

# Databáze Ramanových spekter

Bc. Jiří Barák

---

Diplomová práce  
2014



Univerzita Tomáše Bati ve Zlíně  
Fakulta aplikované informatiky

---

# ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Bc. Jiří Barák**  
Osobní číslo: **A12452**  
Studijní program: **N3902 Inženýrská informatika**  
Studijní obor: **Informační technologie**  
Forma studia: **kombinovaná**

Téma práce: **Databáze Ramanových spekter**

## Zásady pro vypracování:

- 1. Seznamte se se základy a aplikačními možnostmi Ramanovy spektroskopie, popište výstupy měření tj. Ramanova spektra.**
- 2. Zhodnoťte význam spektrálních knihoven.**
- 3. Navrhněte uživatelsky vhodné prostředí a realizujte funkční databázi spektrálních informací pro identifikaci látek.**
- 4. Umožněte vyhledávání a filtraci dat podle předem zadaných kritérií.**
- 5. Naplňte databázi zkušebními daty a otestujte všechny její funkce.**

Rozsah diplomové práce:

Rozsah příloh:

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

Seznam odborné literatury:

1. **SCHMITT, M. aj. POPP. Raman spectroscopy at the beginning of the twenty-first century. Journal of Raman spectroscopy. 37(1-3), 20-28. 2006. ISSN: 1097-4555.**
2. **COLTHUP, N. B., DALY, L. H., WIBERLEY, S. E. Introduction to infrared and Raman spectroscopy. 3rd ed. Boston: Academic Press, 1990, 547s. ISBN 01-218-2554-X.**
3. **HĚL, Radovan. Průzkum databází Ramanových spekter. Zlín, 2011. Diplomová práce. UTB ve Zlíně, Fakulta aplikované informatiky.**
4. **VRÁNA, Jakub. 1001 tipů a triků pro PHP. Vyd. 1. Brno: Computer Press, 2010. ISBN 978-80-251-2940-1.**
5. **KOFLER, Michael a Bernd ÖGGL. PHP 5 a MySQL 5: průvodce webového programátora. Vyd. 1. Brno: Computer Press, 2007, 456 s. ISBN 978-80-251-1813-9.**
6. **PONKRÁC, Miloslav a Bernd ÖGGL. PHP a MySQL: bez předchozích znalostí : [průvodce pro samouky]. Vyd. 1. Brno: Computer Press, 2007, 712 s. Encyklopedie webdesignera. ISBN 9788025117583.**
7. **KOFLER, Michael a Bernd ÖGGL. Mistrovství v MySQL 5: bez předchozích znalostí : [průvodce pro samouky]. Vyd. 1. Brno: Computer Press, 2007, 805 s. Encyklopedie webdesignera. ISBN 978-80-251-1502-2.**

Vedoucí diplomové práce:

**Mgr. Hana Vašková**

Ústav elektroniky a měření

Datum zadání diplomové práce:

**21. února 2014**

Termín odevzdání diplomové práce:

**20. května 2014**

Ve Zlíně dne 21. února 2014

prof. Ing. Vladimír Vašek, CSc.

*děkan*



doc. Mgr. Roman Jašek, Ph.D.

*ředitel ústavu*

## **ABSTRAKT**

Hlavním cílem diplomové práce bylo vytvořit funkční databázi Ramanových spekter, která bude umožňovat vyhledávání podle typických charakteristik uložených spekter a napomáhat tak snazší a rychlejší práci s experimentálními daty a jejich vyhodnocením. V teoretické části je objasněn základní princip Ramanovy spektroskopie a jsou zde také popsány technologie použité při tvorbě databáze. Praktická část obsahuje popis všech možností aplikace a vysvětlení funkčních principů klíčových částí.

Klíčová slova:

PHP, databáze MySQL, framework Nette, Ramanovo spektrum, spektrální analýza

## **ABSTRACT**

The main topic of this thesis was to create a functional database of Raman spectra, which will be searchable by typical values of Raman spectras. In the theoretical part is explained the basic principle of Raman spectroscopy. There are also described technology used in the creation of the database. The practical part contains a description of all possible applications and explanations of the key principles of functional parts.

Keywords:

PHP, MySQL database, Nette framework, Raman spectrum, spectral analysis

Na tomto místě bych rád poděkoval mé vedoucí diplomové práce Mgr. Haně Vaškové za její podnětné připomínky, ochotu při konzultacích, podporu a trpělivost při psaní této diplomové práce.

Prohlašuji, že odevzdaná verze bakalářské/diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

### **Prohlašuji, že**

- beru na vědomí, že odevzdáním diplomové/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 diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/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 diplomovou/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 – diplomovou/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 diplomové/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 diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/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 diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....  
podpis diplomanta

# OBSAH

<b>ÚVOD</b> .....	<b>9</b>
<b>I TEORETICKÁ ČÁST</b> .....	<b>10</b>
<b>1 RAMANOVA SPEKTROSKOPIE</b> .....	<b>11</b>
1.1 PRINCIP RAMANOVA JEVU .....	12
1.1.1 Rayleighův rozptyl .....	13
1.1.2 Neelastický Ramanův rozptyl.....	13
1.2 VÝVOJ METOD A PŘÍSTROJŮ RAMANOVY SPEKTROSKOPIE.....	15
1.3 INTERPRETACE NAMĚŘENÝCH SPEKTER.....	17
1.4 OBLASTI VYUŽÍVAJÍCÍ RAMANOVU SPEKTROSKOPÍ.....	17
1.5 TRENDY VÝVOJE RAMANOVY SPEKTROSKOPIE.....	18
1.6 SPEKTRÁLNÍ KNIHOVNY.....	19
<b>2 PROGRAMOVÉ VYBAVENÍ</b> .....	<b>20</b>
2.1 POUŽITÉ TECHNOLOGIE PŘI TVORBĚ .....	20
2.2 PHP.....	21
2.2.1 Vývoj jazyka PHP .....	21
2.2.2 Výhody a nevýhody PHP.....	24
2.2.3 PHP vs. JAVA.....	25
2.2.4 PHP vs. ASP.....	26
2.2.5 PHP vs. Cold Fusion .....	26
2.2.6 Podpora databází v PHP .....	27
2.3 PHP FRAMEWORK .....	27
2.3.1 Přehled PHP frameworků.....	28
2.3.2 Testování PHP frameworků .....	29
2.4 PHP FRAMEWORK NETTE .....	31
2.4.1 Vývoj.....	31
2.4.2 Architektura MVP.....	32
2.4.3 Presenter a jeho životní cyklus.....	32
2.4.4 Model .....	34
2.4.5 View .....	34
2.4.6 Bezpečnost v Nette.....	34
2.4.6.1 SQL Injection.....	34
2.4.6.2 Cross-Site Scripting (XSS).....	34
2.4.6.3 Cross-Site Request Forgery (CSRF).....	35
2.4.6.4 URL attack.....	35
2.4.6.5 Session hijacking a podobně.....	35
2.4.7 Nette database .....	36
2.4.8 NotORM v Nette.....	36
2.4.9 Composer v Nette.....	36
2.5 DATABÁZE .....	37
2.5.1 Dělení databází.....	37

2.6	DATABÁZE MYSQL .....	38
2.6.1	Historie MySQL .....	38
2.6.2	Architektura MySQL .....	39
2.6.3	MySQL Workbench .....	40
<b>II</b>	<b>PRAKTICKÁ ČÁST .....</b>	<b>42</b>
<b>3</b>	<b>DATABÁZE RAMANOVÝCH SPEKTER .....</b>	<b>43</b>
3.1	POŽADAVKY NA FUNKČNOST APLIKACE .....	43
3.2	NÁVRH DATABÁZE PRO ULOŽENÍ DAT .....	44
3.3	SCHÉMA VÝSLEDNÉ DATABÁZE .....	44
3.3.1	Relace mezi tabulkami .....	44
3.3.2	Popis tabulky data .....	45
3.3.3	Popis tabulky minerals .....	47
3.3.4	Popis tabulky users .....	48
3.3.5	Popis tabulky category .....	50
<b>4</b>	<b>ADMINISTRACE DATABÁZE .....</b>	<b>51</b>
4.1	VNITŘNÍ STRUKTURA ADMINISTRAČNÍHO PROSTŘEDÍ .....	51
4.2	PŘIHLÁŠENÍ DO ADMINISTRACE .....	52
4.3	SKUPINY .....	55
4.4	VZORKY .....	57
4.5	VYTVORENÍ VZORKU .....	61
4.6	HLEDÁNÍ V DATABÁZI .....	64
4.7	UŽIVATELÉ .....	67
4.8	UMÍSTĚNÍ APLIKACE NA HOSTINGOVÝ SERVER .....	68
	<b>ZÁVĚR .....</b>	<b>69</b>
	<b>SEZNAM POUŽITÉ LITERATURY .....</b>	<b>71</b>
	<b>SEZNAM POUŽITÝCH ZDROJŮ OBRÁZKŮ</b> CHYBA! ZÁLOŽKA NENÍ DEFINOVÁNA.	
	<b>SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK .....</b>	<b>74</b>
	<b>SEZNAM OBRÁZKŮ .....</b>	<b>75</b>
	<b>SEZNAM TABULEK .....</b>	<b>77</b>
	<b>SEZNAM PŘÍLOH .....</b>	<b>78</b>

## ÚVOD

Téma této diplomové práce jsem si vybral z důvodu zájmu o principy spektrální analýzy. Pro úspěšné vytvoření této práce jsem se musel na začátku seznámit s principy Ramanovy spektroskopie, o které jsem neměl téměř žádné znalosti. Proto je i teoretická část věnována základním principům, které jsou elementární pro pochopení této experimentální metody, možností jejího využití i další nastavy v podobě funkční spektrální knihovny.

U samotné tvorby databáze jsem se rozhodl vyzkoušet český Framework Nette a to z důvodu jeho rapidním tempem vzrůstající popularity. I u této části jsem musel začít studiem základů frameworku i jazyka PHP.

V teoretické části jsem čtenáři přiblížil základní principy Ramanovy metody a s ní souvisejícího neelastického rozptylu u molekul. Uvádím její klady i zápory, také oblasti, ve kterých se využívá, počátky jejího vzniku i směr, kterým by se mohla v budoucnu ubírat. Objasňuji zde princip funkce moderních spektrometrů. Další část teorie se zabývá vznikem a vlastnostmi použitých technologií, se kterými jsem se během tvorby musel sám seznámit. Jsou zde popsány vlastnosti jazyka PHP, jeho vývoj až k dnešku. Část o frameworku Nette je zaměřena na jeho celkové vlastnosti a je zde zdůvodněna volba jeho použití při tvorbě. Obsažen je zde i test několik frameworků, ze kterého jsou vidět kvality použitého frameworku Nette. V poslední části teorie popisuji i další využití knihovny a pomocné nástroje.

V praktické části se zaměřuji na tvorbu a funkčnost samotné databázové aplikace. Jsou zde popsány klíčové vlastnosti a principy jejich funkce. Tato část je doplněna i názornými obrázky, aby případný uživatel měl co nejmenší problémy s obsluhou a využíváním všech poskytovaných možností této aplikace.

## **I. TEORETICKÁ ČÁST**

## 1 RAMANOVA SPEKTROSKOPIE

Spektroskopie je vědní obor zabývající se měřením emise a absorpce různých vlnových délek viditelných i neviditelných záření. Spektroskopie je často využívána k analyzování substancí na základě emitovaného nebo absorbovaného spektra.

Spektroskopické metody poskytují informace o struktuře na molekulové a atomové úrovni. Každá látka má svou jedinečnou charakteristiku spektra a podle něj může být identifikována.

Chandrasekhara Venkata Raman se narodil ve městě Tiruchirappalli na jihu Indie 7. listopadu 1888. Jeho otec byl učitelem matematiky a fyziky, tudíž už od malička vyrůstal v akademické atmosféře. Ještě během studií začal s pokusy v oblasti optiky a akustiky, kterým pak zasvětil celý život. [1]

V roce 1928, již profesor, Raman společně s K.S. Krishanem popsali jev neelastického optického rozptylu, který je základem metody Ramanovy spektroskopie. [2]

Za tento objev obdržel Chandrasekhara Venkata Raman v roce 1930 Nobelovu cenu.



Obr. 1 - C. V. Raman [3]

Tato metoda vibrační molekulové spektrometrie se používá při analýze pevných látek (krystalické i amorfní materiály, kovy, polovodiče, polymery atp.), kapalin (čisté látky, roztoky vodné i nevodné), plynů, dále též při analýze povrchů (např. sorbenty, elektrody, senzory) či při analýze biologických systémů (od biomolekul až po organismy). Své uplatnění Ramanova spektroskopie nachází od mineralogie a geochemie, přes chemický a farmaceutický průmysl až po medicínu či umění.

## 1.1 Princip Ramanova jevu

Ramanův jev je neelastický rozptyl monochromatického záření. Během tohoto děje dochází k přenosu energie mezi fotony a molekulou tak, že rozptýlený foton má vyšší nebo nižší energii než původní foton. Rozdíly energií způsobené vibračními a rotačními pohyby molekuly po interakci s fotonem poskytují charakteristické informace o zkoumané látce. [4]

Jedná se o dvoufotonový přechod mezi dvěma stacionárními vibračními stavy molekuly, jejichž energie jsou  $E_1$  a  $E_2$ , vyvolaný interakcí s fotonem dopadajícího záření o frekvenci

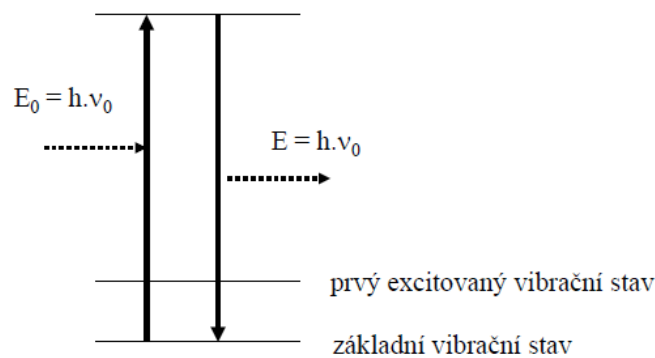
$$f_0 \geq \frac{|E_2 - E_1|}{h} \quad (1)$$

kde  $h$  je Planckova konstanta, a provázený vyzářením fotonu rozptýleného záření o frekvenci  $f_r$ . Tento rozptylový efekt si lze zjednodušeně představit jako současnou absorpci fotonu budícího záření molekulou, kdy molekula přechází na virtuální energetickou hladinu, a emisi sekundárního fotonu, za splnění podmínky zachování energie. [2]

Jinými slovy laserový paprsek může být považován za oscilační elektromagnetické vlny s elektrickým vektorem  $E$  (vektor intenzity elektrického pole dopadajícího záření). Při interakci se vzorkem, se indukuje elektrický dipólový moment (vektorová veličina popisující nesymetrické rozdělení elektrického náboje), který deformuje molekuly. Tyto periodické deformace způsobí vibrace molekul s charakteristickou frekvencí  $f_m$ . Amplitudy těchto vibrací se nazývají jaderný posun. Monochromatické laserové světlo s frekvencí  $f_0$  vybudí molekuly a přemění je na oscilující dipóly. Takové oscilující dipóly vyzařují světlo o určité frekvenci, podle které se dělí různé druhy rozptylu. [2]

### 1.1.1 Rayleighův rozptyl

$$\Delta E = E_0 - E = 0$$

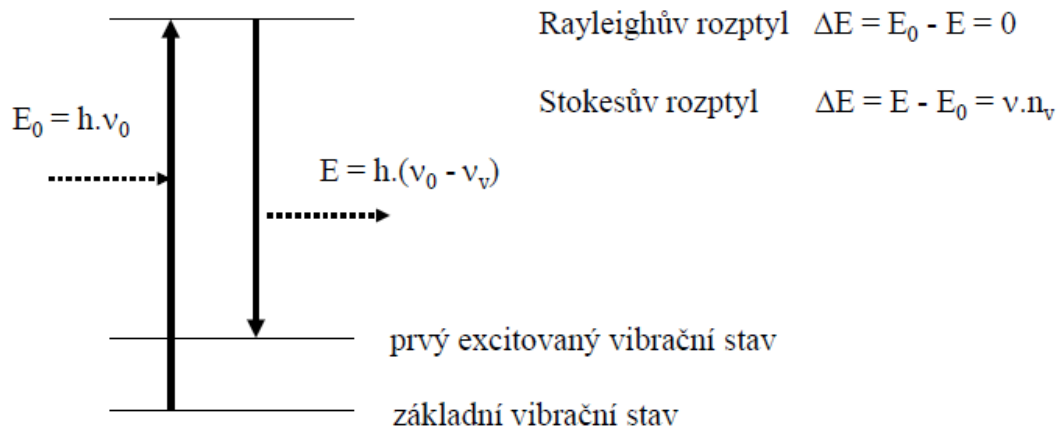


Obr. 2 – Rayleighův rozptyl [2]

Molekula je excitována fotonem ( $E_0 = hf_0$ ) na virtuální energetickou hladinu. Molekula se vrací do základního stavu, přičemž emituje stejné množství energie ( $E_0 = hf_0$ ).

### 1.1.2 Neelastický Ramanův rozptyl

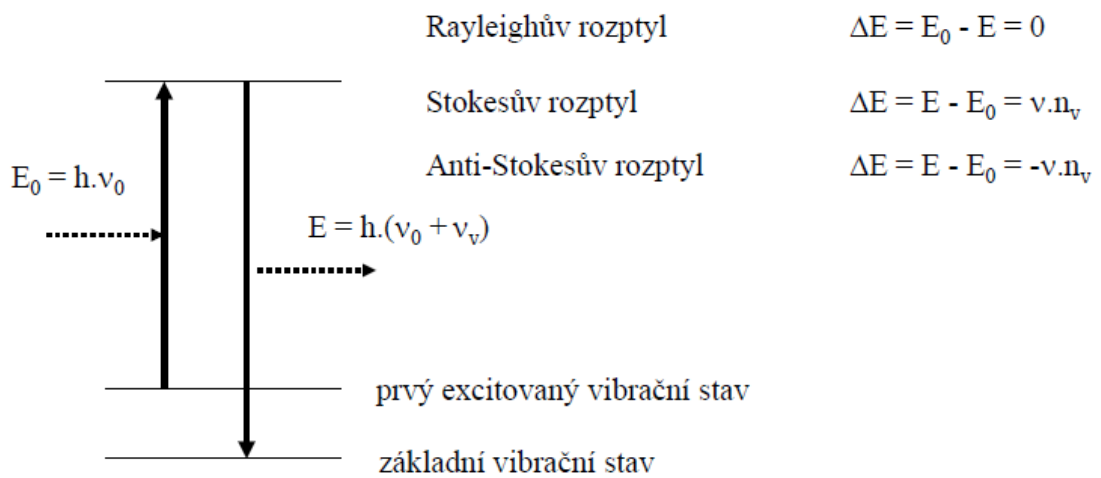
#### Stokesův rozptyl



Obr. 3 – Stokesův rozptyl [2]

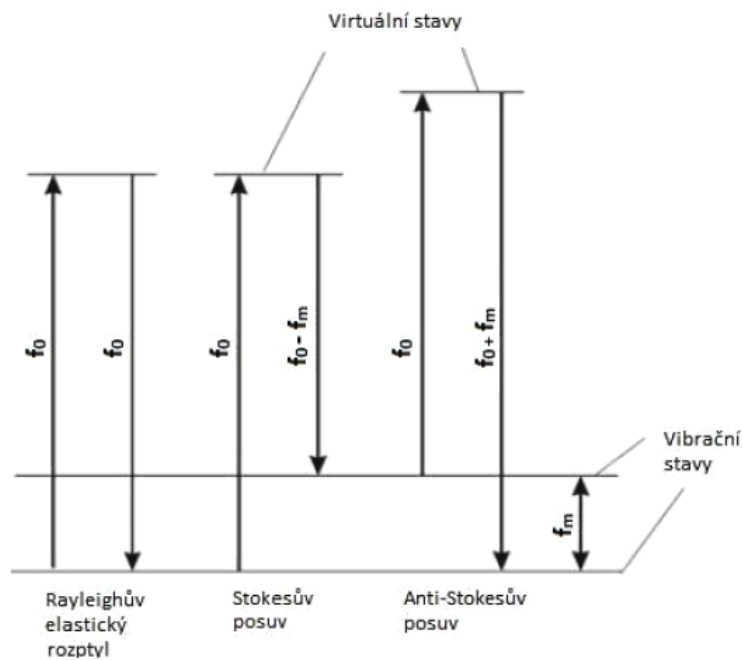
Foton s frekvencí  $f_0$  je absorbován aktivní Ramanovou molekulou, která je v době interakce v základním stavu. Část energie fotonu je převedena na aktivní Ramanův mód s frekvencí  $f_m$  a výsledná frekvence rozptýleného světla je pak snížena na hodnotu  $f_0 - f_m$ . [2]

Anti-Sokesův rozptyl



Obr. 4 – Anti-Sokesův rozptyl [2]

Foton s frekvencí  $f_0$  je absorbován aktivní Ramanovou molekulou, která je v době interakce už ve vybuzeném vibračním stavu. Přebytečná energie vybuzeného aktivního Ramanova módu je uvolněna, molekula se poté vrací do základního stavu a výsledná frekvence rozptýleného světla vzroste na  $f_0 + f_m$ . [2]



Obr. 5 – Přejchody molekul mezi stavy [2]

## 1.2 Vývoj metod a přístrojů Ramanovy spektroskopie

Mezi základní prvky měřicích zařízení pro Ramanovu spektroskopii patří zdroje excitačního záření, sběrná optika, filtry a detektory záření, samozřejmostí je software na zpracování, vyhodnocení a ukládání dat. Díky technickým pokrokům ve zmíněných oblastech (laserová technika, efektivní filtry, velmi citlivé senzory...) současné přístroje mnohonásobně převyšují možnosti, které měla Ramanova spektroskopie ve svých začátcích a díky tomu se stává účinnější metodou.

Zásadním problémem Ramanovy spektroskopie je nízká intenzita záření po neelastickém rozptylu, což je způsobeno tím, že neelasticky se rozptýlí řádově 1 z  $10^6$  fotonů, a mnohem vyšší intenzita záření po rozptylu Rayleighově. Tento problém dlouho omezoval rozvoj Ramanovy spektroskopie.

Ve svých prvních pokusech použili Raman a Krishan filtrované sluneční záření a detekovali pomocí něj Ramanovy linie pro desítky kapalin a plynů. Pozorovali rozptýlené světlo vizuálně a používali barevné filtry pro lepší citlivost.

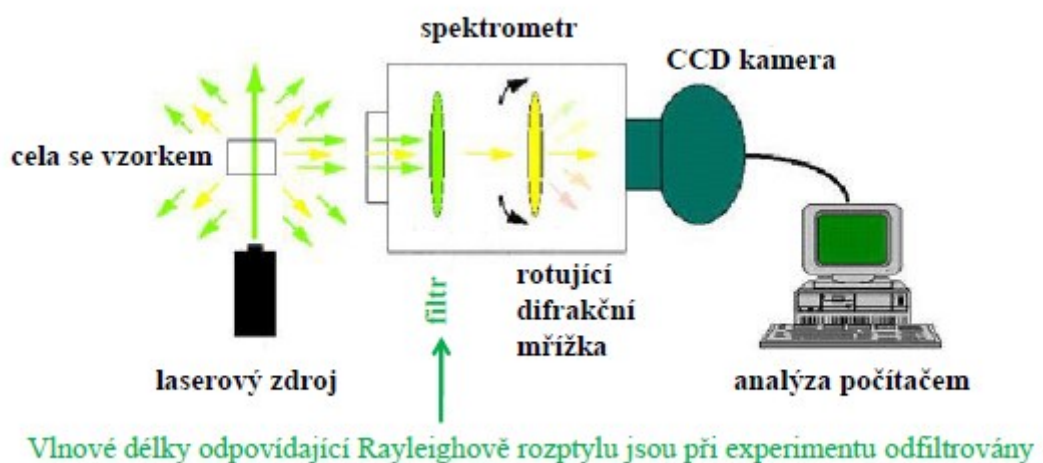
První spektrum chloridu uhličitého se Stokesovými i Anti-Stokesovými liniemi bylo zaznamenáno fotograficky za použití 435,83 nm Hg excitace a bylo publikováno v r. 1929. Logicky se úsilí zaměřilo na zlepšení zdrojů záření. Hg lampy s monochromatickými filtry se používaly v době kolem r. 1930. Později byla zavedena tzv. rtuťová oblouková lampa „Toronto“. Avšak pouze zavedení laseru v r. 1960 umožnilo rozvoj metody. Výhoda byla v soustředění paprsku na malý vzorek, která přinesla zlepšení kvality spekter a možnost měření mikrogramových vzorků. [4]

Od 1970 se používají Argonové lasery pro excitaci linií 488 nm a 514.5 nm. Kr<sup>+</sup>, He-Ne, Cd a rubínové lasery se používají také. Od zavedení Fourierovy transformace Ramanovy spektroskopie v r. 1986 se zde většinou používá Nd:YAG laser s linií 1064 nm.

Ramanova spektroskopie měla vždy problém s elastickým rozptylem, zvláště při zkoumání linií u krátkých Ramanových posunů. Problém byl vyřešen tím, že monochromátor byl opatřen dvěma nebo třemi stupni. První komerčně dostupný dvojestupňový monochromátor byl na trhu r. 1940. Dodnes se používá dvojnásobných a trojnásobných monochromátorů. Tyto monochromátory eliminují Rayleighův rozptyl o 10 i více řádů u Ramanových posunů o několik  $\text{cm}^{-1}$ . [4]



Obr. 6 – Ramanův spektrometr značky Renishaw [5]



Obr. 7 – Uspořádání Ramanova spektrometru [6]

Princip u dnešních spektrometrů popisuje obrázek výše. Vzorek je osvětlen paprskem laseru. Rozptýlené záření proniká do analyzátoru. Filtr odstraní záření o stejné vlnové délce, jako má laserový paprsek a ponechá jen záření odlišné vlnové délky s Ramanovým posunem. Difrakční monochromátor rozdělí záření podle vlnové délky a elektrooptické zařízení je převede na signál, který analyzuje počítač. Výsledkem je graf zobrazující intenzitu záření u každé vlnové délky. [4]

### 1.3 Interpretace naměřených spekter

Interpretace vibračních spekter řeší dva neoddělitelné problémy:

- Přiřazení absorpčních a rozptylových pásů jednotlivým normálním vibračním molekuly.
- Charakterizace normálních vibrací určením symetrie a spřažených vnitřních souřadnic.

Při rozboru vibračního problému musíme analyzovat všechny normální vibrační mody.

Užívá se tzv. koncepce charakteristických vlnočtů funkčních skupin, která umožňuje přiřadit některé význačné pásy ve spektru. Tato metoda si ovšem žádá velkou zkušenost experimentátora. Absorpční pásy mající vrcholy v intervalu 4000–1500  $\text{cm}^{-1}$ , jsou vhodné pro identifikaci funkčních skupin (např.  $-\text{OH}$ ,  $\text{C}=\text{O}$ ,  $\text{N}-\text{H}$ ,  $\text{CH}_3$  aj.). Pásy v oblasti 1500–400  $\text{cm}^{-1}$  se nazývají oblastmi „otisku palce“ (fingerprint region). Pomocí „vyhledávacích programů“ a digitalizovaných knihoven Ramanových spekter je možno identifikovat neznámou analyzovanou látku. Dodatečné informace pro identifikaci látky je možno získat i s infračervené spektroskopie. [7]

### 1.4 Oblasti využívající Ramanovu spektroskopii

Ramanova spektroskopie nachází využití v mnoha oblastech, zde je uvedeno několik z nich:

- **Farmaceutický průmysl** - Ramanova spektroskopie plní důležitou úlohu v mnoha analytických fázích farmaceutického designu výrobku a výrobního procesu. Aplikace zahrnují sledování a řízení rozsáhlých výrobních procesů, k profilování distribuce aktivních farmaceutických látek a pomocných látek v různých fázích ve formulaci cyklu. [8]
- **Materiálové inženýrství** - Ramanova spektroskopie má zásadní význam pro určování a strukturální analýzy téměř všech druhů materiálů (amorfni, částečně krystalické, průhledné, neprůhledné a vzorky s různými povrchovými strukturami). [8]

- **Forenzní vědy a kriminalistika** – díky své rychlosti, nedestruktivní povaze a možnosti bezdotykové analýzy vzorku přímo na místě. Ramanova spektroskopie nachází uplatnění i při odhalování padělků, drog, výbušnin a jiných důkazních materiálů. [9]
- **Výroba elektronických součástek** – zde se Ramanova spektroskopie používá ke zjištění kvality a množství nečistot polovodičů v silikonových směsích a různých uhlíkových vrstvách. Díky rychlosti této metody bývá využívána i k odhalení povrchových vad na materiálech a tím dochází k finančním úsporám při výrobě. [10]

## 1.5 Trendy vývoje Ramanovy spektroskopie

V posledních letech nastal „boom“ přenosných spektrometrů využívajících Ramanovu metodu. Ruční Ramanovy spektrometry se již masivně prosadily v oblasti identifikace látek, zejména ve farmacii, při výrobě léků, ale i rychlé a spolehlivé identifikaci padělků léčiv, identifikaci metanolu v etanolu, v mnoha oborech zajišťující bezpečnost, při detekci výbušnin.

Dosud však Ramanova spektroskopie ve svém standardním provedení (532/785 nm) značně trpí u některých látek fluorescencí, což zásadně prodlužuje měřicí čas (z jednotek vteřin až na hodiny), nebo dokonce zcela znemožňuje analýzu některých látek Ramanovou technologií. Problém fluorescence je nyní odstraněn použitím ručního spektrometru s novou technologií laseru o vlnové délce 1064 nm. Tyto nové Ramanovy spektrometry nyní předepisují přední světové farmaceutické společnosti jako jedinou přípustnou měřicí metodu pro výše popsané účely. [11]



Obr. 8 – Přenosný Ramanův spektrometr s 1064 nm laserem [12]

## 1.6 Spektrální knihovny

V současné době je možno pracovat s komerčními i volně dostupnými spektrálními knihovnami, které obsahují stovky, v některých případech i tisíce naměřených Ramanových spekter mnoha různých látek. Význam budování těchto knihoven spočívá v možnostech jednoduchého porovnávání spekter naměřených na neznámých vzorcích s těmi uloženými v knihovně na základě unikátních charakteristik, jak již bylo uvedeno v kapitole 1.1. Princip Ramanova jevu. Tyto knihovny již dnes bývají běžně implementovány v samotných spektrometrech, kde jsou automaticky naměřená Ramanova spektra porovnávána pomocí speciálních na míru vyvinutých softwarů a výsledky bývají ihned zobrazovány na displeji spektrometru.

## 2 PROGRAMOVÉ VYBAVENÍ

Obsahem teoretické části diplomové práce je zdůvodnění výběru jednotlivých nástrojů použitých při tvorbě webové databáze Ramanových spekter látek. Dále jsou zde popsány nejdůležitější vlastnosti těchto nástrojů a jejich funkce.

Jelikož je cílem diplomové práce zpracování a vytvoření webové aplikace s databází, je nutno použít vhodné nástroje k tvorbě jednotlivých funkčních částí. Webová aplikace se dá obecně rozložit na tři nejdůležitější bloky, a to zdrojový kód samotných webových stránek aplikace a databáze pro uložení dat.

### 2.1 Použité technologie při tvorbě

Ve světě existuje mnoho technologií k tvorbě internetových stránek a aplikací. Rovněž existuje i spousta softwarových nástrojů pro tvorbu a správu databázových nástrojů.

Pro tvorbu zdrojových kódů internetových stránek je v současné době v celosvětovém měřítku nejrozšířenější programovací jazyk PHP, dnes ve verzi 5.0, proto byl jasnou volbou pro vizualizaci, prezentaci dat a interakci s uživatelem. Nicméně dnešní aplikace a webové stránky jsou již málokdy psány přímo v jazyce PHP, ale čím dál častěji je využíván pro tvorbu zdrojových kódů. Nebylo tomu jinak ani u této aplikace, kdy byl využit framework Nette. Tento framework byl použit hlavně z důvodu, že se jedná o velmi rozšířený open source domácí projekt.



Obr. 9 - Logo Nette Framework [13]

Pro potřeby tvorby relační databáze, určené pro uložení naměřených hodnot a dalších potřebných údajů, byla technologie MySQL. A to díky své rychlosti a open licenci, z toho důvodu ji lze považovat za nejoblíbenější systém současnosti. Pro správu a tvorbu databází MySQL existuje řada aplikací, asi nejpoužívanější a zároveň nejjednodušší variantou je

použití aplikace phpMyAdmin. Nicméně práce s touto aplikací není příliš efektivní při časté správě databáze, v tomto případě se vyplatí použít nějakou z mnoha desktopových aplikací jako například Adminer, SQL manager, MySQL Front a MySQL Workbench. Poslední jmenovaný nástroj byl použit při tvorbě databáze pro tuto aplikaci z důvodu jeho existence pod open licenci.

Zároveň zde byla při tvorbě využita řada dalších menších pomocných aplikací, jako jsou Composer, Notorm a Nette database.

Použité výše uvedené technologie jsou popsány v následujících kapitolách.

## 2.2 PHP

PHP Hypertext Preprocessor je skriptovací jazyk běžící na straně serveru (server-side) speciálně navržený pro potřeby webové stránky. PHP je Open Source.

### 2.2.1 Vývoj jazyka PHP

PHP takové jak jej známe dnes, bylo původně vyvíjeno pod produktovým názvem PHP/FI. Vytvořil jej v roce 1997 Rasmus Lerdorf. Úplně první inkarnací PHP byla jednoduchá knihovna funkcí napsaná v programovacím jazyku C. Tuto knihovnu poprvé použil ke sledování návštěvnosti stránek s jeho životopisem, tehdy tuto skupinku skriptů nazval „Personal Home Page Tools“. Tento název se dále ujal jako „PHP Tools“. Postupně Lerdorf přidával další funkce a přepsal „PHP Tools“ do mnohem větší implementace. Tento model PHP již uměl jednodušší spolupráci s databází a jeho pomocí mohli již uživatelé tvořit menší aplikace. [14]

V roce 1995 Rasmus Lerdorf kód PHP vypustil na veřejnost, kdy vyzval veřejnost k užívání. Tito uživatelé mu následně pomohli s odstraněním chyb. V září téhož roku Lerdorf rozšířil funkčnost a změnil název na FI (formulářový interpretor), zde již byly obsaženy některé nové funkcionality PHP, jak je známe dnes. Během roku Lerdorf postupně rozšiřoval funkcionalitu FI, nakonec úplně přepsal zdrojový kód a vrátil se i k původnímu názvu PHP.

V roce 1996 se kód PHP dočkal opětovné úplné přeměny, kdy byly doplněny plné podpory pro různé druhy databází, cookies, uživatelem definované funkce a mnoho dalšího.

V červnu 1996 vyšla PHP verze 2.0. V letech 1997 a 1998 má PHP tisíce uživatelů po celém světě a do konce roku se tento údaj podle organizace Netcraft desetkrát navýšil.

Až verze PHP 3.0 úzce připomínala verzi PHP tak, jak ji známe dnes. Bohužel i v této podobě byl jazyk nedostatečný pro aplikace elektronického obchodu a tak v roce 1997 Andi Gutmans a Zeev Suraski začali spolu s Lerdorfem další kompletní přepsání základního kódu. Tento zcela nový jazyk byl uvolněn pod názvem PHP, jedná se tedy již o zkratku, jak ji známe dnes, „Hypertext Preprocessor“. Jednou z největších předností PHP 3.0 byly jeho obrovské možnosti rozšíření. Kromě toho poskytuje koncovým uživatelům vyspělé rozhraní pro více databází, protokolů a API. Snadné rozšíření jazyka přilákalo desítky vývojářů, kteří dodali řadu modulů. Po zhruba devíti měsících otevřeného veřejného testování, kdy oznámení o oficiálním vydání PHP 3.0 přišlo, bylo již nainstalováno na více než 70.000 doménách po celém světě. V době svého vrcholu bylo PHP 3.0 instalováno na přibližně 10 % webových serverů na Internetu. [14]

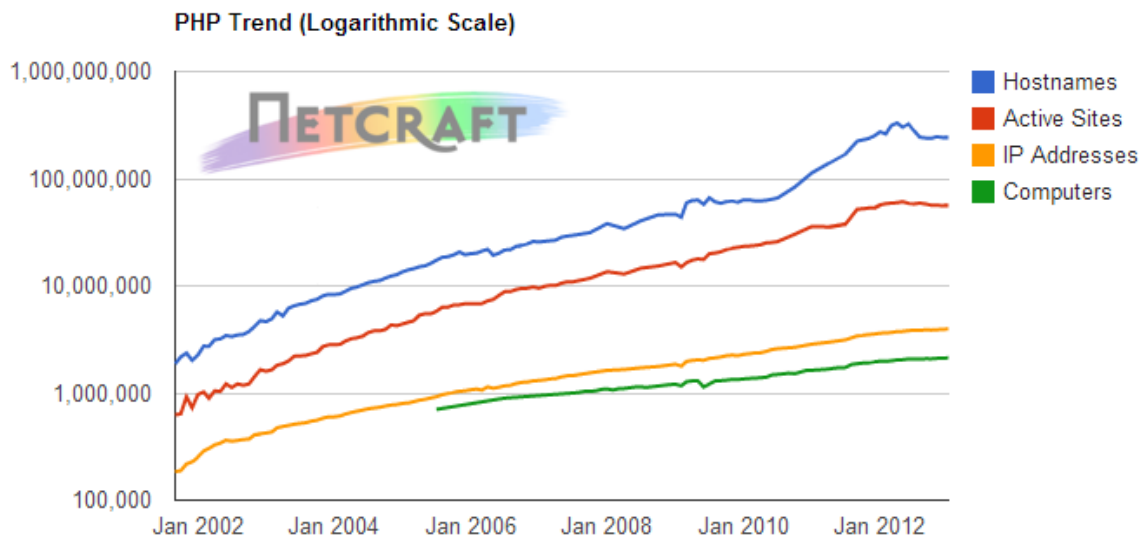
V zimě roku 1998, krátce poté, co PHP 3.0 bylo oficiálně uvolněno, Andi Gutmans a Zeev Suraski začali pracovat na přepsání jádra PHP. Cílem návrhu bylo zvýšit výkon pro složité aplikace a zlepšit modularitu kódové báze PHP. Tyto aplikace byly schopny pracovat s PHP 3.0, s novými funkcemi a podporu pro širokou škálu databází a API od třetích stran, ale PHP 3.0 nebylo navrženo pro práci tak náročných aplikací efektivně. Nový engine, nazvaný Zend Engine (sestaven z jejich křestních jmen, Zeev a Andi ), splnil cíle návrhu úspěšně a byl poprvé představen v polovině roku 1999. PHP 4.0, založené na tomto enginu a doplněné širokou škálou dalších nových funkcí, bylo oficiálně uvolněno v květnu 2000, téměř dva roky po svém předchůdci. Kromě vysoce zvýšenému výkonu této verze, přidává PHP 4.0 další klíčové prvky, jako je podpora pro mnoho WWW serverů, HTTP sessions, buffering výstupu, bezpečnější způsoby zpracování vstupů uživatele a mnoho nových jazykových konstrukcí. [14]

PHP 5 byl uvolněn v červenci 2004 po dlouhém vývoji a řadě předběžných verzí. To je způsobeno především jeho jádrem, Zend Engine 2.0, s novým objektovým modelem a desítkou dalších nových funkcí.

Poslední verze PHP 5 se stále vyvíjí. I když jde jen odhad na základě statistik z minulých let, lze bezpečně předpokládat, že PHP je nainstalováno na desítkách či dokonce i stovkách milionů domén po celém světě. [14]

Řada	Verze	Datum vydání	Přehled některých zásadních změn
1.x	1.0	8. června 1995	Oficiální název „Personal Home Page Tools (PHP Tools)“. Poprvé byl použit název „PHP“.
2.x	2.0	16. dubna 1996	
3.x	3.0	6. června 1998	Rasmus Lerdorf, Zeev Suraski a Andi Gutmans přepsali celý základ pro tuto verzi.
4.x	4.0	22. května 2000	-
	4.1	10. prosince 2001	Byly představeny 'superglobalní proměnné' (\$_GET, \$_POST, \$_SESSION, etc.)
	4.2	22. dubna 2002	
	4.3	27. prosince 2002	
	4.4	11. srpna 2005	
5.x	5.0	13. července 2004	Zend Engine II s novým objektovým modelováním.
	5.1	24. listopadu 2005	Zlepšení výkonu zavedením kompilátoru proměnných v přepracovaném Engine PHP.
	5.2	2. listopadu 2006	Povolen filtr přípon ve výchozím nastavení.
	5.3	30. června 2009	jmenné prostory, oprava chyb, změny ve funkcích a doplňcích
	5.4	1. března 2012	traits, dereference polí, odstranění některých zastaralých funkcí a direktiv
	5.5	20. června 2013	operátor yield, blok finally pro ošetřování výjimek, označení extenze MySQL jako zastaralé
	5.6	není určeno	připravovaná verze
6.x	6.0	není určeno	připravovaná verze

Tabulka 1 – Přehled všech vydaných verzí jazyka PHP [14]



Obr. 10 - Obliba jazyka PHP v posledních letech [15]

## 2.2.2 Výhody a nevýhody PHP

### Výhody PHP

- PHP je specializované na webové stránky.
- Rozsáhlý soubor funkcí v základní knihovně PHP (přes pět a půl tisíce), další funkce v PECL.
- Nativní podpora mnoha databázových systémů.
- Multiplatformost (zejména Linux a Microsoft Windows).
- Možnost využití nativních funkcí operačního systému (možná nekompatibilita s jiným OS).
- Strmá křivka učení.
- Obrovská podpora na hostingových službách – PHP je fakticky standardem, který najdeme všude.
- Obrovské množství projektů a kódů, které lze zdarma využít (WordPress, phpBB a další).
- Poměrně slušná dokumentace.

- Velmi svobodná licence, která (v protikladu k např. GPL) neobsahuje copyleft. [16]

### **Nevýhody PHP**

- Jazyk PHP je definován pouze svou jedinou implementací.
- Nekonzistentní pojmenování funkcí.
- Nejednotné názvosloví skupin funkcí.
- Nejednotné pořadí parametrů.
- Ač jazyk podporuje výjimky, jeho knihovna se používá jen zřídka.
- Slabší podpora Unicode, pouze přes PHP knihovnu.
- Ve standardní distribuci chybí ladící (debugovací) nástroj.
- Po zpracování požadavku neudrhuje kontext aplikace, vytváří jej vždy znovu (oslabuje výkon). [16]

### **2.2.3 PHP vs. JAVA**

Programovací jazyky se dělí do skupin podle různých kritérií. Např. kompilované vs. interpretované, silně vs. slabě typované, staticky vs. dynamicky typované, objektové vs. procedurální. V těchto ohledech bylo porovnáno PHP s Javou.

PHP je skriptovací interpretovaný jazyk, tedy jeho skripty nasazujeme na server tak, jak je programátor napsal a zde se interpretují. (což ne všem vyhovuje – např. se bojí, že někdo jejich program ukradne – a tak se uchylují k různým obstrukcím, které znesnadňují čtení takových skriptů). S Javou je to trochu složitější. Zdrojový kód napsaný programátorem není možné interpretovat – je třeba ho nejdříve zkompilevat do tzv. bajtkódu (mezikód) a až ten se interpretuje pomocí JRE. Java se tedy zároveň kompiluje i interpretuje – oficiálně se pokládá za interpretovaný jazyk, protože její „zkompilevané“ třídy nejsou ve strojovém kódu a neběží přímo na procesoru – je nutné je interpretovat. Formálně tak oba jazyky spadají do stejné kategorie.

Statické typování znamená, že datový typ proměnné uvádíme už při její deklaraci. Tudíž typy jednotlivých proměnných známe už v době kompilace a díky tomu odhalíme řadu chyb dříve než v provozu – kompilátor takový program odmítne přeložit. U dynamicky typovaných jazyků dochází ke kontrole až při běhu aplikace. Díky tomu lze psát pružnější

(ovšem často taky méně přehledný) kód, ve kterém jedna proměnná nabývá hodnot různých datových typů. Java je staticky typovaný jazyk – všechny proměnné musíme deklarovat včetně jejich typu, nedeklarované proměnné nemůžeme používat. Zatímco PHP je jazyk dynamicky typovaný.

Čím se Java a PHP liší zásadněji, je standard a styl vývoje. Java se rozvíjí v rámci tzv. Java Community Process, ve kterém vzniká specifikace tohoto jazyka/platformy. Tato specifikace se následně implementuje – nejznámější je implementace od Sunu, ale svoji implementaci má i IBM (a vedle nich existuje i řada necertifikovaných implementací). Oproti tomu vývoj PHP je poněkud živelnější. Formální specifikace neexistuje a jako standard je třeba brát hlavní implementaci, která stojí na jádru Zend. [17]

#### 2.2.4 PHP vs. ASP

ASP ve skutečnosti není jazyk jako takový, je to zkratka pro Active Server Pages, jazyky nyní používanými k programování ASP jsou Visual Basic Script a JScript. Největší nevýhodou ASP je to, že se jedná o proprietární systém, který je nativně používán pouze na serveru Microsoft Internet Information Server (IIS). To omezuje jeho dostupnost na servery založené na Win32. Existuje několik projektů, které umožňují běh ASP v jiných prostředích a serverech: InstantASP od Halcyon (komerční), Chili!Soft ASP od Chili!Soft (komerční) a OpenASP od ActiveScripting.org (free). O ASP se tvrdí, že je pomalejší a těžkopádnější, a stejně tak i méně stabilní. Z výhod ASP lze uvést to, že primárně používá VBScript, který je poměrně snadno uchopitelný, pokud již víte, jak programovat ve Visual Basicu. Podpora ASP je také standardně zapnuta na IIS, takže se snadno spustí a běží. Komponenty zabudované v ASP jsou opravdu omezené, takže pokud potřebujete použít "pokročilé" prvky, jako interakce s FTP servery, musíte si koupit doplňující komponenty.[18]

#### 2.2.5 PHP vs. Cold Fusion

O PHP se často tvrdí, že je rychlejší a efektivnější pro složité programové úlohy a zkoušení nových myšlenek. PHP je obecně zmiňováno jako stabilnější a méně náročné na systémové prostředky. Cold Fusion má lepší zpracování chyb, databázovou abstrakci a parsování dat, ačkoliv databázová abstrakce je adresována v PHP 4. Jinou věcí, která je uváděna jako silný nástroj Cold Fusion, je jeho výborný vyhledávací engine, avšak s tím, že vyhledávací

engine není něco, co by mělo být součástí skriptovacího jazyka pro web. PHP běží na většině existujících platform; Cold Fusion je k dispozici pouze na Win32, Solarisu, Linuxu a HP/UX. Cold Fusion má dobré IDE a je obecně jednodušší pro začátky, zatímco PHP vyžaduje více programátorských znalostí. Produkt Cold Fusion je navržen s ohledem na neprogramátory, PHP je naopak zaměřeno na programátory. [18]

### 2.2.6 Podpora databází v PHP

Jedna z nejsilnějších a nejvýznamnějších vlastností PHP je jeho podpora pro širokou škálu databází. Vytvoření webové stránky spolupracující s databází je neuvěřitelně jednoduché. V současné době jsou podporovány následující databáze:

Adabas D	InterBase	PostgreSQL
dBase	FrontBase	SQLite
Empress	mSQL	Solid
FilePro (read-only)	Direct MS-SQL	Sybase
Hyperwave	MySQL	Velocis
IBM DB2	ODBC	Unix dbm
Informix	Oracle (OCI7 and OCI8)	
Ingres	Ovrimos	

Tabulka 2 – Přehled databází podporovaných PHP [14]

## 2.3 PHP framework

Pojem Framework se objevuje stále častěji. PHP je velmi jednoduchý jazyk, navíc se jedná o open source projekt, proto je velmi rozšířen. Během let užívání si programátoři osvojili určité techniky tvorby webu pomocí PHP. Ať už jde o vložení samotných stránek nebo konfigurace webového projektu. V poslední době se z těchto důvodů rozvíjí nový trend frameworků, které mají programátorům tyto opakované činnosti usnadnit. Framework se snaží najít novou cestu, poskytnout programátorovi pomoc, nový efektivnější způsob řešení a také ušetřit mnoho kódů, které by musel stále znova psát. Mezi nejznámější frameworky patří Ruby on Rails, ale tento Framework je určen pro jazyk Ruby. Nicméně spousta PHP frameworků v něm nachází inspiraci. [19]

PHP frameworky jsou v podstatě sadou skriptů v PHP, které pracují a fungují úplně stejně jako další kódy v PHP. Nejedná se o žádné kompilované knihovny, které by se musely do PHP přidávat.

U PHP frameworku je nutné dát si pozor při výběru, pro jakou verzi PHP je Framework určen. Někdy se může hodit podpora PHP 4, ale zde zase chybí spousta používaných funkcí z objektové orientace. V dnešní době se už ustupuje od vývoje frameworků pro PHP 4 a většina frameworků je už jen s PHP 5 kompatibilní.

Dalším důležitým faktorem při výběru frameworku PHP je to, k čemu bude Framework potřebný. Obecně se frameworky dělí do dvou skupin:

- Sady skriptů, knihoven – pokrývající velké množství potřeb.
- Skripty vytvářející jednu konkrétní webovou aplikaci, což znamená, že tato aplikace většinou využívá určitý návrhový vzor, nejčastěji MVC. Všechny požadavky pak míří na jeden index.php, jenž následně řídí celou aplikaci, načítá potřebné soubory a podobně.

Již zmiňované MVC je zkratka pro model – view – controller, což je návrhový vzor, tedy objektová struktura, která odděluje data a jejich zpracování od jejich zobrazení. Model představuje datové úložiště, obstarává získání dat a práci s nimi, poté je vrací controlleru, který si o ně zažádal, controller je následně předá view, který je zobrazí. Další možností vzoru je MPS – model – presenter – command. Tento návrhový vzor využívá český framework Nette.

Přínosy, které se získají použitím frameworku:

- Rychlejší vývoj
- Méně kódu
- Univerzální kód – jednodušší přidání změn a nových funkcí
- Pěkná URL adresa

Díky frameworku se můžete opravdu soustředit jen na ten opravdový vývoj. To vše je ale na úkor času, kdy se učíte s frameworkem pracovat.[19]

### 2.3.1 Přehled PHP frameworků

Na první pohled se může zdát, že frameworků pro PHP je velké množství. Pokud se však na výběr zaměříme detailněji, zjistíme, že ne každý vyhovuje. Některé frameworky mají nepřehledné množství knihoven, které obstarávají téměř vše, na co si vzpomenete. To je

ovšem vykoupeno jejich vyššími hardwarovými nároky a nižší rychlostí. Jiné mají knihoven méně, ale dají se, díky své modulárnosti, lehce rozšířit o nové funkce. Velká pozornost se musí věnovat dokumentaci jednotlivých frameworků. Pokud je dokumentace na špatné úrovni, práce s frameworkem trvá mnohem déle, protože je třeba dlouze vyhledávat parametry funkcí či řešení jiných úskalí.

Mezi nejpoužívanější PHP frameworky patří CakePHP, CodeIgniter, DooPHP, Jelix, Kohana, Laravel, Nette, Prado, Qcodo, Recess, Seagull, Symfony, Yii a Zend Framework. Tyto produkty jsou mezi sebou často porovnávány a testovány.

Při výběru vhodného PHP frameworku pro webovou aplikaci datbáze Ramanových spekter pomohl test nejpoužívanějších frameworků. Dle výsledků těchto testů byl vybrán pro použití Framework Nette, a to důvodů, že v testech skončil jako jeden z nevykonnějších a nejrychlejších frameworků. Dále byl důvodem výběru jeho původ, jedná se o český produkt, o čemž svědčilo i množství podpory a dokumentace. [19]

### 2.3.2 Testování PHP frameworků

Každý framework je alespoň trochu odlišný, ať už se jedná o použité programovací jazyky nebo strukturu uspořádání zdrojových souborů. Hlavním aspektem jejich rozdílnosti je ale množství dostupných funkcí a knihoven a případná rozšiřitelnost. Jestliže aplikace používá knihovny, které nepotřebuje přímo k běhu dané aplikace, tak to může mít negativní dopad na její rychlost. [20]

Většina dostupných testů se zabývá pouze vypsáním věty „Hello world“ nebo jednoho jednoduchého SQL příkazu, z tohoto důvodu bylo použito spíše zátěžové testování a různé příkazy. Tento test probíhal na serveru [www.zdrojak.cz](http://www.zdrojak.cz), odkud byly převzaty výsledky.

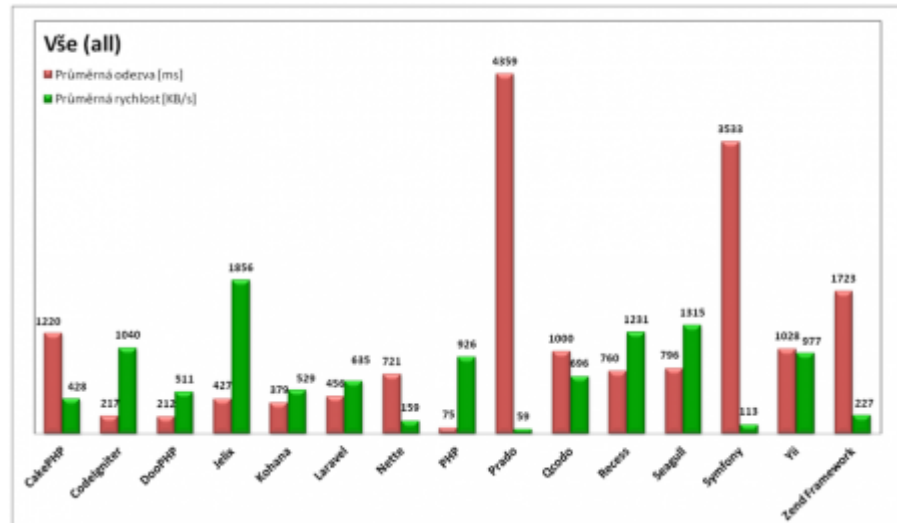
Testy byly prováděny na dvou počítačích v lokální síti, aby byl eliminován vliv zatíženosti případného hostingu. Po konfiguraci obou počítačů byly napsány jednotlivé aplikace ve všech testovaných frameworkech tak, aby si byly co nejvíce podobné. Tyto aplikace měly za úkol:

- Vypsat seznam všech zákazníků (tabulka customer).
- Vytvořit nového herce (tabulka actor) s unikátním ID a jménem a příjmením dle požadavku.

- Zkontrolovat existenci záznamu a následně editovat vytvořeného herce vybraného dle ID.
- Zkontrolovat existenci záznamu a následně smazat vytvořeného herce vybraného dle ID.

Celkem bylo u každého frameworku provedeno pět testů:

- Výběr záznamů (SELECT). Každý uživatel v každém jednotlivém kroku provedl výběr celé tabulky zákazníků a nechal si vypsát informace o každém záznamu (ID, jméno, příjmení).
- Vytvoření nového záznamu (INSERT). Každý uživatel v každém kroku přidal do databáze záznam o novém herci s unikátním identifikátorem a jménem a příjmením složeným z pořadového čísla uživatele, číslem kroku a označením pro danou buňku.
- Úprava záznamu (UPDATE). Každý uživatel zkontroloval dostupnost svých údajů v databázi definovaných aktuálním krokem, a to v závislosti na unikátním ID generovaného dle pořadí uživatele a na aktuálním prováděném kroku.
- Smazání záznamu (DELETE). Každý uživatel zkontroloval dostupnost svých záznamů dle generovaného ID a následně tento záznam smazal.
- Vše zároveň (ALL). V tomto testu byly provedeny všechny dříve zmíněné kroky s tím rozdílem, že byly všechny prováděny najednou. To znamená, že každý uživatel si v jednom svém kroku nechal vypsát tabulku zákazníků, vytvořil nového herce s unikátním ID jménem a příjmením, tento záznam upravil a na závěr odstranil. [20]



Obr. 11 – Výsledky komplexního testu (ALL) frameworků [21]

Výsledky posledního testu ukazují enormní rozdíly v jednotlivých rychlostech zpracování. Z takto naměřených údajů lze vyčíst, že nejrychlejší projekt je napsaný pomocí čistého PHP. Tento výsledek byl předvídatelný, protože tato aplikace neobsahuje žádné dodatečné funkce nebo složitější strukturu, které by omezovaly její výkon.

Nejlepších výsledků dosahuje framework DooPHP. Jeho velmi rychlé zpracování požadavků je způsobeno úsporným návrhem základního jádra, které je možné rozšiřovat o další funkcionalitu. Při zasílání jednoduchých příkazů pro databázi využívá pouze základní funkce a jednoduchý model, což má pozitivní vliv na celkovou rychlost. Podobnou strukturu používají frameworky Jelix nebo CodeIgniter. [20]

Z těchto výsledků je patrné, že Framework Nette není úplně nejrychlejší, ale drží se mezi výkonnostní špičkou. Tudíž o jeho použití rozhodl především fakt, že je zde početná česká komunita programátorů.

## 2.4 PHP framework Nette

### 2.4.1 Vývoj

Nette framework je vyvíjen organizací Nette Foundation, která navazuje svou činností na Davida Grudla, jenž je původním autorem Nette frameworku. Nette framework je český projekt a jeho výhodou i nevýhodou je dokumentace i komunita komunikující převážně v češtině. Nevýhodou je to proto, že se tím komunita okolo Nette stává užší. V minulosti byl problém s jednotlivými verzemi. Mezi verzí 0.9 a 2.0 byl vývoj Nette frameworku

značně turbulentní. Problém byl zejména v tom, že nové verze nebyly zpětně kompatibilní a byla tudíž nutnost přepsat celý kód aplikace, při přechodu na novější verzi. [22]

Nette Framework je napsaný v PHP 5 s plným využitím objektově orientovaného programování. Ačkoliv vznikl už v roce 2004, teprve v roce 2008 byl uvolněn jako open source a zpřístupněn veřejnosti. Jeho licence, která vychází z BSD (Berkeley Software Distribution), patří k těm nejvolnějším. Vyrostla kolem něj jedna z nejaktivnějších komunit českých PHP vývojářů, ne-li nejaktivnější vůbec. Nette používají významné tuzemské společnosti. A podle testu uveřejněném na serveru Root.cz je jedním z nejvýkonnějších frameworků.

Framework je koncipován jako „otevřený“, je ho tedy možné používat i v primitivních webových aplikacích nebo společně s jiným otevřeným frameworkem, jako je například Zend Framework. [23]

#### 2.4.2 Architektura MVP

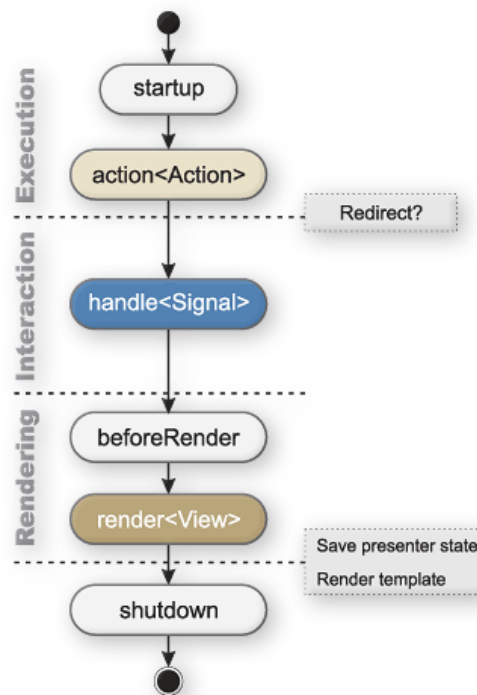
Jak již bylo uvedeno Nette má svou implementaci Model – View – Controller architektury pod označením Model –View – Presenter.

Celý proces je odstartován HTTP požadavkem (HTTP request), který je směřován na view, který daný požadavek zpracuje. Předá jej presenteru, ten získá od modelu potřebná data, která uloží nebo načte. Poté kontaktuje presenter view a předá mu data k zobrazení. View vygeneruje výpis údajů (nejčastěji HTML stránku) a pošle jej do prohlížeče uživatele jako HTTP odpověď (HTTP response). [22]

MVP architektura je standardem ve vývoji webových stránek. Je založena na striktním rozdělení aplikace na tři vrstvy, přičemž každá vrstva se stará o něco jiného. Toto členění je důležité zejména kvůli bezpečnosti, kdy pomocí *htaccess* (konfigurační soubor webového serveru Apache) lze omezit vstup jen do některých adresářů. Případný útočník se tak nedostane přes http protokol do žádného kritického adresáře. [22]

#### 2.4.3 Presenter a jeho životní cyklus

Presenter v Nette vyjadřuje třída, která je potomkem *Nette\Application\UI\Presenter* a tudíž má pevně definovaný svůj životní cyklus. [24]



Obr. 12 – Životní cyklus presenteru [13]

- **Startup** – bývá používán pro ověřování přihlášení a nastavení layoutu, který se konkrétnímu uživateli vykreslí.
- **action<Action>** – metody využívané pro různé akce, které potřebují po provedení přesměrovat – např. zápis do databáze, smazání záznamu. Bez přesměrování by se nám například mohlo stát, že uživatel aktualizuje stránku a právě uložený záznam se v databázi zduplikuje. Dále se zde v deklaraci metody definuje, které parametry se mají při http požadavku presenterem očekávat.
- **handle<Signal>** - jedná se o metody, které zpracovávají signály a díky tomu jsou více než vhodné pro zpracování AJAXových požadavků.
- **beforeRender** – je vhodná pro registraci Latte filtrů, případně pro předání dat globálně do všech view v konkrétním presenteru.
- **render<View>** - používá se pro předávání dat do šablony.
- **shutdown** – spustí se při skončení životního cyklu. [24]

#### 2.4.4 Model

Vrstva model je zodpovědná za datovou vrstvu aplikace. Stará se o zápis a načítání dat. O formě uložení a postupu zpracování dat ví jen model. To znamená, že view a presenter neví o datech nic a spoléhají na předání dat modelem.

#### 2.4.5 View

V MVP view se zpracovává jak požadavek, tak odpověď. Uvnitř by neměla být žádná aplikační logika a view pouze vyvolává jednotlivé uživatelské akce, tedy pokud uživatel například stiskne tlačítko, view vyvolá příslušnou metodu presenteru a předá jí akci.

#### 2.4.6 Bezpečnost v Nette

Bezpečnostní část aplikace je nejdůležitější ze vše hledisek frameworku. Nette je v tomto ohledu zabezpečeno proti většině používaných útoků, což je velkou výhodou toho frameworku. [25]

##### 2.4.6.1 *SQL Injection*

Tento druh útoku manipuluje přímo s databází. Tento útok může získat veškerá data z databáze nebo tato data smazat. Obrana frameworku Nette je jednoduchá, ale přesto velice účinná. Nette sice kód uloží tak, jak byl vložen, ale jako prostý text. [25]

##### 2.4.6.2 *Cross-Site Scripting (XSS)*

Cross-Site Scripting je metoda narušení webových stránek zneužívající neošetřených výstupů. Útočník pak dokáže do stránky podstrčit svůj vlastní kód a tím může stránku pozměnit nebo dokonce získat citlivé údaje o návštěvnicích. Proti XSS se lze bránit jen důsledným a korektním ošetřením všech řetězců. Přitom stačí, aby váš kodér jen jednou jedinkrát toto opomenul, a celý web může být rázem kompromitován. Příkladem útoku může být podstrčení upravené URL uživateli, pomocí které injektujeme do stránky svůj kód. Když aplikace nebude výstupy řádně ošetřovat, vykoná skript v prohlížeči uživatele. Tímto způsobem mu můžeme například zcizit identitu.

Nette Framework přichází s revoluční technologií Context-Aware Escaping, která vás provždy zbaví rizika Cross-Site Scriptingu. Všechny výstupy totiž ošetřuje automaticky

a tak se nemůže stát, že by kodér na něco zapomněl. Nette se samo stará o escapování (vypsání proměnné tak, aby se kód neprovedl). V kódu je výraz pro vypsání proměnné, která byla předána metodou *GET*, už escapován. Pokud je to nutné, je možné escapování vypnout pomocí „!“ . [25]

#### **2.4.6.3 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 takto například pozměnit nebo smazat článek, aniž by si toho uživatel všiml. Proti útoku se lze bránit generováním a ověřováním autorizačního tokenu.

Ochránit formulář před útokem Cross-Site Request Forgery lze v Nette Frameworku pouhým jedním příkazem: `$form->addProtection()`; tím je formulář ochráněn. [25]

#### **2.4.6.4 URL attack**

Různé pojmy související se snahou útočníka podstrčit vaši 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 to dělá automaticky. Nemusí se nastavovat vůbec nic a všechny vstupy budou ošetřené. [25]

#### **2.4.6.5 Session hijacking a podobně**

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 správné konfiguraci serveru a PHP.

Přičemž Nette Framework nakonfiguruje PHP automaticky. Programátor tak nemusí přemýšlet, kterak session správně zabezpečit a může se plně soustředit na tvorbu aplikace. Vyžaduje to však povolenou funkci `ini_set()`. [25]

### 2.4.7 Nette database

Nette Database je velice inovativní ale i „magická“ knihovna, která kombinuje hlavní myšlenku a prvky NotORM od Jakuba Vrány a pozměňuje (příliš) „magické“ API, aby se lépe používala. Tou myšlenkou převzatou z NotORM je inteligentní získávání dat z databáze. Snaží se ušetřit nás od psaní opakujících se SQL dotazů a pokládá minimální množství maximálně efektivních SQL dotazů. Opět můžete porovnat zápis dotazů. Může se tvářit příliš magicky, ale to je právě její výhoda. [26]

### 2.4.8 NotORM v Nette

Většina webových aplikací potřebuje pracovat s propojenými daty uloženými v databázi. Psát SQL dotazy spojující třeba šest tabulek v databázi může být docela zdržující a rozšiřuje to kód. Navíc spojení tabulek nemusí být vždy nejefektivnější, protože se znovu přenáší už jednou přenesená data. Takže někdy je výhodnější položit šest jednoduchých dotazů a propojit až jejich výsledky. Pro získání dat pomocí PHP proto vznikla knihovna, která propojená data z databáze dokáže elegantně a zároveň efektivně získat. Jak už bylo řečeno, tak knihovna je nejen elegantní, ale zároveň i efektivní. Klade tedy jen minimální počet jednoduchých dotazů a z databáze přenáší jen ta data, která jsou potřeba. Dosahuje se toho dvěma způsoby:

- Získání souvisejících záznamů se provádí najednou pro všechny řádky výsledku jedním dotazem.
- Přenášejí se jen ty sloupce, které se nakonec skutečně použijí (při povolení této vlastnosti).

NotORM umí pracovat se všemi databázemi, které podporuje PDO, testované jsou MySQL, SQLite, PostgreSQL a MS SQL. [27]

### 2.4.9 Composer v Nette

Composer je nástroj na správu závislostí v PHP. Dovoluje deklarovat libovolně složité závislosti jednotlivých knihoven a pak je nainstaluje do projektu. Základem pro jakýkoliv projekt nebo knihovnu, která využívá Composer, je soubor *composer.json*, který obsahuje meta informace o projektu či knihovně a jejich závislostech.

Základní *composer.json* tedy může vypadat takto

```
{
  "require": {
    "php" : ">=5.3.2",
    "nette/nette": "~2.1.0"
  }
}
```

Říká zde, že aplikace (nebo knihovna) vyžaduje balíček nette/nette a chce nejnovější verzi, která odpovídá 2.1.0 a taky že běží pouze na PHP větším nebo rovno 5.3.2. [27]

## 2.5 Databáze

S databázemi se setkáváme téměř denně. Obecně je databáze i nákupní seznam, výpis z účtu nebo například výpis uskutečněných telefonních hovorů. V počítačové oblasti se za pojmem databáze skrývá software, který spravuje, ukládá a určitým způsobem umožňuje měnit uložená data. Databáze není jen obyčejné shromaždiště dat, ale slouží i k jejich organizaci, třídění, prohledávání, seskupování a výběru dle kritérií.

Databáze jsou v dnešní době hojně směřovány k vzdálenému přístupu, kdy na jednom centrálním místě může být uloženo více databází pro několik nezávislých odběratelů.

### 2.5.1 Dělení databází

V poslední době rozdělit databáze do jednotlivých skupin je poměrně složité, protože se čím dál více jednotlivá kritéria vzájemně překrývají.

Možné druhy dělení databází jsou:

- **Objektové a relační databáze** - databáze se liší podle způsobu, jak ukládají data. Ve dnešní době převládá relační model databází nad ostatními.
- **Jedno (Více) uživatelské** - podle toho, kolik uživatelů se k databázi může připojit. Pochopitelně, v komerční sféře jednouuživatelským databázím prakticky odzvonilo. Je ale užitečné vědět, že i víceuživatelskou databázi lze většinou nakonfigurovat jako jednouuživatelskou a že to může mít své opodstatnění. MySQL je víceuživatelská databáze.
- **Podle licence a ceny** - Kód databáze může být uzavřený nebo otevřený, šíření software může být svobodné nebo může podléhat nějakým podmínkám, databáze se může využívat bez poplatků nebo může jít o placený software. Licence MySQL

je duální, lze tedy říct, pokud ji použijeme v projektu vytvářeném pod licencí GPL, můžeme ji použít taktéž pod licencí GPL.

- **Podle běhu databáze** - na jednoplatformní a multiplatformní. Jednoplatformní poběží jen na některém systému (třeba na Windows), multiplatformní na více systémech. MySQL je multiplatformní databáze a běží na systémech GNU/Linux, Microsoft Windows, FreeBSD, Sun Solaris, IBM's AIX, Mac OS X, HP-UX, AIX, QNX, Novell NetWare, SCO OpenUnix, SGI Irix, and Dec OSF. [21]

## 2.6 Databáze MySQL

MySQL je velmi populární databáze. Podle mnoha zdrojů se jedná o velmi rychlou databázi. Pro svůj výkon a především díky tomu, že MySQL vychází pod licencí GPL, má vysoký podíl na v současně používaných databázích. Velmi hojně se objevuje kombinace Linux, MySQL, PHP a Apache jako základní software webového serveru („technologie LAMP“). [28]

### 2.6.1 Historie MySQL

Databázi MySQL vyvinula švédská firma MySQL AB v čele s hlavními autory Michalem „Montym“ Wideniusem a Davidem Axmarkem. Od února roku 2008 databázi vlastní Sun microsystems, dceřiná společnost Oracle.

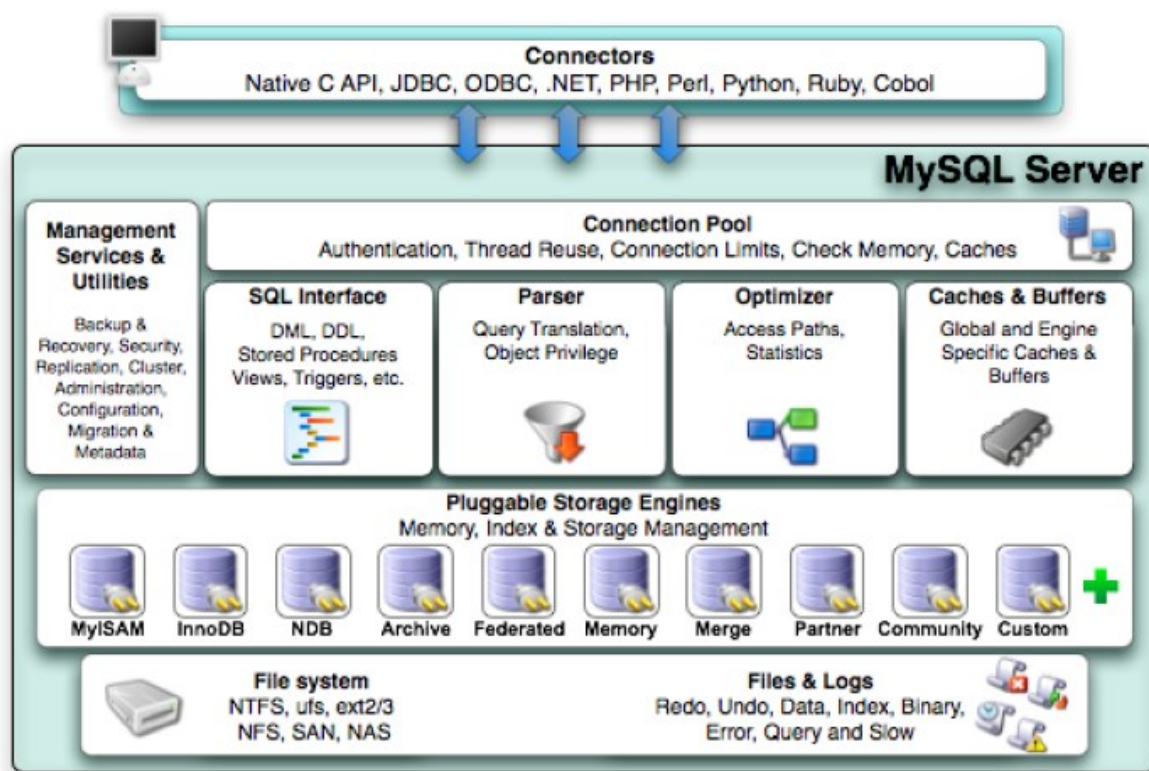
První verze vyšla v roce 1995 a byla určena pro unix a linux. V roce 1998 se objevila verze databáze pro operační systémy Windows společnosti Microsoft. Vývoj pokračoval vydáním verze 3.23 v roce 2001, tato verze se na mnoha místech používá i v dnešní době. V letech 2003 až 2005 se postupně objevily verze 4.0, 4.1 a 5.0, které s sebou přinesly spoustu velkých inovací, jako například R a B stromy, subdotazy, spouštěče, pohledy a další. V následující verzi 5.1, byla zejména rozšířena API, ale objevilo se zde i několik dalších nových funkcí. Ve verzi 6.x již databáze plně podporuje úplnou referenční integritu a paralelní zpracování dotazů. [29]

## 2.6.2 Architektura MySQL

Komunikace v databázi používá, jak již z názvu vyplývá, jazyk sql s některými rozšířeními. Architektura MySQL se velmi odlišuje od ostatních databázových systémů, má široké pole působnosti a lze ji využít pro mnoho různých úloh.

Vrstva v nejvyšší části architektury MySQL pracuje se službami, které nejsou primárně určeny jen pro MySQL, ale obsluhují většinu potřebných nástrojů propojení klient – server.

Ve druhé vrstvě se nachází velká část samotného MySQL. Jsou zde obsaženy části jako kód pro rozbor, analýzu, optimalizaci a ostatní vestavěné funkce. Na této vrstvě se pracuje s veškerou funkční vybaveností jednotlivých úložných enginů.



Obr. 13 – Schématická struktura MySQL serveru [30]

Třetí vrstva obsahuje jednotlivé úložné enginy. Tyto enginy se starají o ukládání a získávání všech potřebných dat. Server komunikuje s enginy pomocí API těchto enginů. Toto API se snaží pomocí množství vnitřních jednoduchých, nízko úroňových funkcí skrýt rozdíly jednotlivých druhů enginů. Tyto nízko úroňové funkce obstarávají jednoduché operace jako zahájení transakce, získání řádku dle primárního klíče a ukončení

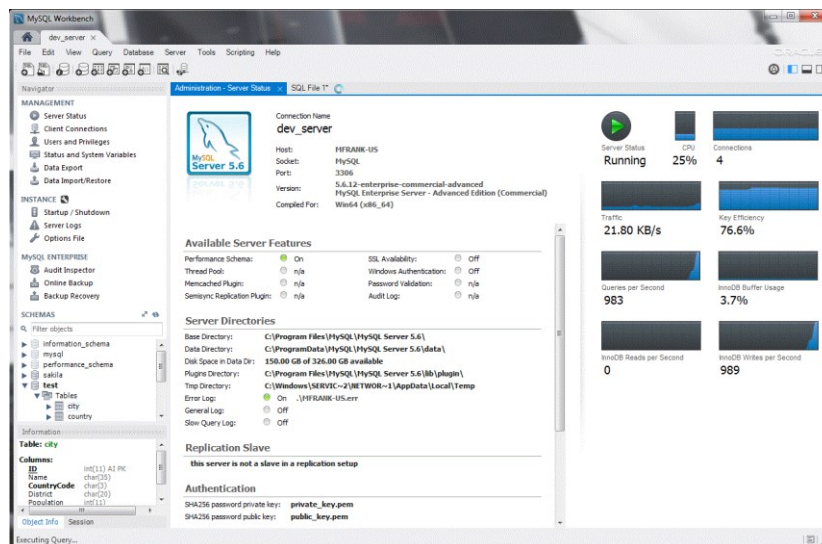
transakce. Zde v úložných enginech neprobíhá žádná analýza dat a tyto enginey ani nekomunikují mezi sebou. Jednotlivá úložiště dat:

- MyISAM
- InnoDB, Berkeley DB (BDB)
- memory, CSV, archive
- Falcon, Merge, Federated, Blackhole, Cluster, Mariam. [29]

Defaultně je v MySQL nastaven úložný engine MyISAM, protože se jedná o velmi rychlý úložný engine s textovými indexy, ale jeho nevýhodou je skutečnost, že neumí provádět transakce ani referenční integritu dat. Stále více se začíná používat úložný engine InnoDB, který se při častém čtení jeví sice jako pomalejší než MyISAM, ale obsahuje plnou podporu transakcí, je vydáván pod duální licenci a od roku 2008 patří společnosti Oracle.

### 2.6.3 MySQL Workbench

MySQL Workbench je unifikovaný vizualizační nástroj pro zobrazení, tvorbu a správu MySQL databáze a serveru. Tento nástroj je podporován na operačních systémech Windows, Linux a Mac OS.



Obr. 14 – základní obrazovka MySQL Workbench [29]

Dále jsou zde obsaženy funkce pro tvorbu, provádění a optimalizaci SQL dotazů. Tyto dotazy jsou barevně odlišovány a je zde tak vidět jednotlivá vazba mezi těmito dotazy. MySQL workbench podporuje i migraci mezi jednotlivými druhy databází tím, že

umožňuje export modelu databáze pro jiné databázové systémy, mezi ně patří Microsoft SQL Server, Sybase ASE, PostgreSQL a další. [29]

## **II. PRAKTICKÁ ČÁST**

### 3 DATABÁZE RAMANOVÝCH SPEKTER

V této části byla vytvořena vlastní databáze Ramanových spekter. Hlavním dodavatelem těchto naměřených spekter byl Ústav elektroniky a měření na Fakultě Aplikované informatiky Univerzity Tomáše Bati ve Zlíně. Na funkčnost databáze bylo několik požadavků, které byly úspěšně splněny. Při tvorbě této databáze a jejího administrátorského webového rozhraní byly použity softwarové technologie uvedené v předchozích kapitolách. V praktické části je nastíněno technické řešení těchto požadavků, celková funkčnost jak databáze, tak i administrátorského rozhraní a jsou rozebrány nejdůležitější části zdrojových kódů. Dále se kapitola zabývá popisem větší části jednotlivých souborů, tvořících strukturu balíku aplikace.

#### 3.1 Požadavky na funkčnost aplikace

Požadavky na funkčnost byly následující:

- Aplikace musí umožňovat uložení naměřených Ramanových spekter.
- Musí existovat více možností pro vložení naměřených dat – z důvodu existence i dalších webových databází Ramanových spekter, z nichž některé neposkytují přesná naměřená data, ale pouze obrázek naměřeného průběhu, musí být umožněno uložit i tento obrázek. V tomto případě je potřeba obrázek spektra doplnit nejdůležitějšími specifickými daty tohoto Ramanova spektra.
- Aplikace musí být přístupná po přihlášení.
- Role uživatelů musí být dvě, tedy administrátor a uživatel. Rozdíl mezi těmito rolami spočívá v možnosti vytvoření, editace a zrušení uživatele. Dále pak není obvyčnému uživateli umožněno smazat naměřené vzorky z databáze.
- Zřejmě nejdůležitějším funkcí je vyhledávání. Tato aplikace měla pomoci při identifikaci neznámého vzorku tím, že zadané naměřené hodnoty uživatelem porovná s hodnotami uloženými v databázi. Tato funkce bude dále podrobně rozebrána v následujících kapitolách.

Toto je stručný přehled základních požadavků na funkčnost této webové databázové aplikace. Detailní popis funkčnosti je uveden v následujících kapitolách.

## 3.2 Návrh databáze pro uložení dat

Na začátku si bylo potřeba ujasnit, co všechno bude třeba do databáze uživateli umožněno uložit, tedy které doplňující informace jsou důležité pro následující analýzu naměřených Ramanových spekter.

Výsledkem byla „karta informací“ jednotlivého naměřeného vzorku, obsahem této karty je několik důležitých údajů:

- **Český název vzorku**
- **Anglický název vzorku**
- **Chemický vzorec**
- **Zdroj** – jedná se u původce naměřených hodnot, tedy kdo naměřil uložená data.
- **Skupina** – tato položka slouží později ke skupinovému výběru při vyhledávání (například skupiny Léčiva, Výbušniny, Minerály, atd.).
- **Popis** - základní měřicí parametry pro získání spektra, tj. např. informace o použitém laseru, výkon laseru, doba expozice aj.
- **Data** – samotná naměřená data Ramanova spektra.

## 3.3 Schéma výsledné databáze

V dnešní době má většina webových hostingů vlastní databazový server i s administrátorským rozhraním pro tvorbu databází. Jelikož během tvorby ještě nebylo rozhodnuto, na kterém hostingu bude aplikace umístěna, byl vytvořen databázový model *czraman.mwb* v programu MySQL Workbench. Tento model byl později aplikován jako sql dotaz na zvolený hosting s databázovým serverem.

Výsledná databáze se skládá ze čtyř tabulek minerals, data, users a category. Schéma databáze je uvedeno na obrázku níže. Ze schématu je patrné, že se v databázi vyskytují mezi tabulkami pouze relace, neboli vztahy, typu 1:N.

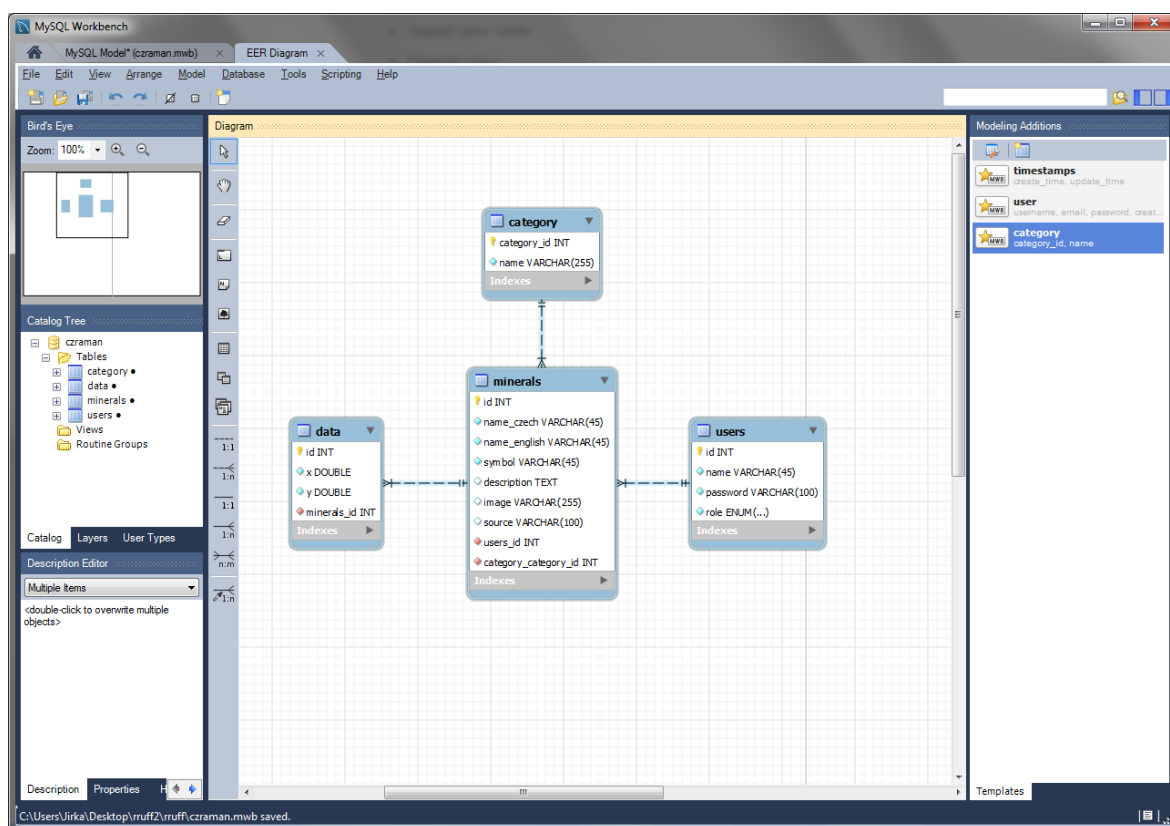
### 3.3.1 Relace mezi tabulkami

Konkrétně mezi tabulkami data a minerals je vztah 1:N myšlen tak, že k jednomu záznamu v tabulce minerals patří více záznamů v tabulce data. Toto tvrzení je vcelku logické, jelikož

pro vykreslení Ramanova spektra je používáno relativně velké množství bodů se souřadnicemi x a y.

Vazba mezi tabulkami minerals a users je opět typu 1:N. Avšak v tomto případě je to myšleno obráceně. To znamená, že jeden uživatel v tabulce users vytvořil více záznamů v tabulce minerals.

Poslední vazbou v databázi je vazba 1:N mezi tabulkami category a minerals, kdy jedna kategorie může náležet několika záznamům v tabulce minerals.



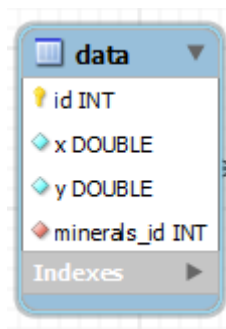
Obr. 15 – Schéma databáze v aplikaci MySQL Workbench

### 3.3.2 Popis tabulky data

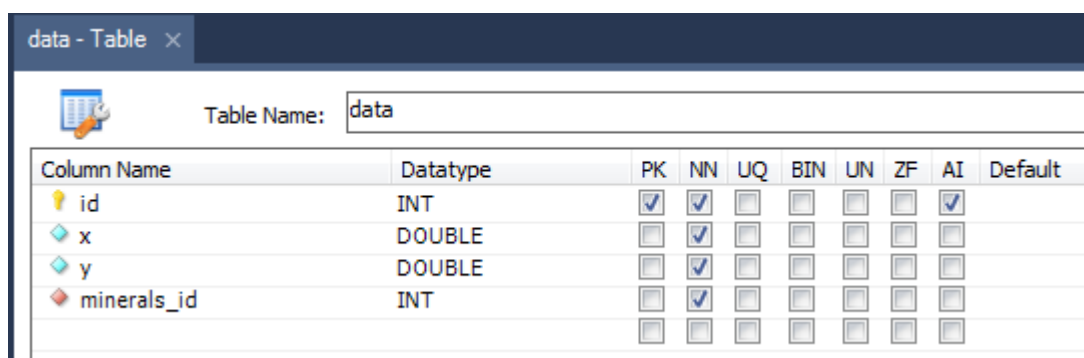
Tabulka data se skládá ze čtyř sloupců.

- Sloupec id - Jedinečný indentifikátor záznamu v tabulce data a je tedy určen i jako primární klíč pro tabulku data. Je automaticky inkrementován, tedy jeho hodnota se automaticky zvyšuje s každým dalším záznamem a jeho datový typ je nastaven na INTEGER, tedy obor celých čísel.

- **Sloupec x** – do tohoto sloupce jsou ukládány x-ové souřadnice každého vloženého bodu Ramanova spektra. V tomto sloupci je nastaven typ DOUBLE, tedy obor reálných čísel. Zároveň je u tohoto sloupce aplikován parametr not null – nesmí být nevyplněn.
- **Sloupec y** – do toho sloupce se ukládají y-ové souřadnice každého vloženého bodu Ramanova spektra. Zde je opět nastaven typ DOUBLE a zároveň aplikován paramter not null.
- **Sloupec minerals\_id** – tento sloupec je typu INTEGER. Chová se jako cizí klíč, to znamená, že odkazuje, ke kterému záznamu v tabulce minerals patří záznam v tabulce data. Opět u něj platí parametr not null.



Obr. 16 – Struktura tabulky data



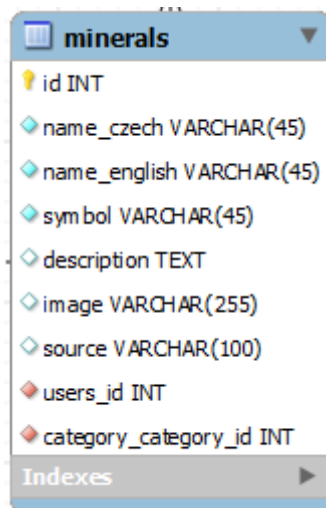
Column Name	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	Default
id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
x	DOUBLE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
y	DOUBLE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
minerals_id	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Obr. 17 – Parametry nastavené u sloupců v tabulce data

### 3.3.3 Popis tabulky minerals

Tabulka data se skládá z devíti sloupců.

- **Sloupec id** - jedinečný indentifikátor záznamu v tabulce minerals je určen i jako primární klíč pro tabulku data. Je automaticky inkrementován, jeho hodnota se automaticky zvyšuje s každým dalším záznamem a jeho datový typ je nastaven na integer - obor celých čísel.
- **Sloupec name\_czech** – do tohoto sloupce jsou ukládány české názvy každého vloženého vzorku. V tomto sloupci je nastaven typ VARCHAR(45), tedy textový řetězec o maximální délce 45 znaků. Zároveň je u tohoto sloupce aplikován parametr not null – nesmí být nevyplněn.
- **Sloupec name\_english** – do tohoto sloupce jsou ukládány anglické názvy každého vloženého vzorku. V tomto sloupci je nastaven typ VARCHAR(45), tedy textový řetězec o maximální délce 45 znaků. Zároveň je u tohoto sloupce aplikován parametr not null – nesmí být nevyplněn.
- **Sloupec symbol** – do tohoto sloupce jsou ukládány chemické vzorce každého vloženého vzorku. V tomto sloupci je nastaven typ VARCHAR(45), tedy textový řetězec o maximální délce 45 znaků. Zároveň je u tohoto sloupce aplikován parametr not null – nesmí být nevyplněn.
- **Sloupec description** – do tohoto sloupce je ukládán popis jednotlivých vložených vzorků. V tomto sloupci je nastaven typ text.
- **Sloupec image** – je nastaven jako typ VARCHAR(255) a ukládá se zde název vloženého obrázku.
- **Sloupec source** – je nastaven na typ VARCHAR(100) a ukládá v sobě popis zdroje dat, tedy odkud data pocházejí, kdo je naměřil.
- **Sloupec users\_id** – zde je uloženo id uživatele, který daný vzorek vkládal do databáze. Jedná se o cizí klíč k tabulce users.
- **Sloupec category\_category\_id** – zde je uloženo id kategorie neboli skupiny prvků. Toto id odpovídá id skupiny v tabulce category, jedná se tedy o cizí klíč.



Obr. 18 – Struktura tabulky minerals

Column Name	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	Default
id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
name_czech	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
name_english	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
symbol	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
description	TEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
image	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
source	VARCHAR(100)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
users_id	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
category_category_id	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

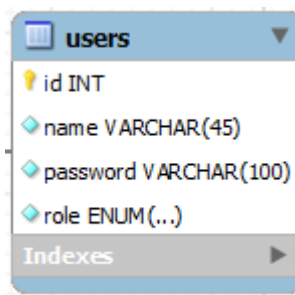
Obr. 19 – Parametry sloupců v tabulce minerals

### 3.3.4 Popis tabulky users

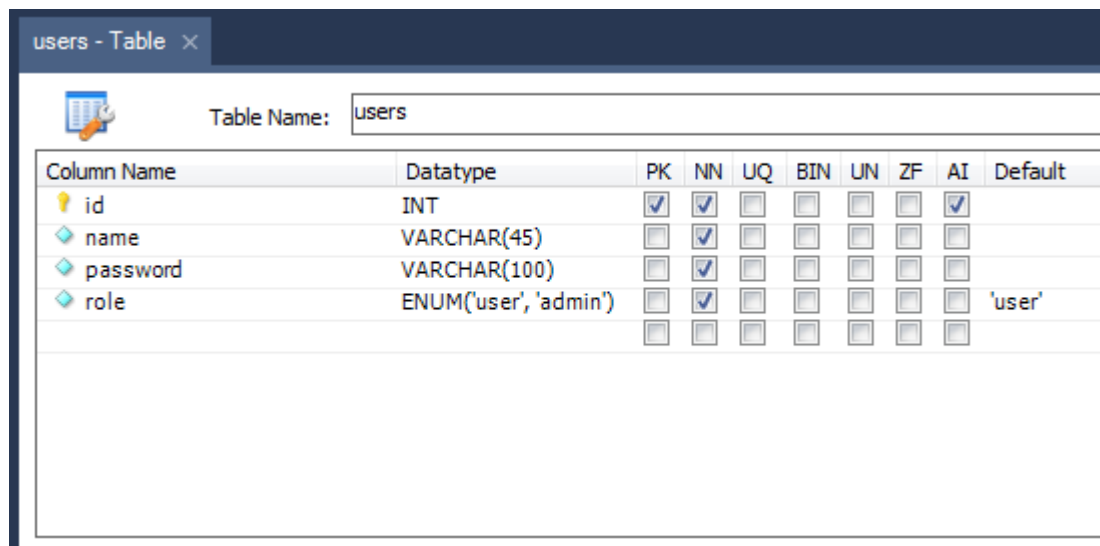
Tabulka data se skládá ze čtyř sloupců.

- Sloupec id - Jedinečný indentifikátor záznamu v tabulce users a je tedy určen i jako primární klíč pro tabulku data. Je automaticky inkrementován, tedy jeho hodnota se automaticky zvyšuje s každým dalším záznamem a jeho datový typ je nastaven na INTEGER.

- Sloupec name – je typu VARCHAR(45), tedy textový řetězec s maximální délkou 45 znaků. Zde jsou uložena jména uživatelů. Je zde i aplikován parametr not null.
- Sloupec password – do toho sloupce se ukládají hashe, tedy šifrované podoby, jednotlivých hesel uživatelů. Je zde nastaven typ VARCHAR(100) a zároveň aplikován parametr not null.
- Sloupec role – tento sloupec je typu ENUM. Je zde tedy umožněn výběr mezi dvěma hodnotami admin a user. Defaultně je zde nastavena hodnota user.



Obr. 20 – Struktura tabulky users



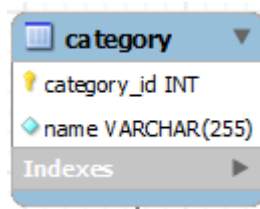
Column Name	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	Default
id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
name	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
password	VARCHAR(100)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
role	ENUM('user', 'admin')	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'user'

Obr. 21 – Parametry nastavené u sloupců v tabulce users

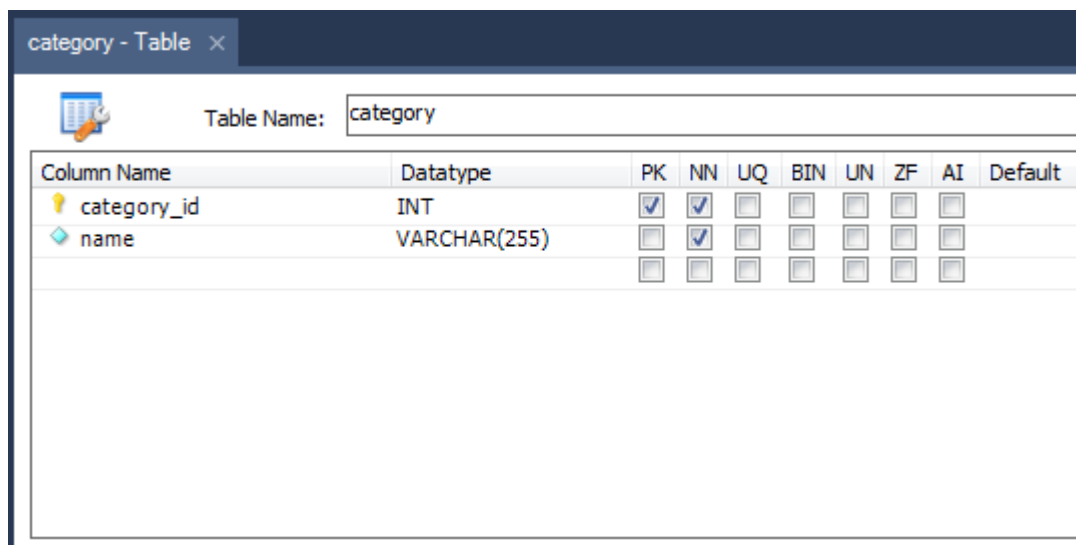
### 3.3.5 Popis tabulky category

Tabulka data se skládá ze dvou sloupců.

- Sloupec id - Jedinečný indentifikátor záznamu v tabulce category a je tedy určen i jako primární klíč pro tabulku data. Je automaticky inkrementován, tedy jeho hodnota se automaticky zvyšuje s každým dalším záznamem a jeho datový typ je nastaven na INTEGER.
- Sloupec name – je typu VARCHAR(255), tedy textový řetězec s maximální délkou 255 znaků. Zde jsou uloženy názvy kategorií vzorků. Je zde aplikován parametr not null.



Obr. 22 – Struktura tabulky category



Column Name	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	Default
category_id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
name	VARCHAR(255)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

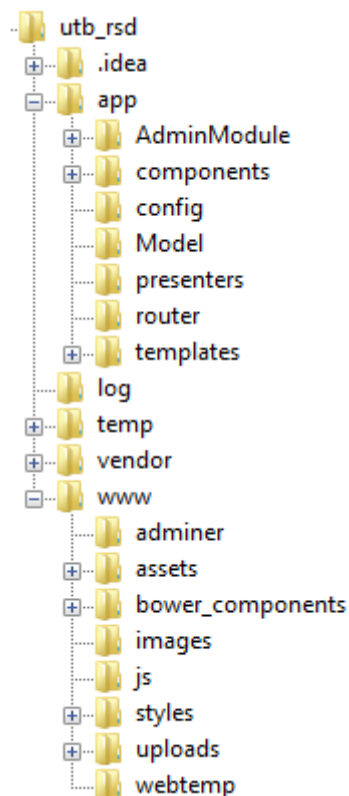
Obr. 23 – Parametry u sloupců v tabulce category

## 4 ADMINISTRACE DATABÁZE

Pro účely administrace databáze lze využívat i administrační rozhraní dodávané poskytovatelem webového hostingu, avšak toto řešení by nespĺňovalo veškerá očekávání a bylo by značně neefektivní. Z tohoto důvodu byla vytvořena webová aplikace pro administraci této databáze a to za pomoci frameworku Nette a dalších tímto frameworkem podporovaných knihoven. Pro vzhled administrace byla zakoupena šablona UnicornAdmin. V následujících kapitolách bude popsána funkčnost jednotlivých částí této administrace.

### 4.1 Vnitřní struktura administračního prostředí

Tato aplikace využívá strukturu souborů doporučenou frameworkem Nette z důvodu její přehlednosti jak pro programátora, tak i pro PHP – skripty, které nemusí při volání funkcí procházet ohromné množství souborů a je tak značně zrychlena samotná funkčnost aplikace.



Obr. 24 – adresářová struktura

- **App** - složka, ve které se vyskytují nejdůležitější části samotné aplikace, tedy aplikační logika. Mimo jiné jsou zde umístěny i důležité soubory jako *bootstrap.php* a *config.ini*. Soubor *bootstrap.php* má na starost zavedení a spuštění frameworku Nette a spuštění aplikace. V této složce jsou také umístěny modely, presentery a templates, jejichž užití vychází z koncepce MVP, která byla popsána v teoretické části. Složka *AdminModule* obsahuje administrátorskou část aplikace, tedy veškeré funkce dostupné po přihlášení uživatele.
- **Vendor** - tato složka slouží pro ukládání ostatních použitých knihoven. Je zde umístěno i samotné Nette.
- **Log** - složka pro ukládání chybových zpráv. Pokud během zpracování vznikne nějaká chyba, Nette automaticky vygeneruje soubor s popisem chyby a uloží jej sem.
- **Temp** - zde si Framework Nette vytváří dočasné soubory – „cache“. Jedná se o dočasná data, šablony. Zejména u šablon se vyplatí je uložit do složky temp, jelikož šablony mají vlastní jazyk a jejich převod do PHP zabírá čas, tudíž se vyplatí pro zrychlení aplikace tento převod provést pouze jednou.
- **www** - do této složky jako jediné má přístup webový prohlížeč. Obsahuje soubory jako CSS, zmíněnou šablonu *UnicornAdmin*, JavaScriptové soubory a obrázky. Obrázky jsou nahrávány pomocí *dropzone* modulu ve složce assets. A poté jsou ukládány ve složce uploads.

## 4.2 Přihlášení do administrace

Pro přístup ke správě databáze a vůbec pro práci s ní je nutné se přihlásit na přihlašovací obrazovce pomocí vytvořeného uživatelského jména a hesla.



```
0 ..... 10 ..... 20 ..... 30 ..... 40 ..... 50 ..... 60 ..... 70 ..... 80
<?php

namespace App\AdminModule;

use Nette\Application\UI\Form;

class SignInForm extends Form
{

    public function __construct()
    {

        parent::__construct();

        $this->addText('login', 'Login:')
            ->setRequired();

        $this->addPassword('password', 'Heslo:')
            ->setRequired();

        $this->addSubmit('submit', 'Přihlásit se')
            ->getControlPrototype()
            ->class('btn btn-primary btn-block');

        $this->getElementPrototype()
            ->class('well form-horizontal');
    }
}
```

Obr. 27 – ukázka souboru SignInForm.php

Samotné vyhodnocení přihlašovacích údajů probíhá pomocí souboru *UserManager.php*. Zde se vytvoří dotaz do databáze, konkrétněji do tabulky users, kde je vyhledáno jméno uživatele (na přihlašovací obrazovce Login a v tabulce uloženo jako name). Při nálezů je vráceno číslo řádku, na kterém byl nalezen uživatel.

Dále je vyhodnocena série podmínek, zda existuje číslo řádku. Pokud ne, je vypsána chybová hláška „Uživatelské jméno není zadáno správně“. Pokud se v databázi nalezne uživatelské jméno, vypočítá se hash ze zadaného hesla a tento hash je porovnán s uloženým hashem v databázi. Pokud nedojde ke shodě těchto hashů, je vypsána chybová hláška „Heslo není zadáno správně“. Pokud nastane přihlášení, je vyhodnocena shoda hashů, jsou funkcí vráceny informace o uživateli. Tedy jeho jméno a role. Toto rozdělení již částečně zachycuje princip struktury MVP.

```
0 10 20 30 40 50 60 70 80 90 100 110 120
/* @param Nette\Database\Context $database
 */
public function __construct(Nette\Database\Context $database)
{
    $this->db = $database;
}

/**
 * Performs an authentication.
 * @param array $credentials
 * @throws \Nette\Security\AuthenticationException
 * @return Nette\Security\Identity
 */
public function authenticate(array $credentials)
{
    list($name, $password) = $credentials;

    $row = $this->db->table('users')->where('name', $name)->fetch();

    if (!$row) {
        throw new Nette\Security\AuthenticationException('Uživatelské jméno není zadáno správně.', self::IDENTITY_NOT_FOUND);
    }

    if ($row->password !== $this->calculateHash($password, $row->password)) {
        throw new Nette\Security\AuthenticationException('Heslo není zadáno správně.', self::INVALID_CREDENTIAL);
    }

    $arr = $row->toArray();
    unset($arr['password']);
    return new Nette\Security\Identity($row->id, $row->role, $arr);
}
```

Obr. 28 – ukázka souboru UserManager.php

Po přihlášení do administrace databáze se ocitnete v hlavní nabídce. Zde je možno zahájit rovnou vyhledávání nebo přejít k jiné z dalších nabízených možností.

Mezi funkčnosti patří základní položka Hledání dále pak Vzorky, kde se nachází celkový přehled všech naměřených vzorků uložených v databázi. Dále je tu možnost tvorby skupin, ze kterých si lze vybírat později při ukládání vzorků do databáze. Poslední položkou k práci je položka Uživatelé, sloužící ke správě uživatelů, k jejich tvorbě i editaci.

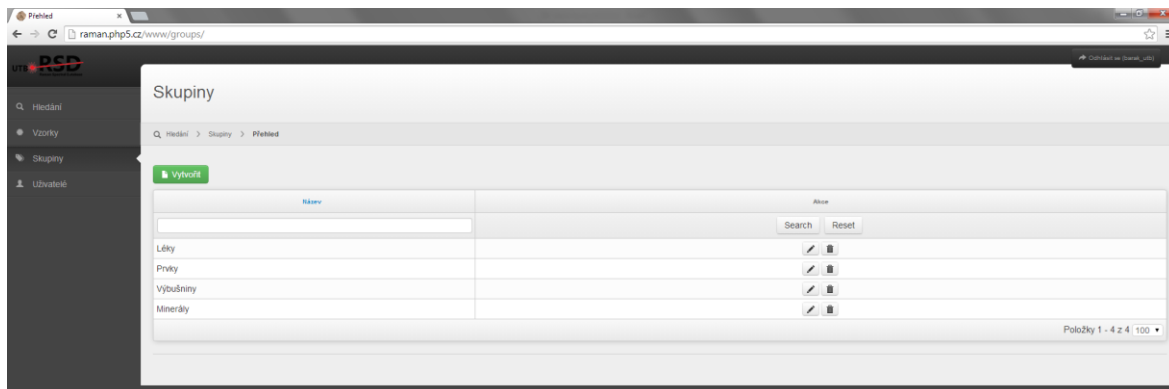
Jednotlivé nabídky budou probírány dle logického sledu použití, jelikož nemůžeme vyhledávat, pokud není nic uloženo v databázi. Protože pro vyhledávání je možné použít i skupinu a tato skupina se musí zadat při ukládání vzorku do databáze, měla by tato skupina být vytvořena jako první.

### 4.3 Skupiny



Tato část administrace patří k těm nejjednodušším, jak již vyplývá ze struktury tabulky category, která je pro tuto nabídku primárně důležitá.

Účelem nabídky skupiny je vytvoření množství skupin látek, které budou měřeny. Jednotlivé kategorie lze později mazat i přidávat dle rozmanitosti měřených látek.

Po kliknutí na nabídku skupiny se objeví přehledové okno se všemi vytvořenými skupinami. Pokud ovšem nejste v roli administrátora, nemáte do této sekce přístup. O tuto kontrolu a vytvoření tohoto okna se stará kód v souboru *GroupsPresenter.php*.



Obr. 29 – Přehled vytvořených skupin

V nabídce skupiny lze skupiny editovat , mazat , vytvářet a vyhledávat konkrétní z nich. Při stisku tlačítka pro vytvoření se vyvolá metoda *actionAdd*, uložená v souboru *BasePresenter.php*. Pro mazání a editaci položek se volají opět metody z tohoto souboru a to *actionEdit* a *handleDelete*.

Při vytváření skupiny stačí pouze kliknout na tlačítko vytvořit a zadat český název skupiny vzorků a kliknout na tlačítko uložit. Po té bude tento záznam uložen do tabulky category. Existuje zde i možnost nahrát obrázek do „dropzone“, což je oblast, kde stačí obrázek přetáhnout myší a bude uložen do složky uploads na FTP server a jeho název do databáze k danému záznamu. Tato funkce zatím u položky skupiny nemá opodstatnění a je využita zejména v dalších částech administrace, nicméně byla i zde ponechána pro možné potřeby dalšího rozšíření aplikace.

```
0      10      20      30      40      50      60      70      80      90      100     110
public function actionAdd()
{
    $result = $this->db->query('SHOW TABLE STATUS LIKE ?', $this->tableName)->fetch();
    $lastId = $result->Auto_increment;
    $this['dropzone'] = $this->dropzoneFactory->create('/uploads/' . $this->tableName . '/' . $lastId );

    if( isset($this['form']['image']) )
        $this['dropzone']->setInput( $this['form']['image'] );

    $this->setView('detail');
}

/**
 * Default action for editing items
 *
 * @param int $id
 */
public function actionEdit($id)
{
    $row = $this->db->table($this->tableName)
        ->wherePrimary($id)
        ->fetch();

    $this->template->row = $row;

    $this['form']->addHidden('id', $id);
    $this['form']->setDefaults($row);

    $this['dropzone'] = $this->dropzoneFactory->create('/uploads/' . $this->tableName . '/' . $id );

    if( isset($this['form']['image']) )
        $this['dropzone']->setInput( $this['form']['image'] );

    $this->setView('detail');
}
```



Obr. 30 – Ukázka ze souboru BasePresenter.php

## 4.4 Vzorky

V této části aplikace je umožněn přehled všech vzorků látek s naměřeným Ramanovým spektrem. Zobrazení tohoto přehledu zajišťuje kód v souboru *MineralsPresenter.php* ve spolupráci se souborem *MineralsDataManager.php*, kde jsou definovány metody používané k uložení, editaci a výběru dat z databáze.



Česká jména	Anglická jména	Chemická značka	Zdroj	Přidáno uživatelem	Skupina	Akce
Křemík	Silicon	Si	UTB	barak_utb	Léky	[Icons]
Acypyrin	Acypyrin	C <sub>9</sub> H <sub>9</sub> O <sub>4</sub>	UTB	barak_utb	Léky	[Icons]
A-00-1	A-00-1	A-00-1	UTB	barak_utb	Výbušniny	[Icons]
Ethanol	Ethanol	C <sub>2</sub> H <sub>6</sub> O	UTB	barak_utb		[Icons]
Methanol	Methanol	CH <sub>4</sub> O	UTB	barak_utb		[Icons]
Olivin	Olivine	(Mg, Fe) <sub>2</sub> (SiO <sub>4</sub> )	UTB	barak_utb	Minerály	[Icons]
Paralen	Paralene	C <sub>8</sub> H <sub>9</sub> NO <sub>2</sub>	UTB	barak_utb	Léky	[Icons]
Polybuten 8640M	Polybutene 8640M	??	UTB	barak_utb		[Icons]
Semtex	Semtex	??	UTB	barak_utb	Výbušniny	[Icons]
Oxid křemičitý	Silicon dioxide	SiO <sub>2</sub>	UTB	barak_utb		[Icons]
Oxid titaničitý	Titanium dioxide	TiO <sub>2</sub>	UTB	barak_utb		[Icons]
Trinitrotoluen	Trinitrotoluene	C <sub>7</sub> H <sub>5</sub> N <sub>3</sub> O <sub>6</sub>	UTB	barak_utb	Výbušniny	[Icons]

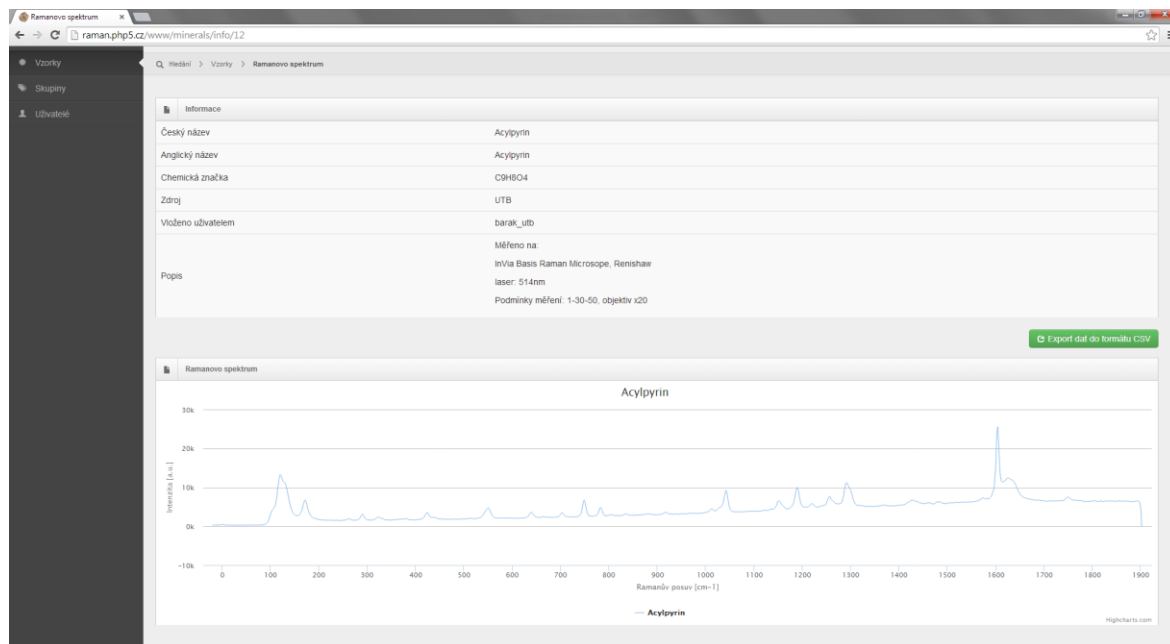
Obr. 31 – Přehled všech zadaných vzorků v databázi

Z této nabídky lze pomocí tlačítka  rozbalit detail jednotlivých prvků, pomocí tlačítka  exportovat uložené naměřené hodnoty Ramanova spektra do CSV souboru.

	A	B	C	D	E
1	1903.2, 23.5305				
2	1901.5, -4.41197				
3	1899.8, 4814.07				
4	1898.1, 6015.78				
5	1896.4, 6317.5				
6	1894.7, 6423.65				
7	1893, 6478.33				
8	1891.29, 6553.56				
9	1889.59, 6445.16				
10	1887.89, 6355.88				
11	1886.19, 6419.34				
12	1884.48, 6378.55				

Obr. 32 – Náhled exportovaného CSV souboru dat

Tlačítkem  lze editovat záznam tabulky nebo jej tlačítkem  smazat. Nakonec je zde umožněna i filtrace dle různých parametrů.



Obr. 33 – Detail dat Acetylpyrinu uložených v databázi

O zobrazení všech dostupných informací ohledně uloženého záznamu se stará šablona *Minerals.info.latte*, které dodávají data *MineralDataManager.php* a *MineralsPresenter.php*. Tento detail prvku je vlastně cílová informace, která je zobrazitelná i při vyhledávání a slouží pro porovnávání naměřených a uložených Ramanových spekter.

Zajímavou částí šablony *Minerals.info.latte* je JavaScript, který se používá k vykreslení Ramanova spektra pomocí souřadnic x a y uložených v databázi. Tento script vychází z opensource knihovny Highcharts, která zde byla pro tento účel použita.

```
0 10 20 30 40 50 60 70 80
{if !$mineral->image || !$imageFinder->imageExists($mineral)}

{define #script}

<script src="{${basePath}/bower_components/highcharts-release/highcharts.js}"></script>

<script>
  $(function () {
    $('#container').highcharts({
      title: {
        text: {$mineral->name_czech},
        x: -20 //center
      },

      xAxis: {
        title: {
          text: 'Ramanův posuv [cm-1]'
        }
      },

      yAxis: {
        title: {
          text: 'Intenzita [a.u.]'
        }
      },

      tooltip: {
        valueSuffix: ' a.u.'
      },

      plotOptions: {
        series: {
          marker: {
            enabled: false
          }
        }
      },

      series: [{
        name: {$mineral->name_czech},
        lineWidth: 1,
        data: [
          {foreach $mineral->related('data')->order('x') as $d}
            [!$d->x], [!$d->y]][sep],{/sep}
          {/foreach}
        ]
      }
    ]
  });
</script>

{/define}
```

Obr. 34 – JavaScript využívající knihovnu Highcharts

Ještě před začátkem scriptu je zde vytvořena důležitá podmínka, a to taková, zda pokud k zadanému vzorku byl nahrán nějaký obrázek, bude tento obrázek upřednostněn před vykreslením Ramanova spektra pomocí scriptu. Tato funkčnost umožňuje uživateli nahrát

místo souboru souřadnic pouze obrázek Ramanova spektra a tento obrázek poté zobrazit uživateli.

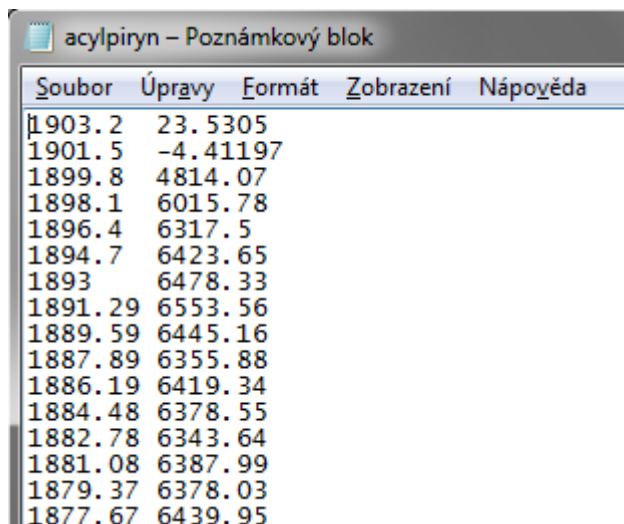
Samotný script si v prvním kontejneru *title* načte proměnnou *name\_czech*, což je název prvku a ten pak ve výsledku použije jako název grafu. V dalším kontejneru *xAxis* se nastaví pouze popisek osy x, počátek a konec osy x se nastavuje automaticky dle vykreslovaných hodnot. To samé se nastavuje v dalším kontejneru *yAxis*, avšak pro osu y. Kontejner *tooltip* slouží pro nastavení zobrazení souřadnic grafu v místě, na kterém nad ním držíte ukazatel myši. Tento kontejner zůstal v defaultním nastavení, pouze zde byla doplněna jednotka pro y-ovou souřadnici bodu. V části *plotOptions* bylo vypnuto zobrazení jednotlivých bodů na průběhu Ramanova spektra z důvodu jejich vysoké četnosti. V posledním kontejneru *series* probírá samotné načtení dat z databáze seřazené od nejmenší x-ové souřadnice po největší.

#### 4.5 Vytvoření vzorku

K vytvoření vzorku v databázi slouží tlačítko vytvořit, které vyvolá okno pro vyplnění údajů u vkládaného vzorku. Zde je nutno vyplnit všechny informace a zvolit metodu jakou budou nahrána data pro vykreslení Ramanova spektra.

V administraci se počítalo s několika variantami možných situací:

- **Nenahrávat data** – tato možnost je zatržena defaultně a používá se pro nahrání pouze popisu daného naměřeného vzorku bez dat Ramanova spektra. Také jde tato varianta využít při editaci již nahraného vzorku, u kterého je například nutné změnit část popisku.
- **Ze souboru** – tato varianta počítá s možností vložit data pro vykreslení Ramanova spektra z textového souboru. Vkládané souřadnice musí být v souboru vždy x i y na jednom řádku mezi sebou odděleny tabulátorem. Tyto souřadnice jsou pak využity výše uvedeným JavaScriptem využívající knihovnu HighCharts pro vykreslení Ramanova spektra. Toto uspořádání dat v textovém souboru bylo zvoleno z důvodu, že Ramanův spektrometr firmy Renishaw, který vlastní Fakulta aplikované informatiky Univerzity Tomáše Bati ve Zlíně, expotruje naměřená data do textového souboru právě v této podobě.

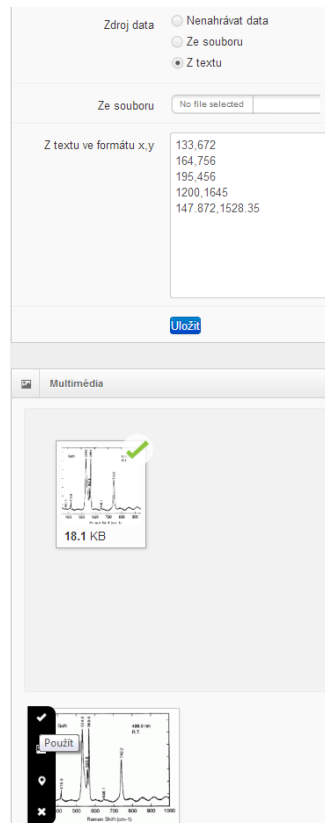


Soubor	Úpravy	Formát	Zobrazení	Nápověda
1903.2	23.5305			
1901.5	-4.41197			
1899.8	4814.07			
1898.1	6015.78			
1896.4	6317.5			
1894.7	6423.65			
1893	6478.33			
1891.29	6553.56			
1889.59	6445.16			
1887.89	6355.88			
1886.19	6419.34			
1884.48	6378.55			
1882.78	6343.64			
1881.08	6387.99			
1879.37	6378.03			
1877.67	6439.95			

Obr. 35 – ukázka vkládaného souboru souřadnic

- **Z textu** – tato varianta byla především zamýšlena pro případy, kdy máme k dispozici pouze obrázek naměřeného Ramanova spektra a chceme jej přidat do databáze. V tuto chvíli je výhodné použít „dropzone“ ve spodní části stránky pro vložení obrázku tohoto Ramanova spektra. Jak již název komponenty napovídá, stačí pouze obrázek přetáhnout nad tuto oblast a označit jej k použití. Samozřejmě samotný obrázek v databázi nám při vyhledávání moc nepomůže a proto je zde možnost připsat několik známých hodnot typických „peaků“ pro toto spektrum a tyto „peaky“ se právě ručně zadají do pole „Z textu“.

Jak již bylo výše uvedeno ve skriptu, který se stará o vykreslování průběhu Ramanova spektra, je na prvním místě podmínka, hlídající existenci obrázku uloženého pomocí této „dropzone“. Tento obrázek má ve výsledku přednost před vykreslováním průběhu Ramanova spektra, přičemž ručně vložené hodnoty „peaků“ lze v databázi stále vyhledat (Obr. 35).



Obr. 36 – Ukázka vložení obrázku spektra s typickými „peaky“

Po zadání všech položek k uložení bude presenter volat funkci `save()` obsaženou v `MineralDataManger.php`, tato funkce obstará uložení všech informací do databáze.

```

0          10         20         30         40         50         60
public function save($mineralId, array $data)
{
    $mineral = $this->db->table('minerals')
        ->wherePrimary($mineralId)
        ->fetch();

    if(!$mineral) {
        throw new BadRequestException;
    }

    $data = array_map(function($item) use($mineralId) {
        return [
            'x' => $item['x'],
            'y' => $item['y'],
            'minerals_id' => $mineralId,
        ];
    }, $data);

    // ----- TRANSACTION BEGIN -----
    $this->db->beginTransaction();
    $rel = $mineral->related('data');
    $rel->delete();
    $inserted = $rel->insert($data);
    $this->db->commit();

    // ----- TRANSACTION END -----

    return $inserted;
}

```

Obr. 37 – Funkce `save()` v `MineralDataManger.php`

## 4.6 Hledání v databázi

Hledání v databázi lze považovat za nejdůležitější funkčnost této administrace, to je taky jeden z důvodů, proč je tato funkce zobrazována hned po přihlášení uživatele.

Vyhledávat v databázi lze podle všech kritérií. Tedy lze vyhledávat podle názvu jak českého tak i anglického, dále lze hledat podle chemické značky, zdroje původu a dokonce i v popisu uloženého vzorku. Navíc je zde možnost i jen tak vyfiltrovat všechny vzorky z dané skupiny.

Ale za nejúčelnější lze považovat vyhledávání v databázi přímo pomocí zadání x-ových souřadnic typických „peaků“ měřených látek.

Obr. 38 – Nabídka vyhledávání v databázi

K vyhledávání se používá funkce *handleSearch()* v *DefaultPresenter.php*. Tato funkce obdrží v parametru název formuláře a z něj si vybere hodnoty – *values*, a poté vybere vše z tabulky *minerals* v databázi.

```
public function handleSearch(SearchForm $form)
{
    $values = $form->values;

    $selection = $this->db->table('minerals');
```

Obr. 39 – Výpis všech údajů z tabulky *minerals*

Poté funkce zkontroluje, zda byly vloženy hodnoty rozptylu pro vyhledávání x-ových souřadnic. Pokud ne, nastaví je rovny nule.

```
$scatterDown = $values->scatter_down ? $values->scatter_down : 0;  
$scatterUp = $values->scatter_up ? $values->scatter_up : 0;
```

Obr. 40 – Nastavení hodnot rozptylů

V další části proběhne kontrola, zda byly zadány i další hodnoty parametrů pro vyhledávání a vybere z výběru tabulky *minerals* pouze ty záznamy, jež obsahují zadané parametry.

```
if($values->groups_id) {  
    $selection->where('groups_id', $values->groups_id);  
}  
  
if($values->name_czech) {  
    $selection->where('name_czech LIKE ?', "%{$values->name_czech}%");  
}  
  
if($values->name_english) {  
    $selection->where('name_english LIKE ?', "%{$values->name_english}%");  
}  
  
if($values->symbol) {  
    $selection->where('symbol LIKE ?', "%{$values->symbol}%");  
}  
  
if($values->source) {  
    $selection->where('source LIKE ?', "%{$values->source}%");  
}  
  
if($values->description) {  
    $selection->where('description LIKE ?', "%{$values->description}%");  
}
```

Obr. 41 – „Filtrace“ výběru z tabulky *minerals*

Samotné vyhledávání uložených x-ových souřadnic začíná deklarací pole *results*. Tento zápis je v PHP podporován od verze 5.4, dříve to bylo možné pouze pomocí `$pole = array ()`. V dalším kroku funkce *explode* vytvoří ze zadaných x-ových souřadnic oddělených mezerou pole a spočítá počet těchto zadaných souřadnic. Pak se vytvoří nové

pole *conds*, kde připočítá k x-ovým souřadnicím rozptyl. Do dalšího pole *ys* vybere z tabulky *minerals* 10 bodů s nejvyšší hodnotou y-ové souřadnice.

```
0 10 20 30 40 50 60 70 80 90
$results = [];

if($values->peaks) {

    $peaks = [];
    foreach(explode(" ", $values->peaks) as $p) {
        $peaks[] = floatval($p);
    }

    $totalPeaks = count($peaks);

    $conds = [];

    foreach ($peaks as $val) {
        $conds[] = 'x >= ' . ($val - $scatterDown) .
            ' AND x <= ' . ($val + $scatterUp) ;
    }

    $ys = [];

    foreach($this->db->table('minerals') as $m) {
        foreach($m->related('data')->select('y')->order('y DESC')->limit(10) as $row) {
            $ys[] = $row->y;
        }
    }
}
```

Obr. 42 – Část kódu pro vyhledání „peaků“

V předposledním kroku z tabulky *minerals* vybere všechny záznamy, které splňují daná kritéria, tedy obsahuje zadané x-ové souřadnice s příslušným rozptylem mezi svými deseti body s nejvyšší hodnotou y-ové souřadnice.

Záznamy splňující podmínky vyhledání poté ohodnotí procenty. Tato procenta vyjadřují, kolik procent z celkových zadaných x-ových hodnot našel v daném záznamu. Pole výsledků pak seřadí sestupně a předá šabloně pro výpis (Obr. 42).

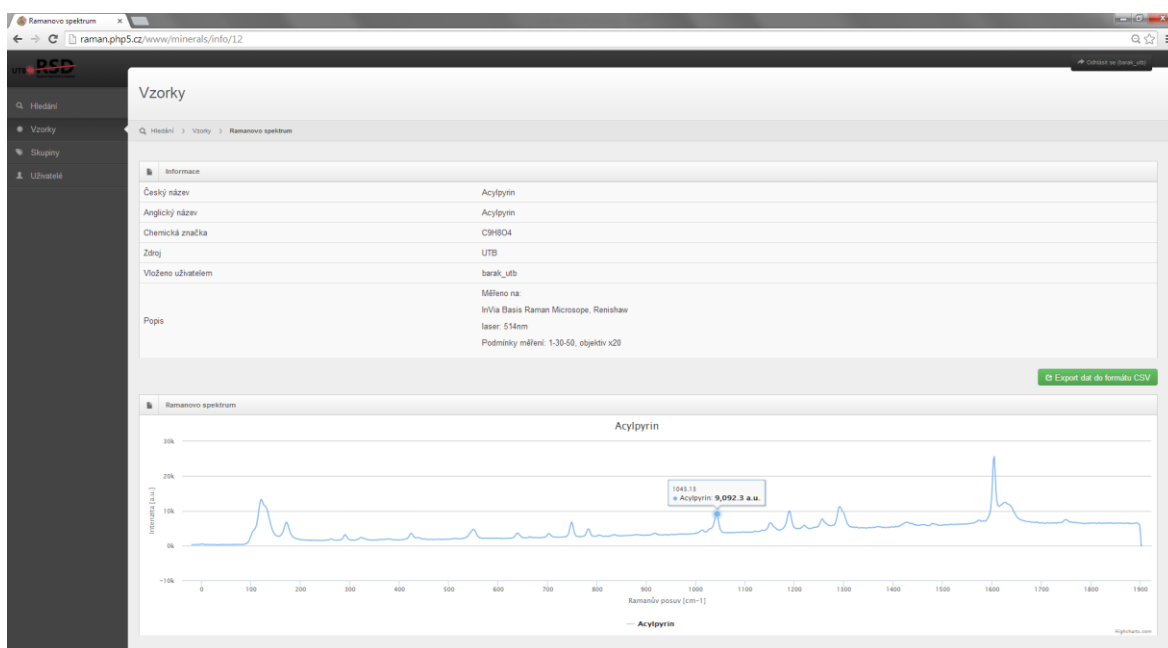
Každý prvek po kliknutí na jeho český název lze rozbalit do detailu, kde jsou vidět veškeré uložené informace včetně vykresleného Ramanova spektra (Obr.43).

Skupina	nerozhoduje
Popis	
Dolní rozptyl	1
Horní rozptyl	1
Hodnoty peaků X oddělené mezerou	520 1602 830 1540 1620

Vyhledávání			
%	Český název	Anglický název	
100	Acylpyrín	Acylpyrin	C9H8O4
80	Olivín	Olivine	(Mg,Fe)2[SiO4]
80	Křemík	Silicon	Si

Obr. 43 – Ukázka výsledku vyhledávání x-ových souřadnic



Obr. 44 – Detail uloženého vzorku

## 4.7 Uživatelé

Poslední funkční položkou této administrace je správa uživatelů. V této administraci existují dvě úrovně přístupu a to administrátor a uživatel. Uživatel má omezený přístup do této sekce správy uživatelů, kde si může pouze změnit heslo. Dalšími povolenými funkcemi pro uživatele jsou vyhledávání v databázi, vytváření nových a editace stávajících vzorků v databázi. Ostatní činnosti jsou mu odepřeny. Ověření role uživatele je provedeno v *UserPresenter.php* a *UserManager.php*. Pro vytvoření nového uživatele slouží funkce *add()* v *UserManager.php*.

```
public function add($name, $password)
{
    $this->db->table('users')->insert(array(
        'name' => $name,
        'password' => $this->calculateHash($password),
    ));
}

/**
 * Computes salted password hash.
 * @param $password
 * @param string
 * @return string
 */
public static function calculateHash($password, $salt = NULL)
{
    if ($password === Strings::upper($password)) { // perhaps caps lock is on
        $password = Strings::lower($password);
    }
    return crypt($password, $salt ?: '$2a$07$' . Strings::random(22));
}
```

Obr. 45 – Funkce pro přidání nového uživatele



Obr. 46 – Správa uživatelů

## 4.8 Umístění aplikace na hostingový server

Pro účely obhajoby této diplomové práce byla vytvořená aplikace nahrána na webový hosting serveru php5.cz. Tento server byl zvolen jako poskytovatel webového hostingu z důvodu podpory PHP verze 5 a možnosti vytvoření MySQL databáze zcela zdarma.

Přihlašovací stránka do administrace databáze je nyní dostupná na adrese:

**[raman.php5.cz/www/sign/in](http://raman.php5.cz/www/sign/in)**

po skončení obhajoby diplomové práce, bude ze serveru php5 odstraněna.

## ZÁVĚR

V teoretické části jsem se snažil objasnit základní principy Ramanovy spektroskopie, její výhody a nevýhody a dále principy, na kterých je založena. Snažil jsem se zvolit způsob podání tak, aby byla srozumitelná zejména pro čtenáře dosud neseznámené s touto spektroskopickou metodou. Uvedl jsem i stručný přehled možnosti využití Ramanovy spektroskopie v nejrůznějších oblastech. Důvody rostoucí popularity a využívání Ramanovy spektroskopie jsou její klíčové vlastnosti, rychlost měření, bezkontaktnost, nedestruktivnost, možnost aplikace na širokou škálu látek a samozřejmě unikátnost Ramanových spekter, jakožto klíčový aspekt pro identifikaci látek.

V této části jsem dále popsal základní vlastnosti použitých technologií, se kterými jsem se při své první tvorbě podobné aplikace musel seznámit.

V praktické části popisují všechny podporované funkce již hotové aplikace, zejména pro bezproblémovou práci uživatele. Dále jsou zde názorně popsány principy funkce všech klíčových součástí funkční aplikace.

Při tvorbě této aplikace byl balík frameworku Nette rozbalen rovnou na hostingový server, kde byla vytvořena doporučená struktura adresářů frameworkem Nette. Tento webový hosting byl využíván po celou dobu pro tvorbu, ladění a dokončování všech nutných zdrojových kódů a uložení použitých knihoven. Z důvodu využití tohoto webového hostingu hned na začátku tvorby aplikace, nebylo nutné využívat a zprovozňovat žádný lokální MySQL a PHP server.

V této části jsou rovněž uvedeny požadavky na funkčnost vytvořené webové aplikace, vycházející z praktických potřeb pro vyhodnocování Ramanových spekter různých materiálů, kterými jsou možnost vstupu do aplikace po přihlášení, vytvoření uživatelských rolí s omezenými právy, tedy administrátor a uživatel. Dále zde byla vytvořena možnost pro několik alternativních způsobů vložení naměřeného vzorku do databáze, přičemž podpora pro vkládání naměřených hodnot byla optimalizována pro Ramanův spektrometr od firmy Renishaw, který je využíván k těmto účelům na Fakultě aplikované informatiky Univerzity Tomáše Bati ve Zlíně. Klíčovou vyžadovanou funkcí pro určování neznámých měřených vzorků byla možnost vyhledávání v databázi dle typických hodnot vrcholů – „peaků“ naměřeného Ramanova spektra, tato možnost vyhledávání byla po větším počtu pokusů implementována správně.

Vytvořená aplikace bude používána pro efektivnější a rychlejší práci s experimentálními daty, jejich porovnáváním a vyhodnocením.

Vytvořená aplikace včetně všech zdrojových kódů a modelu databáze MySQL, vytvořeného v programu MySQL Workbench jsou dostupné v příloze této práce.

## SEZNAM POUŽITÉ LITERATURY

- [1] Venkata Raman - Biographical. *Nobelprize.org* [online]. Dostupné z: [http://www.nobelprize.org/nobel\\_prizes/physics/laureates/1930/raman-bio.html](http://www.nobelprize.org/nobel_prizes/physics/laureates/1930/raman-bio.html)
- [2] MATĚJKA, P. Ramanova Spektrometrie: Úvod – princip Ramanovy spektrometrie. *Výukové materiály VŠCHT : Ústav analytické chemie VŠCHT Praha* [online]. 2002, Dostupný z WWW: <<http://www.vscht.cz/anl/lach2/RAMAN.pdf>>.
- [3] India netzone [online]. [cit. 2014-05-11]. Dostupné z: <http://www.indianetzone.com>
- [4] MATĚJKA, P. Optika: Úvod – Závislost spekter na periodickém pohybu. *Výukové materiály VŠCHT : Ústav analytické chemie VŠCHT Praha* [online]. 2002, [cit. 2011-05-20]. Dostupný z WWW: <<http://www.vscht.cz/anl/matejka/ACH1-6-optika-c.pdf>>.
- [5] Renishaw [online]. [cit. 2014-05-11]. Dostupné z: <http://www.renishaw.cz>
- [6] VSCHT [online]. [cit. 2014-05-11]. Dostupné z: <http://www.vscht.cz>
- [7] TRCHOVÁ, CSc., Doc. RNDr. M. Jak vibrují atomy v molekulách. *Ústav makromolekulární chemie Akademie věd ČR* [online]. [cit. 2011-05-20]. Dostupný z WWW:<<http://archiv.otevrenaveda.cz/users/Image/default/C1Kurzy/NH2006pdf/15.pdf>>.
- [8] Farmaceutický průmysl. [online]. [cit. 2014-05-11]. Dostupné z: <http://www.renishaw.cz/cs/farmaceuticky-prumysl--7977>
- [9] BUZZINI, P.; MASSONNET, G. On the Contribution of Raman Spectroscopy to Forensic Science. [online]. [cit. 2011-05-21]. Dostupný z WWW: <<http://www.icors2010.org/abstractfiles/ICORS20100288.7250VER.1.pdf>>.
- [10] PITT, G. D. Engineering aspects and applications of the new Raman instrumentation. *IEE Proc.-Sci. Meas. Technol.* 2006, Vol. 152, No. 6.
- [11] NOVÉ RAMANOVY SPEKTROMETRY. *CHEMagazín: časopis pro chemicko-technologickou a laboratorní praxi*. Pardubice: Ing. Miloslav Rotrekl, 2012, XXII, č. 6, s. 8-9. Dostupné z: <http://www.spektrometry.cz/analyzator-raman-rigaku/rigaku-raman-spektrometr-bas-rudice-delta-soil-geochem-xrf.pdf>

- [12] Spektrometry [online]. [cit. 2014-05-11]. Dostupné z: <http://www.spektrometry.cz>
- [13] Nette [online]. [cit. 2014-05-11]. Dostupné z: <http://www.nette.org>
- [14] PHP: manual. [online]. [cit. 2014-05-11]. Dostupné z: <http://www.php.net/manual/en/history.php.php>
- [15] Netcraft [online]. [cit. 2014-05-11]. Dostupné z: <http://news.netcraft.com>
- [16] Artic-studio: Co je to PHP? [online]. [cit. 2014-05-11]. Dostupné z: <http://www.artic-studio.net/slovnicek-pojmu/skriptovaci-jazyk-php/>
- [17] Zdroják.cz: Java na webovém serveru: porovnání Javy a PHP [online]. [cit. 2014-05-11]. Dostupné z: <http://www.zdrojak.cz/clanky/java-na-webovem-serveru-porovnanijavy-a-php/>
- [18] Manuál PHP: PHP a jiné jazyky [online]. [cit. 2014-05-11]. Dostupné z: <http://jonatan.spse.pilsedu.cz/doc/php-man/faq.languages.html>
- [19] ŠKRÁŠEK, Jan. Programujte.com: PHP frameworky [online]. 21.2.2008. [cit. 2014-05-11]. Dostupné z: <http://programujte.com/clanek/2008022000-php-frameworky/>
- [20] Zdroják.cz: ORM test PHP frameworků. [online]. [cit. 2014-05-11]. Dostupné z: <http://www.zdrojak.cz/clanky/orm-test-php-frameworku-php-zaver/>
- [21] Zdroják.cz [online]. [cit. 2014-05-11]. Dostupné z: <http://www.zdrojak.cz>
- [22] OLÍŠAR, Petr. Porovnání Nette a Zend Frameworku [online]. 2011 [cit. 2014-05-11]. Bakalářská práce. Vysoká škola ekonomická v Praze, Vedoucí práce Jan Mittner. Dostupné z: <http://theses.cz/id/b0dugh/>.
- [23] Zdroják.cz: Nette Framework: zvyšte svoji produktivitu. [online]. [cit. 2014-05-11]. Dostupné z: <http://www.zdrojak.cz/clanky/nette-framework-zvyste-svoji-produktivitu/>
- [24] Tölg, Jan. Framework Nette [online]. 2013 [cit. 2014-05-11]. Bakalářská práce. Vysoká škola ekonomická v Praze, Vedoucí práce Ing. Pavlíčková Jarmila. Dostupné z: [https://www.vse.cz/vskp/34785\\_framework\\_nette/](https://www.vse.cz/vskp/34785_framework_nette/).
- [25] Nette.org: Zabezpečení před zranitelnostmi. [online]. [cit. 2014-05-11]. Dostupné z: <http://doc.nette.org/cs/2.1/vulnerability-protection>
- [26] Nette.org: Nette\Database vs dibi. [online]. [cit. 2014-05-11]. Dostupné z: <http://pla.nette.org/cs/nette-database-vs-dibi>

- [27] Php.vrana.cz: NotOrm. [online]. [cit. 2014-05-11]. Dostupné z: <http://php.vrana.cz/notorm.php>
- [28] Linuxsoft.cz: MySQL (1) - pestrý svět databází. [online]. [cit. 2014-05-11]. Dostupné z: [http://www.linuxsoft.cz/article.php?id\\_article=731](http://www.linuxsoft.cz/article.php?id_article=731)
- [29] MySQL [online]. [cit. 2014-05-11]. Dostupné z: <http://www.mysql.com>
- [30] Cacak web [online]. [cit. 2014-05-11]. Dostupné z: <http://www.cacak.cz>

**SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK**

API	Application Programming Interface
ASP	Active Server Pages
Cd	kadmium
E	energie
f	frekvence
GPL	General Public Licence
h	Planckova konstanta
He-Ne	Helium – Neonový laser
HTML	HyperText Markup Language
http	Hypertext Transfer Protocol
ID	identifikátor
IDE	Integrated Development Environment
IIS	Internet Information Services
Kr+	kryptonový laser
Nd:YAG	yttrium aluminum garnet laser
nm	nanometry
PDO	PHP Data Objects
PECL	PHP Extension Community Library
PHP	Hypertext Preprocessor
www	World Wide Web

**SEZNAM OBRÁZKŮ**

Obr. 1 - C. V. Raman [3] .....	11
Obr. 2 – Rayleighův rozptyl [2] .....	13
Obr. 3 – Stokesův rozptyl [2].....	13
Obr. 4 – Anti-Stokesův rozptyl [2] .....	14
Obr. 5 – Přejchody molekul mezi stavy [2] .....	14
Obr. 6 – Ramanův spektrometr značky Renishaw [5] .....	16
Obr. 7 – Uspořádání Ramanova spektrometru [6] .....	16
Obr. 8 – Přenosný Ramanův spektrometr s 1064 nm laserem [12] .....	18
Obr. 9 - Logo Nette Framework [13].....	20
Obr. 10 - Obliba jazyka PHP v posledních letech [15].....	24
Obr. 11 – Výsledky komplexního testu (ALL) frameworků [21].....	31
Obr. 12 – Životní cyklus presenteru [13].....	33
Obr. 13 – Schématická struktura MySQL serveru [30] .....	39
Obr. 14 – základní obrazovka MySQL Workbench [29].....	40
Obr. 15 – Schéma databáze v aplikaci MySQL Workbench .....	45
Obr. 16 – Struktura tabulky data .....	46
Obr. 17 – Parametry nastavené u sloupců v tabulce data.....	46
Obr. 18 – Struktura tabulky minerals.....	48
Obr. 19 – Parametry sloupců v tabulce minerals .....	48
Obr. 20 – Struktura tabulky users .....	49
Obr. 21 – Parametry nastavené u sloupců v tabulce users .....	49
Obr. 22 – Struktura tabulky category.....	50
Obr. 23 – Parametry u sloupců v tabulce category .....	50
Obr. 24 – adresářová struktura.....	51
Obr. 25 – Výřez přihlašovací obrazovky do aplikace .....	53
Obr. 26 – ukázka souboru Sign.in.latte.....	53
Obr. 27 – ukázka souboru SignInForm.php.....	54
Obr. 28 – ukázka souboru UserManager.php .....	55
Obr. 29 – Přehled vytvořených skupin.....	56
Obr. 30 – Ukázka ze souboru BasePresenter.php.....	57
Obr. 31 – Přehled všech zadaných vzorků v databázi.....	58

Obr. 32 – Náhled exportovaného CSV souboru dat .....	58
Obr. 33 – Detail dat Acylpyrinu uložených v databázi .....	59
Obr. 34 – JavaScript využívající knihovnu Highcharts .....	60
Obr. 35 – ukázka vkládaného souboru souřadnic .....	62
Obr. 36 – Ukázka vložení obrázku spektra s typickými „peaky“ .....	63
Obr. 37 – Funkce <i>save()</i> v <i>MineralDataManger.php</i> .....	63
Obr. 38 – Nabídka vyhledávání v databázi .....	64
Obr. 39 – Výpis všech údajů z tabulky <i>minerals</i> .....	64
Obr. 40 – Nastavení hodnot rozptylů .....	65
Obr. 41 – „Filtrace“ výběru z tabulky <i>minerals</i> .....	65
Obr. 42 – Část kódu pro vyhledání „peaků“ .....	66
Obr. 43 – Ukázka výsledku vyhledávání x-ových souřadnic .....	67
Obr. 44 – Detail uloženého vzorku .....	67
Obr. 45 – Funkce pro přidání nového uživatele .....	68
Obr. 46 – Správa uživatelů .....	68

**SEZNAM TABULEK**

Tabulka 1 – Přehled všech vydaných verzí jazyka PHP [14] .....	23
Tabulka 2 – Přehled databází podporovaných PHP [14] .....	27

## SEZNAM PŘÍLOH

P I: CD se zdrojovými kódy a modelem databáze MySQL