsaný (nejlépe certifikační autoritou, které důvěřujeme a jejíž veřejný klíč máme v důvěryhodném úložišti, např. THAWTE, VeriSign, RapidSSL, GeoTrust).

Zatímco samotné šifrování ochrání komunikaci před odposloucháváním, bez ověření autenticity veřejných klíčů jsou komunikující strany vystaveny riziku útoku.

Za certifikáty vydané certifikačními autoritami, které mají svůj veřejný klíč v úložišti, které je dodáváno s webovým prohlížečem, je nutné platit. Existuje však možnost vytvoření certifikátu, který si vydavatel sám sobě podepíše (anglicky self-signed certificate), avšak v takovém případě musí protistrana přidat do úložiště veřejný klíč sama (a ověřit ho jinak). [3]

5 PHP

PHP neboli hypertextový preprocesor, je skriptovací programovací jazyk, určený především pro programování dynamických internetových stránek. Nejčastěji se začleňuje přímo do struktury jazyka HTML, XHTML, WML či XML, což lze využít při tvorbě webových aplikací. PHP lze použít i k tvorbě konzolových a desktopových aplikací ale nejčastěji jsou PHP skripty prováděny na straně serveru, k uživateli je přenášen až výsledek jejich činnosti.

Výhody PHP:

- PHP je nezávislý na platformě, skripty fungují bez větších úprav na mnoha různých operačních systémech.
- Je možnost přidat velké množství rozšíření. Jako jsou například knihovny pro práci s grafikou (GD2), pro práci se zip soubory apod.
- Podpora přístupu k většině databázových systému (mj. MySQL, ODBC, Oracle, PostgreSQL, MSSQL)
- Podpora mnoha internetových protokolů (HTTP, SMTP, SNMP, FTP, IMAP, POP3, LDAP apod.)

Syntaxe jazyka je inspirována programovacími jazyky, jako je Perl, C, Pascal a Java. [10]

5.1 PHP5

Od roku 2008 je PHP5 jedinou stabilní verzí, která se vyvíjí. Jedním z hlavních přínosů verze PHP5 je větší podpora OOP (objektově orientované programování).

Pro práci v OOP nabízí PHP5:

- Konstruktory a destruktory
- Interface
- Abstraktní třídy
- Statické proměnné a metody
- Public, protected a private proměnné a metody
- Namespace (jmenné prostory)

5.2 Nette

Nette Framework je napsaný v PHP5 s plným využitím objektů (OOP). Jeho licence, která vychází z BSD, patří k těm nejvolnějším. Jednou z velkých výhod tohoto frameworku je velká a aktivní komunita českých PHP vývojářů. V případě nutnosti je tedy velmi snadné získat českou podporu přímo od vývojářů.

Nette je koncipováno jako otevřený framework, je ho tedy možné používat v primitivních webových aplikacích, složitějších webových aplikacích jako jsou eshopy, wiki, blogy, CMS. Lze jej i používat společně s jiným otevřeným frameworkem, jak je například Zend Framework.

5.2.1 Návrhové vzory MVC a MVP

Zkratka MVC znamená Model-View-Controller. Jedná se o rozdělení aplikace do tří vrstev, které mají na sebe pouze minimální vliv. Jedná se o:

- Datový model (Model) – zajišťuje přístup k datům a manipulaci s nimi
- Uživatelské rozhraní (View) – převádí data reprezentovaná modelem do podoby vhodné k prezentaci uživateli.
- Řídící logiku (Controller) – reaguje na události pocházející od uživatele a zajišťuje změny v modelu nebo v pohledu.

MVP znamená Model-View-Presenter. Je to návrhový vzor, který lépe odpovídá logice Nette Frameworku. Zjednodušeně se dá říct, že presenter je v Nette to samé, co controller v jiných frameworcích [8].

V MVP má platit, že:

- Model nemá vědět o tom, že nějaké view a presentery existují.
- View o modelu vědět také nemusí (pasivní view), nebo naopak může data získávat přímo z něj, záleží na zvolené koncepci.
- Presenter seznámí view s modelem (ne naopak) a realizuje uživatelské akce.
    - Změna view
    - Změna stavu
    - Příkaz pro model

Ideální je, aby se

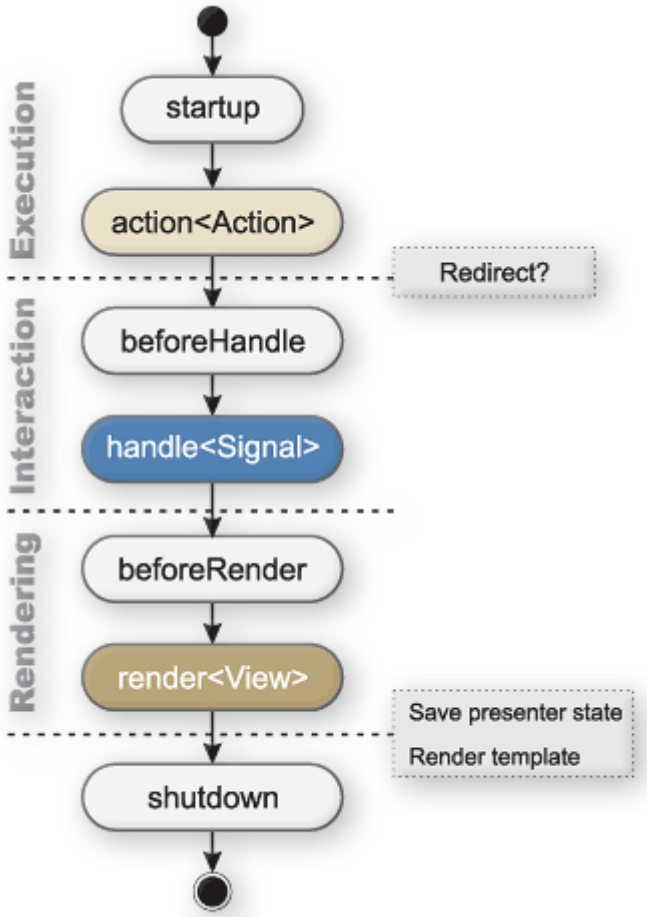
- činnosti modelů omezily jen na získávání data neboli práci s databází
- logika v šablonách omezila na iterace a podmínky
- logika v presenteru omezila na:
    - plnění šablon a registraci helperů a filtrů
    - sestavení stromu komponent
    - úkolování tříd z modelu

5.2.2 Presenter

Životní cyklus presenteru je rozdělen do několika částí představovaných voláním volitelně existujících metod. Jde o action{Action}, handle{Signal} a render{View}. Každá metoda se hodí na něco jiného. Ty, které mají společné znaky, se řadí do společných fází životního cyklu. [9]

Charakteristika fází:

1. výkonná (execution)
2. změny vnitřních stavů (interaction)
3. vykreslovací (rendering)
4. ukončení činnosti (shutdown)



Obrázek 4. Životní cyklus presenteru (Převzato z [9])

Obrázek 4 ilustruje, jak jsou postupně vykonávány metody presenteru v jeho životním cyklu a do jaké fáze se tyto metody začleňují. [9]

- bílé – metody společné pro všechny akce / pohledy
- hnědé – metody pro konkrétní pohled
- modrá – metoda, která má na starosti zpracování konkrétního signálu

5.2.2.1 Popis jednotlivých metod presenteru

Fáze výkonná

1. Startup – je vyvolána na začátku životního cyklu presenteru. Během životního cyklu aplikace se může spustit více presenterů, metoda startup() se může volat vícekrát.
2. action{Action} – by měla obsahovat vykonání operací, po kterých může následovat přesměrování. Také zde může být logika rozhodování pro členění na jednotlivé pohledy.

Fáze změn vnitřních stavů

3. handle{Signal} – zpracování signálů neboli subrequestů. Určeno pro uživatelskou interakci a zpracování AJAXových požadavků.

Fáze vykreslovací

4. beforeRender – může obsahovat například společné nastavení filtrů pro všechny vykreslovače a nastavení společných proměnných pro šablony všech vykreslovačů.
5. render{View} má na starosti vykreslení a s tím spojené nutné činnosti jako jsou tvorba odkazů v šablonách, přiřazení proměnných do jednotlivých šablon apod.

Fyzické vykreslení šablony

6. Uložení vnitřních stavů: dříve než se přejde k další fázi, uloží se stav všech vnitřních stavů a persistentních proměnných.
7. Vykreslení šablony na výstup

Ukončení činnosti

8. shutdown je vyvolána při ukončení životního cyklu presenteru. Zde je možné ukončit databázové připojení, kešování apod. [9]

5.2.3 Nástroje pro zabezpečení

Nette se snaží ušetřit spoustu rutinní práce při programování. Jednou z nutných činností je zabezpečení aplikace. Ve frameworku se nachází spousta nástrojů, které pomáhají zabezpečit aplikaci proti nejrůznějším typům útoků.[7]

5.2.3.1 Cross-site scripting (XSS)

Cross-site scripting je metoda narušení webových stránek zneužívající výstupů. Útočník pak dokáže do stránky podstrčit svůj vlastní kód a tím může stránku pozměnit nebo dokonce získat citlivé údaje o návštěvnících. Proti XSS se lze bránit jen důsledným a korektním ošetřením všech řetězců.

Nette má proti tomuto revoluční technologii Content-aware escaping. Jedná se automatické escapování všech proměnných při vypisování v šabloně. Přitom Nette rozpoznává, zda se proměnná vypisuje v parametru tagu, v definici skriptu a podle toho přizpůsobí ošetření řetězce.

5.2.3.2 Cross-site request forgery (CSRF)

Cross-site request forgery je útok spočívající v tom, že přimějeme uživatele navštívit stránku, která skrytě vykoná útok na webovou aplikaci, kde je uživatel zrovna přihlášen. Lze tak například pozměnit nebo smazat článek, aniž by si toho uživatel všiml. Proti útoku se lze bránit generováním a ověřováním autorizačního tokenu.

Nette Framework obranu formulářů před útokem cross-site request forgery zjednodušuje použitím addProtection().

5.2.3.3 URL attack, control codes, invalid UTF-8

Různé pojmy související se snahou útočníka podstrčit vaší webové aplikaci škodlivý vstup. Následky mohou být velmi různorodé, od poškození XML výstupů (např. nefunkční RSS kanály) přes získání citlivých informací z databáze nebo hesel. Obranou je důsledné ošetřování všech vstupů na úrovni jednotlivých bajtů.

Nette Framework toto ošetřuje zcela automaticky. Není potřeba nic nastavovat.

5.2.3.4 Session hijacking, session stealing, session fixation

Se správou session je spojeno hned několik typů útoků. Útočník buď zcizí anebo podstrčí uživateli své session ID a díky tomu získá přístup do webové aplikace, aniž by znal heslo uživatele. Poté může v aplikaci provádět cokoliv, aniž by o tom uživatel věděl. Obrana spočívá ve správném nakonfigurování serveru a PHP.

Nette Framework nakonfiguruje PHP zcela automaticky. Jediná nutnost je, aby byla povolena uživatelská konfigurace pomocí PHP.

6 MYSQL

MySQL je databázový systém vytvořený firmou MySQL AB, nyní vlastněný společností Sun Microsystems, dceřinou společností Oracle Corporation.

MySQL je multiplatformní databáze, se kterou se komunikuje pomocí jazyka SQL. Podobně jako u ostatních SQL databází se jedná o dialekt tohoto jazyka s některými rozšířeními.

Pro svou snadnou implementovatelnost (lze jej instalovat na Linux, MS Windows, ale i další operační systémy), výkon a především díky tomu, že se jedná o volně šiřitelný software, má vysoký podíl na v současné době používaných databázích. Velmi oblíbená a často nasazovaná je kombinace Linux, MySQL, PHP a Apache jako základní software webového serveru („technologie LAMP“).

MySQL bylo od počátku optimalizováno především na rychlost, a to i za cenu některých zjednodušení: má jen jednoduché způsoby zálohování, a až donedávna nepodporovalo pohledy, triggery, a uložené procedury. Tyto vlastnosti jsou doplňovány teprve v posledních letech, kdy začaly nejčastějším uživatelům produktu – programátorům webových stránek – již poněkud scházet. [6]

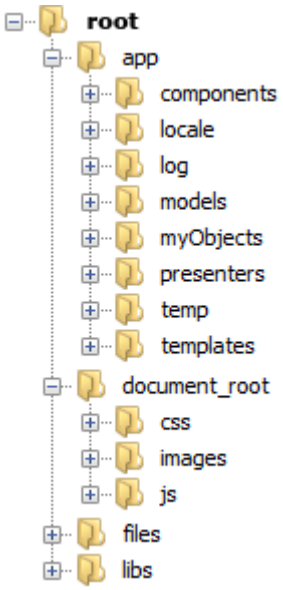
II. PRAKTICKÁ ČÁST

7 NÁVRH APLIKACE

V této části práce bude popsána struktura adresářů, databáze. Také zde bude popsána knihovna pChart, pomocí které jsou v aplikaci generovány grafy.

7.1 Adresářová struktura

Protože je aplikace vytvářena pomocí Nette Frameworku, bude adresářová struktura (Obrázek 5) ovlivněna doporučenou strukturou tohoto frameworku.



Obrázek 5. Adresářová struktura

7.1.1 app

Toto je asi nejdůležitější adresář v aplikaci. Hlavním důvodem umístění tohoto adresáře je bezpečnost. Adresář se nachází mimo složku viditelnou z webu, a proto se k ní případný útočník nemá jak dostat.V tomto adresáři se nachází veškerá aplikační logika a každá její část zde má svůj podadresář. Je to dáno proto, aby byly jednotlivé části logicky seřazeny a aby bylo ihned zcela zřejmé, kde kterou třídu hledat.

Nejdůležitější podadresáře:

- models
- presenters
- temp
- templates

Mimo tyto se zde ještě nachází další podadresáře, které bylo vhodné vytvořit pro zachování přehlednosti v souborech. Jsou to:

- components
- locale
- log
- myObjects

Nette sice má doporučenou strukturu adresářů, ale nic z toho není povinné. V tomto frameworku se totiž nachází velmi užitečná funkce autoloader, která automaticky vyhledává třídy ve všech složkách aplikace a v případě, že jsou tyto třídy potřeba použít je i zpřístupní. Tento přístup má dvě velké výhody a to, že se šetří paměť nutnou k načtení a zpracování aplikace a že není nutné dodržovat žádnou stanovenou adresářovou strukturu.

V této složce se nachází také dva velmi důležité soubory:

- config.ini
- bootstrap.php

7.1.2 document_root

Toto je jediná složka, která je viditelná z webu. Nachází se zde pouze jeden php soubor a to index.php, který se stará o inicializaci konstant, které obsahují cesty ke složce viditelné z webu, ke složce s aplikační logikou a ke složce s knihovnami. Po inicializaci těchto konstant už jen zavede soubor bootstrap.php z adresáře /app.

Do tohoto adresáře patří všechny soubory, které jsou potřeba pro správné zobrazení grafického rozhraní. Jsou to javascripty, kaskádové styly a obrázky.

7.1.3 libs

Do této složky patří všechny php knihovny, které chci použít v aplikaci. V našem případě to jsou tyto knihovny:

- Nette
- Dibi
- pChart

Důvod, proč je tato složka oddělena od složky app je univerzálnost. V případě více aplikací, které jsou ve stejném adresáři, mohou tyto využívat pouze jednu sdílenou složku s knihovnami. V případě aktualizací potom stačí nahradit knihovnu jen na jednom místě a změna se projeví všude. V naší aplikaci toto sice není nutné, ale je dobré myslet do budoucna a ušetřit tak připadnou práci sobě, nebo ostatním programátorům.

7.1.4 files

Složka files je pouze pomocný adresář, do kterého plánuji ukládat nahrané soubory, či soubory, které nemají být vidět z webu. Je to kvůli bezpečnosti a kvůli logice adresářové struktury.

7.2 Databáze

Pro uchovávání dat v aplikaci je použita databáze MySQL5.1. Volba na tuto databázi padla díky její rozšířenosti. Dnes téměř u každého webového hostingu, kde poskytují podporu php se automaticky přidává právě databáze MySQL.

7.2.1 Dibi

Dibi je databázový layer vyvíjený autory Nette Framework. Je přímo vyvíjen pro spolupráci s tímto frameworkem a proto jsem jej začlenil do našeho webu.

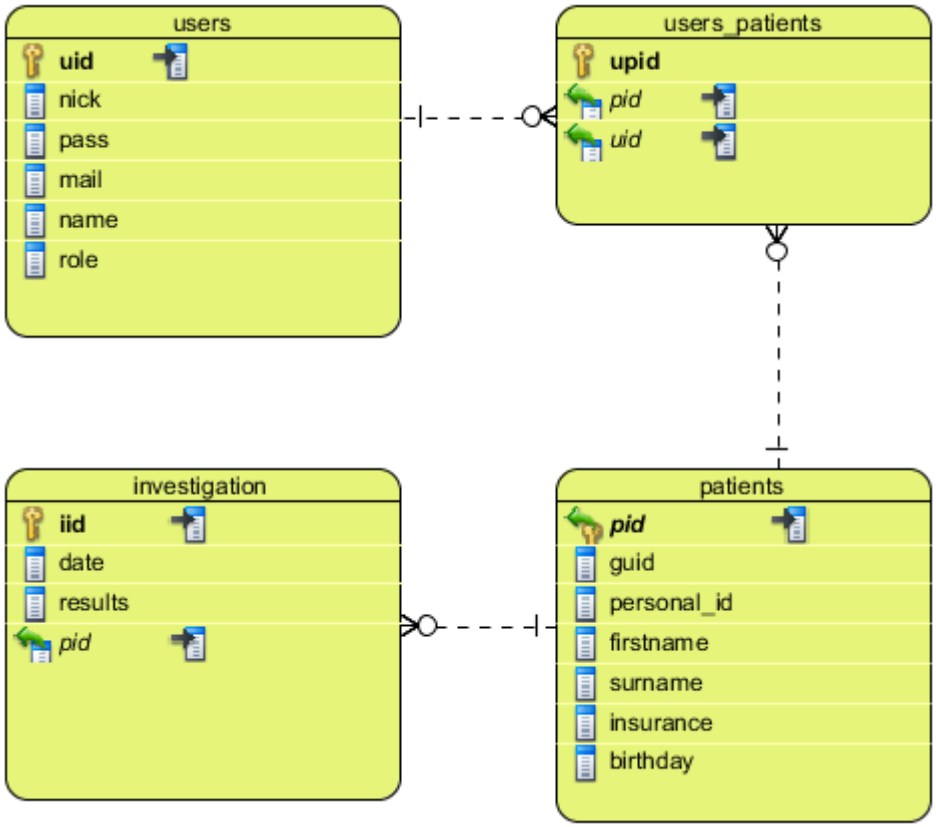
Díky této třídě je použití databáze velmi zjednodušeno. Dibi se stará o spojení s databází, správné formátování dotazů, kontroluje datové typy vstupních proměnných a také umí vrátit výsledek dotazu v optimalizovaném tvaru.

Dibi lze použít i voláním statické třídy. Toto je užitečné i díky tomu, že lze definovat připojení k databázi jen na jednom místě a pak kdekoli v aplikaci volat statické metody této třídy bez nutnosti vytvářet instanci třídy.

7.2.2 Struktura databáze

Vzhledem k potřebám aplikace mi budou stačit pouze 4 tabulky, jak jde vidět na Obrázek 6.

Nyní budou popsány jednotlivé tabulky a to z hlediska obsahu a použití.



Obrázek 6. Struktura databáze

7.2.2.1 Tabulka users

Toto je tabulka, která uchovává informace o lékařích, uživatelích aplikace. Jsou zde uloženy přihlašovací údaje, kontaktní informace a také role uživatele. Položky nick a pass jsou důležité pro přihlášení. Uživatele identifikuje přes celou aplikaci jeho uid a role.

Tabulka 1. Popis sloupců v tabulce users

Název	Datový typ	Popis
uid	int(10)	Unikátní identifikátor uživatele.
nick	varchar(50)	Přihlašovací jméno
pass	varchar(50)	Přihlašovací heslo. Do databáze se ukládá pouze zašifrovaná podoba pomocí algoritmu md5
mail	varchar(255)	Kontaktní email
role	enum(‚admin‘, ‚doctor‘)	Uživatelská role

7.2.2.2 Tabulka users_patients

Toto je pomocná tabulka, která má za úkol spojit tabulky users a patients v relaci N:N. MySQL databáze tuto relaci bohužel nepodporuje, takže se tento problém řeší právě pomocí takové jednoduché tabulky. V této tabulce se tedy nachází pouze tři sloupce. Jedinečný identifikátor záznamu, a pak identifikátory tabulek, které chceme provázat.

Tabulka 2. Popis sloupců v tabulce users_patients

Název	Datový typ	Popis
upid	int(10)	Identifikátor záznamu
uid	int(10)	Identifikátor uživatele – z tabulky users
pid	int(10)	Identifikátor pacienta – z tabulky patients

7.2.2.3 Tabulka patients

V této tabulce se uchovávají všechny informace o pacientovi. Data se dají získat dvěma různými cestami. Buď nahráním xml souboru s vyšetřením, nebo přímo zadáním údajů pomocí formuláře.

Tabulka 3. Popis sloupců v tabulce patients

Název	Datový typ	Popis
pid	int(10)	Jedinečný identifikátor pacienta
guid	varchar(50)	Jedinečný identifikátor pacienta z původního systému. Je obsažen v xml souboru a to v názvu souboru a také jako parametr kořenového elementu
personal_id	varchar(11)	Rodné číslo pacienta
firstname	varchar(200)	Křestní jméno
surname	varchar(200)	Příjmení
insurance	int(4)	Kód pojišťovny
birthday	date	Datum narození

7.2.2.4 Tabulka investigations

Do této tabulky patří data vyšetření. Vyšetřením nemyslíme konkrétní jeden úkon, ale všechny úkony vyšetření provedené v jednom dni. Přitom platí, že každý úkon může být ve vyšetření uložen právě jedenkrát. Proto se všechna data (výsledky) vyšetření ukládají v xml řetězci v takové podobě, v jaké jsou v původním xml souboru. Má to jeden důvod. V případě nového typu úkonu vyšetření by bylo potřeba složitě měnit strukturu databáze. S tímto přístupem stačí pouze přidat nové metody do třídy zpracovávající výsledky vyšetření. Zdroj dat tedy zůstává stejný.

Tabulka 4. Popis sloupců v tabulce investigations

Název	Datový typ	Popis
iid	int(10)	Jedinečný identifikátor vyšetření
date	datetime	Datum a čas vyšetření
results	text	XML data z vyšetření
pid	int(10)	Identifikátor spojující vyšetření s konkrétním pacientem

7.3 Grafy

Jednou z nejdůležitějších částí naší aplikace je grafické zobrazení dat. Tato data jsou dodávána z měřících přístrojů v podobě pro člověka buď nečitelných, nebo velmi těžce interpretovatelných. Pro lékaře jsou nejpřijatelnější dvě podoby: tabulky a grafy.

Tvorba tabulek je dobře řešena již v samotném html. Proto je potřeba zajistit vhodný způsob jak zajistit zobrazování grafů. Existuje mnoho způsobů jak zobrazit grafy. Jednou z možností je použít výbornou php knihovnu pChart (http://pchart.sourceforge.net). Tato knihovna vyžaduje mít na serveru nainstalované php rozšíření GD2. Jedná se o grafickou knihovnu, pomocí které se dají generovat obrázky pomocí php skriptů. [11]

Tato knihovna umí vygenerovat velkou spoustu typů grafů:

- Koláčové (základní, rozpadlé, 3D)
- Sloupcové (základní, skládané, překrývané)

- Spojnicové (základní, výsečové, vyplněné)
- Kubické křivky (základní, vyplněné)
- Bodové
- Radarový graf

Další vlastnosti knihovny pChart

- Možnost zobrazit hodnoty v grafu
- Grafický report případných chyb
- Import dat pomocí CSV souboru
- Zobrazení grafu i v případě chybějících hodnot
- Kešování grafů
- Možnost nastavení pozadí a barev grafů
- Možnost použít vlastní grafické značek a popisky os
- Zobrazení více grafů v jednom výstupu

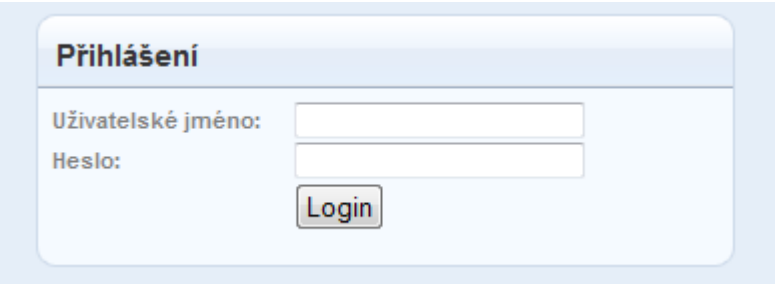
Ve vytvořené aplikaci je tedy pro zobrazení audiogramu použit spojnicový graf spojený s bodovým. Díky možnosti použít vlastní grafické značky mohla být bez problémů dodržena norma pro značení na audiogramu. Bohužel se objevil i drobný problém. A to právě zobrazení grafu i v případě chybějících hodnot. Pokud totiž chybí nějaká položka v datech, tak se tady graf přeruší a u další hodnoty opět pokračuje. Toto je však pro aplikaci nepřípustné chování. Proto bylo nutné poopravit některé metody třídy pChart, aby při vykreslování spojnicového grafu v případě vynechání hodnoty další hodnotu spojila s poslední známou. Toto má bohužel tu nevýhodu, že v případě aktualizace knihovny bude nutné tuto novou opět pozměnit (pokud toto chování nebude již v nové knihovně poopraveno).

8 POPIS APLIKACE

Protože převážná část bakalářské práce bylo programování aplikace, tak bude v této části ukázáno, jak vytvořená aplikace funguje. Postupně budou popsány jednotlivé části aplikace, možnosti, které uživatel má a bude popsán způsob práce s aplikací.

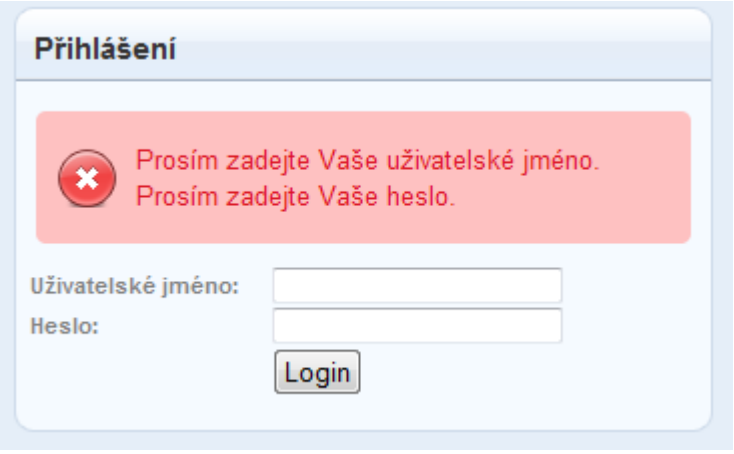
8.1 Přihlášení

Po spuštění aplikace se zobrazí přihlašovací okno (Obrázek 7). Uživatel je vyzván k vyplnění přihlašovacího jména a hesla. Tyto přihlašovací údaje jsou povinné. Po vyplnění je potřeba stisknout tlačítko „Login“.



Obrázek 7. Přihlašovací okno

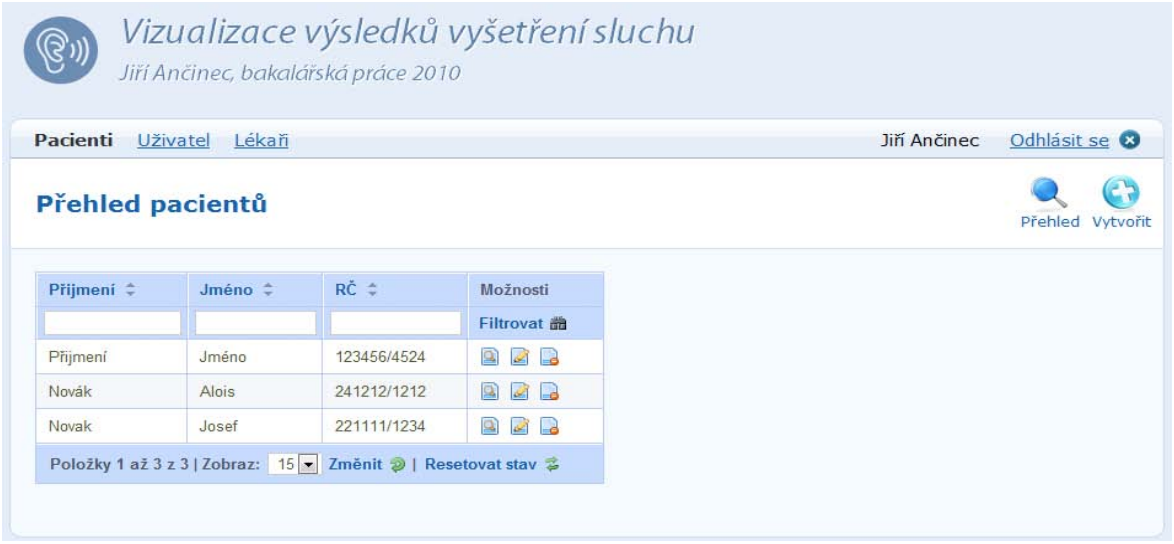
V případě, že uživatel nevyplní některou položku, tak jej prohlížeč pomocí javascriptu upozorní na špatně vyplněný formulář a tento formulář ani neodešle. V případě nefunkčního javascriptu aplikace uživatele nepřihlásí a zobrazí se mu opět přihlašovací formulář i s chybovou hláškou (Obrázek 8). Chybová hláška se zobrazí i v případě, že uživatel zadá chybné přihlašovací údaje.



Obrázek 8. Přihlašovací okno s chybovou hláškou

8.2 Přehled pacientů

Pokud se uživatel úspěšně přihlásí, tak se mu zobrazí úvodní obrazovka – přehled pacientů. V horní části se nachází hlavní menu a submenu. Záleží na oprávnění (roli) uživatele, co se mu zobrazí.



Obrázek 9. Náhled úvodní obrazovky administrátora

Obrázek 10. Náhled úvodní obrazovky lékař

V horní části pod logem se nachází jednoduchá navigační lišta. V případě administrátorského oprávnění (Obrázek 9) v ní najdeme odkaz pro přehled pacientů, lékařů, odkaz pro editaci vlastního profilu a napravo od jména právě přihlášeného uživatele se nachází odkaz pro odhlášení z aplikace. Pokud je přihlášen lékař (Obrázek 10), tak se zde nachází pouze odkazy pro editaci vlastního profilu a odkaz pro zobrazení pacientů, kteří byli tomuto lékaři přiděleni.

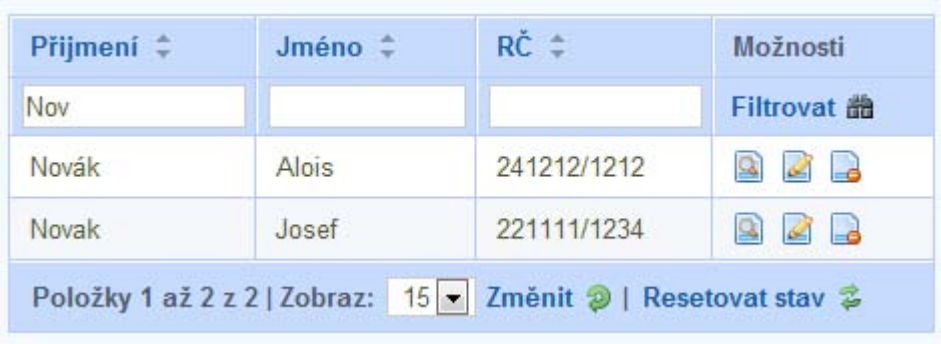
Pod navigační lištou se nachází nadpis stránky, který přehledně informuje, ve které části aplikace se nacházíme. Napravo od nadpisu jsou umístěny navigační ikony, které pomáhají s navigací ve vybrané sekci. Tyto ikony mohou být pro jednotlivé sekce různé. Také záleží na příslušné roli uživatele.

Poslední grafickou částí naší aplikace je hlavní obsah. Toto je část, ve které se zobrazují veškeré informace podle zvolených položek v navigaci. Na úvodní obrazovce neboli přehledu pacientů se v této části nachází pouze tabulka, která přehledně zobrazuje pacienty. Jedná se o speciální komponentu datagrid (Obrázek 11), kterou lze do webové aplikace přidat pomocí Nette frameworku. Tato komponenta má oproti obyčejné tabulce několik výhod:

- Zjednodušené navázání dat z databáze
- Možnost filtrovat zobrazená data
- Možnost zobrazit různé množství záznamů
- Stránkování v záznamech
- Podpora ajaxu

Obrázek 11. Přehled pacientů

Přehled pacientů (Obrázek 11) můžeme filtrovat pomocí formulářových polí, které se nachází ve druhém řádku. Po zadání filtru (například první písmena z příjmení pacienta) můžeme filtr aplikovat pomocí stisknutí klávesy enter, nebo kliknutím na odkaz „Filtrovat“, který se nachází ve druhém řádku datagridu vpravo. Výsledkem jsou tedy data v tabulce, která odpovídají zadanému filtru (Obrázek 12).



Obrázek 12. Datagrid – filtrování dat

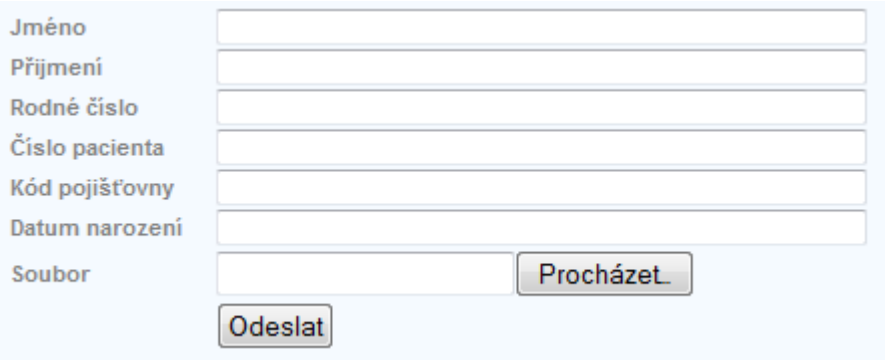
Pokud chceme filtr zrušit, stačí kliknout na odkaz „Resetovat stav“, který se nachází ve spodním řádku datagridu, a opět se načtou původní data, která nejsou omezená filtrem.

Na každém řádku, který odpovídá jednomu pacientovi, se nachází navigační ikony. V případě, že uživatel má roli lékaře, tak se zde nachází pouze jedna ikona – detail pacienta. Administrátor má k dispozici ještě dvě další ikony:

- Editovat pacienta
- Smazat pacienta

8.2.1 Vytvoření nového pacienta

Pro vytvoření nového pacienta je potřeba kliknout kdekoli v části „Pacienti“ na ikonu „Vytvořit“. Zobrazí se formulář (Obrázek 13), pro vytvoření nového pacienta.



Obrázek 13. Formulář pro vytvoření nového pacienta

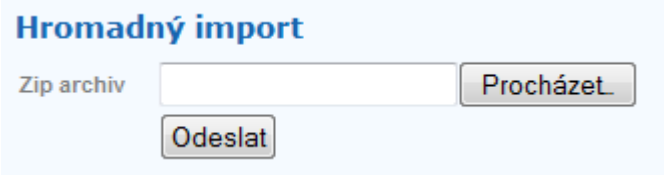
Jedinou povinnou položkou, kterou je nutné vždy vyplnit, je soubor pacienta. Jedná se o xml soubor, ve kterém jsou umístěny informace o pacientovi a všechna jeho vyšetření, která prodělal.

Během testování uploadu xml souborů jsem zjistil, že aplikace Fowler 3 chybně ukládá písmena s diakritikou do souborů. To se projevuje tak, že tato písmena vynechá.

Proto jsem zapracoval do formuláře políčka pro zadání všech hodnot, která lze o pacientovi ze souboru získat. V případě, že je nějaké pole vyplněno, bude příslušný údaj z xml souboru při ukládání do databáze tímto novým údajem nahrazen.

8.2.1.1 Hromadný import dat

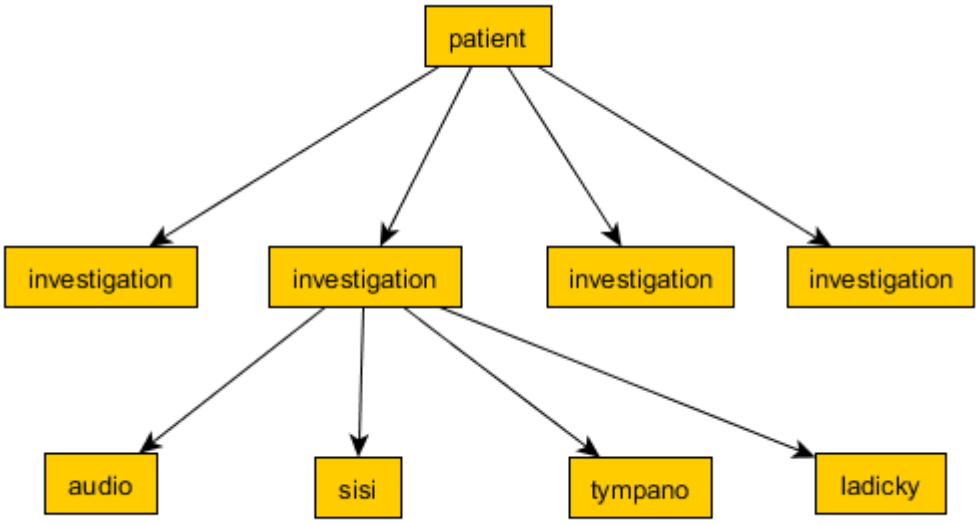
Aby administrátor nemusel nahrávat velké množství pacientů postupně jednoho po druhém, přidal jsem možnost nahrát ZIP archiv (Obrázek 14), který obsahuje xml soubory. Administrátorovi tedy stačí ve svém počítači vytvořit archiv souborů a do aplikace nahrát jen tento jeden.



Obrázek 14. Hromadný import

8.2.1.2 Struktura xml souboru pacienta

XML soubor je generovaný pomocí aplikace Fowler 3, kterou vyšetřující lékař používá k ukládání naměřených dat do počítače.



Obrázek 15. Struktura xml souboru pacienta

Nyní popíšu jednotlivé elementy, které lze najít v xml souboru (Obrázek 15).

patient

Jedná se o kořenový element XML, a proto v celém dokumentu můžeme najít pouze jeden. V tomto elementu můžeme najít tyto atributy:

- guid – jedinečný identifikátor pacienta v aplikaci Fowler 3 a také odpovídá názvu xml souboru.
- id – rodné číslo pacienta
- firstname – křestní jméno
- surname – příjmení
- insurance – kód pojišťovny

investigation

Element investigation odpovídá právě jednomu vyšetření. Rodič tohoto elementu je kořenový element „patient“. Vyšetření může být v souboru uloženo více. Přitom je obsahově od sebe odděluje především údaj o datumu a času vyšetření.

Atributy:

- date – datum vyšetření
- time – čas vyšetření

audio

V tomto elementu se nachází veškerá data potřebná k zobrazení audiogramu. Bohužel jsou jednotlivé křivky uloženy v podobě jednoho řetězce, který je uložen v argumentu tohoto souboru.

Ukázka:
„125:10:250:10:500:15:1000:15:1500:10:2000:10:3000:10:4000:10:6000:20:8000:30“

Data jsou v tomto řetězci uložena všechna pohromadě, přičemž jednotlivé páry hodnot frekvence : decibely jsou odděleny také dvojtečkou. V případě že u dané frekvence nebyla naměřena intenzita zvuku, je tato hodnota vynechána, neboli se objeví dvě

dvojtečky vedle sebe. Vhodnější by bylo, kdyby bylo využito možností XML a místo uložení dat do atributu elementu by tyto byly uloženy v jednotlivých subelementech.

Atributy použité v elementu audio:

- dxphone – vzdušné vedení u pravého ucha
- sinphone – vzdušné vedení u levého ucha
- dxbone – kostní vedení u pravého ucha
- sinbone – kostní vedení u levého ucha
- dxUCL – práh nepříjemného poslechu u pravého ucha
- sinUCL – práh nepříjemného poslechu u levého ucha
- comment – komentář lékaře k audiometrickému vyšetření.

sisi

Toto vyšetření se obvykle zobrazuje pouze v tabulce, kde sloupce odpovídají jednotlivým frekvencím a řádky určují, zda jde o pravé, či levé ucho.

Atributy:

- dx500 – hodnota pro frekvenci 500Hz měřená na pravém uchu.
- dx1000 – hodnota na pravém uchu pro frekvenci 1000Hz
- dx2000 – hodnota na pravém uchu pro frekvenci 2000Hz
- dx4000 – hodnota na pravém uchu pro frekvenci 4000Hz
- dx500 – hodnota na levém uchu pro frekvenci 500Hz
- dx1000 – hodnota na levém uchu pro frekvenci 1000Hz
- dx2000 – hodnota na levém uchu pro frekvenci 2000Hz
- dx4000 – hodnota na levém uchu pro frekvenci 4000Hz
- popis – komentář lékaře k vyšetření SISI

ladicky a tympano

Zde se nachází data z vyšetření s ladičkami a z tympanometrického vyšetření. Tato vyšetření jsem již do aplikace nezapracoval.

8.2.2 Editace pacienta

Editovat pacienta může pouze uživatel s oprávněním administrátora. Proto uživatelé s touto rolí vidí v přehledu pacientů na každém řádku ikonku pro editaci. Po kliknutí na ni se zobrazí podobný formulář jako u vytváření nového pacienta (Obr. 12). Tento formulář se však liší v tom, že při editování není povinné zadat XML soubor pacienta.

8.2.3 Smazání pacienta

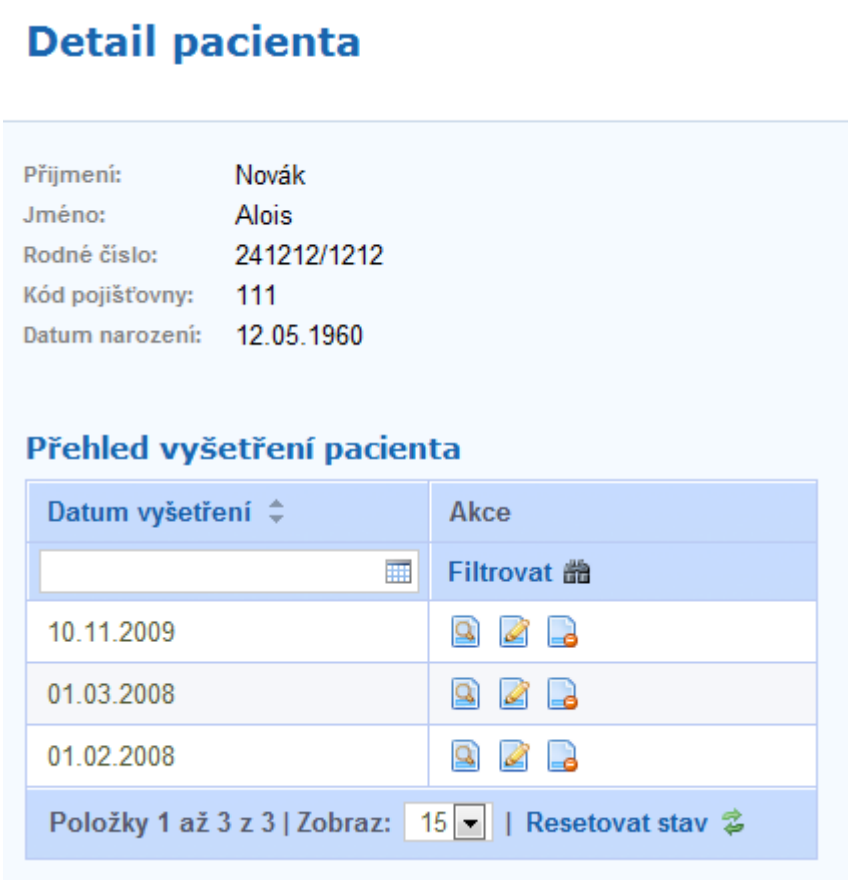
Smazat pacienta může opět pouze uživatel s administrátorským oprávněním. Stačí k tomu vyhledat pacienta v přehledu pacientů a kliknout na ikonu „Smazat pacienta“. Tato akce je nastavena jako ajaxová událost, proto se kvůli tomu nemusí načítat celá stránka, ale jen se znovu načtou data v datagridu.

V případě, že pacient má uložena nějaká vyšetření, jsou vymazána také.

8.3 Detail pacienta a přehled vyšetření

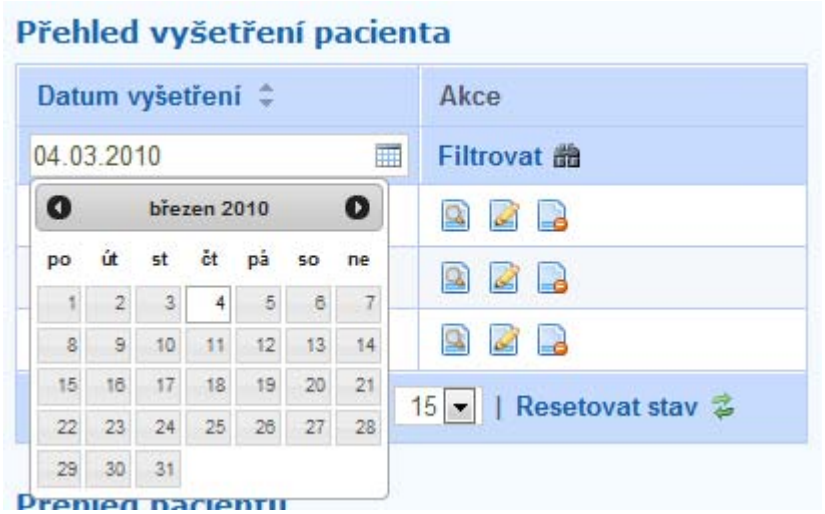
Přehled vyšetření se zobrazí automaticky při zobrazení detailu pacienta. Zobrazí se pouze ta vyšetření, která patří k tomuto pacientovi. Možnost práce s vyšetřeními záleží opět na uživatelském oprávnění.

Administrátor může editovat, mazat, prohlížet vyšetření (Obrázek 16). Lékař může pouze prohlížet vyšetření.



Obrázek 16. Detail pacienta a přehled vyšetření

Do tohoto datagridu jsem přidal možnost filtrovat vyšetření podle datumů (Obrázek 17). Je možné tedy pomocí kalendáře najít odpovídající vyšetření, bez nutnosti složitě prohledávat všechny položky.



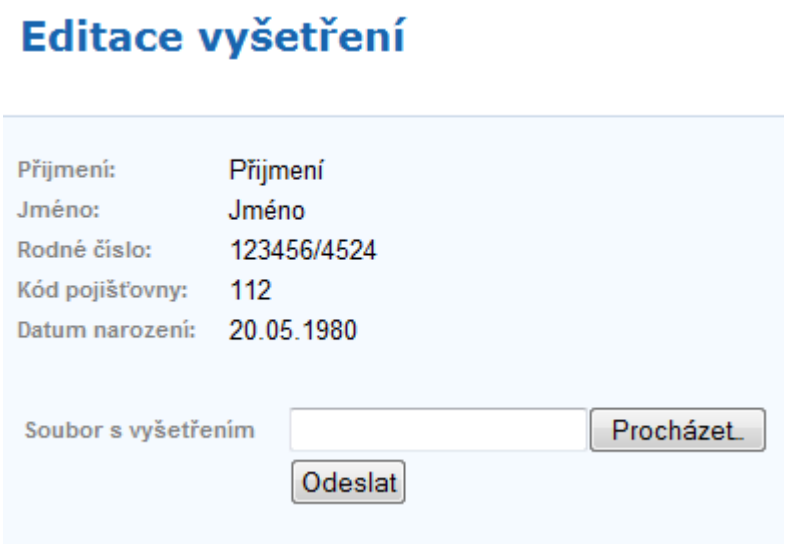
Obrázek 17. Filtrování pomocí kalendáře

8.3.1 Vytvoření nového vyšetření

Vytvořit nové vyšetření lze v editaci pacienta (Obrázek 13) nahráním xml souboru, nebo použitím hromadného importu pomocí zip archivu (Obrázek 14). Při nahrání souboru se projdou všechna vyšetření, a v případě, že nějaké neexistuje, tak se toto uloží. S již existujícími vyšetřeními se nic neděje.

8.3.2 Editace vyšetření

Vyšetření se edituje také pouze nahráním xml souboru, použitím formuláře (Obrázek 18), který se zobrazí po kliknutí na ikonu „Editovat vyšetření“.



Obrázek 18. Formulář pro editaci vyšetření

V případě, že administrátor omylem nahraje soubor, jiného pacienta, než kterého právě edituje, bude na to upozorněn. Mohou nastat dvě situace:

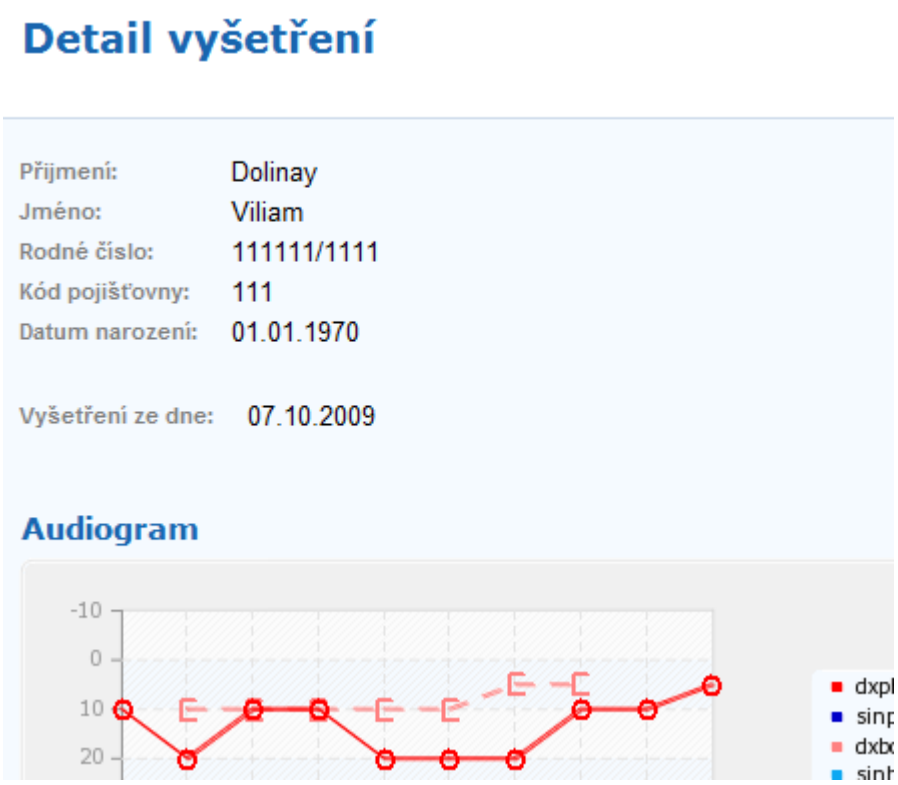
- Nahraje jiného existujícího pacienta – potom bude vrácen na stránku s formulářem, kde může zadat znovu správný soubor
- Nahraje zatím v naší aplikaci neznámého pacienta – potom mu aplikace nabídne, tohoto pacienta vytvořit.

8.3.3 Smazání vyšetření

Ke smazání vyšetření slouží administrátorovi třetí ikonka. V případě, že se rozhodne vyšetření smazat, tak to nemusí být natrvalo. Pokud v příští editaci pacienta nahraje pacientův xml soubor, ve kterém se toto vyšetření bude nalézat, tak jej aplikace znovu vytvoří. Je to proto, aby v případě mylného smazání bylo jednoduché tuto chybu napravit.

8.4 Detail vyšetření

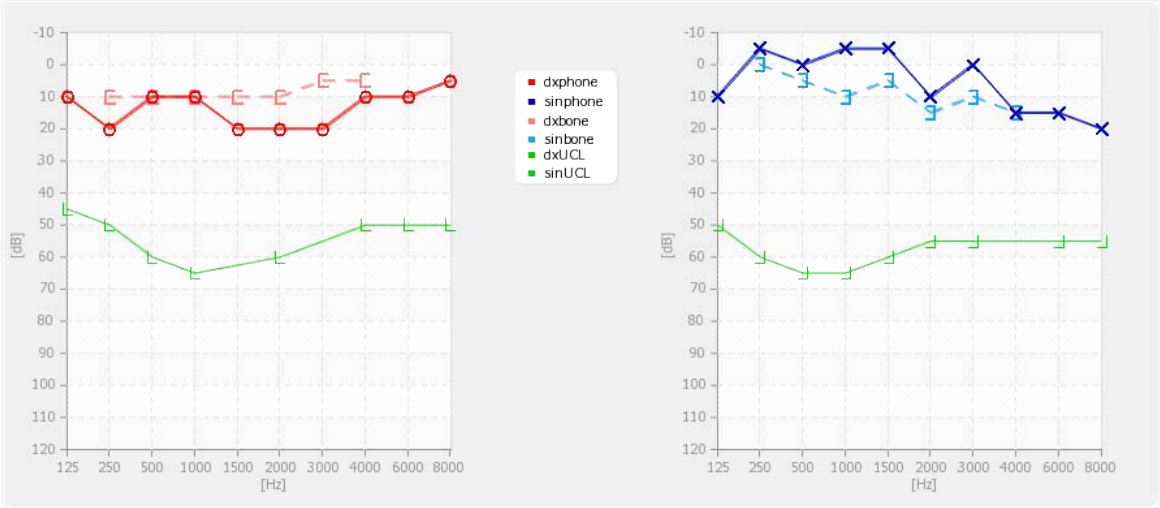
V detailu vyšetření (Obrázek 19) jsou pro přehlednost uvedeny informace o pacientovi, a také datum vyšetření. Je to proto, aby bylo ihned zřejmé, k čemu zobrazené výsledky patří.



Obrázek 19. Detail vyšetření

Naše aplikace umí zobrazit dva typy výsledků vyšetření.

8.4.1 Audiogram



Obrázek 20. Audiogram generovaný aplikací

8.4.2 Sisi

Sisi

	500 Hz	1000 Hz	2000 Hz	4000 Hz
dx	20%	10%	15%	15%
sin	20%	20%	20%	25%

Obrázek 21. Tabulka vyšetření Sisi

8.4.3 Ostatní typy vyšetření

Pokud je ve vyšetření, na jehož detail uživatel kliknul jiné vyšetření, než audiogram či Sisi, bude vrácen zpět na seznam vyšetření pacienta a bude informován, že ve vybraném vyšetření se nachází žádná data, která je možné v této aplikaci zobrazit (Obrázek 22).

Detail pacienta

Ve Vámi zvoleném vyšetření se nenachází žádná data, která je možné v této aplikaci zobrazit.

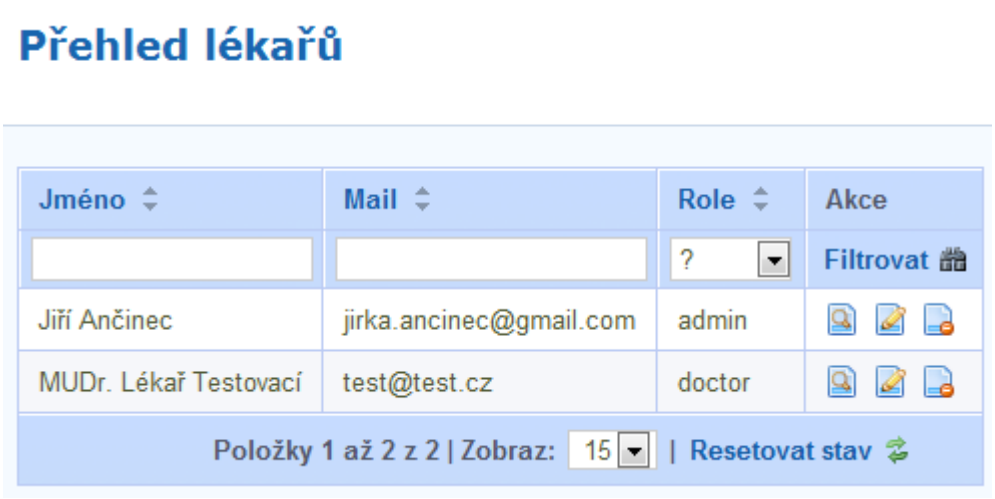
Obrázek 22. Informační hláška po nezobrazení vyšetření

8.5 Přehled lékařů

Toto možnost má v menu pouze administrátor. Může zde vyhledávat a třídit uživatele, editovat je či případně smazat.

Filtrovat uživatele lze

- Podle jména
- Podle emailu
- Podle uživatelské role – zde je na výběr roletový seznam, který je vhodný právě pro položky, kde je jen určité omezené množství možných hodnot.

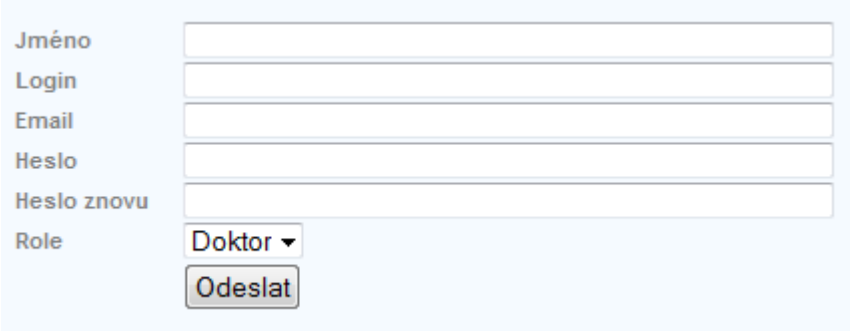


Obrázek 23. Přehled lékařů

8.5.1 Vytvoření a editace lékaře

Odkaz pro vytvoření nového lékaře se nachází napravo od nadpisu, v tzv. submenu. Je to obrázková ikona s nápisem „Vytvořit“. Po kliknutí na tuto ikonu se zobrazí formulář pro vytvoření nového lékaře (Obrázek 24). Zde jsou všechny položky povinné, navíc formulář kontroluje, zda je email napsán ve správném tvaru a zda se hesla shodují.

V případě, že je nějaký prvek nevyplněn, nebo je vyplněn chybně, tak je na to administrátor upozorněn a formulář se neodešle.



Obrázek 24. Formulář pro vytvoření lékaře

Editace lékaře probíhá na stejném formuláři se stejnými pravidly jako vytváření nového lékaře. Je tu pouze jedna výjimka, a tou je, že heslo nemusí být vyplněno. V případě, že do políčka „Heslo“ není nic napsáno, tak se při ukládání změn do databáze heslo přeskočí a zůstane nezměněno.

8.5.2 Detail lékaře a přehled přiřazených pacientů

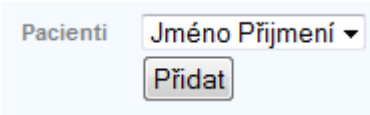
Po kliknutí na ikonku „Zobraz“ v přehledu lékařů se zobrazí detail lékaře (jméno a email) a přehled pacientů přiřazených k lékaři (Obrázek 25).

Obrázek 25. Přehled pacientů přiřazených k lékaři

8.5.3 Přiřazení pacienta k lékaři

Pacienta je možné přiřadit k lékaři po kliknutí na ikonu „Přiřadit pacienta“, která se nachází nahoře, vpravo od nadpisu (v submenu).

Po kliknutí na tuto možnost se zobrazí nad přehledem přiřazených pacientů (Obrázek 25) jednoduchý formulář, se seznamem pacientů, které je možné lékaři přiřadit (Obrázek 26).



Obrázek 26. Formulář pro přiřazení pacienta k lékaři

Jednoho pacienta je možné přiřadit více lékařům. Pokud je ale zvolený lékař administrátorem, tak nabídka přiřazení pacienta není zobrazena. Je to z toho důvodu, že administrátor má práva zobrazit vše.

8.5.4 Smazání přiřazení pacienta k lékaři

Pro tuto volbu je v přehledu přiřazených pacientů k lékaři připravena ikonka s názvem „Smaž“. Po kliknutí na ni se pouze smaže přiřazení k lékaři. Pacient stále zůstává v databázi.

8.6 Editace vlastního uživatelského účtu

Pro editaci vlastního uživatelského účtu je k dispozici možnost „Uživatel“, která se nachází v hlavním menu. Tuto položku má k dispozici každý uživatel a má díky ní možnost jakkoli měnit informace přiřazené ke svému účtu.

Formulář je téměř identický s formulářem pro vytvoření nového lékaře (Obrázek 24), ale chybí zde možnost zvolit uživatelskou roli.

8.7 Odhlášení z aplikace

Odhlášení z aplikace se může stát dvěma způsoby:

- Kliknutím na odkaz „Odhlásit se“
- Vypršením doby platnosti stránky kvůli neaktivitě uživatele

V obou případech je uživatel odpojen ze systému, a není možné zobrazit žádná data ani kliknutím na tlačítko zpět v prohlížeči. Odhlášením také ztrácí uživatel přístup k obrázkům audiogramu (Obrázek 20).

8.8 Praktické využití aplikace

Aplikaci, která je výsledkem této bakalářské práce, je možné nasadit na webový server a je možné ji začít ihned využívat. Je nutné, aby tento webový server splňoval následující minimální požadavky:

- Apache 2.x s podporou SSL
- PHP 5.2 či vyšší s podporou knihoven: GD2, zip, MySQL, mbstring
- MySQL5.1

Aplikace podporuje práci více uživatelů, takže je možné ji nasadit například v nemocnici na počítač umístěný v nemocniční síti a lékaři tak mohou mít přístup k výsledkům ORL vyšetření téměř ihned.

Je také možné nasadit aplikaci na veřejný server. Zde je ale nutné mít podporu šifrovaného spojení (SSL) a také aby tento server byl důvěryhodný. Vyšetřující lékař může pomocí aplikace zveřejňovat například privátním lékařům výsledky ORL vyšetření jejich pacientů.

Další možností využití této aplikace je její nasazení na běžném počítači díky softwarovému řešení XAMPP. Vyšetřující lékař si může do aplikace ukládat výsledky pro své potřeby.

I když má administrátor možnost vkládat pacienty jednotlivě, má k dispozici i nástroj pro hromadné nahrání více pacientů. Tato funkce je otestována nahráním zip souboru, který obsahoval zhruba 50 xml souborů.

ZÁVĚR

Výsledkem této bakalářské práce je funkční webová aplikace, sloužící k zobrazení výsledků vyšetření sluchu, která lze importovat do aplikace pomocí xml souborů. Aplikace byla navržena tak, aby ji bylo možné jednoduše rozšiřovat či aktualizovat. Při návrhu i realizaci této aplikace byl kladen velký důraz na zabezpečení a ochranu osobních dat. Žádná data nejsou volně přístupná z internetu. Obrázky grafů, které obsahují informace o zdraví pacientů, jsou generovány na základě identifikátoru lékaře. V případě že se tato data snaží zobrazit nepřihlášený uživatel, tak se mu nic nezobrazí.

Bylo vytvořeno aplikační rozhraní, které je co nejjednodušší a uživatelsky přívětivé. Ovládacích prvků je v aplikaci co nejméně, aby se v ní uživatel mohl lehce orientovat. Pro větší přívětivost byly v částech webu, kde to bylo vhodné, použity moderní zobrazovací prvky využívající technologii AJAX.

Administrační a uživatelské rozhraní je téměř totožné. Liší se od sebe pouze v množství ovládacích prvků. Možnosti běžného uživatele byly ořezány na minimum, neboli může pouze prohlížet záznamy, které mu administrátor přidělil. Administrátor má možností více. Může vytvářet, editovat a mazat pacienty, vyšetření nebo uživatele. Může přidělovat pacienty lékařům.

Aplikace je v této době nasazena na testovacím serveru a není tedy oficiálně spuštěna. Proto na ní také není použito šifrování SSL. Po oficiálním spuštění bude stačit pouze změnit nastavení zpracování odkazů pomocí Nette na bezpečnou verzi. Finální verze aplikace byla otestována a je k dispozici právě na testovacím serveru i na CD-ROM, které je přiloženo u bakalářské práce. Součástí CD-ROM je rovněž webový server XAMMP a soubor SQL příkazů pro přípravu databáze.

CONCLUSION

The result of this work is functional web application, which is used to display the results of hearing tests. These tests can be import into application using XML files. The application was designed for easy expand or update. During the design and implementation of this application was a great emphasis on security and privacy. No data are free accessible from the Internet. Pictures of graphs, which contain information about the health of patients, are generated by a doctor's identifier. In case that Anonymous is trying to see data, nothing is displayed.

Was created an application interface that is simple and user friendly. In the application are only few controls for user easy navigation. In some parts of the site were used imaging elements, which are using AJAX technology.

Administration and user interface is almost identical. They differ from each other only in the number of controls. Ordinary user can only view records assigned to him by the administrator. The administrator has more options. He can create, edit and delete patients, users or investigations.

At this time the application is placed on test server and is not officially launched. Therefore is not used SSL encryption. After the official launch will only need to change the configuration of Nette links to the secure version. The final version was tested and is available now on test server and on CD-ROM which is included with thesis. On CD-ROM is also web server XAMMP and SQL file with commands to create a database.

SEZNAM POUŽITÉ LITERATURY

[1] Apache HTTP Server. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, [cit. 2010-05-21]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/Apache_HTTP_Server>.

[2] BRÁZA, Jiří. *PHP 5 : začínáme programovat*. První vydání. Praha : Grada Publishing, a.s., 2005. 244 s. ISBN 80-247-1146-X.

[3] HTTPS. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, [cit. 2010-05-21]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/HTTPS>.

[4] KOSEK, Jiří. *PHP a XML*. První vydání. Praha : Grada Publishing, a.s., 2009. 367 s. ISBN 978-80-247-1116-4.

[5] LEJSKA, M., et al. *Základy praktické audiologie a audiometrie*. 1. vyd. Brno : IDVPZ, 1994. 171 s. ISBN 80-7013-178-0.

[6] Mysql. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, [cit. 2010-05-22]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/Mysql>.

[7] Nette Foundation. *Nette Framework* [online]. 2010 [cit. 2010-05-12]. Dokonalé zabezpečení webových aplikací. Dostupné z WWW: <http://nette.org/cs/hlavni-prednosti/bezpecnost>.

[8] Nette Foundation. *Nette Framework* [online]. 2010 [cit. 2010-05-12]. Model-View-Presenter (MVP). Dostupné z WWW: <http://doc.nette.org/cs/model-view-presenter>.

[9] Nette Foundation. *Nette Framework* [online]. 2010 [cit. 2010-05-12]. Nette\Application\Presenter. Dostupné z WWW: <http://doc.nette.org/cs/nette-application-presenter>.

[10] PHP. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, [cit. 2010-05-13]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/PHP>.

[11] POGOLOTTI, Jean-Damien. *PChart : a PHP Charting library* [online]. 2008 [cit. 2010-05-13]. Dostupné z WWW: <http://pchart.sourceforge.net/>.

[12] SSL. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, [cit. 2010-05-12]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/SSL>.

[13] Xml. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, [cit. 2010-05-12]. Dostupné z WWW: <http://cs.wikipedia.org/wiki/Xml>.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

XML	Extensible Markup Language.
XHTML	Extensible Hypertext Markup Language
W3C	World Wide Web Consortium
SVG	Scalable Vector Graphics
RIA	Rich Internet Application
RSS	Really Simple Syndication
WWW	World Wide Web
AJAX	Asynchronous JavaScript and XML
BSD	Berkeley Software Distribution
NCSA	National Center for Supercomputing Applications
PHP	Hypertext Preprocessor
XAML	Extensible Application Markup Language
HTTP	Hypertext Transfer Protocol
SNMP	Simple Network Management Protocol
FTP	File Transfer Protocol
IMAP	Internet Message Access Protocol
POP3	Post Office Protocol
OOP	Object-oriented programming
MVC	Model–view–controller
MVP	Model–view–presenter
XSS	Cross-site scripting
CSRF	Cross-site request forgery
URL	Uniform Resource Locator
LAMP	Linux, Apache, MySQL, PHP

SEZNAM OBRÁZKŮ

SEZNAM TABULEK