Mendelova univerzita v Brně
Provozně ekonomická fakulta

Informační systém pro základní školy

Bakalářská práce

Vedoucí práce:
Ing. Pavel Turčínek, Ph.D.

Lukáš Dubšík

Brno 2015

Rád bych poděkoval Ing. Pavlu Turčínkovi, Ph.D. za cenné rady, podnětné připomínky, vstřícnost při konzultacích a hlavně za trpělivost, kterou se mnou v průběhu tvorby bakalářské práce měl. Další poděkování patří ředitelce základní školy v Ketkovicích, Mgr. Petře Burešové, za nápady a průběžné konzultace, které pomohly tuto práci dokončit. A v neposlední řadě nesmím opomenout podporu ze strany rodiny, která mi dala zázemí a prostor pro její vypracování.

Čestné prohlášení

Prohlašuji, že jsem tuto práci: **Informační systém pro základní školy**
vypracoval/a samostatně a veškeré použité prameny a informace jsou uvedeny v seznamu použité literatury. Souhlasím, aby moje práce byla zveřejněna v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách ve znění pozdějších předpisů, a v souladu s platnou *Směrnicí o zveřejňování vysokoškolských závěrečných prací.*

Jsem si vědom/a, že se na moji práci vztahuje zákon č. 121/2000 Sb., autorský zákon, a že Mendelova univerzita v Brně má právo na uzavření licenční smlouvy a užití této práce jako školního díla podle § 60 odst. 1 Autorského zákona.

Dále se zavazuji, že před sepsáním licenční smlouvy o využití díla jinou osobou (subjektem) si vyžádám písemné stanovisko univerzity o tom, že předmětná licenční smlouva není v rozporu s oprávněnými zájmy univerzity, a zavazuji se uhradit případný příspěvek na úhradu nákladů spojených se vznikem díla, a to až do jejich skutečné výše.

V Brně dne 20. května 2015 _____

Abstract

Dubšík, L. Information system for elementary schools. Bachelor thesis. Brno: Mendel University, 2015.
This bachelor thesis describes development of new information system for elementary school, in this case, it is an elementary school in Ketkovice. Listed here are the technologies that were used in its creation, a description of the development, and then compare the original with the new solution. Beyond the creating of new information system, the text describes the additional options that elementary schools have.

Keywords

Information system, elementary school, PHP, Nette Framework, web application, MySQL, MVC.

Abstrakt

Dubšík, L. Informační systém pro základní školy. Bakalářská práce. Brno: Mendelova univerzita v Brně, 2015.
Tato bakalářská práce popisuje vývoj nového informačního systému pro základní školy, v tomto případě se jedná o základní školu v Ketkovicích. Jsou zde uvedeny technologie, které byly použity při jeho tvorbě, popis jakým způsobem vývoj probíhal a následné porovnání původního řešení s řešením novým. Mimo tvorby nového informačního systému, text popisuje i další možnosti, které základní školy mohou využít.

Klíčová slova

Informační systém, základní škola, PHP, Nette Framework, webová aplikace, MySQL, MVC.

Obsah

Seznam obrázků

1 Úvod a cíl práce

1.1 Úvod

Informační systémy jsou na základních školách poměrně běžné, ale ne vždy splňují veškeré požadavky, které si školy nárokují. Problém je většinou u malých a středních základních škol, kde je organizace a struktura mírně rozdílná od klasických základních škol a je těžké využít některé z již fungujících dostupných řešení nebo zaplatit vývoj nového, který by zahrnul veškeré potřeby a požadavky, protože na tento vývoj by většinou tyto školy neměli dostupné finanční prostředky.

Další skutečností je fakt, že mnohdy základní školy nemají přehled a platí za určitou službu a řešení vyšší finanční částku, než je její skutečná hodnota. U větších škol tento problém není tak výrazný, jak právě u menších škol, kde finančních prostředků není nazbyt a mohly by se využít lépe nebo využít právě na takové řešení, které by odpovídalo ceně, jakou školy platí.

Mimo informační systém je zde i samotná prezentace na internetu, každá webová aplikace by měla být přehledná a použitelná, aby se s ní lehce pracovalo. Webová stránka by měla také být graficky uhlazená a příjemná na pohled. U stránek základních školy by toto mělo platit vždy.

Stěžejním bodem je komunikace žáků s učiteli, případné přidání doplňujících informací ke vloženému příspěvku. Pro neveřejnou komunikaci by rodiče měli být schopni lehce nalézt kontakt na daného učitele či zaměstnance školy a u obecné otázky nebo připomínky by měla být možnost ji k příspěvku, například prostřednictvím komentáře vložit.

1.2 Cíl práce

Cílem této práce je navržení a vytvoření informačního systému pro základní školu v Ketkovicích, který nahradí aktuální řešení, neodpovídající současným požadavkům. Informační systém bude navržen jako moderní webová aplikace, která bude sloužit, jak k prezentaci základní školy, tak i jako komunikační kanál mezi rodiči žáků a učitelů, případně ostatních zaměstnanců školy. Tato webová aplikace by měla být uživatelsky přívětivá, tzn. přehledná tak, aby každý, kdo webové stránky navštíví, lehce nalezl přesně to, co hledá. Aplikace by také měla být jednoduchá na ovládání, jelikož s daným systémem budou pracovat osoby ve všech věkových kategoriích a s různými znalostmi.

2 Informační systémy a webové aplikace

Informační systémy jsou důležitým nástrojem podniků, pomáhají jim efektivně fungovat a ulehčovat práci. K vývoji a tvorbě informačních systémů lze přistupovat několika způsoby, v každém případě záleží na potřebách organizace, na tom k čemu má být informační systém uzpůsobený a kdo s ním bude pracovat.

Webovou aplikaci v tomto případě můžeme popsat jako možnost, kterou můžeme využít pro informační systém.

2.1 Definice informačního systému

Před definicí informačního systému (IS), je důležité, definovat samotný pojem systém. Mnohé složité věci jsou jako celek více než jen souhrn částí, ze kterých se skládají (Aristoteles). Pro tyto složité věci používáme pojem systém. Celek složitých věcí mívá, na rozdíl od pouhého souhrnu částí, svou kvalitu, obvykle se jeví například tak, že má svou podstatu, svůj účel nebo cíl, anebo specifické účelové či cílové chování (Bruckner, 2012). Jednodušeji řečeno systém je uspořádaný celek nebo soustava věcí.

Informační systém je tedy komplex informací, lidí, použitých informačních technologií, organizace práce, řízení chodu systému (zabezpečuje propojení na prostředí) a metod sloužících ke sběru, přenosu, uchování a dalšímu zpracování dat za účelem tvorby a prezentace informací (Rábová, 2008).

2.2 Přístupy k analýze k IS

Pro přístup k analýze a návrhu informačních systému máme dva základní přístupy:

- Strukturovaný přístup.
- Objektově orientovaný přístup.

Strukturovaný přístup je zde zmíněn jen okrajově, protože při vývoji nebude využit. Tento přístup je tedy klasickou metodou analýzy a návrhu informačních systémů, navržený v 70. letech. Popisuje systém pomocí dvou modelů, funkčního (popis funkcí a transformace dat) a datového (struktura dat v databázi). Oba modely rozkládají daný problém na jednodušší části. Při vývoji jsou modely tvořeny odděleně, ale závisle na sobě tzn., změna u jednoho modelu, ovlivňuje model druhý a naopak. (Mikulášek, 2008).

Objektově orientovaný přístup, nahlíží na svět jako systém objektů, které jsou ve vzájemné interakci (Procházka a kol., 2010). Jelikož OOP bude použit při vývoji nového IS, tak je blíže popsán v následující kapitole 2.3.

2.3 Objektově orientované programování (OOP)

Základní myšlenkou objektově orientovaného programování je pohled na program jako na systém objektů. Tento přístup je často více intuitivní, protože svět je přirozeně chápán jako systém objektů. Objekt v OOP je samostatnou částí, která je vhodným způsobem propojena s ostatními objekty. Jednotlivé objekty jsou tedy částmi systému, které ve vzájemné interakcí vytvářejí jeho funkčnost.

Při využití tohoto přístupu se naskytuje mnoho příležitostí a výhod, které lze využít. Obecná výhoda je lepší využitelnost kódu, než například u přístupu strukturovaného přístupu. Tuto využitelnost popisují následující myšlenky a koncepce. (Procházka a kol., 2010).

Objekty

Neboli také instance tříd jsou konkrétní části systému, definované příslušnou třídou a konkrétními hodnotami vlastností. Objekt by měl být samostatnou funkční jednotkou programu (Procházka a kol., 2010).

Třídy a abstrakce

Třída objektu definuje všechny jeho vlastnosti a přípustné hodnoty a rozsahy. Definuje také funkce objektu, tedy jeho chování a reakce na okolí. Třída je tedy takový abstraktní vzor (šablona) objektu, která definuje jeho vlastnosti (atributy) a metody (funkce) společné pro každý objekt (Procházka a kol., 2010).

Abstraktní třída (abstrakce) je třída, která obsahuje jak abstraktní metody, tak hotové metody. Tyto abstraktní třídy nesmějí vytvářet instance, ale mohou být děděny a jejich potomci po implementaci potřebných metod instance vytvářet mohou. Abstraktní nemusejí být jen třídy, abstraktní mohou být i metody, ale nemohou definovat obsah, pouze specifikují název a případné parametry.

Třídy, které obsahují abstraktní metody a uvádějí jejich definici, musejí být označeny se stejnou nebo menší viditelností než abstraktní metody. Je-li tedy metoda označena jako protected, potomek abstraktní třídy může metodu uvádět pouze jako public nebo protected. (Mrozek, 2006).

Zapouzdření

Skrývání implementace se používá pro úplnou ochranu stavových atributů objektu. Tyto atributy by měli mít povolenou modifikaci pouze přes metody k tomu určené. Atributy a metody v rámci třídy jsou přístupné vždy. Pokud se ovšem přistupuje k metodám a atributům objektu mimo danou třídu z jiných metod a funkcí, atributy a metody již přípustné být nemusí. K nastavení přístupu slouží následující modifikátory viditelnosti:

- Public (veřejné) – Takto označené atributy může kdokoliv číst a modifikovat, metody označené tímto modifikátorem, mohou být volány z jiných metod a funkcí.

- Private (soukromé) – Atributy mohou být čtené a modifikované pouze z metod dané třídy, podobně i metody mohou být volané pouze z jiných metod dané třídy,
- Protected (chráněný) – třetí modifikátor viditelnosti povoluje přístup z metod vlastní třídy a vlastních potomku.

Skrývání implementace je velice důležitý rys objektového návrhu a je velmi důležité jej dodržovat. (Procházka a kol., 2010).

Dědičnost

Jeden z klíčových nástrojů OOP je dědičnost, která umožňuje vytváření nových tříd, jen nastavením změn oproti jiné třídě. Tento způsob vývoje zásadně přispívá ke zrychlení vývoje aplikací a bývá nazýván diferenciální programování (Procházka a kol., 2010).

Když tedy občas potřebujeme vytvořit třídu podobnou té, kterou již máme definovanou, není potřeba vytvářet třídu novou, stačí pouze pozměnit stávající třídu, proces vytváření nové třídy, která přebírá libovolné metody a atributy se právě nazývá dědičnost (Mrozek, 2006).

Polymorfismus

Polymorfismus bývá označován jako „pozdní vazba“. Pro bližší definici pozdní vazby je nutné nejdříve specifikovat, co je to „časná vazba“. Časná vazba je stav, kdy překladač přesně ví, jaká metoda se bude volat. Vytvoří se objekt a zavolá jeho metoda.

Pozdní vazba se objevuje v souvislosti s děděním. Platí vlastnost, že potomek je vždy specializovanou verzí předka. V určitém slova smyslu může být řečeno, že potomek je „kompatibilní“ s předkem. Tato vlastnost je využívána takovým způsobem, že po vytvoření odkazu na předka, lze tento odkaz „naplnit“ instancí potomka. (Procházka a kol., 2010).

Polymorfismus umožňuje používat jednotné rozhraní pro práci s různými typy objektů. V praxi to znamená, že u předka je definována určitá metoda a u potomka (podtřídy) se tato metoda může přepsat.

Podstatou polymorfismu je tedy metoda nebo metody, které mají všichni potomci definované se stejnou hlavičkou, ale jiným tělem. (Čápka, 2015).

2.4 Životní cyklus informačního systému

Od prvotního požadavku až k ukončení prochází informační systém několika různými fázemi. Souhrnně jsou tyto fáze nazývány životním cyklem informačního systému. Jejich sled v praxi neprobíhá lineárně, tak jak je zde následně uvedeno. Některé fáze se mohou přerývat či opakovat.

K návrhu a řízení vývoje informačního systému existuje řada modelů, například model vodopád, iterativní model a další. Ovšem při reálném využití často není

využíván jeden konkrétní model, ale spíše kombinace modelů podle potřeby daného IS. (Rybička, 2009).

2.4.1 Analýza

Analýza je nezbytným krokem před návrhem IS a počíná prvotním požadavkem na jeho tvorbu. Je prováděna kvůli zjištění potřeb a požadavků dané organizace, seznámení se s její strukturou a získaní potřebných informací pro návrh. Při analýze jsou vytvářeny různé modely a diagramy, které slouží pro názornou ukázku a komunikaci mezi zadavateli a analytiky. Mohou být využity například tyto:

- Use Case – Diagram případů užití
- Funkční model – reprezentace a popis funkcí a vazby mezi nimi
- Datový model – ERD (entitně-relační diagram)

K reprezentaci modelů se využívá různých forem zápisu například grafické znázornění, matice chování, popis slovy a jiné.

Výsledkem analýzy je tedy kompletní rozbor daného problému a poukázání na problematické části, které by mohli být problémem pro návrh. (Rybička, 2009).

2.4.2 Návrh

Tato fáze je výsledkem analýzy systému. Výsledkem bývá dokument, jako podklad pro obsah smlouvy s externí firmou o návrhu a realizaci IS, časový harmonogram, cena vyvíjeného projektu, konkrétní implementace systému (zde je zahrnut logický a fyzický datový model), podmínky zavádění v organizaci, záruka a podmínky pro konečné předání IS. Tato studie musí obsahovat:

- Základní informace o tvůrcích systému.
- Základní informace o organizaci, pro kterou je systém vyvíjen.
- Popis současného stavu organizace.
- Globální návrh IS – logický datový model, je návrhem funkcí dat bez ohledu na technologické prostředí.
- Detailní návrh IS – fyzický datový model obsahuje funkční a datovou analýzu systému, popis datových toků v organizaci a popis funkcí řízených událostmi. Výstupem je návrh funkcí a dat budoucího systému.
- Detailní popis nasazení IS v praxi.
- Detailní popis testovacího provozu systému.
- Celkový harmonogram spolupráce.

Při tvorbě studie se nesmí opomenout uvést veškerá fakta v dostatečně detailní podobě a provedení, která bude pochopitelná všem členům vedení, kteří provádí závěrečná rozhodnutí.

V případě dohodnutím mezi organizací a tvůrci systémem slouží tato studie jako podklad realizace systému a podklad pro podmínky předání a testování. (Šmíd, 2002).

2.4.3 Implementace

Tato část životního cyklu IS je vlastním programováním, kterého se účastní vybraní experti v programování a analytik nesoucí zodpovědnost za správnost řešení. Jako podklady pro jejich práci slouží veškeré informace shromážděné předchozími etapami a fyzický návrh systému (Šmíd, 2002).

Je třeba vytvořit databázi a naplit ji částečnými daty. V této fázi mohou být využívány testovací data, ale je potřeba připravit se na převod dat z původního nahrazovaného systému do systému nového. Je také nutné zvolit metodu zavádění informačního systému.

Programové moduly a aplikace jsou vytvářeny podle vzorových řešení nebo jsou použity nástroje pro jejich tvorbu. Průběh tvorby aplikací by také měl být provázen doprovodnými dokumentacemi – uživatelskými (pro školení uživatelů pracující se systémem), provozní (pro zajištění správného chodu systému) a programátorské (pro dodatečné úprav či oprav chyb). Aplikace jsou průběžně testovány, zda poskytují korektní výsledky a je kontrolována rychlost odezev. Výsledky testů jsou následně zohledňovány při dalším vývoji. Implementace také může být doprovázena základním školením uživatelů. (Rybička, 2009).

2.4.4 Testování a zavedení systému

V etapě testování se provádí přípravné testy na hotovém IS. Je nutné vyzkoušet veškeré možné situace, které mohou nastat při zadávání dat a následné zjištěné nedostatky opravit. Testování se často provádí na systému, který ještě není reálně nasazen a neprobíhá v reálném prostředí, neboť selhání by mohlo mít rozsáhlé následky. Příkladem mohou být systémy ve zdravotnictví, letectví, jaderném průmyslu, apod.

Zaváděním systému je především míněna jeho instalace, a postupné zavedení do provozu organizace, transformace původní datové základny tak, aby byla přístupná novému systému, poskytují se manuály a školení uživatelům. Tato etapa se nesmí v žádném případě podcenit, jelikož její zanedbání by mohlo způsobit u budoucích uživatelů averzi vůči novému systému a tím neúspěch celého projektu.

Zavádění systému může být řešeno několika způsoby:

- Souběžná strategie – Při použití souběžné strategie jsou po určitou dobu v provozu oba systémy (původní i nový) současně. Provoz obou systémů trvá několik pracovních cyklů, dokud nový systém nepracuje spolehlivě a uživatelé s ním nejsou dostatečně seznámeni. Tato strategie je bezpečná, ale náročná pro zaměstnance jelikož musejí provádět dvakrát totéž, což by mohlo vést k averzi vůči systému novému. Proto se na toto období najímají externí pracovníci.

- Pilotní strategie – Při uplatnění pilotní strategie se systém zavede a vyzkouší v jednom oddělení podniku, po odsouhlasení je zaveden v celém podniku. Jako pilotní část se vybírá taková, které je poměrně náročná a je možné si na ni ověřit co nejvíce problémových oblastí.
- Postupná strategie – Tato strategie se využívá zejména u složitějších systémů, kde jsou složité vnitřní vazby. Nejprve se zavádějí primární části IS, na kterých ostatní části závisí a po jejich ověření se podobným způsobem zavedou ostatní části až po zavedení celého systému.
- Nárazová strategie – Ve výjimečných případech lze využít i nárazovou strategii, kdy je původní systém vystřídán novým takřka bez přechodové fáze, tato strategie se využívá pro ušetření času a pracovní síly.

(Šmíd, 2002).

2.4.5 Zkušební provoz a údržba

Zkušební provoz je celková realizace projektu, ve které je poskytovatel povinen zajistit okamžitý servis, odstranit chyby zjištěné během provozu, nebo dořešit dodatečné požadavky uživatelů v rámci původního návrhu (Šmíd, 2002).

Při běžném provozu systému by mělo docházet k pravidelné údržbě hardwaru a softwaru. Hardwarové závady jsou řešeny prostřednictvím servisních oprav nebo nákupem nového zařízení. Hardwarem může být také myšlen pronajatý webhosting[1], v tomto případě je údržba hardwaru prováděna na dálku a při nalezené chybě či nutnosti rozšíření se komunikuje s poskytovatelem. Při provozu může probíhat postupné posilování výkonu nebo vytváření záložních serverů.

Zásahy do softwaru IS jsou prováděny z následujících důvodů:

- Oprava nalezených chyb
- Modifikace IS dle přání zákazníka
- Rozšiřování IS – doplňování IS o požadované funkce

Po dokončení oprav a modifikací může být součástí údržby také přeškolení uživatelů. (Rybička, 2009).

2.4.6 Reengineering, ukončení provozu

Důvodem ukončení provozu IS často může být tzv. reengineering. Ten v podstatě znamená zásadní přehodnocení a radikální rekonstrukci (redesign) podnikových procesů tak, aby mohlo být dosaženo dramatického zdokonalení z hlediska kritických měřítek výkonnosti, jako jsou náklady, kvalita, služby a rychlost. Jde tedy o nový začátek ne jen o vylepšení struktury nebo provedení dílčích změn. Provést reengineering firmy znamená odhodit staré systémy a začít znovu. Jeho součástí je návrat k počátku a k nalezení lepších způsobů práce. (Hammer, 2010).

[1] Webhosting je pronajmutí prostoru pro provoz webových stránek (aplikací) na cizím serveru.

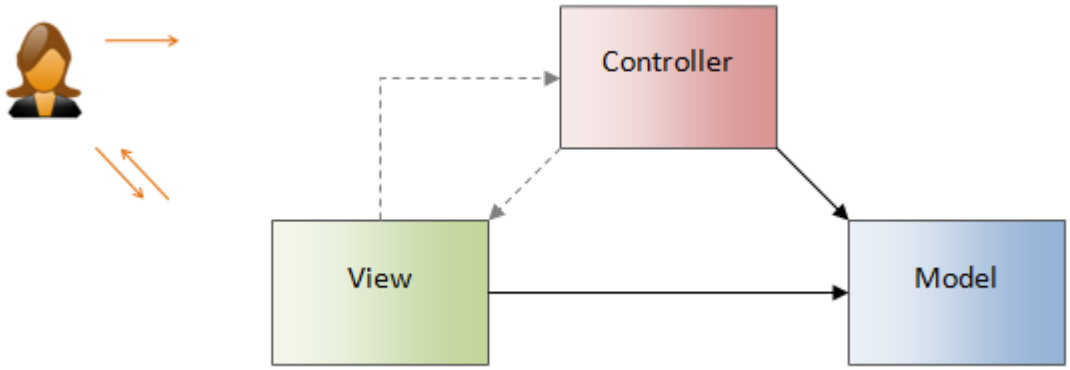
2.5 Architektura MVC a MVP

Architektura MVC dělí aplikaci na 3 logické části tak, aby mohli být samostatně upravovány a dopad změn byl na ostatní části co nejmenší. Části jsou následující:

- Model – reprezentuje data a business logiku aplikace.
- View – zobrazuje uživatelské rozhraní.
- Controller – má na starosti tok událostí v aplikaci a obecně aplikační logiku.

Přístup k MVC je potřeba rozdělit na obecnou architekturu (přístup k tvorbě aplikací) a možné variace MVC, protože variací může být několik.

Obecnou strukturu a návaznost jednotlivých komponent představuje následující obrázek 1:



Obr. 1 Obecný MVC přístup

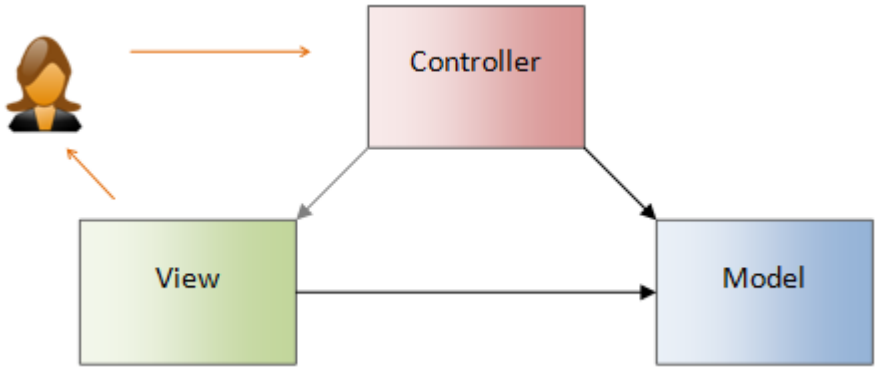
Jak lze na obrázku 1 vidět, tak controller má přímý odkaz na Model, aby mohl upravovat jeho data a view má přímý odkaz na model, aby mohl tyto data zobrazovat. Vazba mezi view a controllerem je někdy jednosměrná a jindy obousměrná. Dále je zde znázorněno to, jak může probíhat komunikace mezi uživatelem a systémem.

Pro variace existuje toto rozdělení, jednotlivé vzory budou popsány následně.

- Na platformách, kde je oddělený vstup a výstup je aplikovatelný vzor MVC, kde C (controller) zpracovává vstup.
- Když je platforma založená na konceptu ovládacích prvků je aplikovatelný vzor MVP, kde vstup zpracovává view a úloha presenteru se mění.

(Bernard, 2009).

Vzor MVC



Obr. 2 Vzor MVC

Schéma, tedy obrázek 2 popisuje vzor MVC. Tok událostí vypadá následně:

1. Uživatel vykoná akci na uživatelském rozhraní.
2. Tato akce je zachycena controllerem.
3. Controller rozhodne, jak na akci zareagovat a změní hodnoty v modelu nebo ovlivní view.
4. View se zobrazí uživateli.

Jednotlivé komponenty vypadají následovně:

Model je nejjednodušší částí, obsahuje data a business logiku, s konkrétní prezentací nemá nic společného.

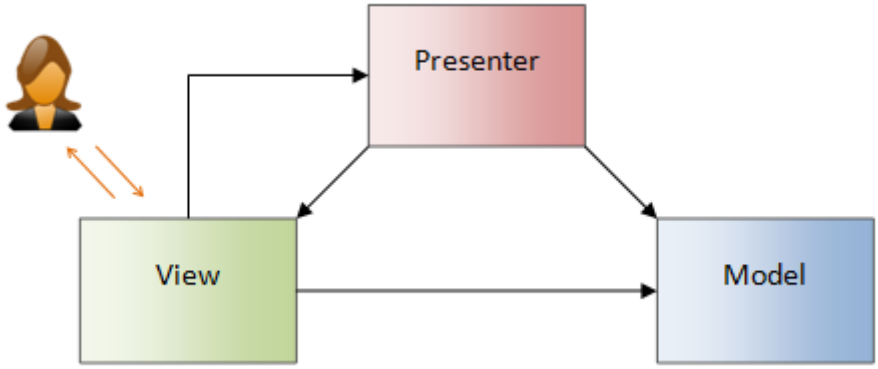
View je v tomto případě serverový kód, který se stará o generování HTML, XML nebo JSON.

Controller se ve webovém prostředí nejčastěji skládá ze dvou částí. První je tzv. Front Controller, který zachytává veškeré HTTP požadavky a následně je zpracuje a přeposílá konkrétním controllerům. Druhou částí jsou konkrétní controllery, které tyto požadavky zpracují, uloží je do modelu a prováží s konkrétním view, které už se stará o vygenerování HTML nebo jiného formátu.

(Bernard, 2009).

V následujících schématech jsou popsány vzory MVP, obrázek 3 představuje Supervising Controller a obrázek 4 Passive View, tyto vzory představuje a popisuje Bernard ve svém článku na webu (zdrojak.cz, 2009).

Vzory MVP

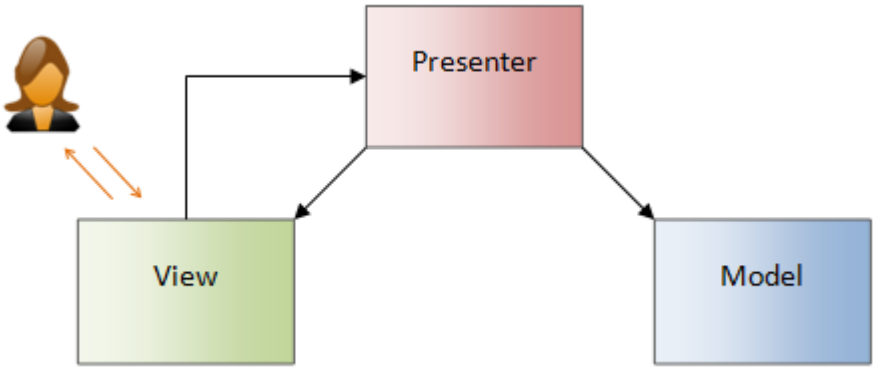


Obr. 3 MVP - vzor Supervising Controller

Oproti MVC uživatelský vstup a výstup komunikuje plně s view pomocí uživatelského rozhraní. Primárním důvodem bylo oddělení view a presenteru.

Část view zde navíc oproti MVC zpracovává uživatelský vstup. Například v reakci na kliknutí myši určitou metodu presenteru (deleguje uživatelské akce).

Presenter obsahuje aplikační a prezentační logiku. Manipuluje s modelem a to nám zajistí aktualizací nebo přímo ovlivnění view.



Obr. 4 MVP – vzor passive view

Jediným podstatným rozdílem oproti předchozímu vzoru obrázku 3, je chybějící vazba z view na model. Presenter má tak daleko větší zodpovědnost jelikož je zde veškerá logika. Hlavním důvodem bylo pokrytí automatický testů. (Bernard, 2009).

2.6 Webová aplikace

Webovými aplikacemi jsou chápány aplikace, které běží na webovém serveru a uživatelé se k nim dostanou prostřednictvím internetu či intranetu. Ke spuštění webových aplikací je využíván klient (především webový prohlížeč), který umí zobrazovat a zpracovávat přijatá data ze serveru. (Darie, 2006).

3 Metodika

3.1 Východiska řešení

Před každým nasazením nového informačního sytému nebo aplikace je nutné mít povědomí o aktuálním stavu. Je tedy třeba provést kompletní analýzu, získat veškeré potřebné informace a rozhodnout se, zda je lepší využít některé z již fungujících řešení nebo vytvořit řešení nové.

V tomto případě se jedná o malou základní školu v Ketkovicích, jejíž struktura je odlišná od běžných základných škol. Tato škola má dvě třídy žáků, kde v jedné třídě jsou žáci 1. a 3. ročníků a ve druhé jsou žáci 2., 4. a 5. ročníků. Dále škola obsahuje také školku, družinu a jídelnu.

Po konzultaci s vedením školy, mi bylo řečeno, že aktuální řešení webových stránek je již nevyhovující. Jedna z nevyhovujících věcí je již nepodporovaný redakční systém, tento redakční systém má navíc nepřehlednou administrační část a mnoho uživatelů se v ní ztrácí. Dalším nedostatkem je struktura stránek, na stránkách nejsou relevantní odkazy nebo odkazy nefungující. To je nejspíše způsobeno proto, že systém byl převzat z jiné školy a nebyl úplně uzpůsoben pro tuto školu.

Za úkol tedy je navrhnout řešení, které bude splňovat všechny požadavky a zahrnovat veškeré potřeby, které tato základní škola bude mít.

3.1.1 Využití dostupných informačních systémů

Při návrhu řešení zde byla otázka, zda nevyužít některý z již dostupných informačních systémů. Fungujících systému informačních systémů pro IS je poměrně mnoho, zde jsou uvedeny některé z nich:

- aSc Rozvrhy (Applied Software Consultants)
- Bakaláři (Bakaláři Software)
- dm Vysvědčení, dm Evidence, dm Knihovna (dm Software)
- Etřídnice
- iškola (Computer Media)
- RELAX KEŠ (Alis)
- SAS (MP-Soft)
- Škola OnLine (CCA Group)

(Neumajer, 2010).

Pro bližší popis byl vybrán následující systém *Škola OnLine*. Jedná se tedy o školní informační systém, který zpracovává veškerou školní agendu, tento IS byl také vytvořen jako webová aplikace. Umožňuje správu zaměstnanců, integraci se službami Office 365, propojení s docházkovým systémem. Učitelé mají k dispozicí hodnocení, elektronickou třídní knihu a různé výukové zdroje a další. Pro žáky je

k dispozici přehled docházky, hodnocení, rozvrh, výukové materiály, komunikace a další. (ŠKOLA ONLINE a.s., 2015).

Hlavním důvodem, proč nebude vybrán žádný z uvedených informačních systému, je ten, že tyto informační systémy jsou poměrně nákladné a obsahují mnoho funkcí, které by byly nevyužité. Proto vyvíjený informační systém bude zdarma a volně rozšiřitelný.

3.1.2 Nástroje pro správu webu

Dalším východiskem bylo využít některý z redakčních systémů a následně jej upravit. Tyto redakční systémy jsou určené převážně pro prezentační stránky, a i když existuje mnoho rozšíření (pluginů/modulů), které se mohou využít, tak ne vždy je vhodné jít právě touto cestou. Vybrané redakční systémy:

- **WordPress** – je nejpoužívanějším open source redakčním systémem vyvíjen pod licencí GNU GPL[2]. Mezi jeho hlavní výhody patří pravidelné aktualizace, jednoduchá modifikovatelnost a bezplatnost. Jednou z diskutovaných otázek je bezpečnost WordPressu. Kvůli jeho popularitě a velké nabídce šablon a doplňkových modulů je jeho kód považován za zranitelný. Na druhou stranu pravidelně vycházejí opravy chyb, které mají uživatele k dispozici zdarma. Pokud jde o výkonnost, tak dle několika měření se snižuje s rostoucí velikostí databáze a počtem stránek. Mnoho volně dostupných šablon a modulů není aktualizováno a jsou bez aktivní uživatelské podpory, takže je mnohdy obtížné sestavit z nich kvalitní funkční celek (iParťák s.r.o., 2015).
- **Joomla!**
- **Drupal**

Podobně jako WordPress jsou i Joomla a Drupal populárními redakčními systémy, každý z vyjmenovaných redakčních systémů má svoje přednosti, ale všechny jsou volně dostupné a vyvíjeny pod licencí GNU GPL, nechybí jim podpora češtiny a běží na PHP s využitím MySQL.

Redakční systémy jsou velmi dobrými a užitečnými nástroji, nicméně, nebude využit žádný se zmíněných, protože požadavky od základní školy jsou poměrně specifické. A využití redakčního systému by spíše přitížilo, než ulehčilo tyto požadavky splnit.

3.2 Metodika řešení

Vývoj informačního systému bude probíhat následovně.

Jako první proběhne seznámení se strukturou základní školy, od vedení se získají potřebné informace a požadavky na nový IS, proběhne tedy analýza. Na základě této analýzy bude navrhnuto vhodné řešení a následně bude vytvořen use-case diagram, popisující to, jak budou uživatelé se systémem komunikovat a pra-

[2] GNU GPL je licence pro svobodný software. Je to nejpopulárnější licence a software šířený pod licencí GPL je možno volně používat, modifikovat a šířit.

covat, bude popsána funkcionalita systému a po celkovém schválení tohoto řešení bude navrhnut datový model databáze. Když bude schválen i datový model, tak se následně vybere volně dostupná šablona nebo případně se navrhne nový vzhled, který bude využíván a po odsouhlasení celého návrhu se přejde k samotné implementaci. Implementace je v tomto případě programování. Toto programování aplikace bude probíhat v jazyce PHP s využitím Nette Frameworku. Veškerá funkčnost se převede z funkčního návrhu do aplikace. K uchování dat bude využita MySQL databáze, převede se tedy ERD model do databáze vytvoří se potřebné indexy a relace.

Základní vývoj bude probíhat odděleně, po dokončení implementace se aplikace přesune na testovací server a bude volně dostupná, zde bude běžet do doby, než bude kompletně otestována a schválena.

Testování aplikace bude mít několik fází, první testování bude pouze testování vlastní, plnění zkušebními daty, testování bezpečnosti apod. Druhá fáze testování bude probíhat společně s prvním zaškolováním, budou zaznamenány základní nedostatky a budou poznamenány doplňující či chybějící části. Třetí a poslední fází testování bude zapojení zaměstnanců školy případně i rodičů žáků a po kompletním schválení a opravě chyb bude aplikace připravená k nasazení.

3.3 Využité nástroje a technologie

3.3.1 Use case – Diagram případů užití

Tento diagram znázorňuje funkcionalitu a chování systému z pohledu uživatele. Vypracování tohoto diagramu je obsahem use case analýzy. Tento diagram má 4 základní stavební prvky:

- **Aktor** – může být uživatel, komponenta, jiný systém nebo čas. Je to tedy kdokoliv nebo cokoliv mimo systém, kdo s ním nějakým způsobem komunikuje nebo pracuje.
- **Use case** – případ užití, je to jakákoliv funkčnost, kterou aktor může vykonávat, reprezentuje ucelenou funkcionalitu, daného problému
- **Vztah** – vztahy mezi aktorem a use case (vazba udávající jakým způsobem může aktor komunikovat se systémem), dalšími vztahy jsou mezi use case (jednotlivé use case mohou mezi sebou spolupracovat, tyto vztahy slouží ke zjednodušení modelu) máme k dispozici tyto vztahy
    - Vazba include – vztah, který zahrnuje použití dalších use case, používá se například tam, kde se část use case může opakovat.
    - Vazba extend – rozšiřující vztah za určitých podmínek se vykonávají oba use case, používá se pro volitelné chování za specifických podmínek.
    - Vazba gen / spec – tato vazba se užívá pro zobecnění nebo odvození funkcionality.
    - Vztahy mezi samotnými aktory jsou vazby gen / spec, které odvozují typy aktorů.

- **Hranice systému** – vymezuje use case a aktory, zobrazení hranice je nepovinné.

(Rábová, 2008).

3.3.2 ERD – Entitně relační diagram

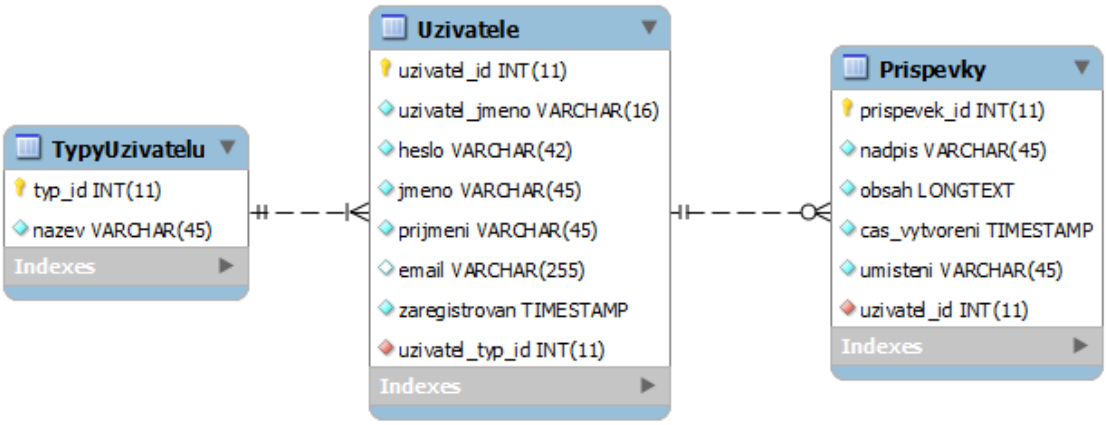
Datový model neboli ERD slouží k zobrazení datové struktury systému, je několik způsobů jakým mohou být tyto diagramy prezentovány. V této práci však bude využito následující zobrazení (notace).

Entity

Entity jsou v databázi jednotlivé tabulky a jejich atributy označují sloupce tabulky.

Obr. 5 ERD - entita a její atributy

Vztahy



Obr. 6 ERD – znázornění vztahů

Vztahy jsou zakreslovány přerušovanými čarami. Povinnost výskytu určují tyto značky, prázdně kolečko (nepovinné) nebo kolmá čára (povinné). Tzv. vidlice určují kardinalitu (násobnost) vztahu.

V tomto příkladu na obrázku 6, uživatel nemusí vlastnit žádný příspěvek nebo příspěvků může mít několik, ovšem příspěvek má vždy přiřazeného uživatele.

3.3.3 Nette Framework

Nette je kompletním Framework pro PHP, který výrazně zjednodušuje tvorbu webových aplikací. Autorem tohoto frameworku je český vývojář David Grudl. Nejrozšířenější je tento framework převážně v Čechách fungují na něm velké projekty jako např. GE Money, Slevomat, ČSFD.

Nette je tedy klasickým MVC frameworkem, i když je často představován jako MVP, tak v tomto případě je presenter spíše synonymem pro controller.

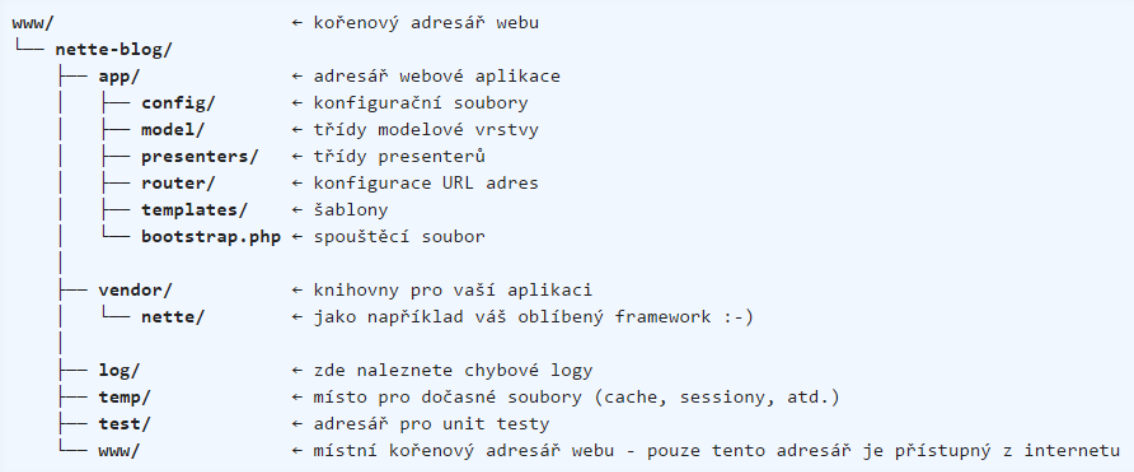
Pro využití framevorku je obecně mít dobré zkušenosti s PHP a objektově orientované programování a znát alespoň základy MVC architektury. (Čápka, 2015).

MVC v Nette.

MVC architektura již byla představena, ale není špatné si ji ještě jednou zopakovat jinými slovy MVC je tedy softwarová architektura, která u aplikací s grafickým rozhraním odděluje kód obsluhy (controller) od kódu aplikační logiky (model) a od kódu zobrazení (view). Tím aplikaci zpřehledňuje a usnadňuje vývoj a umožňuje testování jednotlivých částí:

- Model – je datový a zejména funkční základ celé aplikace obsahuje aplikační logiku, jakoukoliv uživatelskou akci (přihlášení, změna hodnoty v databázi, apod.). Model si spravuje svůj vnitřní stav a ven nabízí pevně dané rozhraní a voláním jeho funkcí můžeme zjišťovat či měnit jeho stav. Model o existenci view nebo controlleru neví.
- View – je vrstva aplikace, která má na starost zobrazení výsledku požadavku. Obvykle je použit šablonovací systém a view tak ví, kterou komponentu nebo získání výsledek z modelu má zobrazit.
- Controller – Zpracovává požadavky od uživatele a tyto požadavky zpracovává, na jejich základě volá požadovaný model a poté view požádá o vykreslení dat obdobou controllerů v Nette jsou presentery.

(Nette, 2015).



Obr. 7 Nette 2.1 – sandbox
Zdroj: http://doc.nette.org/cs/2.1/quickstart/getting-started

Tento obrázek 7 znázorňuje kostru aplikace, tzv. *sandbox*. Při použití nette se na této kostře staví požadovaná webová aplikace. Tato kostra je mnohdy označována jako *skeleton*.

3.3.4 PHP

PHP (Hypertext Preprocessor) je skriptovací programovací jazyk. Mnoho ze syntaxe jazyka vychází z C, Java a Perl a několika unikátních specifik pro PHP. Cílem tohoto jazyka je především umožnit webovým vývojářům rychle psát dynamicky generované stránky (PHP, 2015).

3.3.5 SQL a MySQL

SQL je standardizovaný strukturovaný dotazovací jazyk, který je využíván pro přístup k relačním databázím a pro práci s jejich daty (w3schools, 2015).

Jedním z relačních databázových systémů je MySQL. MySQL databáze je velmi populární, má velmi rychlý a snadno použitelný RDBMS[3], je používána v mnoha malých i velkých podnicích. Tato databáze byla vyvinuta a uvedena na trh a podporována švédskou společností MySQL AB (TutorialsPont, 2015).

3.3.6 HTML5 a CSS3

HTML (Hyper Text Markup Language) je značkovací jazyk. Využívá se k tvorbě struktury webových stránek. V minulosti se HTML používalo i pro formátování vzhledu stránek, V dnešní době vzhled a formátování zajišťují CSS neboli kaskádové styly. CSS tedy odděluje vzhled stránek od její struktury.

[3] Relational DataBase Management System (RDBMS) – neboli relační databáze umožňuje implementovat tabulky, sloupce a indexy, zaručuje integritu mezi řádkami různých tabulek, automaticky aktualizuje indexy a interpretuje SQL dotazy a kombinuje informace z různých tabulek.

HTML 5 je aktuální verze, rozšiřuje jazyk o nové elementy sémantické například značky *<header>, <footer>, <nav>* přidání nových formulářových atributů, grafických elementů a elementů pro multimédia, zjednodušuje deklaraci a má nové HTML 5 programátorské prostředí.

CSS 3 je poslední standart CSS, je kompletně kompatibilní s předchozími verzemi. CSS3 je rozšířeno o nové moduly. Tohle jsou některé z nejdůležitějších:

- Selektory
- Blokové modely
- Hodnoty obrázků a přepsání obsahu webu
- Textové efekty
- Animace
- 2D/3D transformace

(w3schools, 2015).

3.3.7 JavaScript (JS)

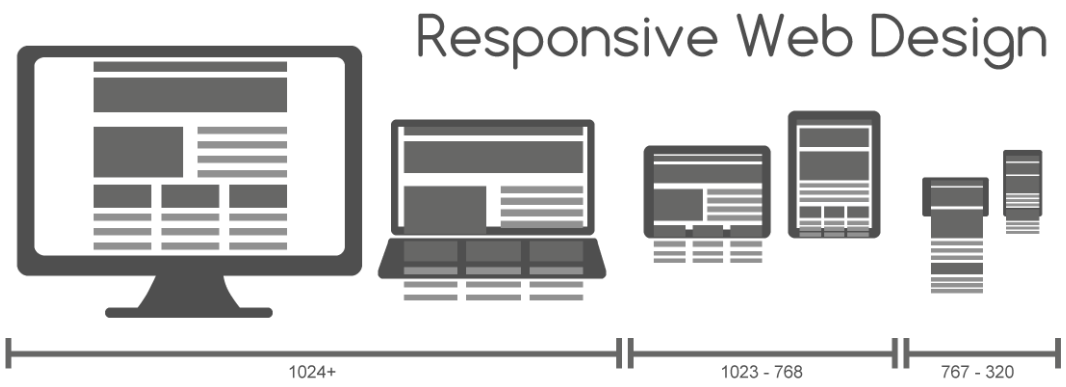
JavaScript je programovací jazyk pro HTML. Přidává statickým stránkám dynamiku. JS dokáže měnit strukturu HTML kódu, upravovat a plnit obsah. Dále také dokáže měnit atributy HTML elementů a přidávat či upravovat styly (CSS). (w3schools, 2015).

3.3.8 jQuery

jQuery je rychlá, malá a bohatá JavaScriptová knihovna. Umožňuje průchod a manipulaci s HTML dokumentem, zpracovává události, animace a AJAX v jednoduchém a použitelném rozhraní, jQuery je tedy všestranným nástrojem ulehčující práci s JS. (jQuery, 2015).

3.3.9 Responzivní design

Responzivní design je v dnešní době velmi skloňovaný pojem, díky velkému posunu mobilních zařízení jako jsou tablety a smartphony. Co tento pojem vlastně znamená, je to přizpůsobení webové stránky různým zařízením při různých rozlišeních tak, aby s stránka byla použitelná a mohlo se s ní pracovat na jakémkoliv zařízení. V praxi to znamená navrhnout dvě až tři rozdílná zobrazení a hned od začátku vývoje myslet na zobrazování dané stránky. Tuto práci běžně vykonávají kodéři ve spolupráci s grafiky webových stránek.



Obr. 8 Názorná ukázka responze
Zdroj: http://commons.wikimedia.org/, 2013. Responsive Web Design for Desktop, Notebook, Tablet and Mobile Phone.png.

3.3.10 NetBeans IDE

Je velmi rozsáhlý a pokročilý editor, určený pro vývoj jak desktopových, mobilních tak i webových aplikací ve všech možných programovacích jazycích. Mimo jiné má obrovskou podporu komunity z celého světa, je volně dostupný a má řadu výhod.

Pokud se budeme bavit o podpoře PHP, tak NetBeans disponuje okamžitou detekcí chyb implementovanou dokumentací, takže může ihned zobrazit popis PHP funkce, nebo usnadňovat pohyb v kódu, například pokud někde v kódu je volána metoda třídy, tak tato metoda může být ihned zobrazena a případně upravena.

3.3.11 Modelovací nástroje

Modelovacích nástrojů je celá řada, ale právě tyto nástroje budou využity pro návrh informačního systému.

- MySQL Workbench
- Visual Paradigm

MySQL Workbech umožní namodelovat ERD schéma a toto schéma pak lze jednoduše transformovat přímo do databáze, takže tabulky a reference nemusí být vytvářeny ručně. Tento nástroj také umožnuje z již fungující databáze vygenerovat ERD schéma, toto schéma upravit a posléze jej upravení importovat do databáze. Tato synchronizace je v mnoha případech velmi užitečná a ulehčuje práci.

Visual Paradigm je silný modelovací nástroj, který umožňuje vytvořit snad jakýkoliv model či schéma, na které se dá vzpomenout. Především slouží pro modelování UML[4] a CASE[5].

[4] Unified Modeling Language je nejpoužívanější specifikace a způsob jakým popsat nejen strukturu aplikace, chování a architekturu ale také podnikové procesy a strukturu dat.

4 Návrh řešení a implementace

4.1 Seznámení se strukturou školy

Seznamování se strukturou a organizací základní školy v Ketkovicích probíhalo na základě schůzek s paní ředitelkou.

Struktura této školy se poměrně liší od běžných základních škol, jelikož se jedná o malou školu s malým počtem žáků do pátého ročníku. Tato škola má dvě třídy žáků to znamená, že v jedné třídě jsou žáci druhého, třetího a pátého ročníku dohromady a ve třídě druhé jsou dohromady žáci ročníku prvního a čtvrtého. Takto škola funguje již několik let, a i když by se mohlo zdát, že takto strukturovaná škola nemůže fungovat, tak opak je pravdou.

Tímto se dostávám k požadavku, na nové webové stránky, jejich aktuální web nesplňuje všechna očekávání, je postaven na starém redakčním systému „phprs[6]“, který se nevyvíjí, má poměrně složitou administraci a vzhled stránek neodpovídá představě, kterou paní ředitelka má.

Byly tedy zadány následující požadavky, vytvořit nenáročný informační systém jako webovou aplikaci, která nahradí současné řešení a bude mít a umět následující:

- Webová aplikace bude rozdělena do kategorií pro části základní škola, mateřská škola, školní družina a jídelna.
- Obsah těchto kategorií budou plnit zaměstnanci školy na základě přidávání příspěvků. Do jaké kategorie zaměstnanci mohou příspěvek umístit, zajistí jejich role.
- Na příspěvky budou moci reagovat rodiče žáků pomocí komentářů, přes účet vytvořený každému žákovi.
- Kategorie pro základní školu bude obsahovat rozvrhy, které půjdou snadno měnit a domácí úkoly, zadávány učiteli pro žáky.
- Jídelna bude disponovat jídelníčkem. Tento jídelníček budou doplňovat a upravovat právě kuchařky. Pomoci připraveného formuláře.
- Ředitel školy a určený správce budou mít na starosti přidávání nových uživatelů, jak žáků, tak i zaměstnanců školy, jelikož žáků a zaměstnanců není mnoho, budou se vkládat jednotlivě.

[5] Computer-aided software engineering, počítačem podporované SW inženýrství

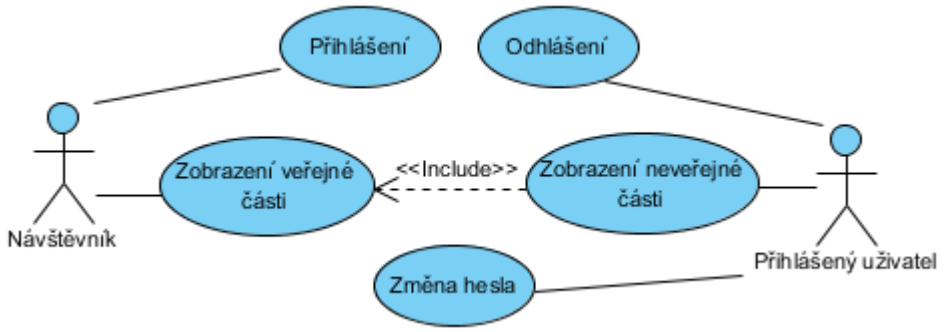
[6] Redakční phpRS dostupný systém na http://www.supersvet.cz/phprs/

- Ovládání webové aplikace by mělo být snadné a intuitivní, jednoduší, než u současného řešení, kde vybraný redakční systém byl poměrně složitý a bylo poměrně náročné s ním pracovat.
- Bude vybrána šablona, která zajistí konečný vzhled stránek. Šablona se později bude moci případně podle potřeby upravit.

4.2 Use Case diagram

Po provedení analýzy a seznámení se, se základní strukturou základní školy byl navrhnut use-case diagram, který znázorňuje, jakým způsobem bude probíhat komunikace uživatelů s daným systémem.

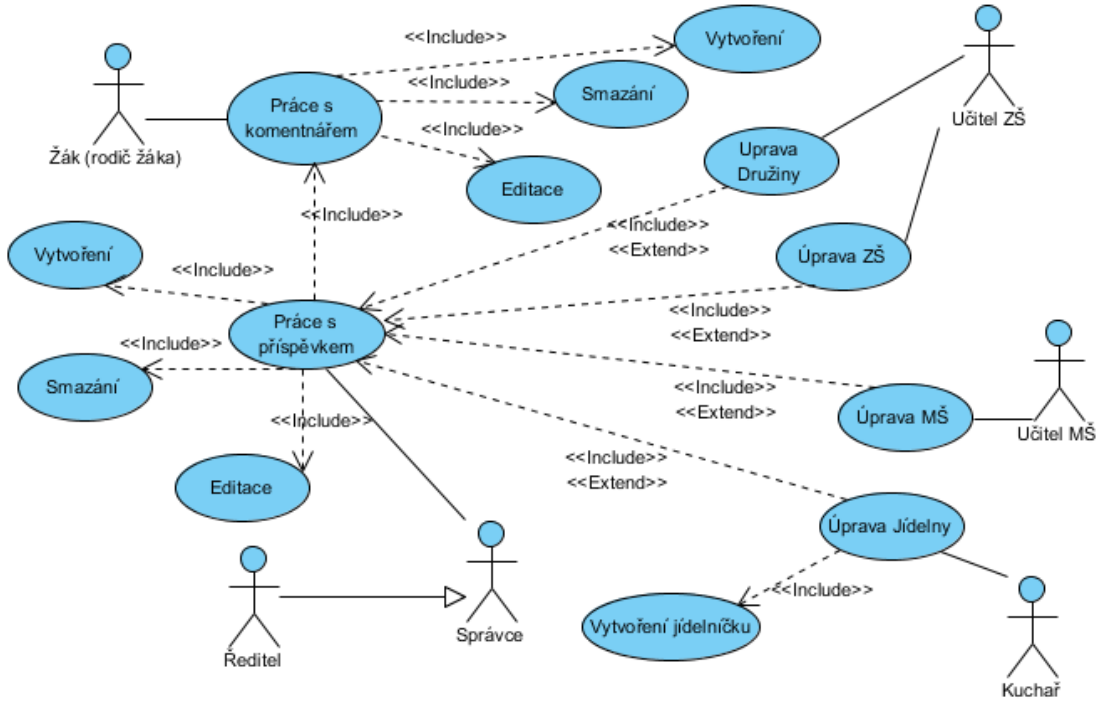
Tento diagram je rozdělený do několika částí, jelikož systém bude komplexnější a celkový diagram by nebyl přehledný.



Obr. 9 Use-Case první pohled přihlášený/nepřihlášený uživatel

První částí diagramu (obrázek 9) je pohled na přihlášeného a nepřihlášeného uživatele. Nepřihlášený uživatel může prohlížet veřejné části webové aplikace nebo se přihlásit pokud má přihlašovací údaje.

Přihlášený uživatel má kromě přístupu do veřejné, také přístup do části neveřejné. Tento uživatel si také bude moci změnit své heslo nebo se odhlásit.



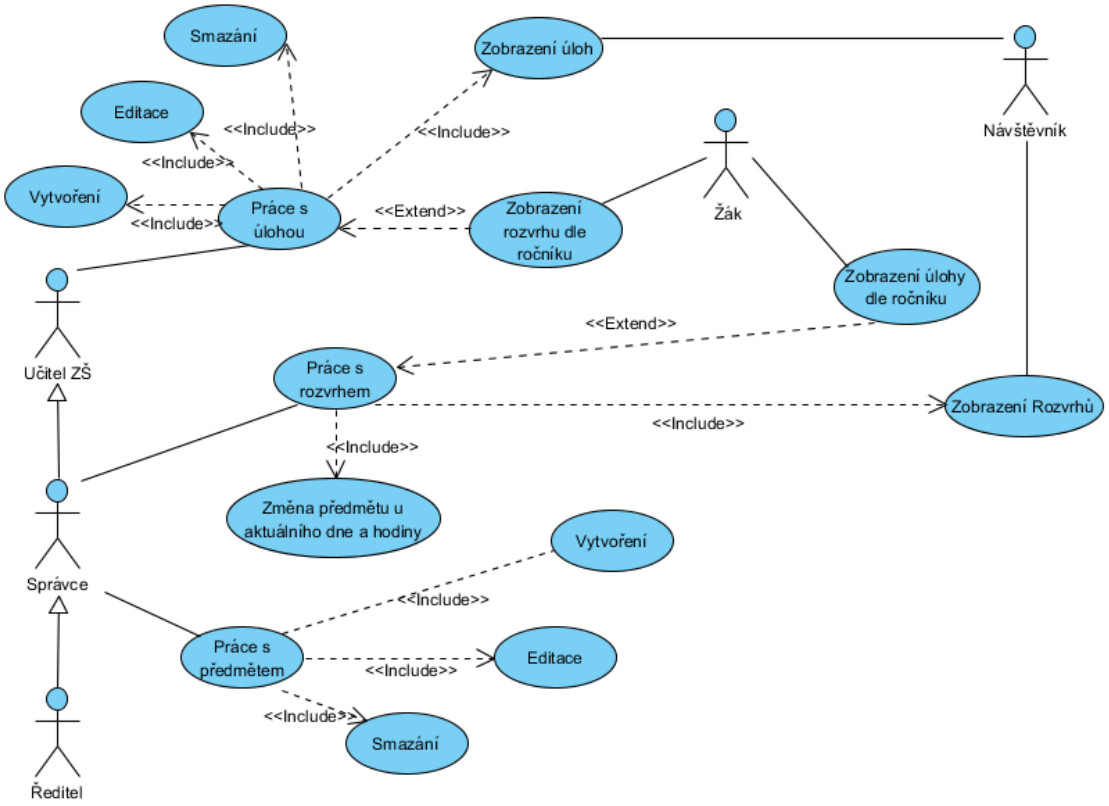
Obr. 10 Use-Case práce uživatelů s příspěvky

Tento diagram neboli obrázek 10 popisuje, jak přihlášení uživatelé pracují s příspěvky a komentáři. Slovně řečeno, se systémem pracují zaměstnanci školy a žáci respektive rodiče žáků. Každý přihlášený uživatel může přidávat k příspěvkům komentáře, vlastní komentáře pak může upravovat nebo mazat. Rozdělení rolí zaměstnanců školy zaměstnanci mimo komentářů mohou přidávat a upravovat příspěvky v kategoriích:

- Učitel MŠ – kategorie Mateřská škola
- Učitel ZŠ – kategorií Družina a Základní škola
- Kuchař – kategorie Jídelna
- Ředitel a Správce – uživatelé s těmito rolemi mohou přidávat a upravovat veškeré kategorie

V diagramu je navíc zobrazeno vytváření jídelníčku, tento jídelníček mohou vytvářet všichni uživatelé s rolí Kuchař, Správce a Ředitel.

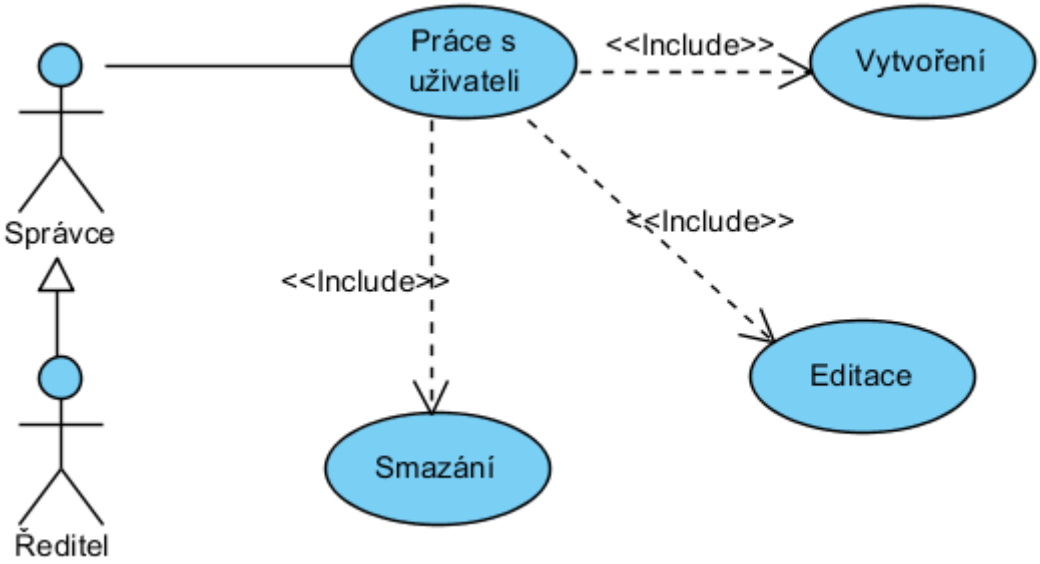
Mimo jiné je zde jedna zajímavost a to taková, že vztahy mezi úpravou kategorií a práci s příspěvky jsou popsány jako *include* a *extend* zároveň, znamená to to, že tento vztah rozšiřuje a zároveň zahrnuje práci s příspěvky.



Obr. 11 Use-Case práce uživatelů s úlohami předměty a rozvrhem

Další z diagramů, obrázek 11 popisuje vytváření a úpravu předmětů, domácích úloh a tvorbu samotného rozvrhu. Uživatelé, kteří mají roli Učitel ZŠ, mohou vytvářet, upravovat a mazat domácí úkoly pro žáky. Uživatelé s rolí Správce a Ředitel mohou dále vytvářet a upravovat aktuální rozvrh podle ročníků a upravovat předměty, které se na škole vyučují.

Když uživatel není přihlášen, tak si může zobrazit jakýkoliv rozvrh či úlohu, když je uživatel přihlášen jako Žák tak vidí svoje aktuální úlohy a svůj rozvrh podle ročníku, ve kterém se nachází.

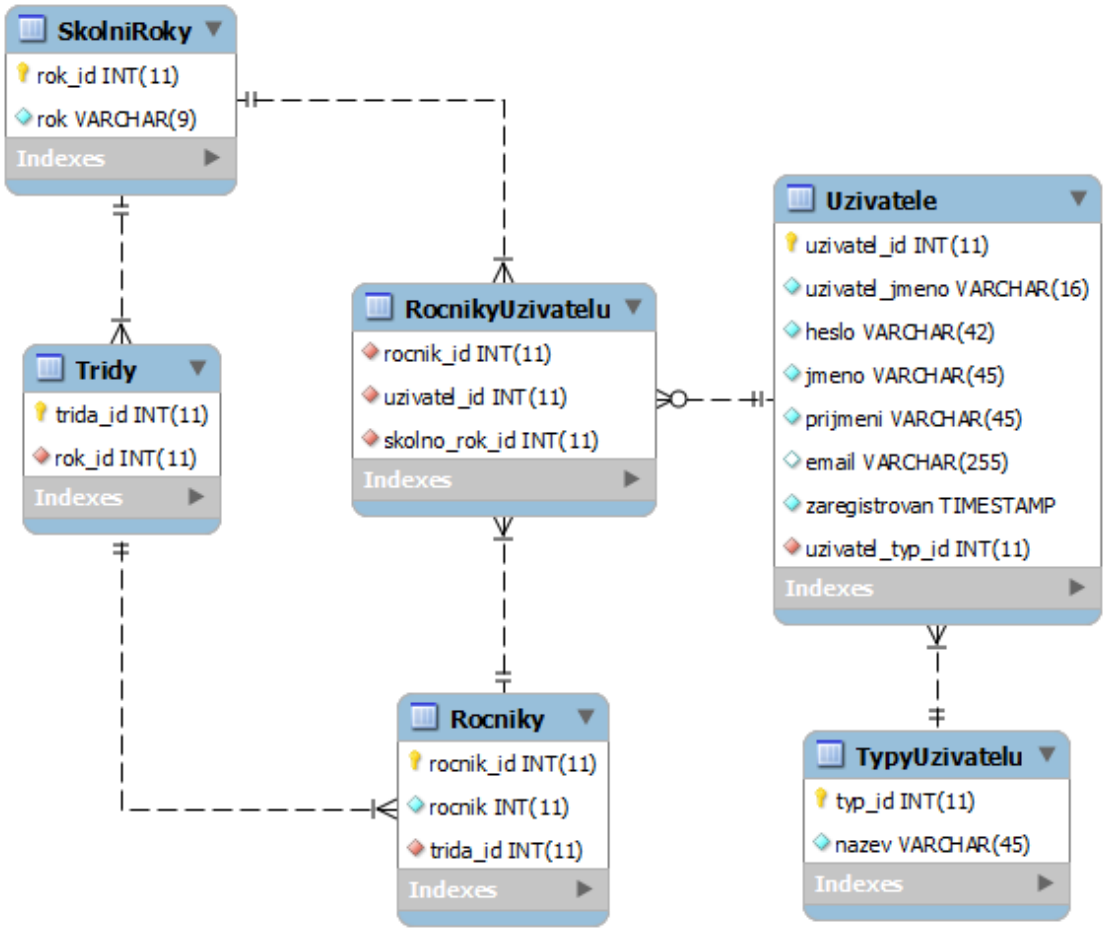


Obr. 12 Use-Case práce s uživateli

Poslední z vytvořených Use Case diagramů (obrázek 12), je diagramem pro vytváření nových uživatelů, nové uživatele můžou vytvářet pouze uživatelé s rolemi Ředitel nebo Správce.

4.3 Návrh ERD

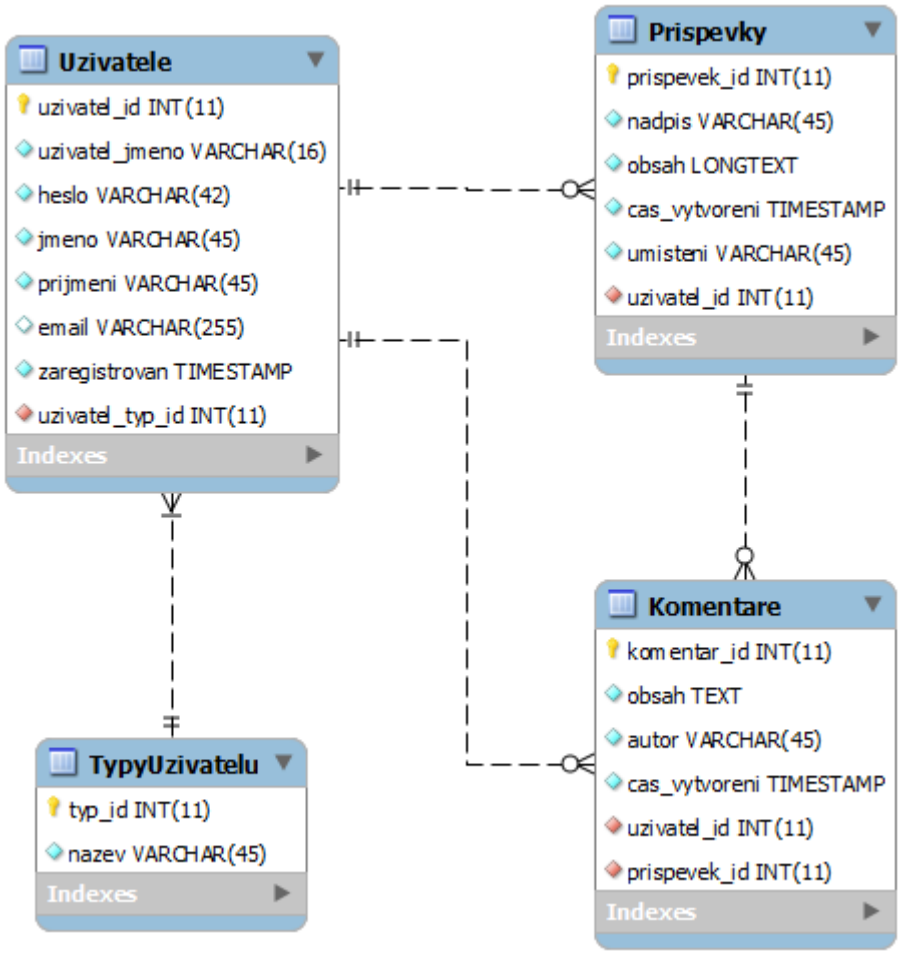
Navržený informační systém bude využívat následující datový model. Tento model popisuje nový IS základní školy v Ketkovicích a podobně jak use-case je rozdělený do částí, které jsou podrobněji popsány.



Obr. 13 ERD – školní rok a rozdělení uživatelů

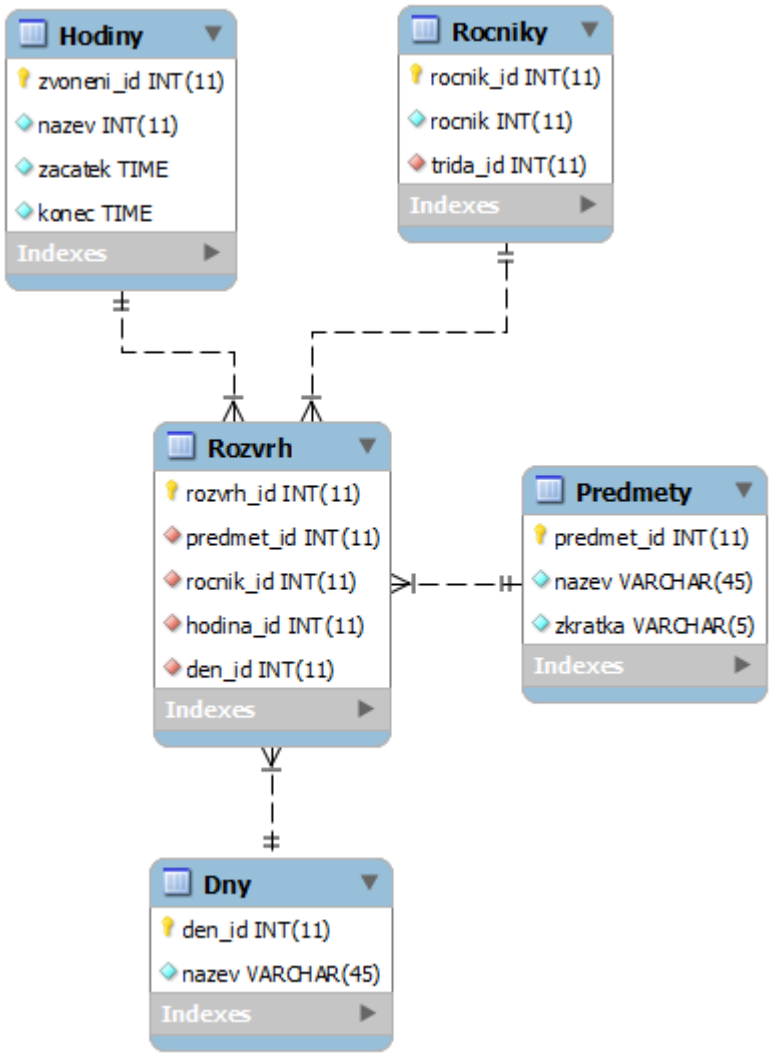
Tento obrázek 13 znázorňuje školní rok a žáky v jednotlivých ročnících. Jelikož se jedná o malou školu, tak datový model má takovouto strukturu, tedy ve dvou třídách jsou rozmístěni žáci prvních až pátých ročníků.

Tabulka *RocnikyUzivatelu* tedy představuje žáky a jejich ročník v daném školním roce.



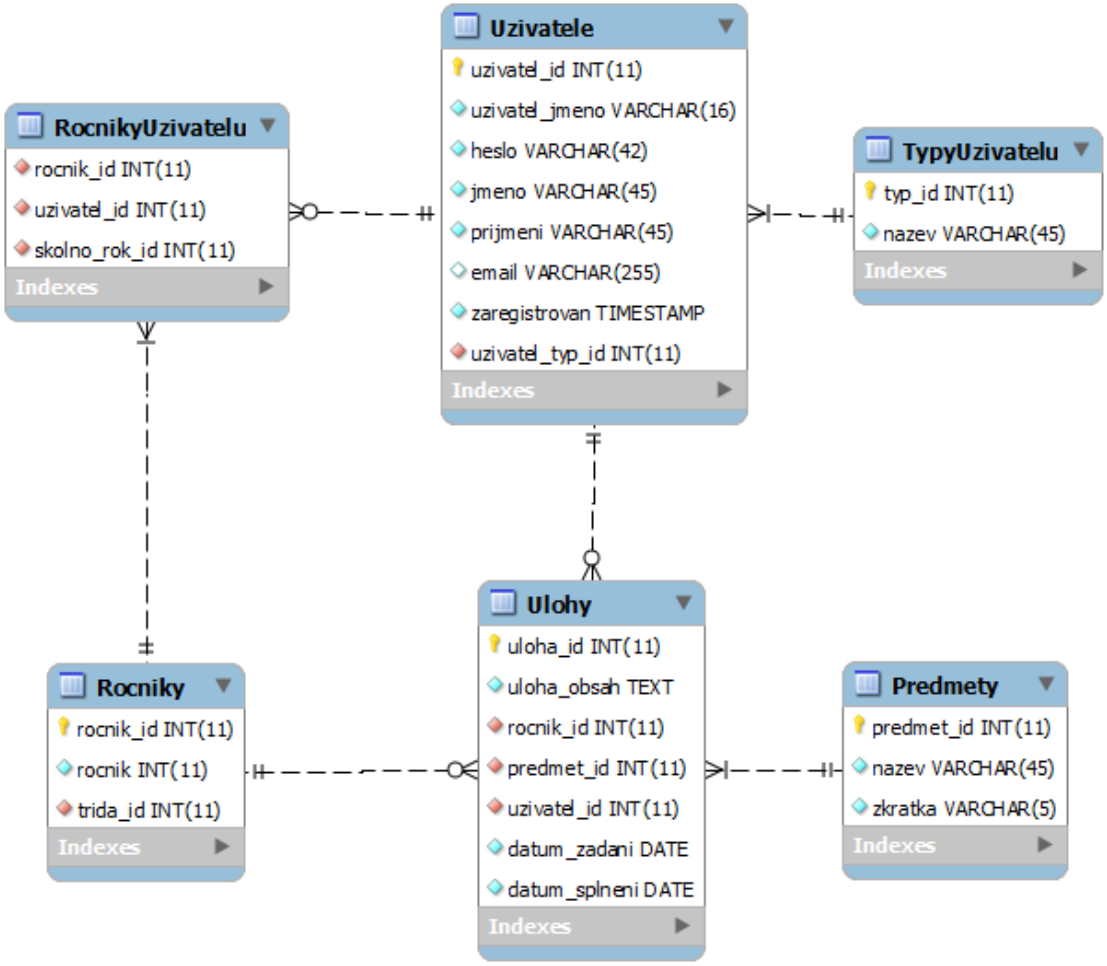
Obr. 14 ERD – příspěvky a komentáře uživatelů

Datový model pro uživatele, jejich příspěvky a komentáře je zobrazen na obrázku 14. Zaměstnanci školy mohou podle své role přidávat příspěvky do jednotlivých kategorií, jako je základní škola, školka, jídelna, aj., komentáře se přidávají k jednotlivým příspěvkům, mohou je přidávat všichni přihlášení uživatelé IS a mohou sloužit ke komunikaci s rodiči žáků nebo k přidání doplňující informace od jiného zaměstnance.



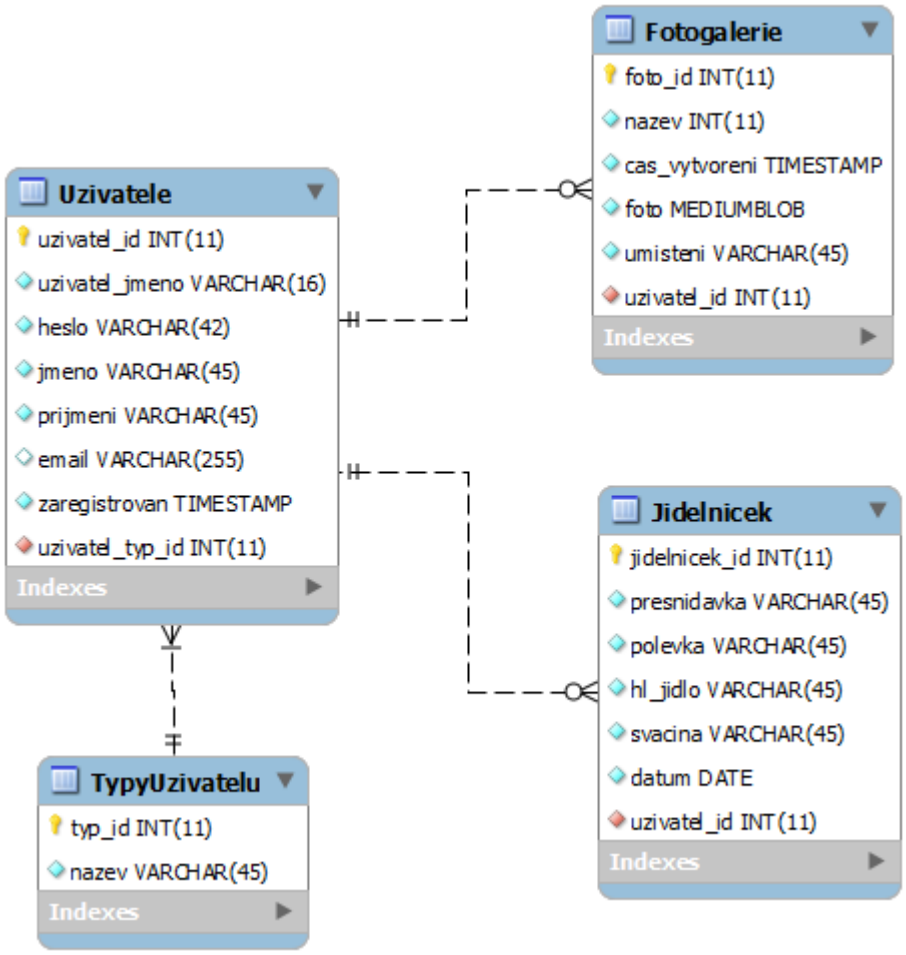
Obr. 15 ERD – Rozvrh jednotlivých ročníků

Zde na obrázku 15 je znázorněn školní rozvrh, každý ročník žáků má rozvrh vlastní. Jeden řádek tabulky rozvrhu tedy představuje, jaký předmět daného ročníku se vyučuje konkrétní hodinu v konkrétní den.



Obr. 16 ERD – Domácí úlohy

Tato část datového modelu, obrázek 16 zobrazuje ukládání domácích úloh. Úloha je tedy zadávána pro konkrétní ročník a předmět. Úloha má také uchováno to, který uživatel úlohy zadal.



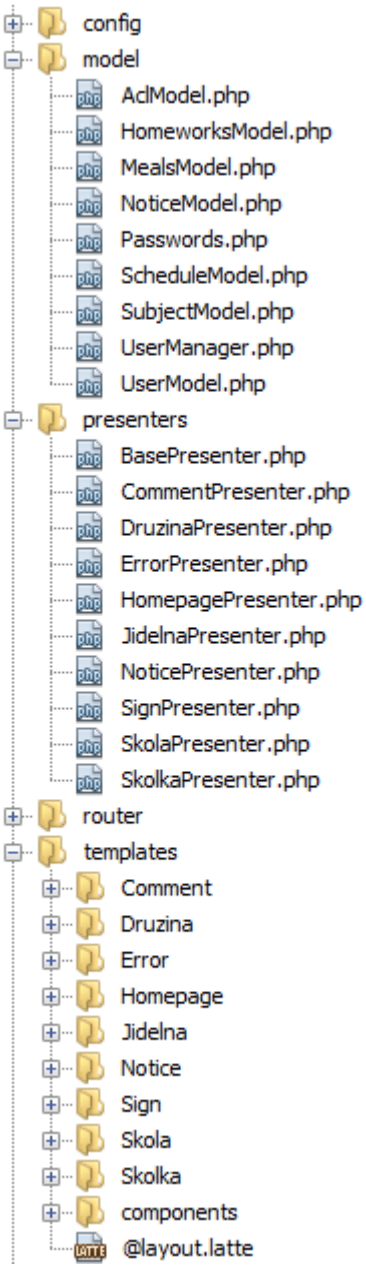
Obr. 17 ERD – fotogalerie a jídelníček

Jako poslední část datového modelu (obrázek 17), je model pro fotogalerie a jídelníček. Vazby na tyto kategorie znázorňují uživatele, kteří je vytvořili.

4.4 Tvorba IS s využitím Nette

4.4.1 Návrh struktury aplikace

S návrhem struktury aplikace pomáhá nette, bude tedy využit sandbox. Popis sandboxu byl uveden dříve v textu v podkapitole 3.3.3 Nette Framewok. Následující obrázek 18 zobrazuje navrženou strukturu a níže pod obrázkem jsou struktura a vytvořené třídy blíže popsány.



Obr. 18 Struktura nového informačního systému

Adresář **config** a jeho soubory zajišťují mimo jiné připojení k databázi, toto připojení pomocí „injection“ poskytuje jednotlivým modelům.

Modely představují komunikaci s databází výjimkou je třída *AclModel.php*, kde jsou nastaveny role uživatelů a těmto rolím jsou nastavena práva a třída *Password.php*, která má za úkol ověřovat „zahashovaná“ hesla, nebo hesla naopak „hashovat“. V dalších modelech jsou funkce s následujícími předponami:

- *get* – metody takto označený vyberou z databáze data, parametr blíže specifikuje, která data se mají z databáze vybrat.

- *create* – tyto metody mají za úkol vytvořit v databázi nový záznam, parametr nám tento záznam blíže specifikuje.
- *edit* – metody touto předponou, editují již existující záznamy.
- *delete* – jako poslední z předpon je předpona delete označují nám metody, které mají za úkol smazat položku z databáze.

Další adresář je **presenter**, ten obsahuje třídu *BasePresenter.php,* která je rodičem všech ostatních tříd v tomto adresáři. Obsahuje metodu *startup(),* ta se zavolá a po vytvoření presenteru se ověří uživatelská oprávnění. Presentery mají za úkol provádět různé akce, na základě požadavků od uživatele.

Adresář **templates** obsahuje jednotlivé stránky (latte soubory), struktura tohoto adresáře také určuje zobrazovanou url, tato url se dá nastavit či přepsat v adresáři router.

4.4.2 Funkcionalita aplikace

Hlavní funkcionalitou systému jsou příspěvky a komentáře. Zaměstnanci školy mohou dle své role vkládat příspěvky do povolené kategorie, například kuchaři mohou přidávat příspěvky pouze do kategorie Jídelna. Příspěvky a komentáře budou sloužit především ke komunikaci rodičů žáků se zaměstnanci školy a k přidávání novinek a aktualit na stránky školy.

Další funkcionalitou jsou domácí úlohy. Úloha se vytváří následovně, vyplní se zadání úlohy, vybere se ročník, pro který je domácí úloha určena, dále se vybere předmět. Úlohy pak mohou zobrazit i nepřihlášení uživatelé, pokud uživatel bude přihlášený s rolí žák, uvidí úlohy právě pro svůj ročník. Se zobrazením rozvrhů je to stejné, jak se zobrazením úloh. Rozvrhy mohou vytvářet uživatelé s rolí správce a vyšší.

Další funkcionality víceméně vyplývají z navrhnutého use-case diagramu kapitola 4.2 a ERD modelu kapitola 4.3.

Obr. 19 Zobrazení příspěvku s komentáři

Na obrázku 19 je vidět příspěvek a vlastní komentáře z pohledu rodiče. Rodič může svůj komentář upravit, smazat nebo přidat další. Všichni uživatelé také mají k dispozici přehled všech vlastních komentářů a reakcí na vytvořené komentáře.

4.4.3 Zabezpečení a přístupy do aplikace

Jak vypadají jednotlivé přístupy do aplikace z pohledu uživatele, bylo již zobrazeno v navrhnutém use-case diagramu kapitole 4.2 a jejich podkapitolách. Nyní budou popsány jednotlivé role uživatelů a jejich možnosti z pohledu aplikace.

Nastavení rolí uživatelů a jejich práv zajišťuje třída v modelu *AclModel.php*. Je využit dostupný autorizátor, který je v Nette již implementovaný a to ve třídě *Nette\Security\Permission*. Nastavení jednotlivých rolí je znázorněno na obrázku 20.

```
$this->acl = new Nette\Security\Permission();

$this->acl->addRole("guest");
$this->acl->addRole("Žák","guest");
$this->acl->addRole("Kuchař","Žák");
$this->acl->addRole("Učitel MŠ","Žák");
$this->acl->addRole("Učitel ZŠ","Žák");
$this->acl->addRole("Správce","Učitel ZŠ");
$this->acl->addRole("Ředitel","Správce");
$this->acl->addRole("Admin","Ředitel");
```

Obr. 20 Role uživatelů

Funkce *addRole* má může mít 2 parametry, první parametr je nastavení nové role uživatele a druhý parametr (nepovinný) je role, ze které se dědí povelené přístupy či akce.

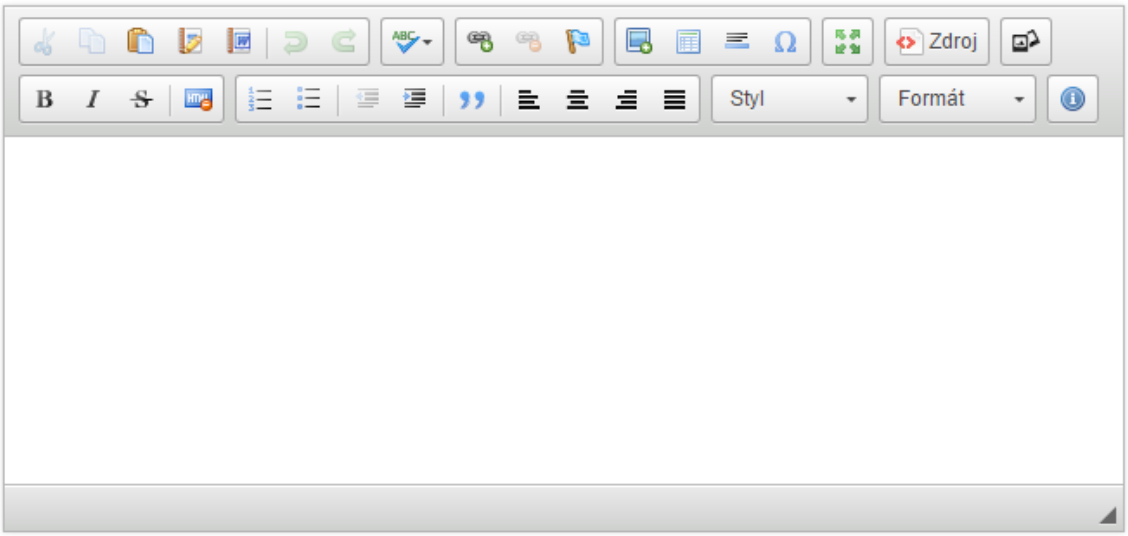
```
$this->acl->allow("guest",array(
        "druzina",
        "homepage",
        "sign",
        "comment",
        "skola",
        "skolka",
        "jidelna"),array("default","oSkole","in","rozvrh","ulohy","jidelnicek"));
$this->acl->allow('Žák',array('sign','comment'),array('out','show'));
$this->acl->allow('Kuchař','jidelna');

$this->acl->allow("Učitel ZŠ","skola",array('ulohy','deleteHom'));

$this->acl->allow("Žák","comment");
$this->acl->allow("Žák","homepage","komentare");
$this->acl->allow("Žák","homepage","heslo");

$this->acl->allow("Kuchař", 'notice');
$this->acl->allow("Učitel ZŠ", 'notice');
$this->acl->allow("Učitel MŠ", 'notice');

$this->acl->allow("Správce");
```

Obr. 21 Nastavení přístupu rolí

Tento obrázek 21 znázorňuje základní nastavení přístupů jednotlivým uživatelů. Funkce *allow* může mít 3 parametry, první je vybrání role, druhý parametr je sekce aplikace a třetí parametr jsou jednotlivé akce v sekci. Když je metoda *allow* zavolána bez parametru, nastaví se všem rolím neomezený přístup, pokud je vyplněn pouze první parametr nastaví se neomezený přístup k dané roli.

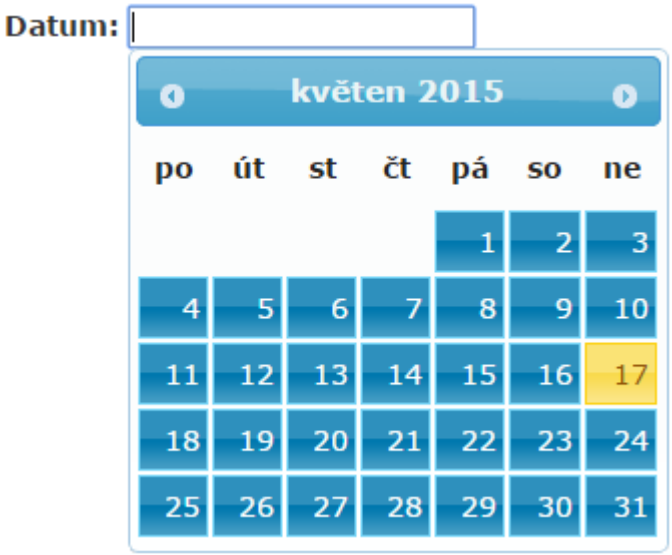
4.4.4 Využití freewerových nástrojů a dostupné šablony

Ckeditor – pro psaní příspěvků byl využit volně dostupný web textový editor, konečný vzhled editoru byl přizpůsoben obrázek 22 (CKSource, 2015).



Obr. 22 Přispůsobený ckeditor

DatePicker – tento nástroj byl využit pro výběr data (Tvrdík, 2012).



Obr. 23 DatePicker s vybraným vzhedem

Šablona – byla vybrána následující šablona obrázek 24, původně určená pro wordpress, tato šablona byla upravena a přizpůsobena pro novou aplikaci, obrázek 25 (Free Wordpress Themes, 2015). V patičce stránky je uveden odkaz na tuto šablonu.

Obr. 24 Back To School Again Wordpress Theme

Obr. 25 Upravená šablona – aktuální vzhled stránek

4.5 Testování aplikace

Testování aplikace probíhá ve 3 fázích.

- První fází - je testování a hledání chyb na základě vlastních podnětů. Tedy po vytvoření funkčního bloku probíhá jeho testování, jedná se například o plnění databáze zkušebními daty, zkoušení situací, které mohou nastat nebo, co vše je uživatel schopen provést. Toto testování není úplně dokonalé, ale pro zajištění základní funkčnosti je to dostačující.
- Druhá fáze testování – probíhá informativní schůzka s ředitelkou školy, tedy zadavatelem. Je představován běžící systém na testovacím serveru. Probíhá školení a seznamování se systémem. Během představování a prvního testování reálným uživatelem je poukázáno na následující:
    - Navigace – logika přepínání stránek je mírně zmatečná, na úvodní do levého panelu nutno přidat odkazy na domácí úlohy a na rozvrh.
    - Chybějící přehled vlastních komentářů.
    - Pro zaměstnance školy, mít přehled příspěvků – dříve zaměstnanec, který neměl roli správce nebo vyšší, měl možnost příspěvek pouze přidat.
    - Výpis domácích úloh obrátit řazení, a u úloh byl zbytečný nadpis, domácí úlohy nemají mít datum splnění, pouze datum zadání.
- Třetí fáze – Informační systém již běží a je postupně testován a naplňován daty, aplikace je schválena jak rodiči žáků, tak i personálem školy, probíhá komunikace a připravuje na nasazení systému. Opravují se drobné nedostatky.

5 Přínosy informačního systému

5.1 Využití v praxi

Tento informační systém byl určen převážně pro základní školu v Ketkovicích, nyní je systém schválen, probíhají závěrečné úpravy a připravuje se jeho nasazení. Reálné využití systému je v udržování aktuálnosti a povědomí o škole pomocí příspěvků, které se dají snadno upravovat. Po dokončení a přidání drobností budou tyto stránky plnohodnotným řešením pro prezentaci školy a uchovávání dat. Velkou výhodou tohoto systému je jeho snadná použitelnost.

Přestože byl informační systém převážně určen pro Ketkovickou základní školu, tak se systém bude dát využit i pro mnohé jiné základní školy. Systém se dá poměrně snadno modifikovat a lze jej rozšiřovat.

5.2 Porovnání s předchozím řešením

Minulé řešení informačního systému bylo nevyhovující díky složité administraci, nekorektním odkazům a ne moc příjemnému vzhledu. Stránky jsou postaveny na zastaralém a již nepodporujícím redakčním systému. Škola za následující řešení platí poměrně vysokou částku, ale kvalita stránek tomu neodpovídá.

Nové řešení stránek a informačního systému, má oproti snadnou administraci, aplikace byla navržená na míru přímo podle daných požadavků škola si vybrala vlastní šablonu (design stránek), která byla předělána podle přání. Stránky jsou postaveny na moderním, neustále se vyvíjejícím Nette Frameworku a jsou poměrně dobře rozšiřitelné.

Následující obrázek 26 zobrazuje porovnání původního (napravo) řešení s řešením novým (nalevo).

Obr. 26 Nové vs. původní řešení

6 Závěr

Motivací této práce bylo vytvoření informačního systému pro základní školu v Ketkovicích, tento systém postupně dovést do stavu, kdy bude plně vyhovujícím a bude splňovat, nejlépe všechny požadavky, které škola má. Po splnění všech kritérií a otestování systému, má následně nahradit původní nevyhovující řešení.

Byl tedy vytvořen informační systém na webovém rozhraní, tedy moderní webová aplikace, která umožňuje zaměstnancům přidávat aktuality do různých sekcí školy. Na tyto aktuality lze přispívat komentáři, které mohou psát rodiče žáků, nebo jiní zaměstnanci školy, tedy všichni přihlášení uživatelé. Aplikace dále umožňuje vytvoření rozvrhu, vytvoření domácí úlohy a jejich specifické zobrazení pro žáky, tzn., žák uvidí rozvrh a domácí úlohy podle svého aktuálního ročníku. Systém je také dobře rozšiřitelný a editovatelný, i když bude několik funkcionalit systému nejspíše předěláno a vyřešeno lépe. Jde o to, že se aplikace vyvíjela poměrně dlouhou dobu a za tu dobu narůstaly znalosti využívané při její tvorbě. Jak již bylo zmíněno jednou z velkých předností této aplikace, je snadné a intuitivní ovládání, jak v administrační části, tak i v části veřejné. Další z výhod je využití moderního Frameworku Nette, který zajišťuje bezpečnost a poměrně snadnou rozšiřitelnost.

Aplikace v této době ještě není nasazena na požadované doméně, zatím je spuštěna pouze pronajatém testovacím serveru. Nicméně stránky byly odsouhlaseny, jak vedením školy, tak i rodiči žáků a nyní již chybí pouze provést doplnění drobností, přidání všech uživatelů a umístění potřebných dokumentů. Také je důležité na základě podané žádosti vyčkat na udělení licence od centra pro transfer technologií. Lze tedy říci, že hlavní cíl práce byl splněn.

K aplikaci byli vytvořeny tři uživatelé pro účely odzkoušení, tyto přístupové údaje budou funkční minimálně do konce července 2015.

- URL testovací aplikace - http://lukasbakalarka.javeismy.name/
- Testovací uživatelé (login - heslo):
    - mendeluZak - mendelu
    - mendeluUcitel - mendelu
    - mendeluReditel - mendelu

7 Literatura

ARISTOTELES. *Metafyzika, V, 11 a VIII, 6* [online]. Brno [cit. 2015-05-14]. Dostupné z: http://classics.mit.edu/Aristotle/metaphysics.html.

BERNARD, Borek. *Úvod do architektury MVC, Prezentační vzory z rodiny MVC*. [online]. 2009 [cit. 2015-05-17]. Dostupné z: http://www.zdrojak.cz/clanky/uvod-do-architektury-mvc/ a http://www.zdrojak.cz/clanky/prezentacni-vzory-zrodiny-mvc/.

BRUCKNER, Tomáš. *Tvorba informačních systémů: principy, metodiky, architektury.* 1. vyd. Praha: Grada, 2012, 357 s. Management v informační společnosti. ISBN 978-80-247-4153-6.

CKSOURCE. *CKEditor*. [online]. 2015 [cit. 2015-05-17]. Dostupné z: http://ckeditor.com/.

ČÁPKA, David. *1. díl - Úvod do Nette frameworku pro PHP*. *Polymorfismus, finální prvky a autoloader v PHP*. [online]. 2015 [cit. 2015-05-17]. Dostupné z: http://www.itnetwork.cz/uvod-do-php-frameworku-nette, http://www.itnetwork.cz/tutorial-php-polymorfismus-final-autoloader.

DARIE, Cristian. *AJAX a PHP: tvoříme interaktivní webové aplikace profesionálně*. Vyd. 1. Brno: Zoner Press, 2006, 320 s. Encyklopedie webdesignera. ISBN 80-86815-47-1.

FREE WORDPRESS THEMES. *Back To School Again Wordpress Theme*. [online]. 2015 [cit. 2015-05-17]. Dostupné z: http://www.free-wordpress-theme.net/theme/back-to-school-again.html.

HAMMER, Michael. 2000. *Reengineering - radikální proměna firmy: manifest revoluce v podnikání*. 3. vyd. Praha: Management Press, 212 s. ISBN 80-726-1028-7.

IPARŤÁK S.R.O. *O wordpressu*. [online]. 2015 [cit. 2015-05-16]. Dostupné z: http://www.iwp.cz/o-wordpressu/.

JQUERY. *What is jQuery?*. [online]. 2015 [cit. 2015-05-17]. Dostupné z: http://jquery.com/.

MIKULÁŠEK, Adam. *Modely strukturované analýzy pro nevidomé*. Bakalářská práce. Masarykova univerzita. [online]. Brno, 2008 [cit. 2015-05-08]. Dostupné z: http://is.muni.cz/th/134645/fi_b/Bakalarska_prace.txt.

MROZEK, Jakub. *OOP v PHP: Abstraktní třída, OOP v PHP: Dědičnost*. [online]. 2006 [cit. 2015-05-15]. Dostupné z: https://www.interval.cz/stitek/oop-php/.

NETTE. *MVC aplikace & presentery*. [online]. 2015 [cit. 2015-05-17]. Dostupné z: http://doc.nette.org/cs/2.3/presenters.

NEUMAJER, Ondřej. *Školní informační systémy*. [online]. 2010 [cit. 2015-05-16]. Dostupné z: http://clanky.rvp.cz/clanek/c/u/8019/SKOLNI-INFORMACNI-SYSTEMY.html/.

PHP. *General Information*. [online]. 2015 [cit. 2015-05-17]. Dostupné z: http://php.net/manual/en/faq.general.php.

POCHÁZKA, David, a kolektiv. *Objektově orientované programování v jazyku C++*. [online]. 2010 [cit. 2015-05-15]. Dostupné z: http://ui.pefka.mendelu.cz/files/ZOO_opora_2.pdf.

RÁBOVÁ, Ivana. *Podnikové informační systémy a technologie jejich vývoje*. V Tribun EU vyd. 1. Brno: Tribun EU, 2008, 139 s. ISBN 978-80-7399-599-7.

RYBIČKA, Jiří. *Informační systémy*. [online]. 2009 [cit. 2015-05-07]. Dostupné z: https://akela.mendelu.cz/~rybicka/prez/infsyst.pdf.

ŠKOLA ONLINE a.s. *Školní informační systém Škola OnLine*. [online]. 2015 [cit. 2015-05-15]. http://www.fi.muni.cz/~smid/mis-zivcyk.htm.

ŠMÍD, Dalibor. *Životní cyklus informačního systému*. [online]. 2002 [cit. 2015-05-15]. http://www.fi.muni.cz/~smid/mis-zivcyk.htm.

TUTORIALSPOINT. *MySQL Introduction*. [online]. 2015 [cit. 2015-05-17]. http://www.tutorialspoint.com/.

TVRDÍK, Jan. *DatePicker*. [online]. 2012 [cit. 2015-05-17]. Dostupné z: http://nette.merxes.cz/date-picker/.

VRÁNA, Jakub. 2010. *1001 tipů a triků pro PHP*. Vyd. 1. Brno: Computer Press, 456 s. ISBN 978-80-251-2940-1.

W3SCHOOLS. *SQL, HTML5, CSS5*. [online]. 2015 [cit. 2015-05-17]. http://www.w3schools.com/.