FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Modul pro tvorbu úvazků a výběr volitelných předmětů pro informační systém střední školy
Student:	Jan Vožeh
Vedoucí:	Mgr. Petr Matyáš
Studijní program:	Informatika
Studijní obor:	Webové a softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2018/19

Pokyny pro vypracování

Vytvořte modul pro tvorbu úvazků a výběr volitelných předmětů připravovaného informačního systému školy.
1) Proveďte analýzu potřeb pro evidenci úvazků a zapisování volitelných předmětů studenty školy.
2) Aplikace musí:
- využívat sdílenou databázi s ostatními moduly informačního systému,
- umožnit studentům výběr volitelných předmětů (primární a sekundární) podle oboru studenta, navazujících předmětů,
- umožnit přiřadit třídě a předmětu počet vyučovaných hodin a učitele, vypsat kapacity předmětů dle úvazku učitele,
- sumarizovat výpisy pro učitele, předměty a třídy, kontrola kapacit a úvazků učitelů,
- exportovat výpisy.
3) Poskytnutí API není vyžadováno, všechny části systému budou využívat sdílenou databázi.
4) Veškeré softwarové výstupy důkladně otestujte.
5) Funkčnost modulu ověřte integrací do výsledné webové aplikace.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 3. ledna 2018

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalářská práce

Modul pro tvorbu úvazků a výběr volitelných předmětů pro informační systém střední školy

Jan Vožeh

Katedra softwarového inženýrství
Vedoucí práce: Mgr. Petr Matyáš

11. května 2018

Poděkování

Rád bych poděkoval vedoucímu této bakalářské práce Mgr. Petrovi Matyášovi za cenné rady, věcné připomínky a vstřícnost při konzultacích a vypracování práce.

Dále bych rád poděkoval Tomášovi Trbolovi za rady při vývoji a následné integraci mé práce do jádra systému, které vytvořil v rámci své bakalářské práce *Webová administrace informačního systému pro střední školu.* [1]

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo"), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 11. května 2018 .....

České vysoké učení technické v Praze
Fakulta informačních technologií

Odkaz na tuto práci

Vožeh, Jan. *Modul pro tvorbu úvazků a výběr volitelných předmětů pro informační systém střední školy.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2018.

Abstrakt

Cílem práce je zanalyzovat potřeby Střední školy a vyšší odborné školy reklamní a umělecké tvorby Michael a do připravovaného infromačního systému naprogramovat modul pro tvorbu úvazků učitelů a výběr volitelných předmětů studenty školy.

První část práce se zabývá analýzou požadavků a již existujících řešení. Druhá část práce popisuje již samotnou implementací modulu na základě požadavků vedení školy. Připravovaný informační systém je realizován jako webová aplikace, proto byl pro tvorbu modulu použit PHP framework Symfony.

Modul přiřadí učitelům předměty a vyučovací hodiny dle jejich pracovního úvazku a aprobace, dále v rámci vypsaných kapacit zapisuje studentům volitelné předměty dle jejich preferencí. Veškeré změny modul verzuje a výstupy lze exportovat pro potřeby školy.

Funkčnost modulu byla ověřena integrací do prototypu informačního systému a následně proběhlo uživatelské testování.

Klíčová slova informační systém, střední škola, Symfony, tvorba úvazků, zápis předmětů, webová aplikace

Abstract

The main purpose of this bachelor thesis is to analyze the needs of the Michael school - Secondary school and College of Advertising and Art and to create a module, designed to assign schedules to teachers and to select optional courses for students, which will be implemented into the school's information system currently under development.

The first part of the thesis deals with the analysis of requirements and existing solutions. The second part of this thesis describes the implementation of the module in regard to the school's management's requirements. The information system is implemented as a web application, therefore the PHP framework Symfony was used to create the module.

The module assigns subjects and teaching hours to teachers according to their work schedule and requirements. The module also assigns optional courses to students according to their preferences. The module supports versioning and outputs can be exported for the school's internal use.

The functionality of the module was verified by integration into the prototype of the information system and subsequently the module was tested by users.

Keywords information system, high school, Symfony, time jobs creation, subjects enrollment, web application

Obsah

Seznam obrázků

Seznam ukázek kódu

Úvod

V současné době Střední škola a vyšší odborná škola reklamní a umělecké tvorby Michael využívá pro správu žáků, předmětů, učitelů, tříd a rozvrhů, podobně jako většina škol v České republice, informační systém Bakaláři.

Tento systém však často nesplňuje konkrétní požadavky jednotlivých především specializovaných škol, protože se jedná o univerzální nástroj. Zároveň není tento software volně rozšiřitelný a proto jej nelze jednoduše a nezávisle na tvůrcích systémů rozšířit o požadované funkcionality.

Konkrétně se u školy Michael jedná o výběr volitelných předmětů žáky školy, proces tvorby úvazků učitelů včetně zahrnutí volitelných předmětů, správu praxí nebo závěrečných prací, které jsou pro některé studenty povinné. Proto se vedení školy rozhodlo pro vlastní informační systém. V první fázi je třeba vytvořit jádro systému, modul pro tvorbu úvazků a zápisů volitelných předmětů a v neposlední řadě modul pro správu praxí a závěrečných prací.

Cílem práce je provést analýzu požadavků vedení školy, navrhnout vhodné řešení a vypracovat modul pro správu úvazků a výběr volitelných předmětů studenty školy, funkčnost modulu ověřit integrací do prototypu systému a následně provést testování.

Výběr volitelných předmětů studenty probíhá papírovou formou a studenti často dělají chyby při výběru předmětů. Dále je třeba zefektivnit tvorbu úvazků na základě počtu studentů ve volitelných předmětech. Téma jsem si zvolil především proto, že se jedná o výpomoc střední škole a pokud bude moje práce splňovat všechny požadavky, bude modul integrován do systému a reálně využíván.

Kapitola 1

Rešeršní část

Tato kapitola pojednává o současném stavu realizace procesů na Střední škole a Vyšší odborné škole reklamní a umělecké tvorby Michael[1] (dále jen škola Michael). Analyzuje již existující řešení na trhu a popisuje funkční a nefunkční požadavky, které vyplynuly z rozhovoru se zaměstnanci školy.

1.1 Současný stav

V současné době škola využívá v České republice velmi rozšířený systém Bakaláři[2], který svou funkčností pokrývá potřeby většiny škol. Ale díky specializaci školy nemůže komerční software pokrýt všechny tyto potřeby. Konktrétně se u školy Michael jedná o požadavky na správu závěrečných prací a praxí, na které studenti školy dochází, dále pak na volbu volitelných předmětů studenty školy, vyhodnocení výsledků voleb studentů a následné vytvoření úvazků učitelů dle vypsaných předmětů.

Celkem vzniknou tři moduly, které ve výsledku pokryjí potřeby školy Michael. V rámci práce „*Webová administrace informačního systému pro střední školu*" [1] vznikne administrace, databáze studentů, tříd a učitelů. V rámci práce „*Modul pro evidenci praxí a závěrečných prací pro informační systém střední školy*" vznikne modul pro správu praxí a závěrečných prací a součástí této práce je implementace modulu pro tvorbu úvazků a výběr volitelných předmětů.

1.1.1 Výběr volitelných předmětů

V praxi nyní funguje výběr volitelných předmětů papírovou formou. Zaměstnanci školy připraví papírový dotazník 1.1, který následně rozdají studentům školy. Vyplněný dotazník poté studenti odevzdají třídnímu učiteli.

[1] Michael - Střední škola a Vyšší odborná škola reklamní a umělecké tvorby, s.r.o., Machkova 1646, Praha 4, 149 00; dostupné z: `http://www.skolamichael.cz`.

[2] Dostupné z: `https://www.bakalari.cz`.

Z odevzdaných dotazníků se vytvoří statistika preferencí volitelných předmětů studenty a vyberou se nejvíce žádané předměty, které budou následující školní rok vyučovány.

Přihláška - povinně volitelné semináře pro školní rok 2017/2018

4. ročník

Fotografická tvorba

Žáci si vyberou z následujících seminářů tak, aby celkový počet hodin byl **minimálně 8 vyučovacích hodin**. **Povinně zvolte (1 hodina) Seminář z DTR nebo Tmt dle zaměření maturity*.**

V tabulce zvolte prioritní volitelný seminář a náhradní seminář (v případě malého počtu zájemců).

Při zapisování do seminářů se bude přihlížet k datu odevzdání přihlášky. Přihlášku je nutné odevzdat třídnímu učiteli **nejpozději do 17. 5. 2017**. Po tomto termínu budou semináře přiřazeny školou bez nároku na změnu. Počet žáků v jednotlivých seminářích je limitován.

Předmět	Hodinová dotace týdně	Volba 1 - prioritní	Volba 2 - náhradní
Grafické programy II	2		
Počítačové zpracování obrazu II	2		
Flash a animace	2		
Figurální kresba	3		
Filmový seminář	2		
Seminář z DTR*	1		
Seminář z Tmt*	2		

Povinně jsou zavedeny pro všechny žáky seminář z ČJL (1 hodina), Fotografický seminář (1 hodina) a Seminář z dějin výtvarné kultury (1 hodina). Tyto hodiny nejsou obsaženy v tabulce.

Jméno a příjmení žáka:

Podpis žáka:

Obrázek 1.1: Papírová forma dotazníku pro výběr volitelných předmětů

Toto řešení s sebou nese mnoho komplikací, které celý proces vypisování volitelných předmětů ztěžují. Studenti si mohou vybírat z několika skupin předmětů, kdy se typicky jedná o výběr určitého minimálního počtu předmětů z nabídky a/nebo výběr určitého minimálního počtu hodin týdně z nabídky předmětů. Studenti v dotazníku vyplní svou preferovanou volbu a také volbu náhradní, pokud by se jimi preferovaný předmět z kapacitních důvodů následující školní rok nevyučoval.

Studenti často dotazníky neodevzdají včas, vůbec, nebo je odevzdají neúplně vyplněné, což komplikuje jejich zpracování.

Z analýzy tedy vyplynulo, že je potřeba implementovat rozhraní, které umožní zaměstnancům školy vytvořit pro každý studijní obor a ročník nabídku předmětů, ze které si studenti po přihlášení jimi preferované předměty vyberou. Je potřeba, aby aplikace validovala informace zadané studenty a neumožnila jim odeslat neúplné nebo chybné údaje.

1.1.2 Tvorba úvazků

V současné době tvorba úvazků probíhá pomocí papírových tabulek, které si připravují zaměstnanci školy. Toto řešení s sebou přináší mnohé komplikace, jako je například nepřehlednost a neefektivita. Navíc je na škole potřeba zavést systém pro sběr požadavků učitelů jak na svůj rozvrh, tak například na třídy, ve kterých chtějí vyučovat. Zavedení sběru požadavků by v současné době ještě více zkomplikovalo práci a přineslo více papírových formulářů, což není žádoucí. Proto je třeba vytvořit rozhraní pro sběr požadavků od učitelů v elektronické podobě a zohlednit je poté při tvorbě úvazků, která by měla nově probíhat také elektronickou formou.

1.2 Existující řešení

Na trhu existují mnohá řešení informačních systémů pro školy. Většina z nich jsou modulární a pokrývají tak požadavky většiny základních a středních škol.

„Ústav pro informace ve vzdělávání uskutečnil v roce 2004 dotazníkového šetření, kterého se zúčastnilo 4 172 ZŠ, SŠ, VOŠ a pomocných škol. Z šetření vyplynulo že, 52 % škol vede evidenci žáků s využitím evidenčního SW, tedy nějakého elektronického IS.“[2]

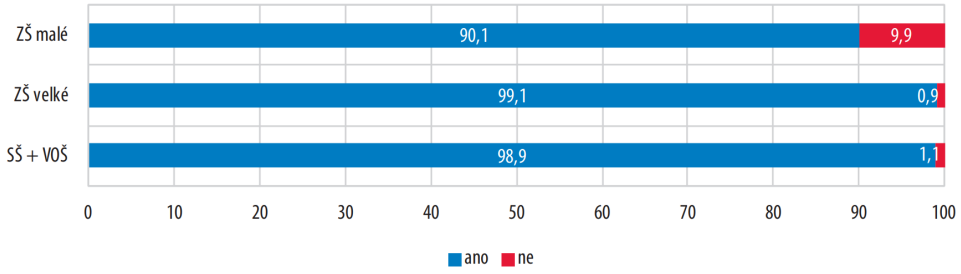
Toto číslo lze porovnat se zjištěním, které ve své výroční zprávě za školní rok 2016/2017[3] prezentovala Česká školní inspekce[3].

Česká školní inspekce rozdělila zkoumané školy na tři skupiny – malé ZŠ, velké ZŠ a SŠ+VOŠ. Ve skupině „malé SŠ“ využívá interní informační systém 90,1 % škol, u skupiny „velké ZŠ“ činí podíl 99,1 % a u „SŠ+VOŠ“ 98,9 %. Celkově se tedy podíl škol, které využívají informační systém k vedení vnitřní agendy, pohybuje nad 90 %. Tento nárůst má podle Česká školní

[3]Česká školní inspekce, dostupné z: `http://www.csicr.cz`.

inspekce na svědomí jednak rozšíření informačních technologií, jednak nárůst agendy a výkazů, které musejí školy vést a uchovávat. Podle ČŠI lze užívání informačních systémů ve školství hodnotit jako významně pozitivní.[3]

„*Téměř všechny velké základní, střední a vyšší odborné školy využívají pro vedení svých agend k tomu určené specifické informační systémy. Přibližně desetina malých základních škol pak tyto systémy nevyužívá. Významně jsou zastoupeny čtyři produkty konkrétních dodavatelů informačních systémů pro správu agend školy (pokrytí více než 10 % škol v daném segmentu).*“[3]



Obrázek 1.2: Využití informačních systémů pro vedení agend školy – podíl ZŠ, SŠ a VOŠ (v %)[3]

Podle ankety, která byla spuštěna 2. listopadu 2015 a běží dosud v diskusním fóru na portálu `rvp.cz`[4], který je určen pro podporu vzdělávání, jsou nejčastěji využívané systémy s podílem zastoupení v % následující:

Bakaláři [5] – 39 %

Škola OnLine [6] – 19 %

dm Software [7] – 12 %

aSc Rozvrhy [8] – 8 %

iŠkola [9] – 7%

SAS [10] – 7 %

[4]Metodický portál RVP.CZ, dostupné z: `https://rvp.cz`
[5]Dostupné z: `https://bakalari.cz/`.
[6]Dostupné z: `https://www.skolaonline.cz/`.
[7]Dostupné z: `https://portal.dmsoftware.cz/`.
[8]Dostupné z: `https://www.asctimetables.com/`.
[9]Dostupné z: `https://www.iskola.cz/`.
[10]Dostupné z: `http://sas.edookit.cz/`.

V následující části budou popsány dva systémy, které mají podle ankety nejvyšší podíl zastoupení – Bakaláři a Škola OnLine. Další řešení v pořadí od firmy dm Software je velmi podobné systému Škola OnLine, použité moduly jsou téměř identické po stránce funkční i vzhledové.

1.2.1 Bakaláři

Informační systém Bakaláři patří mezi nejrozšířenější v českých školách, jak uvádí sám výrobce.[4] Jedná se o modulární systém, který lze provozovat na školní síti, plně v tzv. cloudu[11] nebo kombinovaně, tedy běžící na vlastní infrastruktuře školy a plně zálohované na vzdálené úložiště. Celý systém se skládá z modulů, které školy mohou podle potřeby dokupovat od výrobce. Mezi hlavní moduly systému patří:

Evidence žáků a zaměstnanců, školní matrika
Modul umožňuje správu a evidenci veškerých osob, o kterých škola uchovává osobní údaje, včetně osobních údajů o rodičích žáků. Modul umožnuje vést u každého žáka průběžnou i pololetní klasifikaci. Nabízí export dat pro MŠMT a také sumarizaci údajů pro tisk.

Internetová žákovská knížka
Jedná se o samostatnou webovou aplikaci, která žákům a jejich rodičům zpřístupňuje údaje o klasifikaci, docházce a rozvrhu studenta. Aplikaci lze využít jako nástroj pro komunikaci rodičů se zaměstnanci školy jako je např. omlouvání žáků z vyučování.

Rozvrh hodin, suplování, plán akcí školy, rozpis maturit
Poskytuje prostředí pro tvorbu rozvrhů, které hlídá kolize i využití jednotlivých učeben. Pro rozšíření této funkcionality obsahuje systém generátor, který se snaží předejít kolizím. Umožňuje vypisovat suplování, akce školy a plánovat časy a využití učeben při maturitách.

Třídní kniha, tematické plány
Elektronická třídní kniha umožnuje zapisovat absenci žáků učiteli přímo v počítači v učebně. Tato data jsou okamžitě přístupná také v internetové žákovské knížce, kam mají přístup studenti i jejich rodiče. Na systém lze rovněž napojit i systém elektronické evidence příchodů a odchodů z budovy školy.

Přijímací zkoušky, knihovna, inventarizace
Je skupina jednotlivých drobných modulů, které zajišťují evidenci uchazečů o studium, evidenci výtisků a vypůjček ve školní knihovně nebo inventarizaci školního majetku.

[11] na vzdáleném úložišti, na které mohou uživatelé přistupovat odkudkoliv

K dispozici jsou další drobné moduly, které usnadňují například evidenci úrazů, hospitací, cestovních příkazů, přijímacích zkoušek, skladu nebo pracovních dohod. Pro žáky a jejich rodiče existuje také aplikace pro mobilní telefony - `Bakaláři - oficiální aplikace`[12], která napomáhá ke zpřístupnění údajů.

1.2.2 Škola OnLine

„Škola OnLine je moderní školní informační systém, který umožňuje rychle a efektivně zpracovávat veškerou školní agendu při zachování vysokého uživatelského komfortu. Jedná se o webovou aplikaci, což znamená, že je dostupná 24 hodin denně prostřednictvím Internetu, a to při využití pouze běžného webového prohlížeče bez nutnosti jakékoliv další instalace.“[5]

Podobně jako systém `Bakaláři` i zde se jedná o modulární systém, jehož cena se odvíjí právě od počtu zakoupených modulů. Narozdíl od `Bakalářů` se jedná kompletně o webovou aplikaci, což umožňuje přistupovat k funkcím systému pouze pomocí webového prohlížeče bez nutnosti instalace proprietárního software na klientský počítač.

Moduly tohoto systému téměř identicky kopírují funkcionalitu i rozdělení modulů v systému `Bakaláři`, proto je níže uveden pouze výčet hlavních modulů bez jejich popisu.

Jádro systému

Školní matrika, evidence osob

Rozvrh a suplování

Třídní kniha a evidence docházky

Evidence průběžného hodnocení

Uzávěrky známek a tisk vysvědčení

Návrhář tiskových sestav a dokumenty

Export dat ze školní matriky pro MŠMT

Systém taktéž nabízí mobilní aplikaci[13] pro podporu přístupnosti dat pro žáky a rodiče. Mezi další moduly patří například: učební plány, zápis do 1. ročníku, knihovna, sklad. Navíc systém obsahuje modul pro správu veřejných webových stránek školy.

[12] Dostupná pro Android, iOS, Windows 10
[13] Dostupná pro Android, iOS, Windows Phone

Kapitola 2

Analýza a návrh

2.1 Funkční požadavky

Ke správě a tvorbě nabídky volitelných předmětů k zápisu, ale i k vytvoření úvazků vyučujících je potřeba v databázi evidovat veškeré informace o učitelích, třídách a studentech - tuto funkcionalitu poskytuje modul administrace vytvořený v rámci práce [1]. Autorem vytvořený modul bude navíc evidovat všechny předměty vyučované na škole, zápisy volitelných předmětů a vztahy mezi nimi a studenty. Všechna data o zápisech a odpovědích studentů musejí být držena v historii a snadno dohledatelná.

2.1.1 Vypsání nabídky volitených předmětů

Aplikace musí umožňovat vytvořit nabídku volitelných předmětů k zápisu (dále jen zápis) pro studenty daného studijního oboru v daném ročníku. Například tedy vytvořit zápis pro studenty oboru „Filmová tvorba“, kteří jsou aktuálně ve druhém ročníku studia na škole. Dále je třeba některé předměty definovat jako navazující na jiné a neumožnit studentům si takový předmět zapsat, pokud nestudovali požadované předchozí předměty.

Každý zápis se skládá z několika skupin předmětů, které mohou být trojího typu:

Automaticky zapsané předměty pro všechny
: Předměty obsažené v této skupině jsou specifické pro daný obor a ročník, ale jsou zároveň i povinné. Proto jsou zapsány automaticky bez nároku studentů na výběr.

Výběr minimálně *n* hodin z nabídky
: Z této skupiny předmětů si student musí vybrat minimální daný počet hodin za týden. Zadáno je také maximum hodin, které volba studenta musí splňovat.

Výběr minimálně *n* předmětů z nabídky
Z této skupiny předmětů je student povinnen si vybrat minimální daný počet předmětů. Volba studenta musí také respektovat zadané maximum.

Zápis musí být pro studenty dostupný pouze v daném časovém období a součástí zápisu musí být stručný popis a návod pro studenty.

2.1.2 Volba volitelných předmětů studenty

Studenti školy se pomocí vygenerovaného kódu přihlásí do webové aplikace, kde uvidí seznam zápisů, které se jich týkají. U vybraného zápisu se zobrazí doprovodný text k zápisu a také nabídka předmětů formou formuláře. Studenti si vybírají z vypsaných skupin předmětů tak, aby jejich volba splňovala příslušná kritéria. Každý student provede v dané skupině předmětů celkem dvě volby:

Primární volba studenta jsou předměty, které student preferuje k zápisu.

Náhradní volba studenta jsou předměty, které budou studentovi přiřazeny, pokud některý předmět, který si zvolil jako primární, nebude vypsán.

Systém nesmí umožnit studentovi odeslat a uložit volbu, která nesplňuje požadavky na minimální a maximální počty primárně vybraných předmětů a hodin.

Po ukončení zápisu a vyhodnocení výsledků musí systém studentovi po přihlášení zobrazit, které předměty mu byly přiřazeny.

2.1.3 Vyhodnocení zápisu volitelných předmětů

Po ukončení zápisu je potřeba vyhodnotit volby studentů a vytvořit statistiku zájmu o předměty a zároveň zobrazit počty preferovaných a náhradních voleb ke každému předmětu.

Administrátor poté každému studentovi dle kapacity předmětu, zájmu o předmět a podle volby studenta přiřadí předměty tak, aby byly splněny požadavky na minimální a maximální počty hodin nebo předmětů.

Systém by měl administrátora, který provádí vyhodnocení zápisu upozornit, pokud některý z žáků má přiřazeno více nebo méně předmětů a hodin. Dále je požadováno zachování původních odpovědí studentů v systému.

2.1.4 Evidence předmětů a aprobací

Evidence předmětů je nezbytnou součástí systému. Systém musí evidovat název předmětu, jeho kód a popis předmětu, který by měl pomoci studentům při zapisování předmětů. Každý předmět může obsahovat libovolný počet skupin, do kterých jsou studenti rozděleni. Systém musí umožnit ke každému učiteli evidovaném v systému přiřadit předměty, které smí vyučovat.

2.1.5 Sběr požadavků od učitelů

Systém musí každému přihlášenému učiteli zobrazit formulář pro výběr preferovaných tříd a předmětů. Také musí umožnit vyplnit požadovaný počet vyučovacích hodin a zobrazit tabulku reprezentující rozvrh hodin, ve které si učitel vybere hodiny, které preferuje vyučovat. Systém by měl kontrolovat, zda se preference na vyučovaný předmět shodují s požadovaným týdenním úvazkem učitele.

2.1.6 Tvorba úvazků

Po vyhodnocení zápisu a dokončení sběru požadavků na výuku a úvazky od učitelů je potřeba přiřadit jednotlivé třídy a předměty k učitelům. Nejprve se v systému přiřadí vyučované předměty k jednotlivým třídám.

V dalším formuláři se zobrazí takto vypsané paralelky spolu s vypsanými volitelnými předměty ze zápisů a seznam učitelů. Systém umožní přiřadit paralelky k učitelům a přitom zobrazovat aktuální a požadovaný úvazek učitele. Systém by měl umožnit filtrování podle předmětů jak mezi aprobacemi učitelů, tak mezi vypsanými paralelkami. Zároveň systém musí upozornit administrátora, pokud přiřadí předmět k učiteli, který nemá aprobaci jej vyučovat.

2.1.7 Test dat

Pro zjištění aktuálního stavu zápisů budou sloužit testy dat. Je vhodné testovat stav vyplněnosti zápisů, vypsat studenty, kteří dosud zápis nevyplnili, vypsat nevyhodnocené zápisy a zkontrolovat úvazky učitelů.

Při porušení některých podmínek vhodně upozornit administrátora nebo samotné studenty, aby zápis vyplnili.

2.1.8 Export dat

Další nepostradatelnou funkcionalitou je vytváření výpisů dat z databáze. Výpisy by se měly zobrazit v aplikaci, ale měly by být k dispozici i pro tisk a ke stažení. Dostupné by měly být minimálně tyto výpisy:

- učitel a předměty, které vyučuje
- student a jeho předměty
- učitel a úvazky
- předmět, vyučující učitel a zapsaní studenti podle tříd

2.2 Nefunkční požadavky

Z důvodu dostupnosti zápisu předmětů studenty je vhodné implementovat systém jako webovou aplikaci dostupnou on-line. Dále je žádoucí její fungování na běžném ne příliš drahém webhostingu, aby nedošlo k zbytečnému zatížení rozpočtu školy.

2.2.1 Webová aplikace

Responzivní design
: Webové rozhraní musí být korektně zobrazováno jak na osobních počítačích, tak mobilních telefonech i tabletech.

Kompatibilita se všemi běžnými prohlížeči
: Internet Explorer 10+, Microsoft Edge, Opera 10+, Firefox 35+, Chrome 55+

2.2.2 Technologie

Zvolené technologie řešení vyplývají z technologií použitých v jádře informačního systému, které musí modul pro zachování kompatibility také využívat.

PHP 7+

Symfony 3.4.6

MySQL 5+

Doctrine 2.5

HTML 5

CSS 3

Sass

Composer

Gulp

Yarn

JavaScript

jQuery 1.9+

Bootstrap 3+

Twig

Validace formulářů musí probíhat také na straně serveru, aby nebylo možné ji obejít pouhým vypnutím JavaScriptu v prohlížeči.

2.3 Uživatelské role

Protože se jedná o uzavřený systém pouze pro studenty a učitele školy Michael, neexistuje žádná veřejná část pro anonymní uživatele a každý účastník se musí přihlásit. Studenti mají přístup pouze do rozhraní pro zápis. Ostatní uživatelé do plné administrace systému. Přihlášení studentů je řešeno v rámci této práce, přihlašování učitelů a administrátorů obstarává administrace, do které se tento modul po dokončení integruje.

Role přihlášených studentů – prostředí zápisu:

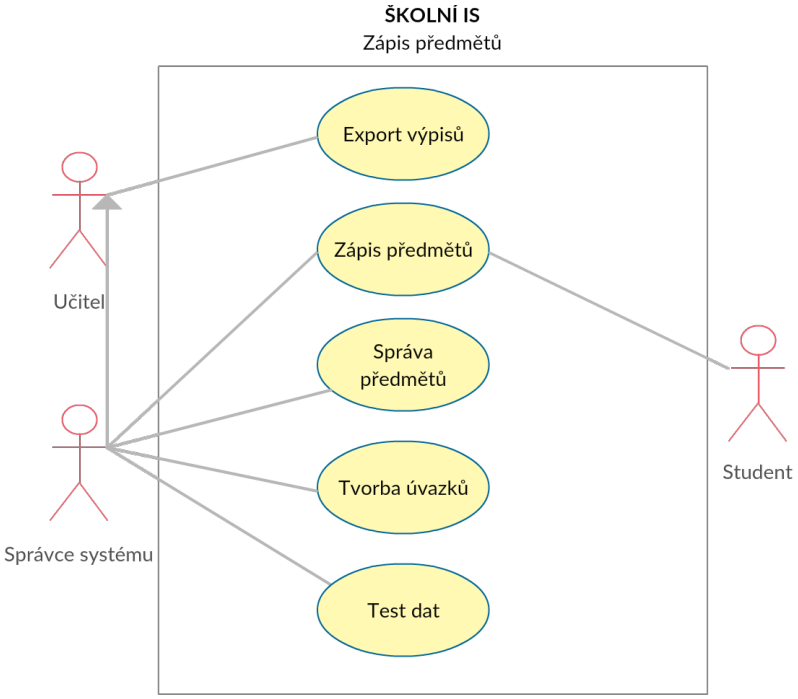
- Student

Následující role vždy přejímá oprávnění skupin pod sebou. Role přihlášených uživatelů – prostředí administrace:

- Administrátor
  Administrátor v rámci administrace má veškerá oprávnění kromě oprávnění pro vyplnění požadavků na úvazek, pokud nemá zároveň roli učitele. Oprávnění umožňuje kompletní správu zápisů, aprobací a předmětů. Dále je zpřístupněna funkce pro zpracování výsledků zápisu. Uživatel spravuje úvazky učitelů, přiřazuje učitelům vyučované předměty, vyhodnocuje zápisy a provádí kompletní správu systému.
- Učitel
  Učitel má oprávnění vyplnit své požadavky na úvazek, preferované předměty a třídy pro další školní rok.

2.4 Případy užití

Následující kapitola obsahuje popis případů užití vytvářeného modulu tak, jak vyplynuly z rozhovorů se zaměstanci školy a funkčních požadavků. Na obrázku 2.1 jsou znázorněny hlavní případy užití, které jsou dále rozpracovány v samostatných podkapitolách.

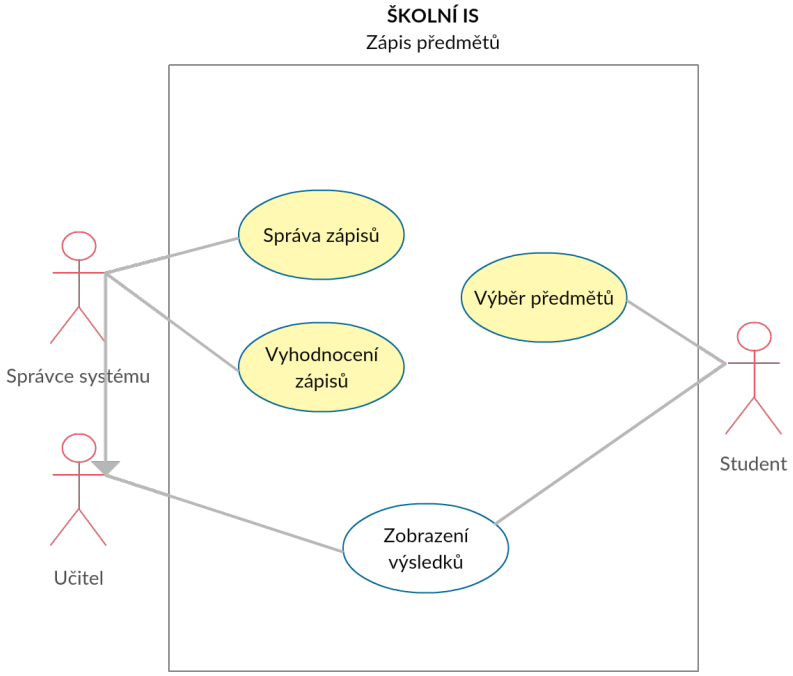


Obrázek 2.1: Případy užití modulu pro tvorbu úvazků a výběr volitelných předmětů

2.4.1 Zápis volitelných předmětů 2.2

Z důvodu přehlednosti je diagram 2.2 pro obsáhlé případy užití rozdělen na jednotlivé diagramy, které budou popsány v následujících podkapitolách.

Uživatel vybere v administraci položku „*Zápis předmětů*“ a následně se zobrazí podmenu umožňující výběr konkrétní akce. Pokud je uživatel přihlášen jako učitel, v celém zápisu předmětů se mu budou zobrazovat pouze zápisy související se třídami, které vyučuje.



Obrázek 2.2: Případy užití: Zápis předmětů

- Zobrazení výsledků
  Správce systému, učitel, nebo student vybere u příslušného zápisu předmětů možnost „Zobrazit výsledky“ a systém následně zobrazí vypsané předměty a jejich hodinovou dotaci pro daný školní rok.

2.4.1.1 Správa zápisů 2.3

- Vytvoření zápisu
  Administrátor vybere možnost „Vytvořit zápis“ a vyplní formulář, který systém zobrazí. V případě korektního vyplnění zápisu systém přesměruje administrátora na detail zápisu 2.19. V opačném případě vyzve administrátora k opravení nevalidních dat.

- Změna viditelnosti zápisu
  Administrátorovi se dle aktuálního stavu zápisu zobrazí možnosti „Označit dokončený“ nebo „Označit rozpracovaný“. Administrátor příslušnou akci vybere a potvrdí systémem zobrazené výstražné okno. Při potvrzení varování se akce provede, v opačném případě nikoliv.

- Archivace zápisu
  Administrátor vybere tuto možnost, pokud zápis již není aktuální. Výběrem této možnosti dojde k archivaci zápisu.

- Editace zápisu
  Administrátor vybere možnost „Upravit zápis“ a systém následně zobrazí formulář s poli dostupnými pro editaci v závislosti na stavu zápisu. Pokud odeslaný formulář obsahuje nevalidní data, systém na tuto skutečnost upozorní a vyzve administrátora k opravě.

- Vytvoření validačních skupin
  Na detailu zápisu 2.19 vybere administrátor požadovanou skupinu, která má být vytvořena, pokud to stav zápisu dovoluje. V zobrazeném formuláři vyplní příslušné vlastnosti skupiny. Po odeslání se v systému vytvoří nová validační skupina a přiřadí k danému zápisu.

- Přiřazení předmětů
  Administrátor vybere možnost „Přidat předmět“ pro již předem vytvořenou validační skupinu. Systém zobrazí formulář pro výběr předmětu a přiřazení hodinové dotace. Po odeslání validního formuláře dojde k přiřazení předmětu k validační skupině a tím i k příslušnému zápisu.

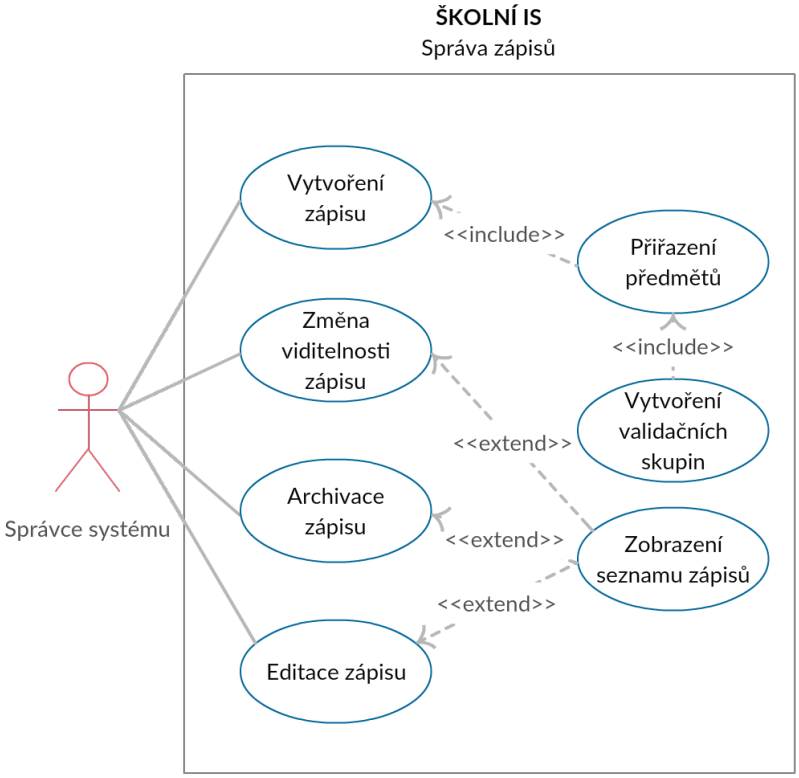
- Export dat
  Administrátor vybere ze seznamu položky, které budou exportovány a odešle formulář. Aplikace uloží data do souboru a v klientském prohlížeči zahájí stahování.

2.4.1.2 Výběr předmětů 2.4

- Zobrazení seznamu zápisů
  Studentovi se po přihlášení zobrazí seznam aktivních a vyhodnocených zápisů.

- Vyplnění zápisu
  Student vybere možnost „Vyplnit zápis“ u zápisu a systém zobrazí interaktivní formulář, který studenta provede výběrem předmětů. Student vyplní jednotlivé formuláře pro každou skupinu a pro primární i náhradní volby. V případě odeslání validních dat se studentova volba uloží, v opačném případě je student vyzván k opravě zadaných údajů.

- Editace odpovědi
  Pokud již student dříve zápis vyplnil a znovu vybere možnost pro vyplnění zápisu, systém zobrazí formulář s předvyplněnými minulými odpověďmi. Student je může upravit a znovu formulář odeslat.



Obrázek 2.3: Případy užití: správa zápisů

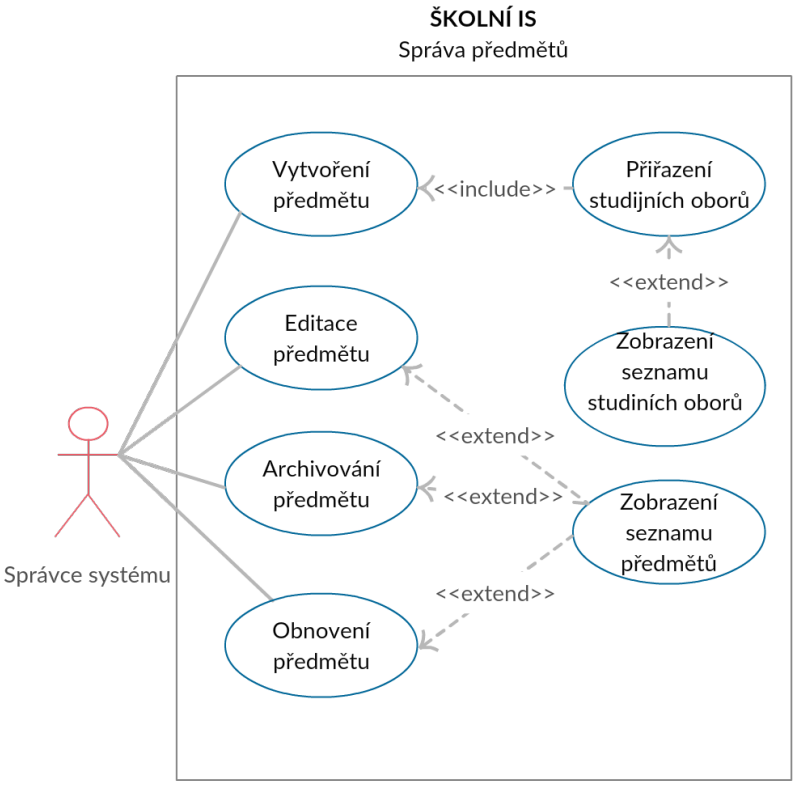
2.4.1.3 Vyhodnocení zápisu 2.5

- Vyhodnocení zápisu
  Administrátor vybere možnost vyhodnit zápis a systém zobrazí příslušný formulář. Administrátor vybere předměty, které mají být otevřeny a přiřadí je jednotlivým studentům. Administrátor odešle formulář kliknutím na tlačítko „Uložit změny“ a tím data uloží.

- Označení za rozpracovaný
  Pokud administrátor poprvé vybere možnost „Vyhodnotit zápis“, označí jej systém jako rozpracovaný.

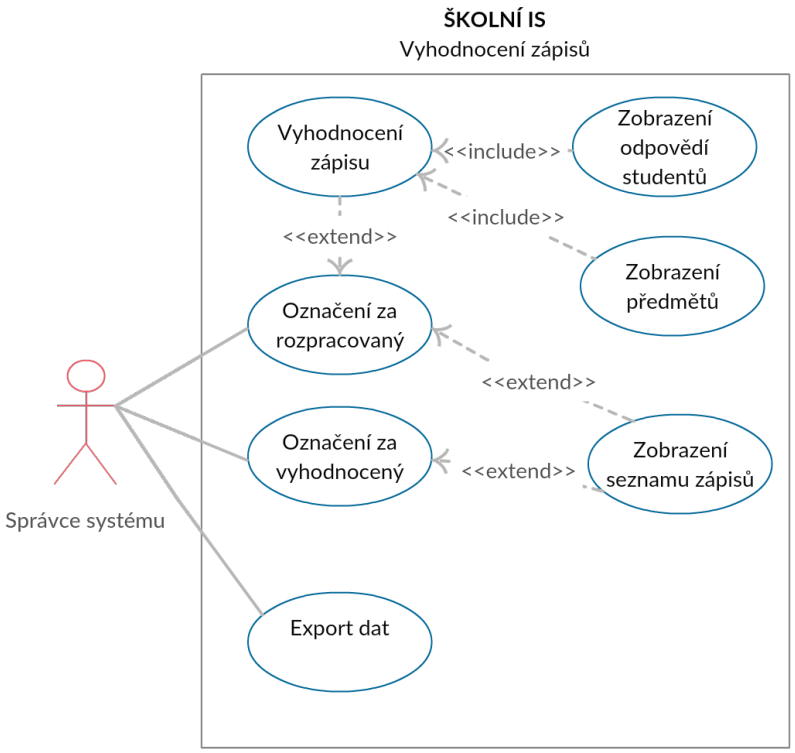
- Označení za vyhodnocený
  Pokud administrátor vybere možnost „Uzavřít zápis“, označí jej systém jako vyhodnocený.

- Odeslání emailů
  Administrátor vybere možnost „Odeslat email studentům“ a systém odešle studentům výsledky zápisu. O výsledku systém administrátora informuje.



Obrázek 2.4: Případy užití: výběr předmětů

- Export dat
  Administrátor vybere ze seznamu položky, které budou exportovány a odešle formulář. Aplikace uloží data do souboru a v klientském prohlížeči zahájí stahování.

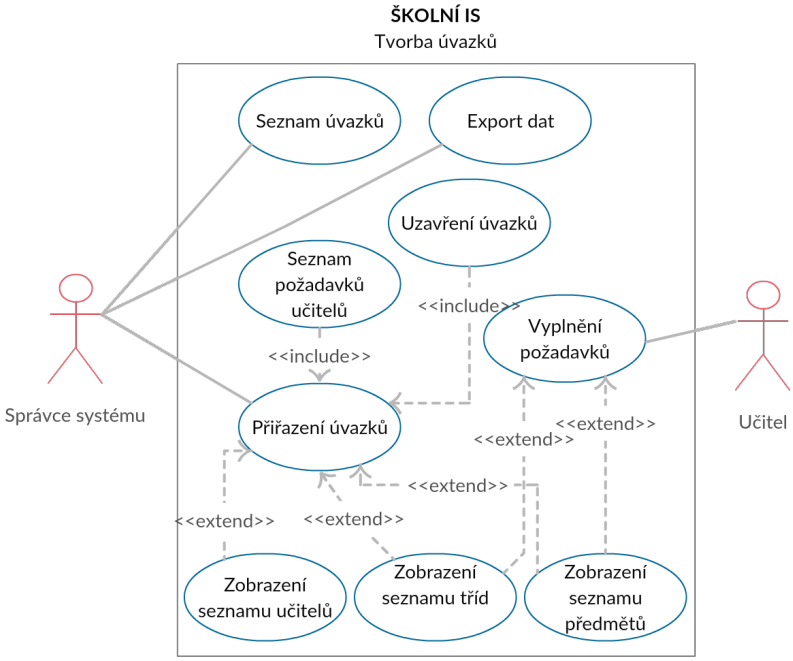


Obrázek 2.5: Případy užití: vyhodnocení zápisu

2.4.2 Tvorba úvazků 2.6

- Vyplnění požadavků
  Učitel vybere možnost „Mé požadavky“ a systém zobrazí interaktivní formulář pro zadání potřebných dat pro vyhodnocování zápisů. Učitel vyplní požadovaná pole formuláře a v tabulce reprezentující rozvrh hodin vybere preferované časy své výuky. V případě, že je formulář vyplněn korektně, data se uloží, jinak je učitel vyzván k opravě.

- Přiřazení úvazků
  Administrátor vybere možnost „Vytvořit úvazky“ a systém následně zobrazí první formulář pro vypsání předmětů s hodinovou dotací v jednotlivých třídách. Administrátor údaje vyplní a uloží změny. Poté administrátor vybere možnost „Další krok“ a systém zobrazí interaktivní formulář pro přiřazení vypsaných dvojic – třída a předmět – k jednotlivým učitelům. Administrátor údaje vyplní a uloží změny. Poté administrátor vybere možnost „Další krok“ a zobrazí souhrn úvazků. Administrátor vybere možnost „Uzavřít úvazky“ nebo „Zpět“.



Obrázek 2.6: Případy užití: tvorba úvazků

2.4.3 Správa předmětů 2.7

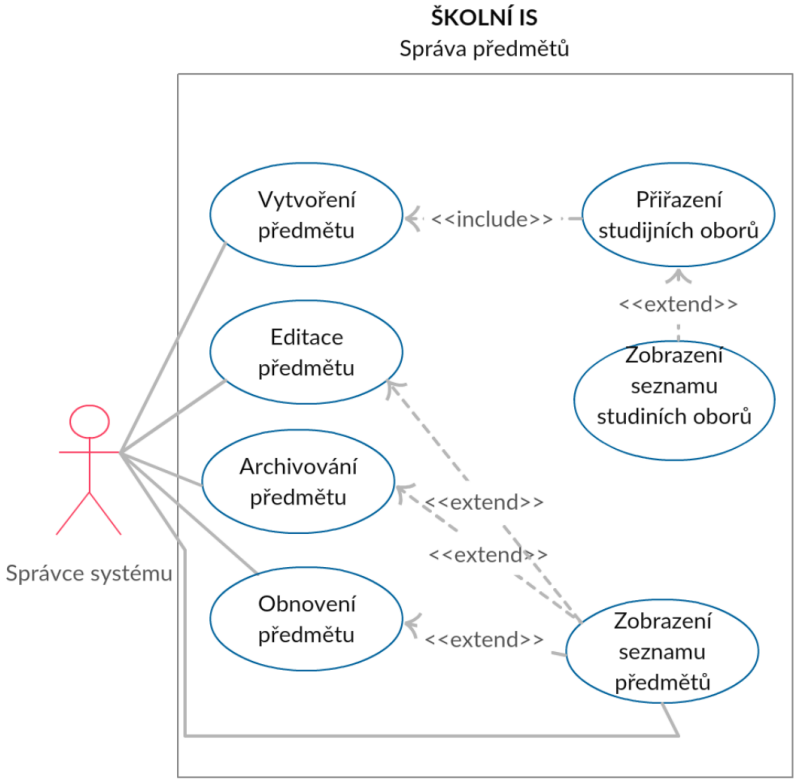
- Vytvoření předmětu
  Administrátor vybere možnost „Vytvořit předmět“ a systém zobrazí příslušný formulář. V případě vyplnění validních dat se předmět po odeslání formuláře administrátorem vytvoří. V opačném případě systém vyzve administrátora k opravě chybně zadaných údajů.

- Přiřazení studijních oborů
  Administrátor vybere možnost „Přidat studijní obor“, systém zobrazí formulář pro výběr konkrétího studijního oboru. Následně administrátor formulář odešle.

- Editace předmětu
  Administrátor vybere možnost „Upravit předmět“, systém zobrazí formulář s vyplněnými hodnotami. Administrátor formulář upraví a odešle, nové hodnoty se uloží.

- Archivování předmětu
  Administrátor vybere možnost „Archivovat předmět“, systém zobrazí výstražnou obrazovku. Administrátor potvrdí archivaci, nebo se vrátí zpět.

- Obnovení předmětu
  Administrátor vybere možnost „Obnovit předmět“, systém zobrazí výstražnou obrazovku. Administrátor potvrdí obnovení, nebo se vrátí zpět.

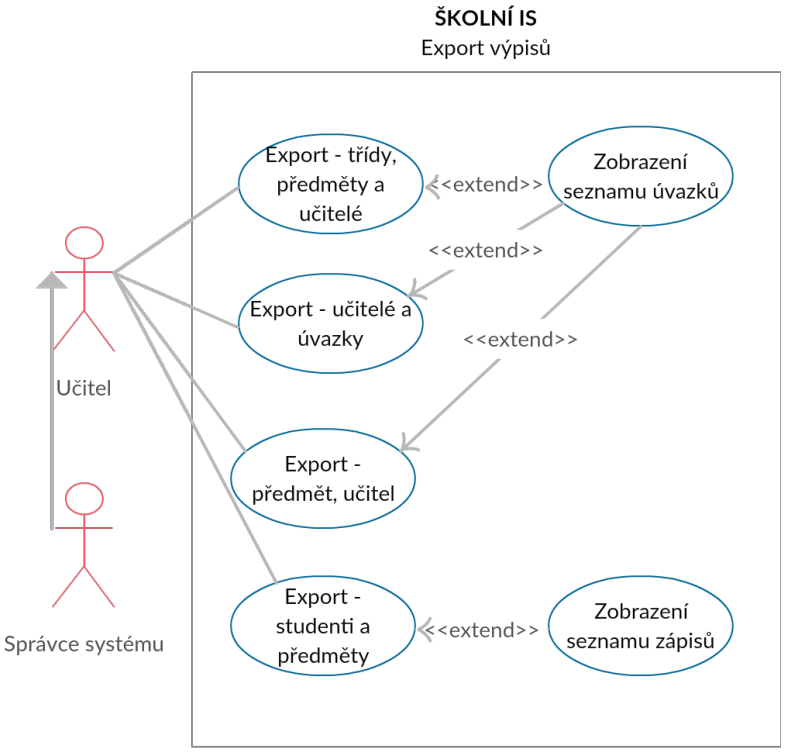


Obrázek 2.7: Případy užití: správa předmětů

2.4.4 Export dat 2.8

- Export - třídy, předměty, učitelé a úvazky
  Administrátor vybere u detailu úvazků pro daný školní rok požadovaný formát dat a systém následně spustí stahování souboru.

- Export - studenti a předměty
  Administrátor nebo učitel mající oprávnění vybere u detailu vyhodnoceného zápisu možnost „Zobrazit výsledky“ a následně vybere, zda požaduje exportovat data vztahů mezi předmětem a studentem indexovaná podle studenta nebo podle předmětu. Následně administrátor klikne na tlačítko „Exportovat“ a systém spustí stahování souboru.



Obrázek 2.8: Případy užití: export dat

2.5 Databázové schéma

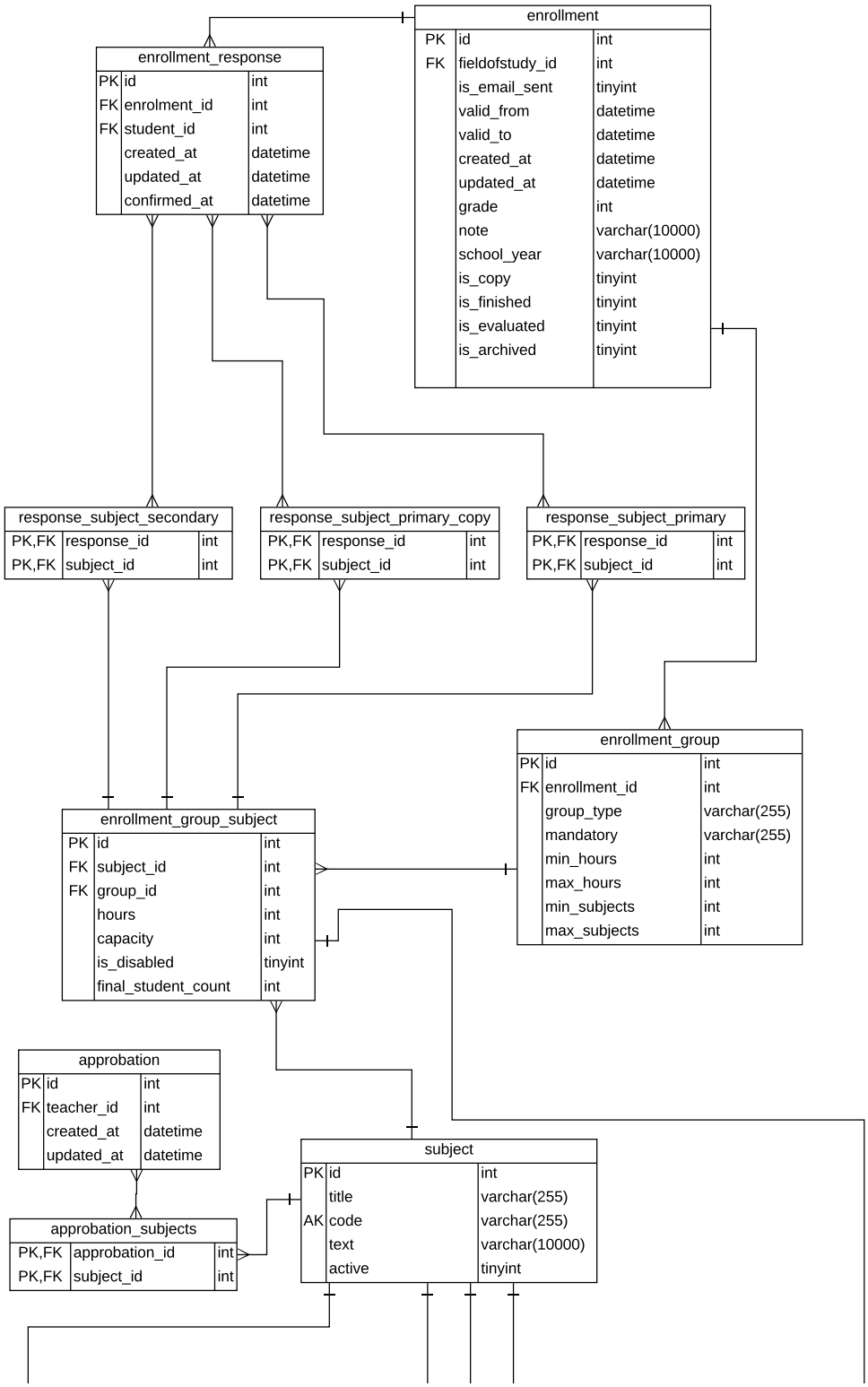
Následující kapitola popisuje databázové schéma, které bylo v rámci implementace modulu vytvořeno. Samotná databáze obsahuje navíc tabulky jádra informačního systému, které modul také pro svojí funkci využívá, nejsou však níže popsány, ani zobrazeny na diagramu 2.9.

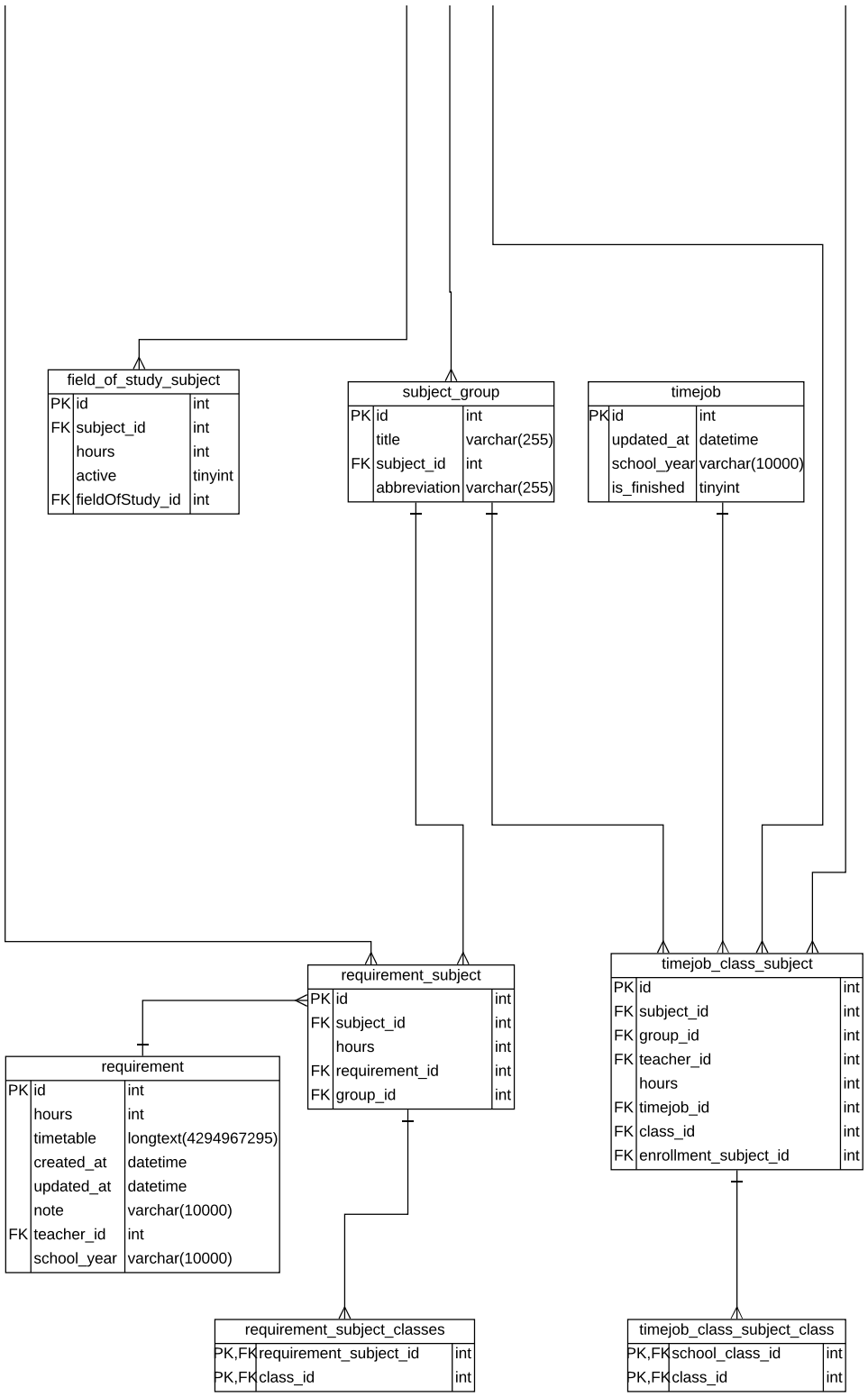
- Tabulka zápisů (enrollment)
  - **id:** identifikátor
  - **fieldofstudy_id:** identifikátor studijního oboru, ke kterému zápis patří
  - **is_email_sent:** příznak odeslání emailu s výsledky studentům
  - **valid_from:** datum vpuštění studentů do zápisu volitelných předmětů
  - **valid_top:** datum ukončení zápisu pro studenty
  - **created_at:** datum vytvoření zápisu
  - **updated_at:** datum poslední editace zápisu
  - **grade:** ročník, pro který je zápis vypsán
  - **note:** informace k zápisu pro studenty
  - **school_year:** školní rok, pro který je zápis vypsán
  - **is_copy:** příznak, zda byly odpovědi zápisů již duplikované
  - **is_finished:** příznak, zda je zápis označen jako připravený
  - **is_evaluated:** příznak, zda je vyhodnocení zápisu uzavřeno
  - **is_archived:** příznak, zda je zápis archivován
- Tabulka odpovědí studentů na zápis (enrollment_response)
  - **id:** identifikátor
  - **enrollment_id:** identifikátor zápisu, ke kterému odpověď studenta patří
  - **student_id:** identifikátor studenta, který odpověď odeslal
  - **created_at:** datum vytvoření odpovědi
  - **updated_at:** datum poslední editace odpovědi

- Tabulka validačních skupin zápisu (enrollment_group)
  - **id:** identifikátor
  - **enrollment_id:** identifikátor zápisu, ke kterému validační skupina přísluší
  - **group_type:** příznak typu skupiny
  - **mandatory:** příznak, zda mají být předměty ve skupině povinně zapsány
  - **min_hours:** minimální počet hodin, které si student musí mezi předměty vybrat
  - **max_hours:** maximální počet hodin, které si student musí mezi předměty vybrat
  - **min_subjects:** minimální počet předmětů, které si student musí mezi předměty vybrat
  - **max_subjects:** maximální počet předmětů, které si student musí mezi předměty vybrat
- Tabulka předmětů patřících k validačním skupinám (enrollment_group_subject)
  - **id:** identifikátor
  - **subject_id:** indetifikátor předmětu
  - **group_id:** identifikátor validační skupiny
  - **hours:** počet hodin
  - **capacity:** kapacita předmětu
  - **is_disabled:** příznak vyřazení předmětu ze zápisu během vyhodnocování
  - **final_student_count:** počet zapsaných studentů po vyhodnocení
- Tabulka předmětů (subject)
  - **id:** identifikátor
  - **title:** název předmětu
  - **code:** kód předmětu
  - **text:** informace o předmětu
  - **active:** příznak, zda je předmět aktivní

- Tabulka aprobací (approbation)
  - **id:** identifikátor
  - **teacher_id:** identifikátor učitele
  - **created_at:** datum vytvoření aprobace
  - **updated_at:** datum poslední editace aprobace
- Tabulka vztahů předmětů a oborů studia (field_of_study_subject)
  - **id:** identifikátor
  - **subject_id:** identifikátor předmětu
  - **hours:** počet hodin předmětu v daném oboru studia
  - **active:** příznak, zda je předmět stále vyučován do oboru
  - **fieldOfStudy_id:** identifikátor studijního oboru
- Tabulka skupin předmětů (subject_group)
  - **id:** identifikátor
  - **title:** název skupiny předmětu
  - **subject_id:** identifikátor předmětu
  - **abbreviation:** zkratka názvu skupiny
- Tabulka skupin úvazků pro školní rok (timejob)
  - **id:** identifikátor
  - **updated_at:** datum poslední editace úvazků
  - **school_year:** školní rok, do kterého jsou úvazky zařazeny
  - **is_finished:** příznak, zda jsou úvazky dokončené

- Tabulka přiřazení učitele ke třídě, předmětu a skupině (timejob_class_subject)
  - **id:** identifikátor
  - **subject_id:** identifikátor předmětu
  - **group_id:** identifikátor skupiny předmětu
  - **teacher_id:** identifikátor učitele
  - **hours:** počet hodin přiřazených učiteli
  - **timejob_id:** identifikátor skupiny úvazků
  - **class_id:** identifikátor třídy
  - **enrollment_subject_id:** identifikátor předmětu ze zápisu volitelných předmětů
- Tabulka požadavků učitelů (requirement)
  - **id:** identifikátor
  - **hours:** počet hodin
  - **timetable:** preferovaný rozvrh učitele
  - **created_at:** datum vytvoření požadavku
  - **updated_at:** datum poslední editace požadavku
  - **note:** poznámka učitele k požadavkům
  - **teacher_id:** identifikátor učitele
  - **school_year:** školní rok, ke kterému se požadavky vztahují
- Tabulka vztahů mezi předmětem, skupinou, třídou a požadavkem učitele (requirement_subject)
  - **id:** identifikátor
  - **subject_id:** identifikátor předmětu
  - **hours:** počet hodin/týden
  - **requirement_id:** identifiktor požadavku
  - **group_id:** identifikátor skupiny předmětu


enrollment
PK id int
FK fieldofstudy_id int
is_email_sent tinyint
valid_from datetime
valid_to datetime
created_at datetime
updated_at datetime
grade int
note varchar(10000)
school_year varchar(10000)
is_copy tinyint
is_finished tinyint
is_evaluated tinyint
is_archived tinyint
enrollment_response
PK id int
FK enrolment_id int
FK student_id int
created_at datetime
updated_at datetime
confirmed_at datetime
response_subject_secondary
PK,FK response_id int
PK,FK subject_id int
response_subject_primary_copy
PK,FK response_id int
PK,FK subject_id int
response_subject_primary
PK,FK response_id int
PK,FK subject_id int
enrollment_group
PK id int
FK enrollment_id int
group_type varchar(255)
mandatory varchar(255)
min_hours int
max_hours int
min_subjects int
max_subjects int
enrollment_group_subject
PK id int
FK subject_id int
FK group_id int
hours int
capacity int
is_disabled tinyint
final_student_count int
approbation
PK id int
FK teacher_id int
created_at datetime
updated_at datetime
subject
PK id int
title varchar(255)
AK code varchar(255)
text varchar(10000)
active tinyint
approbation_subjects
PK,FK approbation_id int
PK,FK subject_id int



Obrázek 2.9: Databázové schéma modulu

2.6 Návrh grafického uživatelského rozhraní

V následující kapitole jsou realizovány návrhy obrazovek pomocí wireframe[14] modelu, který slouží pro náhled hotového řešení. Jde o modely definující rozložení, funkce a obsah jednotlivých stránek aplikace. V návrhovém modelu wireframe se zpravidla nedbá na grafické prvky, barvy nebo obrázky. [6]

Byly realizovány návrhy všech důležitých obrazovek vyjma formulářů, jejichž formulářové prvky prakticky kopírují sloupce jednotlivých entit a také některé seznamy entit, které budou ve výsledné práci realizovány jako jednoduchá tabulka s proklikem na detail entity.

Při návrhu obrazovek byl kladen důraz zejména na to, aby odpovídaly funkčním požadavkům definovaným v kapitole 1.3.

Jelikož se jedná o interní aplikaci, všechny obrazovky, kromě přihlašovací obrazovky, předpokládají přihlášeného uživatele. V levém horním rohu uživatel vidí, ve které části se aktuálně nachází – *Administrace* nebo *Zápis volitelných předmětů*. V pravé části je umístěn odkaz pro změnu jazykové mutace a odhlášení z aplikace. Hlavní navigační menu se nachází v levém sloupci. Položky menu mohou být víceúrovňové.

2.6.1 Přihlášení studenta pro výběr volitelých předmětů

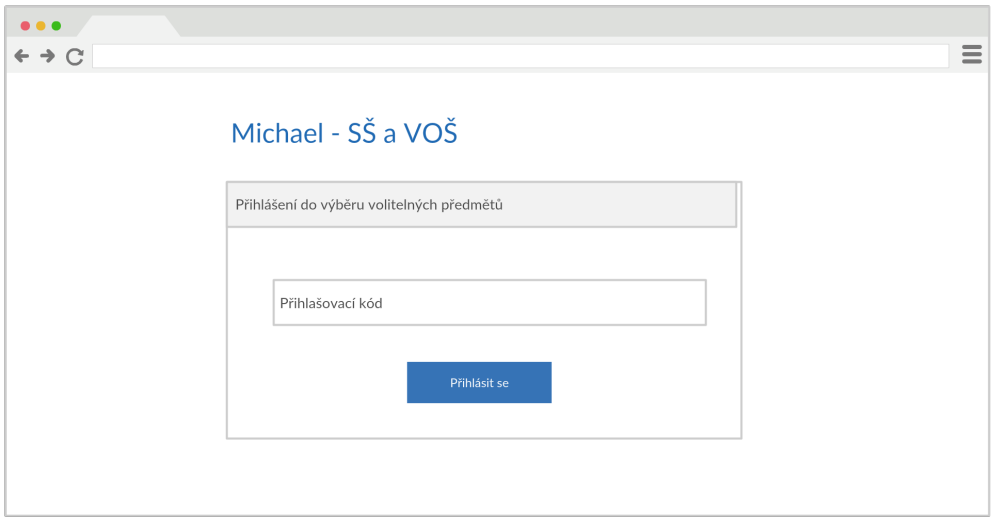
Dle funkčních požadavků je požadováno přihlášení studenta pouze pomocí přihlašovací kódu. Wireframe 2.10 reprezentuje veřejně dostupnou obrazovku, tudíž nemá klasickou strukturu popsanou výše a obsahuje pouze název školy, formulářové pole pro zadání přihlašovacího kódu a tlačítko pro odeslání formuláře. V případě neúspěchu bude nad formulářovým polem zobrazena chybová hláška.

2.6.2 Přehled zápisů studenta

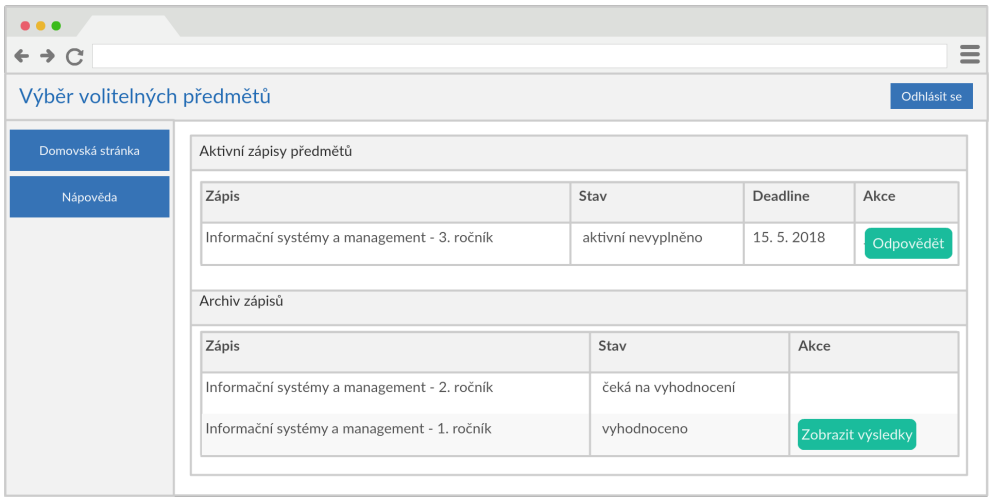
V hlavní části layoutu obrazovky jsou umístěny dvě tabulky. První tabulka obsahuje seznam zápisů, které jsou aktivní a student má zpřístupněný formulář 2.12 pro vybrání volitelných předmětů. K přesměrování na tento formulář slouží tlačítko „*Odpovědět*“ v posledním sloupci tabulky. Tabulka obsahuje důležitý údaj o termínu do něhož je možné zápis vyplnit, informaci o tom, zda student zápis vyplnil a zda je zápis aktivní. Po kliknutí na název zápisu bude student přesměrován na obrazovku 2.13 s detailem zápisu.

Druhá tabulka obsahuje seznam zápisů, jejichž datum platnosti již skončilo a v ideální případě je student také vyplnil. Název zápisu v prvním sloupci opět odkazuje na obrazovku 2.13 s detailem zápisu a voleb studenta. V momentě, kdy se stav zápisu změní z *čeká na vyhodnocení* na *vyhodnoceno*, se studentovi zpřístupní tlačítko pro zobrazení výsledků 2.14 zápisu.

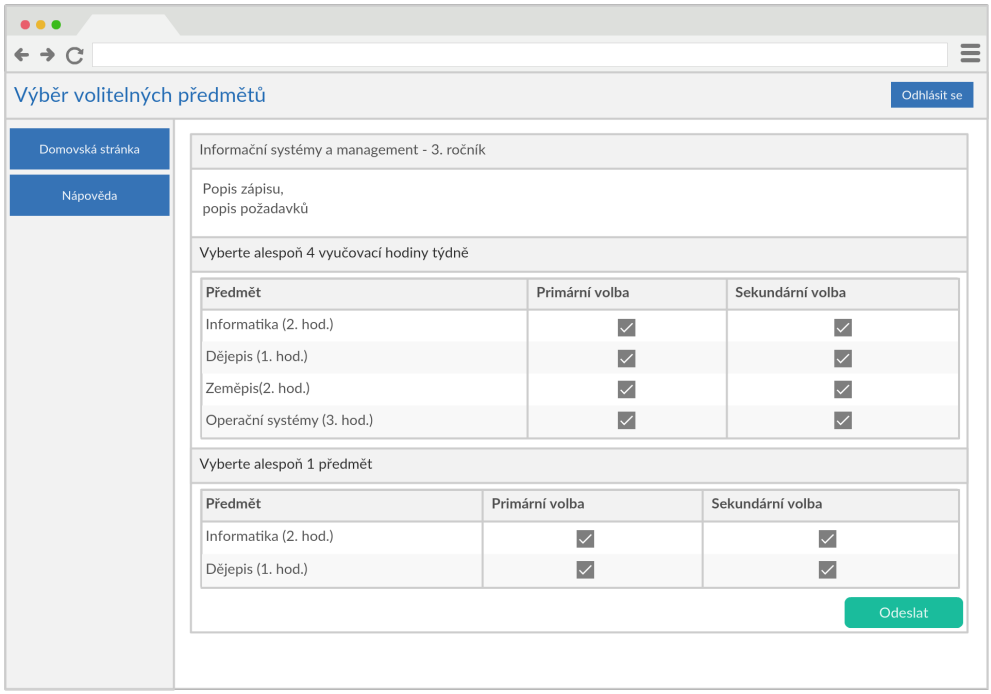
[14] drátěný model



Obrázek 2.10: Wireframe přihlášení studenta pro výběr volitelých předmětů



Obrázek 2.11: Wireframe s přehledem zápisů studenta



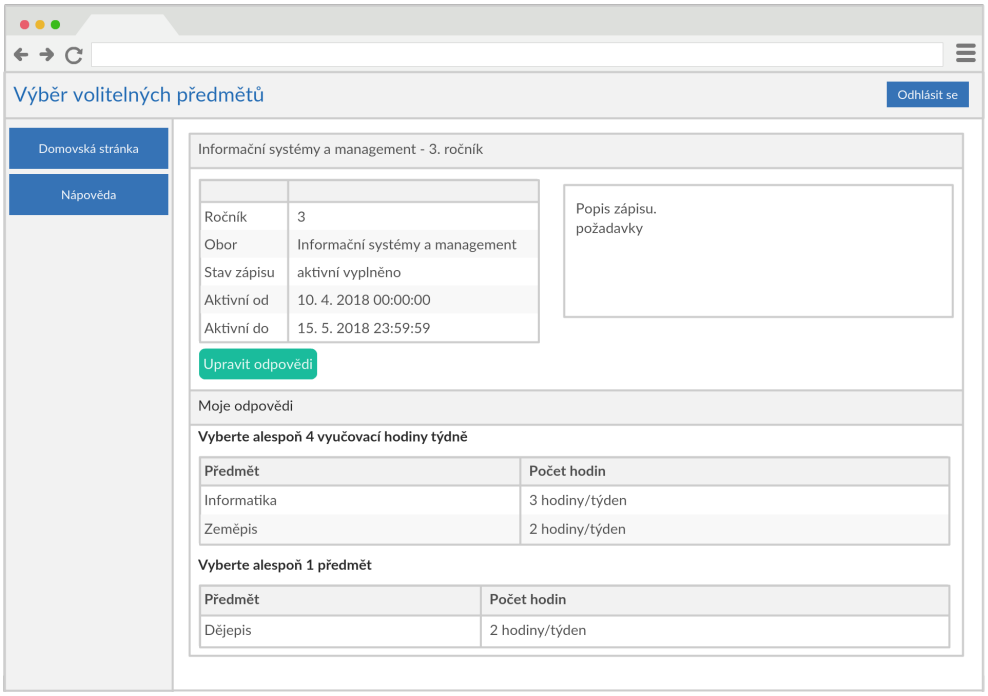
Obrázek 2.12: Wireframe s formulářem pro výběr volitelných předmětů

2.6.3 Vyplnění formuláře zápisu – výběr předmětů

Stěžejní součást studentské části je obrazovka 2.12, která slouží k vybrání vypsaných volitelných předmětů studentem. Hlavní sekce je rozdělena na popis zápisu, sekci s předměty, které jsou zapsané pro všechny povinně, pokud je v zápisu definovaná, a dále následují tabulky pro každou skupinu předmětů. Dle funkčních požadavků jsou skupiny dvojího typu: výběr určitého minimálního počtu předmětů a výběr určitého minimálního počtu hodin týdně. Typy skupin reflektuje způsob validace formuláře.

Každá skupina je reprezentována tabulkou, která obsahuje názvy předmětů včetně hodinové dotace a dále dva sloupce. Druhý sloupec obsahuje checkboxy pro označení předmětu na daném řádku jako primární volbu studenta, třetí sloupec pak checkboxy pro označení sekundární volby. V každém řádku tabulky, neboli pro každý předmět, může být označena pouze volba primární nebo volba náhradní, nikoli obě najednou.

Po stisknutí tlačítka pro uložení odpovědí se provede validace a pokud studentova volba nesplňuje požadavky na počty hodin nebo předmětů ve skupině, zobrazí se u dané skupiny chybová hláška.



Obrázek 2.13: Wireframe s detailním zobrazením zápisů a odpověďmi studenta

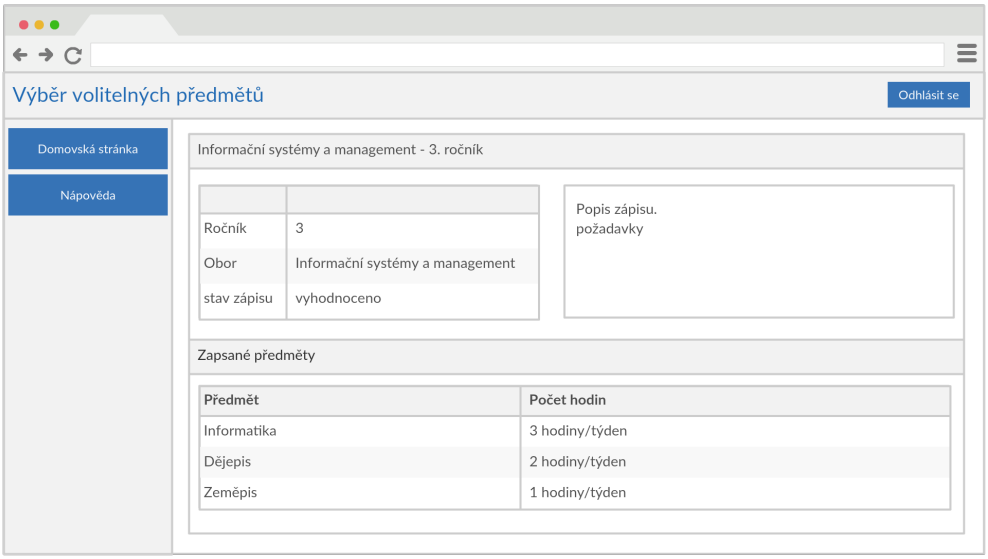
2.6.4 Detail zápisu a odpovědi studenta

Obrazovka 2.13 obsahuje detailní informace o zvoleném zápisu volitelných předmětů. Dále, pokud student již odeslal formulář 2.12, se pod tabulkou s detaily zápisu zobrazí tlačítko pro možnou úpravu odpovědí, v opačném případě se zobrazí tlačítko pro vyplnění formuláře 2.12 s výběrem volitelných předmětů. Ve spodní části jsou zobrazeny dvě tabulky, první obsahuje seznam předmětů, které si student zvolil jako primární, druhá pak předměty zvolené jako náhradní. V případě, že student dosud volbu neprovedl, je tato skutečnost zapsána v tabulce.

2.6.5 Detail zápisu s výsledky

Po vyhodnocení zápisu administrátorem se studentům zpřístupní obrazovka 2.14, na níž jsou kromě detailů zápisu zobrazeny předměty, které po zásahu administrátora budou skutečně v následujícím školním roce vyučovány.

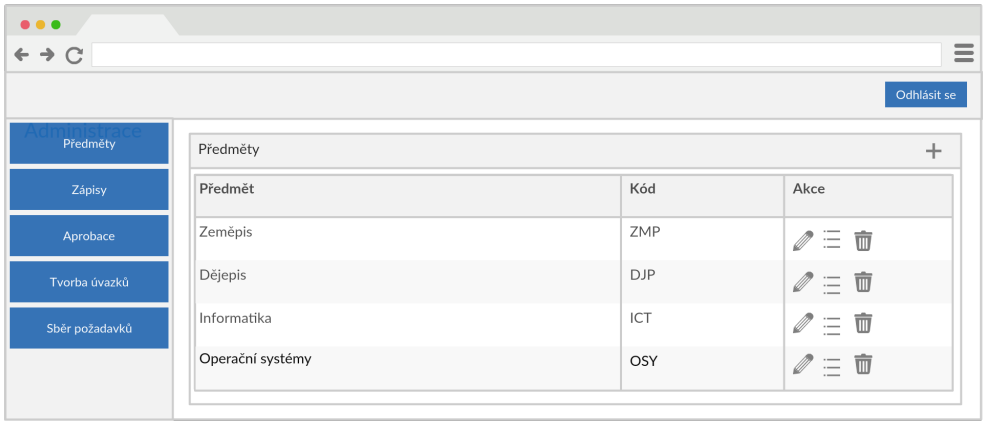
Předměty jsou vypsány v tabulce, kde každý předmět je doplněn o počet vyučovacích hodin týdně.



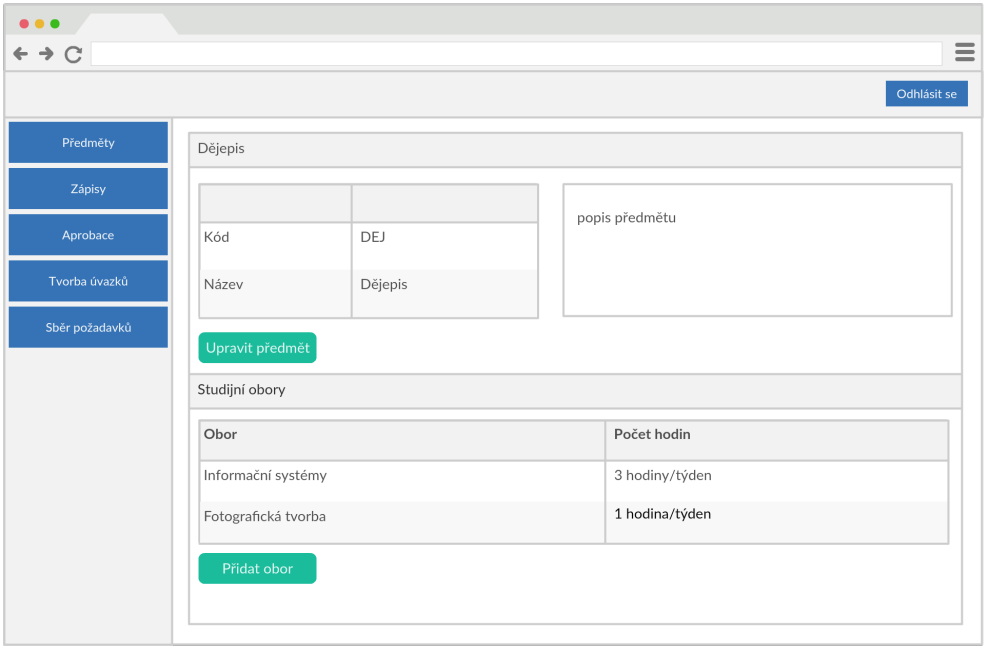
Obrázek 2.14: Wireframe s výsledky zápisu

2.6.6 Seznam předmětů

Tato obrazovka 2.15 slouží pouze jako rozcestník pro správu jednotlivých předmětů. Obsahuje proklik z názvu předmětu na jeho detail. Pro zjednodušení orientace lze předměty v tabulce řadit abecedně nebo podle kódu. Z důvodu usnadnění jsou v posledním sloupci tabulky pro každý předmět odkazy na příslušné akce - editace, detail a archivace předmětu.



Obrázek 2.15: Wireframe seznamu předmětů



Obrázek 2.16: Wireframe s výsledky zápisu

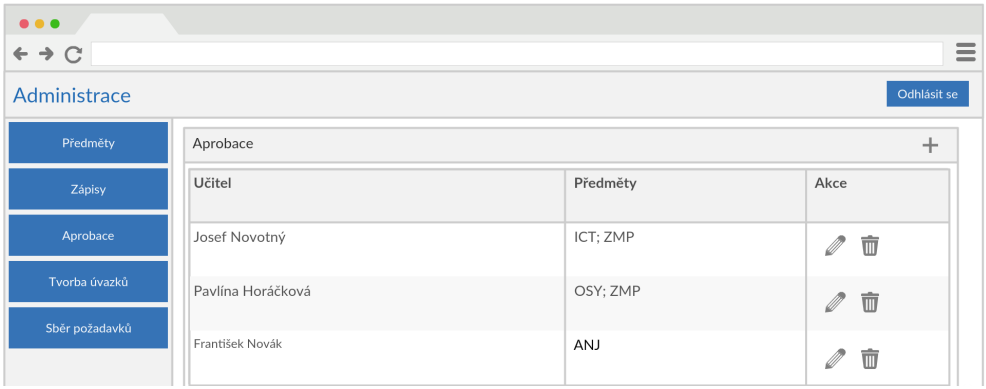
2.6.7 Detail předmětu

Obrazovka 2.16 slouží jednak jako rozcestník pro další nezbytné akce s předmětem, jednak pro zobrazení potřebných detailů o předmětu. V horní části se nachází box s popisem předmětu, který je dostupný také studentům, aby jim pomohl při rozhodování o výběru předmětu. V tabulce se nachází detaily o předmětu - název a kód. Pod tabulkou je umístěno tlačítko editace předmětu.

V dolní části se nachází tabulka se studijními obory, ve kterých se předmět vyučuje. Studijní obory lze přidat tlačítkem pod tabulkou, odebrat nebo editovat pomocí ikon v posledním sloupci tabulky.

2.6.8 Seznam aprobací

Hlavní část obrazovky 2.17 obsahuje tabulku se seznamem všech učitelů. Data v tabulce lze řadit abecedně podle jména učitele a vyhledávat podle předmětů, které učitel vyučuje. V druhém sloupci jsou kódy předmětů a v posledním odkazy ve formě ikon na akce pro archivování a úpravu aprobací učitele.

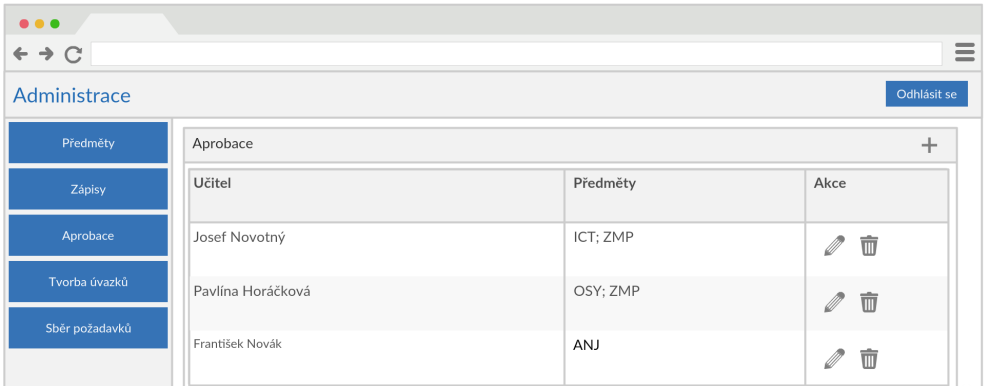


Obrázek 2.17: Wireframe se seznamem učitelů a jejich aprobací

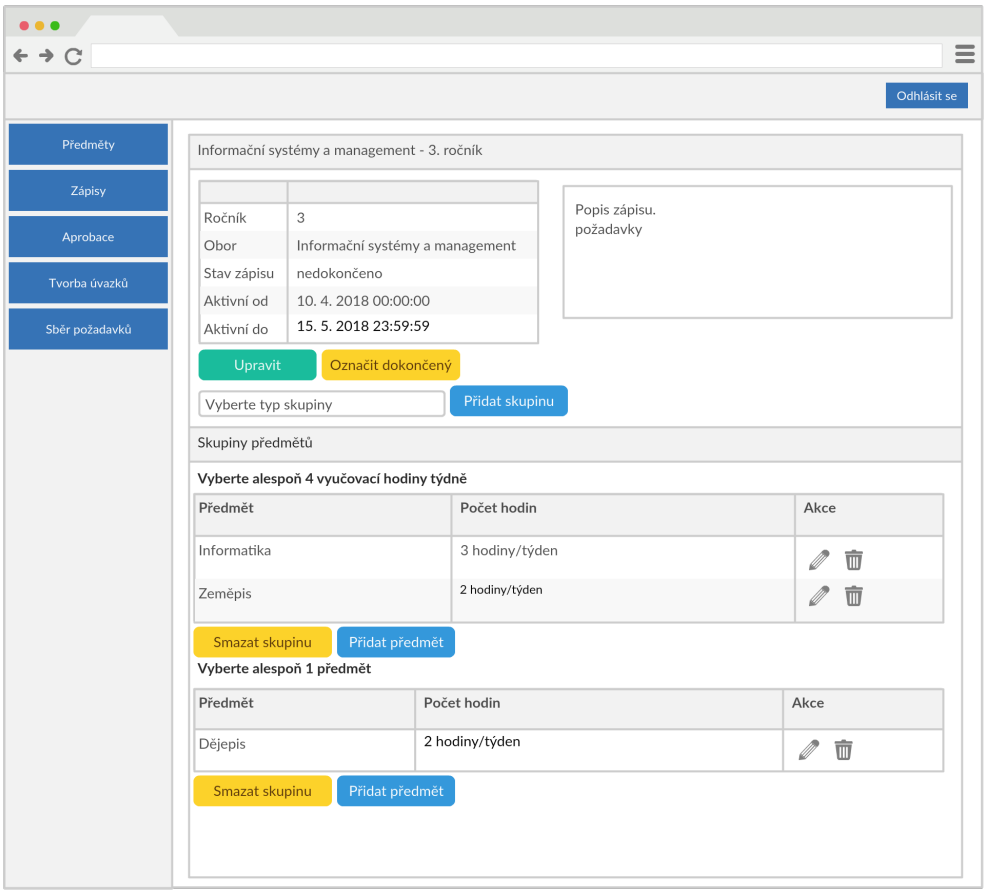
2.6.9 Seznam zápisů

Klíčovým rozcestníkem pro správu zápisů v modulu je obrazovka 2.18 seznamu zápisů. Jedná se o tabulku obsahující ve svých řádcích jednotlivé zápisy. Zápis je textově reprezentován jako název oboru, ročník a školní rok, pro který je zápis platný.

Po kliknutí na název zápisu v prvním sloupci dojde k přesměrování na obrazovku 2.19 s detailem zápisu. Dále seznam obsahuje školní rok, stav zápisu a sloupec s odkazy na akce detail, editace a archivace. V horní části záhlaví karty s názvem stránky je umístěn symbol *„plus“* pro vytvoření nového záznamu.



Obrázek 2.18: Wireframe se seznamem zápisů



Obrázek 2.19: Wireframe se detailem zápisu

2.6.10 Detail zápisu

Obrazovka 2.16 obsahuje všechny informace o daném zápisu a také možnost zápis upravovat. V horní části je zobrazen popis zápisu viditelný pro studenty, který obsahuje obecné informace a pokyny pro výběr předmětů. V tabulce nalevo je kromě položek ročník, obor a platnosti zápisů také položka „*stav zápisu*“, která indikuje ve které fázi se zápis aktuálně nachází a měla by být výrazně odlišena tak, aby byl stav na první pohled patrný.

Pod tabulkou se nachází tlačítko „*upravit*“ pro přesměrování na formulář editace základních údajů o zápisu a tlačítko pro změnu stavu zápisu, toto tlačítko přepíná mezi stavy dokončený/nedokončený. Níže je umístěn jednoduchý formulář pro výběr skupiny předmětů, která má být k zápisu přidána. Po jejím přidání se zobrazí jako samostatná tabulka v dolní části obrazovky.

V části „*Skupiny předmětů*“ jsou vypsané jednotlivé skupiny předmětů, které jsou vždy identifikovány nadpisem a typem skupiny. Pro přidání předmětu do skupiny slouží tlačítko pod tabulkou s předměty, vedle kterého se nachází také tlačítko pro smazání celé skupiny předmětů ze zápisu. Jednotlivé přidané předměty se zobrazují jako řádek v tabulce příslušné skupiny a je u nich také zobrazena hodinová dotace na týden a odkazy na smazání předmětu ze skupiny a editaci záznamu daného předmětu.

2.6.11 Formulář pro vyhodnocení zápisu

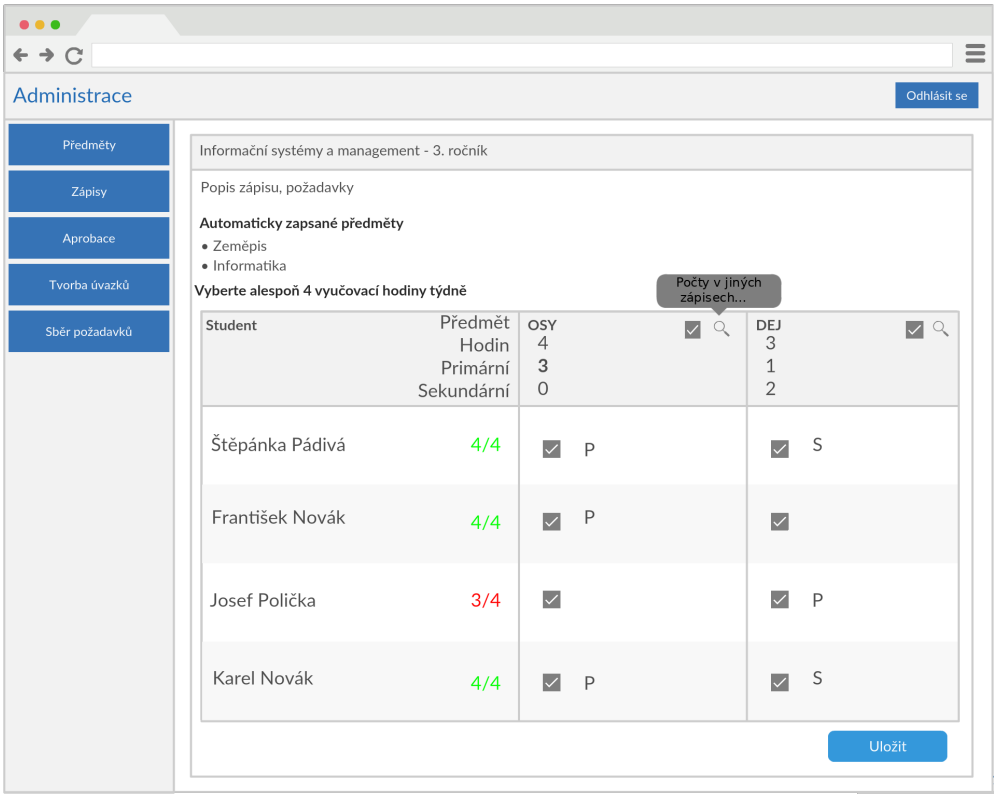
Dle funkčních požadavků jedna ze stěžejních částí systémů je samotné vyhodnocení voleb předmětů studentů. Návrh formuláře na obrazovce 2.20 zobrazuje jeden zápis a pro zjednodušení pouze jednu skupinu předmětů. V případě, že zápis bude mít více skupin předmětů, budou vypadat obdobně.

V horní části je zobrazen popis zápisu a požadavky na výběr předmětů, který je také dostupným studentům, aby měl administrátor, který zápis vyhodnocuje, přehled o vypsaných předmětech a minimálních požadavcích, které jsou kladeny na volby studentů. Pokud byla u zápisu definována skupina automaticky zapsaných předmětů, následuje jejich seznam.

Níže je pro každou skupinu předmětů umístěn formulář. Ten je jedinečný pro každou skupinu předmětů a má formu tabulky, jejíž řádky tvoří jednotliví studenti, kteří spadají pod daný zápis a to i takoví, kteří zápis nevyplnili. Sloupce tabulky odpovídají vypsaným předmětům. V záhlaví tabulky je u každého předmětu uveden počet hodin za týden, počet studentů, kteří mají aktuálně předmět zvolený jako primární – tedy bude jim zapsán a počet studentů, kteří jej v zápise zvolili jako sekundární. Po najetí kurzorem na symbol lupy vedle názvu předmětu se zobrazí okno s počty zapsaných studentů z jiných zápisů pro ten stejný předmět. Checkbox má funkci zrušení předmětu, tedy pokud administrátor checkbox zaškrtne, dojde ke zrušení výběru daného předmětu všem studentům a administrátor jim bude muset přiřadit náhradní předmět. Takto označený předmět nelze žádnému studentovi zapsat.

V buňce se jménem studenta je zobrazen počet hodin/předmětů, které si měl student zapsat, a počet, který má aktuálně zapsaný. Pro přehlednost by měl být výrazně označen student, který nesplňuje požadavky zápisu. Pokud je v dalších buňkách vždy na souřadnici student - předmět, zobrazeno písmeno P nebo S, značí to skutečnost, že se jedná o studentovu primární nebo sekundární volbu. Zaškrnutí checkboxu indikuje zapsání předmětu danému studentovi. Při prvním otevření formuláře administrátorem zaškrtnutý checkbox odpovídá tomu, že si student vybral předmět jako primární.

Studenti jsou v tabulce řazeni podle data odeslání odpovědi. Pokud počet studentů přesáhne kapacitu předmětu, budou takoví studenti v tabulce výrazně označeni tak, aby administrátor mohl na tuto skutečnost reagovat.



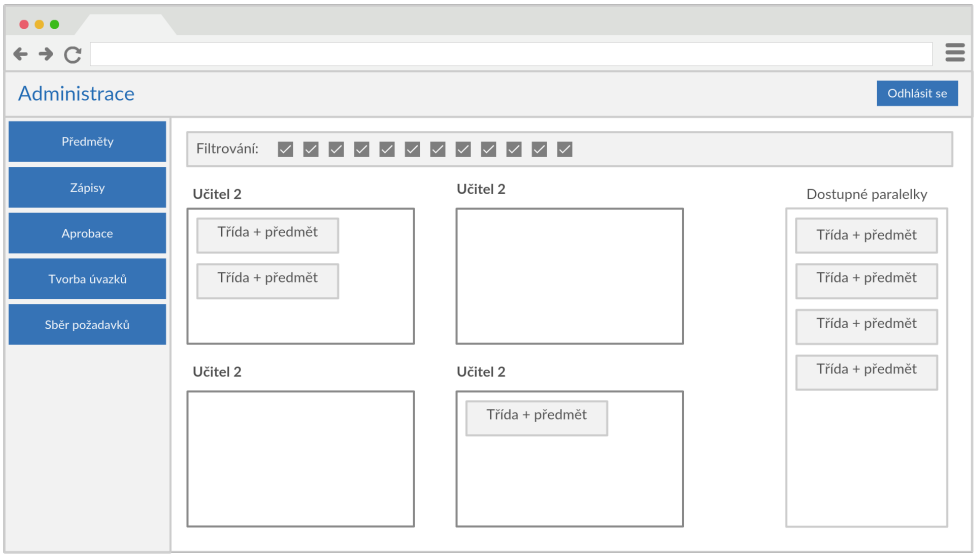
Obrázek 2.20: Wireframe s formulářem pro vyhodnocení zápisu

2.6.12 Formulář pro přiřazení úvazků

Druhým stěžejním formulářem modulu je 2.21, který umožňuje k jednotlivým učitelům přiřadit vypsané paralelky (trojice předmět – skupina – třída). V pravé straně hlavní časti obrazovky se nachází seznam s dosud nepřiřazenými paralelkami. V levé části pak seznam učitelů a ke každému učiteli jeden box, do kterého lze paralelky přesunovat ze seznamu nepřiřazených paralelek na pravé straně.

Během přesouvání by měl systém zobrazit, který z učitelů může daný předmět vyučovat. Po přiřazení paralelky k učiteli systém zobrazí aktuální a požadovaný úvazek učitele.

Jak mezi zobrazenými učiteli, tak mezi vypsanými paralelkami by měl systém umožnit administrátorovi filtrovat a zobrazit jen určité předměty nebo třídy a k nim relevantní učitele. Rozhraní pro filtrování se nachází v horní části stránky.



Obrázek 2.21: Wireframe s formulářem pro tvorbu úvazků

Kapitola 3

Implementace

Následující kapitola nejprve stručně popisuje použité technologie při implementaci a následně se zabývá již samotnou architekturou a implementací modulu do informačního systému. Jelikož se jedná o mnoho řádků kódu, budou níže popsány pouze stěžejní části aplikace s ukázkami kódu. Veškeré zdrojové kódy jsou dostupné na přiloženém CD [B].

3.1 Použité technologie

Z důvodu požadavku na jednotnou administraci modulu spolu s administrací informačního systému, byly pro implementaci modulu zvoleny stejné technologie, které jsou použity pro informační systém. Pro vývoj modulu byl tedy použit jazyk PHP a framework Symfony[15]. Toto rozhodnutí přináší maximální kompatibilitu a integraci modulu. Dále dle požadavků musí modul využívat sdílenou databázi s informačním systémem střední školy, proto byla současná databáze pouze rozšířena o potřebné tabulky a pro svou funkčnost využívá také tabulky informačního systému. Protože se nepředpokládá nasazení modulu mimo informační systém školy Michael, využívá přímo entity z informačního systému, což usnadňuje a zajišťuje koherenci výsledné aplikace.

V následující kapitole budou nejprve představeny všechny použité technologie a stručně popsány jejich základní charakteristiky. Autor se blíže zaměří na samotný framework Symfony, u kterého bude detailněji popsáná struktura a princip jeho fungování.

[15] Dostupné z: `https://symfony.com`

3.1.1 Framework Symfony

Následující podkapitola popisuje framework Symfony a to ve verzi 3.4.8, která byla použita v projektu.

Framework Symfony, dále jen Symfony, je dle [7] balík samostatných a znovupoužitelných nástrojů, které řeší nejčastější problémy při vývoji webových aplikací. Také se jedná o kompletní webový framework napsaný v jazyce PHP. Symfony nabízí velmi mnoho komponent, které lze použít samostatně nebo využít úzké integrace všech komponent a používat kompletní framework.

Samotný tvůrce frameworku Symfony jej nedefinuje jako MVC framework. Korektně je Symfony definován jako HTTP framework, který má za úkol zpracovat požadavek a vrátit požadovanou odpověď. [7] Protože je však MVC v dnešní době velmi často skloňované slovo a jednotlivé komponenty dle doporučení [8] mají rozdělovat aplikaci na tři pomyslné vrstvy, jedná se obecně o framework inspirovaný architekturou MVC.

Jak již bylo zmíněno výše, mezi hlavní výhody frameworku patří úzká provázanost jednotlivých komponent. O toto provázání se starají třída *AppKernel*, která registruje všechny komponenty a zajišťuje jejich načtení při startu aplikace. Mezi stěžejní komponenty patří zejména HttpKernel, který tvoří jádro celého Symfony frameworku a zajišťuje přijetí, vyhodnocení a vrácení adekvátní odpovědi na přijatý *http požadavek*.

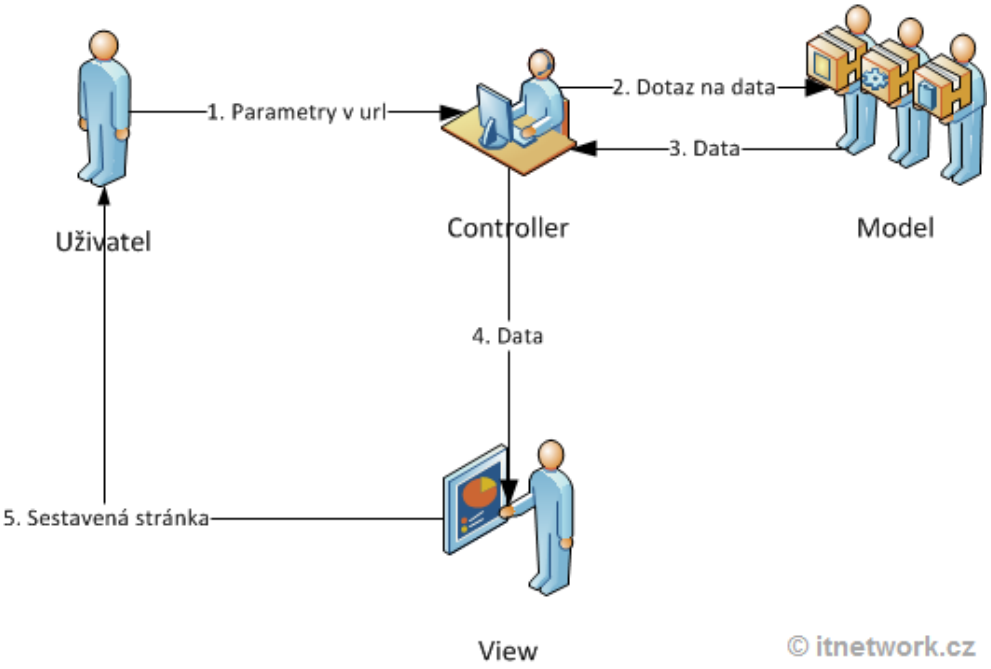
DependencyInjection slouží k propojení a vytvoření závislostí mezi jednotlivými službami aplikace. Třída HttpFoundation vytváří vrstvu nad samotným PHP a zajišťuje objektově orientovaný přístup k *http požadavkům* a odpovědím. ClassLoader, který poskytuje aplikaci autoloader implementující standarty PSR-4[16] a v neposlední řadě samozřejmě také třídy zajištující routování a vykreslování šablon.

Do Symfony je v základní instalaci integrován ORM Doctrine 2, který není nutné používat, jak již bylo zmíněno. Každý vývojář si může zvolit jen určitou část framewroku, ale v mnoha případech tato integrace dokáže ušetřit práci.

3.1.1.1 Architektura MVC

Architektura aplikace vychází z požadavku na rozšíření informačního systému napsaného ve frameworku Symfony. Jedná se o architektonický vzor aplikace, který rozděluje celou aplikaci na tří zásadní části. Tedy Model, který obsahuje business logiku aplikace a pod kterou se zpravidla ještě nachází databáze uchovávající data. Controller, který reaguje na požadavky uživatele, dotazuje se a spouští funkce Modelu a výsledky prezentuje uživateli prostřednictvím prezentační vrsty – View. Cílem této architektury je rozdělit aplikaci tak, aby kód rozdělený do částí byl přehledný a znovupoužitelný. [9]

[16] specifikace pro autoloading tříd z adresářů. Dostupné z: `https://www.php-fig.org/psr/psr-4/`.



Obrázek 3.1: Diagram struktury architektury MVC [9]

3.1.1.2 Konzole

Pro běžné operace při práci se Symfony lze využít připravenou konzoli, která je dostupná z `bin/console` a poskytuje vývojáři základní operace pro práci s cache a debugovací nástroje.

Debugovací nástroje v konzoli umožňují například vypsat veškéré dostupné třídy, nastavení jednotlivých balíčků nebo vypsat aktuální URI k jednotlivým akcím kontrolerů.

Využití Doctrine konzole umožňuje pracovat s vlastní databázovou vrstvou, generovat Entity, vytvářet a importovat data do databáze, kontrolovat, zda schéma databáze koresponduje s modelovým schématem aplikace. [10]

V neposlední řadě Symfony konzole umožňuje implementaci vlastních příkazů a je tak možné vytvářet plně konzolové aplikace v jazyce PHP. [11]

3.1.1.3 Bundly

Veškeré komponenty frameworku Symfony jsou rozděleny do tzv. bundlů. Jedná se o adresář obsahující zdrojové kódy jednotlivých komponent. Dle [8] by měly zachovávat strukturu MVC. Jednotlivé bundly obsahují vlastní konfigurační soubory, pohledy nebo testy.

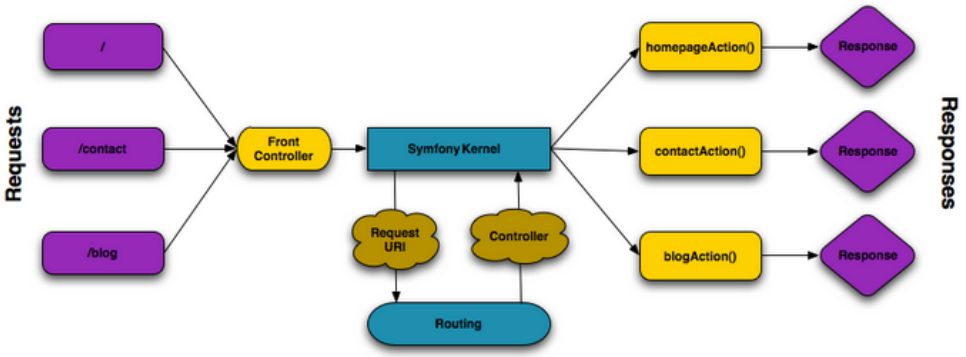
Celý framewrok Symfony se skládá z bundlů, jako je například FrameworkBundle, SecurityBundle nebo DebugBundle. [12] Jednotlivé bundly jsou do aplikace zaregistrovány v souboru `app\AppKernel.php` a mohou být instalovány například pomocí nástroje Composer.

3.1.1.4 Zpracování a směrování požadavku

Hlavním cílem frameworku Symfony je přijmout, korektně zpracovat a vrátit odpověď na *http požadavek*. [7] Samotný požadavek odesílá klientský počítač na server, kde běží Symfony aplikace. V základním nastavení je požadavek předám souboru `web/app.php`, neboli front-controlleru, ve kterém se vytvoří instance kernelu Symfony a následně je jí předán požadavek.

Symfony nabízí dvě základní třídy pracující HTTP komponentami. První z nich je třída *Symfony Request Object*, která zaobaluje požadavek a práci s ním zpřístupňuje pomocí metod. Tato třída pro Symfony poskytuje veškerou infrastrukturu potřebnou pro práci s požadavky. Druhou třídou je *Symfony Response Object*, což je třída, která poskytuje metody pro práci s *http odpovědí* serveru. Jedná se o základní funkcionalitu framewroku Symfony.

Jak již bylo řečeno výše, kernel aplikace obdrží požadavek a na základě konfigurace a požadované URI klientem rozhodne, kterou konkrétní akci kontroleru zavolá. Kontroler provede požadované operace, a protože vlastní kontrolery aplikace rozšiřují třídu *Controller*, musejí dle deklarace vracet objekt typu *Symfony Response Object*. Nejčastěji se jedná o odpověď s vyrenderovanou html stránkou, ale může se samozřejmě jednat také o JSON, přesměrování nebo binární soubor. Kernel tento objekt zpracuje a výstupem je klasická *http odpověď* odeslaná zpět klientovi.



Obrázek 3.2: Diagram zpracování požadavku ve framewroku Symfony[13]

Pro směrování požadavku na základě požadované URI využívá Symfony componentu *Routing*, která je volaná z *kernelu* a skládá se ze třech hlavních tříd. Třída *RouteCollection*, která v sobě uchovává instance tříd *Route* vytvořených podle konfigurace aplikace. Třída *RequestContent*, která drží informace o požadavku a třída *UrlMatcher*, která zajišťuje zavolání správné akce kontroleru pomocí předpisu cesty definované v aplikaci. [14]

3.1.1.5 Adresářová struktura

Adresářova struktura základní Symfony aplikace obsahuje složku `app`, ve které se nacházejí soubory `AppCache.php` obsahující implementaci zjednodušené *reverse proxy* [17] cache pro cachování *http požadavků* a `AppKernel.php` zajišťující nahrání komponent aplikace. Dále se zde nacházejí důležité soubory pro konfiguraci celé aplikace, ať již jde o konfiguraci připojení k databázi, konfiguraci zabezpečení nebo routování.

Dalším stěžejním adresářem je `src`, kde se nacházejí složky s jednotlivými bundly. V případě této práce je to složka `TimejobBundle` a dva další bundly obsahující implementaci samotného informačního systému školy – `AdminBundle` a `DefaultBundle`.

Další adresáře struktury souborů aplikace Symfony jsou popsány na obrázku 3.3. Jedná se pouze o ilustraci, v mnoha projektech se struktura může dle požadavků a použitých komponent lišit.

```
/
├── app
│   ├── config ..................... konfigurační soubory
│   ├── Resources ..................... pohledy, překlady
│   ├── AppCache.php
│   └── AppKernel.php
├── bin ..................... příkazy
│   └── console ..................... symfony konzole
├── src ..................... zdrojové soubory, vlastní bundly
├── var ..................... generované soubory
│   ├── cache
│   ├── logs
│   └── sessions
├── vendor ..................... obsahuje bundly třetích stran
├── web ..................... výchozí složka pro soubory dostupné z webu
│   └── app.php ..................... třída přímající http požadavky
└── composer.json ..................... soubor závislostí pro composer
```

Obrázek 3.3: Ilustrační adresářová struktura Symfony aplikace

[17] provádí cachování odpovědí z aplikace a vrací je klientovi [15]

3.1.2 HTML

Dle [16] HTML je tzv. markup (značkovací) jazyk, který slouží k určení logické struktury webové stránky. Správná a validní HTML struktura je důležitá pro přístupnost webu, správné zobrazení na všech zařízeních a také kvůli optimalizaci webu pro vyhledávače. Validní stromová struktura výsledné stránky je důležitá pro správné přistupování k jednotlivým elementům pomocí CSS, JavaScriptu a dalších jazyků. Pomocí HTML lze do stránky vložit například nadpisy, tabulky, obrázky nebo formuláře. Dle [17] aktuální verze HTML 5 je již podporovaná všemi moderními prohlížeči. Veškeré vlastnosti HTML jsou dostupné v oficiální specifikaci standardů jazyka HTML.[18]

3.1.3 CSS

Cascading Style Sheets neboli ve volném překladu kaskádové styly je tzv. stylovací jazyk, který popisuje zobrazení HTML prvků na webové stránce. [18] Samotné výrazy jazyka CSS lze zapisovat přímo do zdrojového souboru webové stránky nebo do zvláštního souboru. Elementy webové stránky se vybírají pomocí selektorů a na tyto elementy se poté aplikují vlastnosti deklarované v zápise CSS. Samotné vykreslení a aplikování stylů probíhá na straně klienta a záleží tak na prohlížeči, jak daný element vykreslí. Nejnovější verze CSS 3, kterou již z velké části podporují všechny moderní webové prohlížeče, [19] přináší rozšíření např. o animace a transformace prvků, rozšířené selectory, přechody pozadí, stíny či průhlednost. [20] Oficiální dokumentace je dostupná z webu konsorcia W3C.[19]

3.1.4 PHP

PHP je hojně využívaný server-side skriptovací jazyk, který se používá při tvorbě webových aplikací. Byl vytvořen v roce 1995 jako jazyk pro tvorbu webů. Na serveru běží často v podobě modulu a je zpracováván PHP interpretrem. [21] Jedná se o dynamicky typovaný, objektově orientovaný jazyk, který již v základu obsahuje mnoho funkcí, což také pozitivně přispělo k jeho rozšíření. [22] V současné době dle [23] podíl PHP na poli webových aplikací dosahuje 83,4 %. Poslední dostupná stabilní verze je 7.2.5[20], nejčastěji používané verze jsou 5.6+. PHP je šířen pod svobodnou licencí. Další informace včetně dokumentace jsou dostupné na `http://php.net`.

[18] Dostupné z: `https://www.w3.org/TR/html5/`.
[19] Dostupné z: `https://www.w3.org/Style/CSS/`.
[20] Ke dni 30. 4. 2018

3.1.5 JavaScript

JavaScript je objektově orientovaný, dynamicky typovaný, funkcionální programovací jazyk, který se typicky interpretuje na straně klienta ve webovém prohlížeči. [18] Ale existují i implementace JavaScriptu běžící na straně serveru, které se v poslední době dostaly mezi trendy ve vývoji webových aplikací. [24] Na webových stránkách se používá pro vytvoření interaktivních a dynamických prvků a obsluhu elementů v reálném čase např. různé efekty, animace, validace formulářů před odesláním.

3.1.6 jQuery

jQuery je velmi rozšířená JavaScriptová knihovna, která nabízí funkce pro zjednodušení práce s HTML elementy. Umožňuje odchytávání událostí a staví na principu odděleného JavaScriptu od samotného HTML kódu. Nabízí funkce pro AJAX a práci s atributy a styly elementů stránky. jQuery používá v současnosti přes 70 % webových vývojářů a jedná se tak jednoznačně o nejrozšířenější knihovnu pro JavaScript. [25] Je vydávána pod svobodnou licencí MIT[21].

3.1.7 Bootstrap

Twitter Bootstrap aktuálně ve verzi 4.1.0[22] je dle [26] hojně rozšířený a současně nejpoužívanější CSS framework. Jedná se o sadu šablon a stylů využitelných při tvorbě webu. Původně vznikl ve firmě Twitter[23], kde měl za úkol sjednotit vzhled a rozložení interních nástrojů firmy, ale velmi rychle se rozšířil, k čemuž přispěla i jeho svobodná licence MIT.[24] K CSS se později přidala i JavaScriptová knihovna rozšiřující funkce frameworku, který se dnes používá pro jeho jednoduchost a dobrou dokumentaci. Umožňuje vývojářům snadno tvořit responzivní design a používat již předpřipravené elementy. Kompletní dokumentace je dostupná na `www.getbootstrap.com`.

3.1.8 Twig

Twig je šablonovací systém navržený pro použití s jazykem PHP a nejčastěji spolu s frameworkem Symfony, jehož je součástí. [27] Software je šířen pod licencí BSD[25]. Kompletní dokumentace je dostupná na `twig.symfony.com`.

[21] Licence dostupná z: `https://jquery.org/license/`
[22] Ke dni 24. 4. 2018
[23] Dostupné z: `https://twitter.com/`
[24] Licence dostupná z: `https://github.com/twbs/bootstrap/blob/v4.0.0/LICENSE`
[25] Dostupné z: `https://opensource.org/`

3.1.9 Doctrine 2

Doctrine 2 je webový ORM framework, který zajišťuje mapování objektů (entit) na relační databázi. Umožňuje pracovat jak s entitami samotnými, tak vytvářet mezi nimi vazby, pracovat s kolekcí entit. Pro složitější dotazy probíhá dotazování v jazyce DQL, který však nepoužívá názvy tabulek a sloupců v databázi, ale příslušné proměnné a názvy entit. [28] Požadované vlastnosti entit lze nastavit pomocí anotací, YML nebo XML souborů. Doctrine je také integrovaný v základni verzi frameworku Symfony, což také přispívá k jeho rozšíření. V kontextu aplikace Symfony se využívá služeb třídy *EntityManager*, která implementuje rozhraní pro práci s jednotlivými repozitáři. Kompletní dokumentace je dostupná na `www.doctrine-project.org`.

3.1.10 MySQL

MySQL je multiplatformní relační databáze, která je dostupná pod bezplatnou licencí GPL[26]. Využívá jazyka SQL a jedná se o jeden z nejpoužívanějších databázových systémů. [29] Své oblibě se těší především díky jednoduchosti používání, rychlosti a spolehlivosti. Databáze je snadno rozšiřitelná a knihovny pro práci s ní jsou zabudovány například i v jazyce PHP. Kompletní dokumentace je dostupná na `www.mysql.com`.

3.1.11 Composer

Composer je nástroj pro správu závislostí v PHP. Pomocí souboru ve formátu JSON nebo příkazové řádky umožňuje instalovat jednotlivé knihovny pro jazyk PHP. Composer vytvoří strom závislostí a knihovny dle konfigurace nainstaluje. Composer také zajišťuje autoloading[27] PHP tříd. [30] Kompletní dokumentace je dostupná z `www.getcomposer.org`.

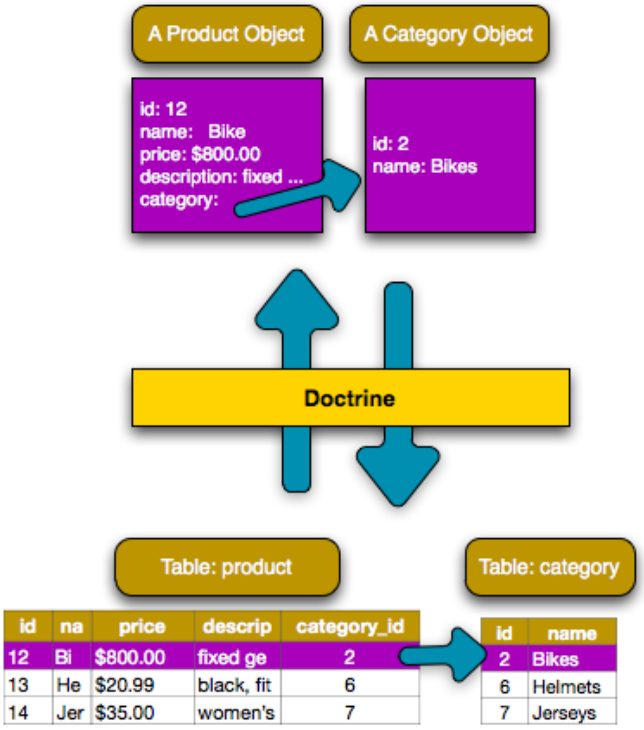
[26] Dostupné z: `https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html`.
[27] automatické načítání

3.1.12 Použité bundly

Pro rozšíření funkcionalit frameworku Symfony je často užíváno knihoven třetích stran – bundlů. K implemetaci modulu byly kromě bundlů dodávaných ve standardní Symfony instalaci a bundlu informačního systému použity:

PhpSpreadsheet

Jedná se o bundle, který poskytuje třídy pro práci se soubory pro tabulkové procesory typu Microsoft Excel a podobné. Umožňuje vytvářet soubory a jednotlivým listům v soboru přesně definovat obsah určitých buňek. V aplikaci je PhpSpreadsheet[28] bundle využit pro vytváření a export výpisů dat ve formátu *.xls*.



Obrázek 3.4: Diagram mapování entit a fungování Doctrine [31]

[28] Dostupné z: `https://github.com/PHPOffice/PhpSpreadsheet`.

3.2 Architektura aplikace

Vzhledem k požadavku na praktickou část této práce, kterým je vytvoření modulu pro informační systém střední školy, bude modul implementován jako bundle pro Symfony aplikaci informačního systému. Z požadavku na modul pro framework Symfony vyplývá také požadovaná architektura modulu – MVC. [8]

V následující podkapitole budou popsány veškeré důležité části aplikace doplněné o ukázky kódu, které budou z důvodu zachování přehlednosti zkráceny a upraveny.

3.2.1 Entity

Ve frameworku Symfony se pro reprezentaci tabulek používají zvláštní třídy – entity. [32] Jejich mapování na data uložená v databázi standardně obstarává ORM framework Doctrine. Tyto entity jsou zpravidla nezávislé na ostatních částech aplikace, které pouze využívají vlastností entit. Entity tak vytvářejí datový model celé aplikace včetně reprezentace vazeb mezi entitami a validačních pravidel. Na obrázku 3.4 je znázorněno mapování entit na tabulku v databázi včetně vazby mezi produktem a kategorií.

Entitní třídy lze generovat pomocí konzolové aplikace nebo vytvořit přímo soubor s danou třídou. Výsledkem je PHP soubor obsahující třídu, která je opatřena anotacemi tak, aby framework Doctrine mohl vygenerovat databázové schéma. Dále je pomocí anotace k třídě přiřazena pomocná třída `repository`, která rozšiřuje standardní třídu frameworku Symfony a může obsahovat pokročilé dotazy k získávání dat z databáze. [32]

Na ukázce kódu 3.1 níže lze vidět anotaci definující název příslušné tabulky v databázi a přiřazení třídy *EnrollmentRepository*. Pomocí dalších anotací lze kromě validací a vztahů mezi entitami definovat i další vlastnosti daných proměnných, respektive sloupců v databázi. Například anotace `@OrderBy({"updatedAt" = "ASC"})` u proměnné `$responses` definuje řazení záznamů při výpisu z databáze.

```php
/**
* @ORM\Table(name="enrollment")
* @ORM\Entiy(repositoryClass="EnrollmentRepository")
*/
class Enrollment {
    /**
    * @ORM\Column(name="id", type="integer")
    * @ORM\Id
    * @ORM\GeneratedValue(strategy="AUTO")
    */
    private $id;

    /**
    * @OneToMany(
    *    targetEntity="EnrollmentResponse",
    *    mappedBy="enrollment_id")
    * @OrderBy({"updatedAt" = "ASC"})
    */
    private $responses;
```

Ukázka kódu 3.1: Příklad entitní třídy a použití anotací

3.2.2 Validace dat

Kontrola vstupních dat, která může být dále použita například pro validaci ve formulářových polích, je obvykle definována u členských proměnných jednotlivých tříd reprezentující entity. [33] Data musí být validována při odeslání formuláře, ale také před zapsáním do databáze. Základní validační pravidla se zapisují pomocí direktivy `\Assert()` a příslušných pravidel. Validace je ve frameworku Symfony možno realizovat mnoha způsoby, pro ty složitější lze využít vlastní metodu v dané třídě a pomocí anotace kontrolovat její návratovou hodnotu a případně zobrazit chybu.

V následující ukázce kódu 3.2 je použita jednak validace pomocí regulárního výrazu, jednak pravidlo požadující vyplnění pole a definující rozsah vloženého čísla. Každému validačnímu pravidlu lze také přiřadit chybovou zprávu, která bude vrácena v případě nevalidního vstupu.

```
1  class Enrollment {
2      /**
3       * @Assert\NotBlank()
4       * @Assert\Range(
5       *  min = 1,
6       *  max = 4)
7       * @ORM\Column(name="grade", type="integer")
8       */
9      private $grade;
10
11     /**
12      * @Assert\Regex(
13      *  pattern = "/\d{4}\/\d{2}/",
14      *  message="Format ex. 2015/16")
15      * @ORM\Column(
16      *  type="string",
17      *  length=10000,
18      *  nullable=false)
19      */
20     private $schoolYear;
```

Ukázka kódu 3.2: Příklad použití validačních pravidel

3.2.3 Asociace

Vazby mezi entitami se v Symfony, díky použití Doctrine, zapisují pomocí anotací nebo ve zvláštním YML či XML souboru. V této práci je použit a bude na příkladech předveden zápis pomocí anotace v PHP souboru s danou třídou. [34]

ManyToOne

Je to velmi často používaná vazba. V kontextu vytvořeného modulu je použita například u vazby mezi vypsanou paralelkou (tedy entitě spojující předmět a třídu) a třídou. To znamená, že v rámci jedné třídy je vypsáno více paralelek. U vazby typu ManyToOne je vazba definována vždy pouze na vlastnící straně. [34]

```
1  class TimejobClassSubject {
2      /**
3      * @ManyToOne
4      *  (targetEntity="AdminBundle\Entity\SchoolClass")
5      * @JoinColumn
6      *  (name="class_id",
7      *  referencedColumnName="id")
8      */
9      private $class;
10 }
```

Ukázka kódu 3.3: Příklad použití vazby ManyToOne

OneToMany

Vazba musí být implemetována jako obousměrná, pokud není realizována pomocí vztahové tabulky. Pokud je realizována pomocí vztahové tabulky, implementace v Doctrine odpovídá vazbě ManyToMany bez určení směru (tzv. ManyToMany undirectional). Vlastnící strana u této vazby je vždy entita s kardinalitou „Many“. Doctrine vyžaduje použití anotací `inversedBy` na vlastnící straně a `mappedBy` na druhé straně vztahu.[34]

V této práci se vazba vyskytuje například mezi předmětem a jeho skupinami (začátečníci, pokročilí), kdy jeden předmět je rozdělen na více skupin, ale každá skupina patří vždy k jednomu předmětu.

```
class Subject {
    /**
     * @var SubjectGroup[]
     * @ORM\OneToMany
     *   (targetEntity="SubjectGroup",
     *    mappedBy="subject",
     *    cascade={"persist","remove"})
     */
    private $subjectGroups;
}

class SubjectGroup {
    /**
     * @var Subject
     * @ORM\ManyToOne
     *   (targetEntity="Subject",
     *   inversedBy="subjectGroup")
     * @ORM\JoinColumn
     *   (name="subject_id",
     *   referencedColumnName="id")
     */
    private $subject;
}
```

Ukázka kódu 3.4: Příklad použití vazby OneToMany

ManyToMany

Vazba ManyToMany bez určení směru je implementována pomocí vztahové tabulky s anotací a uchování klíče na záznam v tabulce na vlastnící straně. V případě obousměrné vazby nemusí být implicitně definována vztahová tabulka a v podstaně dojde k dekompozici vztahu na dvě jednodušší vazby. [34]

Příklad použití vazby ManyToMany bez určení směru nalezneme v práci mezi entitou reprezentující aprobaci učitele a předmět. Ke konkrétnímu učiteli je přiřazeno více předmětů a předmět může vyučovat více učitelů. V tomto

konkrétním případě je, dle požadavků na funkčnost systému, definice vazby i na druhé straně redundantní.

```
class Approbation {
    /**
    * @ManyToMany(targetEntity="Subject")
    * @JoinTable
    *  (name="approbation_subjects",
    *   joinColumns={@JoinColumn
    *       (name="approbation_id",
    *        referencedColumnName="id")},
    *   inverseJoinColumns={@JoinColumn
    *       (name="subject_id",
    *       referencedColumnName="id")})
    */
    private $subjects;
```

Ukázka kódu 3.5: Příklad použití vazby ManyToMany

OneToOne

Poslední typ vazby OneToOne implementovaný ve frameworku Doctrine, který přiřazuje jedné entitě právě jednu další, nebyl v práci využit.

3.2.4 Controllery

Kontroler je třída, která v Symfony frameworku obsahuje *akce* pro zpracování požadavku a vrácení požadované odpovědi. Framework vybere metodu, která se má volat podle cesty nejčastěji uvedené v anotaci. V Symfony frameworku musejí metody kontroleru vracet objekt třídy *Response*, což může být HTML stránka, soubor ke stažení, přesměrování nebo chyba.

Akce kontroleru by dle [35] měly být krátké, přehledné a neobsahovat logiku programu, která má být oddělena. Proto byl v práci využit princip „Operation, functionality, repository“, který odděluje logiku aplikace od kontroleru a je popsán v následující podkapitole.

Kontroler v následující ukázce 3.6 přijímá požadavky z URI, v tomto případě začínající na */admin/requirements* a volá příslušnou akci. V případě, že není uživatel přihlášen nebo nemá přidělenou roli *role_user*, jak je definováno v anotaci kontroleru, ten vrátí výjimku a zamezí takovému uživateli provést akci. Akce *indexAction* v ukázce nepřijmá žádné parametry a je volána při zadání cesty popsané výše bez dalšího suffixu. Akce vrací pohled s parametrem `requirements`.

```
/**
 * @Security("has_role('role_user')")
 * @Route("/admin/requirements")
 */
class RequirementController extends BaseController {
    /**
     * @Route("/", name="requirements")
     */
    public function indexAction(Request $request) {
        $requirements = $this->em
                        ->getRepository(Requirement::class)
                        ->findAll();

        return $this->render
        ('@Timejob/Default/requirement/index.html.twig', [
            'requirements' => $requirements
        ]);
}
```

Ukázka kódu 3.6: Příklad použití kontroleru

3.2.5 Views

Jak již bylo zmíněno v předchozí kapitole, kontrolery zpracují požadavek a nejčastěji vracejí právě HTML stránku. Ve frameworku Symfony je pro tyto účely již v základní instalaci integrován šablonovací systém Twig. Funkcí pohledů je předat zákazníkovi výstupy aplikace, neměly by zbytečně obsahovat logiku a použití šablonovacího systému Twig přispívá k oddělení PHP kódu od samotných HTML stránek. [27]

Šablonovací systém nabízí mnoho funkcí a filtrů včetně možnosti vkládání šablon do ostatních, což zvyšuje přehlednost a znovupoužitelnost částí kódu. V této práci jsou šablony uloženy v adresáři `Resources/Views` pro zpřehlednění a oddělení šablon od zdrojových kódů aplikace. Zároveň integrace systému Twig do Symfony umožňuje vykreslovat objekty typu *Form* přímo pomocí šablony bez nutnosti explicitně definovat každý prvek formuláře.

3.2.6 Repository

Repository jsou podpůrné třídy pro vytváření složitějších dotazů, na které nepostačují standardní funkcionality poskytované ORM Doctrine. Jedná se o třídy rozšiřující třídu *EntityRepository*, která se vždy váže k jedné konkrétní entitě. Třídu repozitáře standardně volá *EntityManager* pomocí metody `getRepository()`.

V ukázce kódu 3.7 vidíme repozitář pro třídu *Enrollment*, jehož metoda `findAllActive()` provede dotaz v jazyce DQL, který vrátí všechny aktuálně aktivní zápisy předmětů. Čtenář si může povšimnout použití parametru `$now`

reprezentující aktuální časovou stopu. Metoda vrací pole, které je následně v kontroleru předáno do Twig šablony pomocí parametru.

```
class EnrollmentRepository extends EntityRepository {
    public function findAllActive(): array {
        $now = new \DateTime('now');
        $qb = $this
            ->createQueryBuilder('e')
            ->where(':now BETWEEN e.validFrom AND e.validTo')
            ->andWhere('e.isFinished = true')
            ->andWhere('e.isArchived = false')
            ->setParameter('now', $now)
            ->getQuery();
        return $qb->execute();
    }
```

Ukázka kódu 3.7: Příklad rozšíření třídy EntityRepository

3.2.7 Formuláře

Formuláře jsou jednou z nezbytných součástí všech webových aplikací a i na toto je framework Symfony připraven. Formuláře lze vytvořit a zpracovat klasickým způsobem, nebo rozšířit třídy integrované v Symfony. Jedná se především o třídu *formBuilder*, která poskytuje metody pro vytvoření polí a práci s formuláři. Formulář je vytvářen metodou `createForm()`. Výsledkem této metody je objekt typu *Form*, se kterým lze dále v kontextu Symfony aplikace pracovat.

Třídy formulářů jsou v projektu uloženy ve složce `/Forms` a v metodě `buildForm()` přidávají jednotlivá pole formulářům. Standardně jsou jim přiřazeny vlastnosti definované v anotacích jednotlivých proměnných v entitě včetně validačních pravidel. Tato pravidla jsou automaticky přidána jednak k samotných prvkům formuláře ve vygenerováném HTML kódu, jednak kontrolována na straně serveru.

Pro zpracování formuláře slouží metody pro zjištění stavu odeslání formuláře `isSubmitted()` a validaci formuláře `$form->isValid()`. Obě jsou volány po zpracování HTTP požadavku metodou `handleRequest()` objektu *Form*.

Následující ukázka kódu 3.8 obsahuje volání metody pro vytvoření formuláře včetně předání příslušného objektu pro naplnění formuláře daty, dále ukázku třídy reprezentující formulář a ukázku kontroly odeslání formuáře.

```
class SubjectController extends BaseController {
    //...
    public function editAction (
        Request $request, Subject $subject) {

        $form = $this->createForm(SubjectType::class,
                                  $subject);

        $form->handleRequest($request);
        if ($form->isSubmitted() && $form->isValid()) {
            //...
        }
        //...
    }
}

class SubjectType extends AbstractType {

    public function buildForm(
        FormBuilderInterface $builder, array $options) {

        $builder->add('title')
                ->add('code')
                ->add('description', TextareaType::class));
    }
```

Ukázka kódu 3.8: Příklad vytvoření a zpracování formuláře

3.2.8 Princip Repository, Functionality, Operation

Vzhledem ke struktuře informačního systému školy a použití frameworku Symfony, který svou strukturou vychází z architektury MVC, [7] byl pro rozdělení jednotlivých logických operací a přístup k datům zvolen tzv. princip *Repository, Functionality, Operation*, rozdělující strukturu projektu na tři pomyslné vstvy.

První a nejnižší vrstvou jsou třídy *repository*, které vždy náleží právě jedné entitě a základní implementaci metod v Doctrine lze rozšířit o další specializované a složitější. Tyto třídy rozšiřují třídu `EntityRepository`.

Druhou vstvu představují *functionality*. Do *functionality* patří třídy, které pracují s danou entitou a využívají metody tříd *repository*. *Functionality* by měly obsahovat metody pro základní operace s entitou a neměly by pracovat s dalšími třídami z vrstvy *functionality* ani *repository*.

Poslední a nejvyšší vrstvou jsou *operation*. Tyto třídy obsahují metody pro práci s více entitami a mohou pracovat i s dalšími třídami *operation* i *functionality*, neměly by však pracovat přímo s *repository*. Operation mohou provádět složitější operace, v případě této práce například zpracování formulářů nebo odesílání emailů.

3.2.9 Autorizace a autentizace

Přihlášení do aplikace pro učitele a administrátory není v tomto modulu řešeno, neboť je implementováno v informačním systému, do kterého je tento modul integrován. Nicméně stále musí modul provádět autorizaci k přístupu uživatelů k funkcím modulu. Pro tento účel má v informačním systému každý přihlášený uživatel přidělené role, které jsou na kontrolovány i v modulu této práce.

Kontrola probíhá na úrovni kontrolerů a jednotlivé role potřebné k přístupu k daným akcím, neboli jednotlivým URI, jsou definované v anotacích. 3.6 Samotný šablonovací systém Twig obsahuje funkce pro testování uživatelských rolí, které je vhodné použít například ke skrývání odkazů na časti aplikace, kam by uživatel neměl mít přístup.

V případě přihlášení žáků bylo potřeba vytvořit vlastní rozhraní pro přihlášení a autorizaci, protože standardní nástroje Symfony neumožňují přihlášení pouze pomocí přístupového kódu.

Pro tento účel byla vytvořena třída rozšiřující třídu *AbstractGuardAuthenticator* dostupnou ve frameworku Symfony. Třída obdrží HTTP požadavek, který zpracuje a získá kód, který student odeslal v přihlašovacím formuláři. Proběhne validace, kontrola platnosti a správnosti kódu a třída buďto vrátí instanci třídy *UserInterface*, nebo příslušnou chybovou hlášku. Díky použití třídy rozšiřující standardní Symfony autentikátor je možné dále v aplikaci používat všechny tradiční postupy pro řízení přístupu přihlášeného uživatele k akcím kotrolerů.

3.2.10 Adresářová struktura modulu

Adresářová strukturu modulu 3.5 koresponduje s většinou doporučení v [8] a konvencemi při vývoji ve frameworku Symfony. Adresářová struktura `Resources` za účelem zvýšení přehlednosti kopíruje strukturu téže složky použité v informačním systému školy.

```
src/TimejobBundle
├── Controller ..... kontrolery
├── Entity ..... entity
├── Form ..... třídy formuláře FormType
├── Functionality ..... třídy vrstvy Functionality
├── Operation ..... třídy vrstvy Operation
├── Repository ..... třídy vrstvy Repository
├── Resources
│   ├── config ..... konfigurační soubory
│   ├── translations ..... soubory s překlady
│   └── views ..... soubory šablon systému Twig
│       ├── auth
│       ├── components
│       ├── default
│       ├── emails
│       ├── base.html.twig ..... základní šablona administrace
│       └── base_student.html.twig ..... základní šablona zápisů
├── Security ..... třídy autentikátoru
└── TimejobBundle.php ..... třída rozšiřující základní Symfony Bundle
```

Obrázek 3.5: Použitá adresářová struktura

3.3 Integrace modulu do informačního systému školy

Během dokončování tohoto modulu byla již dokončena implementace prototypu informačního systému střední školy. Vzhledem k tomu, že modul je bundle pro framework Symfony, bylo by možné jej připravit tak, aby byl instalovám pomocí Composeru do aplikace informačního systému. Ale vzhledem k požadované úzké integraci a propojení funkcionalit do jednoho celku tak, aby běžný uživatel administrace nepoznal rozdíl mezi jádrem a tímto modulem, nebyla tato možnost zvolena. Struktura modulu jakožto bundlu pro Symfony byla zachována, bylo třeba ovšem modul úzce propojit a doplnit do informačního systému školy odkazy na funkce modulu.

3.3.1 Integrace modulu

Do složky `/src` je potřeba nahrát složku `TimejobBundle` se zdrojovými soubory modulu. V Symfony aplikaci informačního systému školy je třeba nejprve zaregistrovat modul jako bundle do kernelu aplikace v souboru `app/AppKernel.php` včetně používaného bundlu pro tvorbu *.xls* souborů.

```
public function registerBundles() {
    $bundles = [
        new TimejobBundle\TimejobBundle(),
        new Yectep\PhpSpreadsheetBundle\PhpSpreadsheetBundle()
        //...
        new AdminBundle\AdminBundle(),
        ];
    //...
```

Ukázka kódu 3.9: Zaregistrování modulu jako bundlu v AppKernelu

Dále je třeba zajistit autoloading potřebných souborů v konfiguračním souboru nástroje Composer v souboru `composer.json`.

```
    "autoload": {
        "psr-4": {
            "DefaultBundle\\" : "src/DefaultBundle",
            "AdminBundle\\" : "src/AdminBundle",
            "TimejobBundle\\" : "src/TimejobBundle"
        },
        "require": {
            "yectep/phpspreadsheet-bundle": "^0.0.4"
        }
```

Ukázka kódu 3.10: Zaregistrování modulu pro composer

Následně je třeba importovat konfigurační soubor modulu, který zajistí přidání následujícího řádku pod nastavení importu v `app\config\config.yml`.

```yaml
import:
- {resource: "@TimejobBundle/Resources/config/config.yml"}
```

Ukázka kódu 3.11: Import konfigurace bundlu

Následuje přidání konfigurace pro routování v souboru `app\config\routing.yml`.

```yaml
//...
timejob:
    resource: "@TimejobBundle/Controller/"
    type:     annotation
    prefix:   /
```

Ukázka kódu 3.12: Konfigurace routování

Konfigurace přihlašování studentů je třeba nastavit v souboru `app/config/security.yml`.

```yaml
//...
providers:
    //...
    code_provider:
    entity:
        class: AdminBundle:AccessCode
        property: username
    //...
firewalls:
    enrollment:
        anonymous: ~
        logout_on_user_change: true
        provider: code_provider
        logout:
            path: /enrollment/logout
            target: /zapis
        guard:
            authenticator: access_code_authenticator
        pattern: ^/(zapis|enrollment)
    //...
```

Ukázka kódu 3.13: Konfigurace přihlašování studentů

Posledním krokem je přidání překladů pro jednotlivé zprávy v systému. To lze jednoduše provést sloučením souborů `messages.*.yml` a `breadcrumb.*.yml` informačního systému a modulu ve složce `src/Resources/translations/`.

Po tomto kroku chybí pouze spustit příkaz `composer update` a pro aktualizaci databáze `bin/console doctrine:schema:update -f`.

Modul je nyní připraven k používání a plně integrován do informačního systému. V případě potřeby lze přidat odkazy na jednotlivé akce modulu kamkoliv do pohledů informačního systému. Tento postup byl zvolen především proto, že se nejedná o klasický bundle, který se bude běžně instalovat pomocí nástroje composer, ale o modul, který již bude natrvalo součástí informačního systému a proto je jeho co nejužší integrace se systémem žádoucí.

3.3.2 Nasazení

Prototyp informačního systému školy momentálně běží v testovacím prostředí v kontejneru volně dostupné verze služby `Heroku`[29], která podporuje velké množství nástrojů včetně integrace `GitHubu`[30], který byl intenzivně využíván při tvorbě tohoto modulu i prototypu informačního systému.

Samotné nasazení výsledného systému probíhá příkazem `git push heroku master` a následně jsou již automaticky spuštěny veškeré potřebné skripty pomocí buildovacího nástroje `yarn`[31]. První nasazení a samotná integrace projektu proběhla sloučením vývojové větve modulu na githubu s větví informačního systému. Pokud by z nějakého důvodu automatické nasazení selhalo, systém se sám vrátí do předchozího funkčního stavu.

[29] Dostupné z: `https://www.heroku.com`.
[30] Dostupné z: `https://www.github.com`
[31] Dostupné z: `https://www.yarn.com`.

3.4 Bezpečnost

Otázka bezpečnosti je u webových aplikací naprosto klíčová, neboť taková aplikace je dostupná doslova všem a proto je její napadení mnohem snažší než například jiné aplikace běžící na interní infrastruktuře firmy. Většina moderních frameworků pamatuje na zapezpečení a často je využití frameworku správným krokem právě vzhledem k již integrovaným bezpečnostním opatřením a ani Symfony není výjimkou.

Podle sdružení OWASP[32], které každoročně zveřejňuje statistiky nejčastějších bezpečnostních rizik, je nejčastějším bezpečnostním rizikem vložení kódu, často SQL příkazu, do požadavku a tím provedení změn bez autorizace uživatele.

OWASP Top 10 - 2013		OWASP Top 10 - 2017
A1 – Injection	→	A1:2017-Injection
A2 – Broken Authentication and Session Management	→	A2:2017-Broken Authentication
A3 – Cross-Site Scripting (XSS)	↘	A3:2017-Sensitive Data Exposure
A4 – Insecure Direct Object References [Merged+A7]	U	A4:2017-XML External Entities (XXE) [NEW]
A5 – Security Misconfiguration	↘	A5:2017-Broken Access Control [Merged]
A6 – Sensitive Data Exposure	↗	A6:2017-Security Misconfiguration
A7 – Missing Function Level Access Contr [Merged+A4]	U	A7:2017-Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	☒	A8:2017-Insecure Deserialization [NEW, Community]
A9 – Using Components with Known Vulnerabilities	→	A9:2017-Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	☒	A10:2017-Insufficient Logging&Monitoring [NEW,Comm.]

Obrázek 3.6: Žebříček TOP 10 dle OWASP pro roky 2013 a 2017 [36]

Dle žebříčků OWASP[33] 3.6 došlo oproti roku 2013 k vyřazení zranitelnosti vůči CSRF ze seznamu deseti nejvýznamějších hrozeb. Dle [36] je to především díky rozšíření frameworků při vývoji aplikací, které často ochranu proti CSRF obsahují. Zároveň došlo k významnému poklesu XSS, který ve značné míře způsobil dle [36] nástup frameworků.

V následující kapitole bude vybráno několik hrozeb z OWASP žebříčku a analyzováno, jakým způsobem je proti nim aplikace chráněna. Dále bude věnována samostatná podkapitola validaci vstupních dat z formulářů.

[32] Dostupné z: `https://www.owasp.org`.
[33] Dostupné z: `https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project`.

3.4.0.1 XSS

„XSS (Cross-site scripting) je metoda využívající bezpečnostní chyby webu, konkrétně nedostatečné ošetření dat. Typicky spočívá ve vložení škodlivého JS kódu.“ [37] Symfony díky použití Doctrine dle [38] vstupy uživatelů ošetřuje tak, aby nebylo možné vložené zdrojové kódy spouštět, proto je tedy modul proti XSS dostatečně chráněn. V modulu jsou sice implementovány i formuláře, které nezpracovává a nevykresluje standardní funkcionalita frameworku, takové formuláře jsou ale pouze v části pro administrátory nebo učitele a i takové vstupy jsou upraveny a ochráněny.

Samotný šablonovací systém Twig podporuje automatické ošetření proti XSS při výpisu dat v šabloně, které nahradí nebezpečné znaky a zamezí tak spuštění cizího kódu v prohlížeči uživatele. [39]

3.4.0.2 CSRF

„Cross-site Request Forgery (CSRF) je zranitelnost webových aplikací, která za určitých podmínek umožňuje autentizované uživatele donutit k tomu, aby provedli nezamýšlenou akci. Podstata zranitelnosti tkví v tom, že aplikace nedokáže rozlišit požadavky, které uživatel zavolal neúmyslně.“ [40]

I vůči této technice jsou formuláře v Symfony ochráněny již v základní instalaci frameworku. [39] Aplikace vygeneruje tzv. CSRF token, který je unikátní a náhodný. Platí vždy pro konkrétního uživatele a pro daný formulář. Takový token je pak uložen ve skrytém poli formuláře a při zpracování formuláře se kontroluje jeho správnost. Pro formuláře negenerované Symfony je poskytnuta dodatečná funkcionalita, která umožňuje bezpečností tokeny generovat o ověřovat.

```
<input
    type="hidden"
    id="timejobbundle_subject__token"
    name="timejobbundle_subject[_token"
    value="v5dYtdKRN3d2fNGhQoiSxWGsWD7Vu1GEDxCLwNyZbz0">
```

Ukázka kódu 3.14: Vygenerovaný CSRF token

3.4.0.3 Injection

„SQL Injection je bezpečnostní chyba založená na možnosti manipulovat s daty v databázi bez nutnosti mít k nim legitimní přístup. Principem je vkládání nových/rozšiřujících SQL dotazů do již existujících SQL dotazů.“ [41]

Díky použití ORM Doctrine je aplikace chráně proti vykonání SQL kódu útočníka, pokud budou metody frameworku používány v souladu s dokumentací[34]. Základní metody Doctrine, jako je například `persist()`, jsou chráněny, protože ORM vrstva vygeneruje dotaz do databáze a automaticky jej parametrizuje. [38]

Bezpečností rizika mohou nastat tam, kde aplikace pracuje s uživatelským vstupem a zároveň vytváří vlastní dotazy pomocí třídy *QueryBuilder* a jazyka DQL. V takovém případě je nežádoucí používat neparametrizované dotazy a ty přímo výkonávat. Místo toho je dle [38] doporučeno použít parametr v dotazu, který je poté inicializován pomocí metody `SetParametr()`. Takto vytvořený dotaz na ORM Doctrine je již řádně zabezpečen.

Nedodržení předchozích doporučení je velmi nebezpečné a důrazně na něj upozorňuje oficiální dokumentace Symfony i ORM Doctrine.

3.4.1 Validace formulářů

Validace formulářů v Symfony probíhá standardně na straně serveru a standardně ji provádí framework za použití validačních pravidel, která byla definována programátorem buďto v anotacích (YML nebo XML souborech) u jednotlivých entit, nebo ve třídách reprezentující formuláře. Tato třída přidává validační pravidla k vygenerovaným formulářovým polím, ale pouze na úrovni jazyka HTML.

Toto řešení je dostačující, neboť k validaci dochází na straně serveru, a v případě, že to webový prohlížeč podporuje, tak i na straně klienta. Pro zvýšení uživatelského komfortu byla v modulu u důležitých formulářů, např. vyhodnocení zápisu a výběru předmětů studenty, implementována validace také pomocí JavaScriptu. Tato kontrola jde snadno obejít vypnutím JavaScriptu v prohlížeči, nicméně je i zde doplněna o validaci vstupů na straně serveru.

[34] Dostupné z: `https://www.doctrine-project.org/`.

Kapitola 4

Testování

Tato kapitola obsahuje popis použitých technologií a postupů při testování kódu pomocí nástroje PHPUnit a také popis provedeného uživatelského testování. Došlo k otestování klíčových částí modulu, k provedení heuristické analýzy a uživatelskému testování.

4.1 Jednotkové testy

Jednotkové testy, neboli *unit testy*, jako nejnižší možnou úroveň testují jednotlivé třídy a správnost návratových hodnot jejich metod. V jednotkových testech by se měl programátor vyvarovat použití databáze, volání metod jiných tříd, používání filesystému nebo přistupování k síti. [42]

V aplikaci byl pro testování využit framework PHPUnit[35], který byl zvolen především z důvodu přímé integrace ve frameworku Symfony. Jednotlivé testy jsou umístěny ve složce `tests`, jejíž struktura kopíruje strukturu modulu `TimejobBundle`. Tedy například ve složce `tests/Entity` jsou testy jednotlivých entitních tříd.

Jednotkové testy jsou samostatné třídy, které rozšiřují třídu *TestCase*. Jednotlivé metody reprezentující testy pracují s objekty dané třídy a vždy vytváří novou instanci téhož objektu, neboť jednotkové testy by neměly zasahovat do databáze nebo ovlivňovat jiné třídy. Na vytvořeném objektu s požadovanými vlastnosti je tedy zpravidla zavolána testovaná metoda a návratová hodnota porovnána s očekávanou. K tomu framework PHPUnit nabízí především funkci *AssertEqual* a podobné, které vždy testují hodnotu předanou parametrem těmto metodám.

V ukázce 4.1 je proveden test metody `isActive()` entitní třídy Enrollment, která vrací informaci o tom, zda je zadaný zápis aktivní a studenti na něj mohou odpovídat. V tomto případě bylo vytvořeno hned několik testů, každý simulující odlišnou situaci.

[35] Dostupné z: `https://phpunit.de/`.

```php
class EnrollmentTest extends TestCase {
    public function testIsActive() {
        $from = new DateTime('now');
        $from->modify('-10 days');

        $to = new DateTime('now');
        $to->modify('+1 day');

        $enrollment = new Enrollment();
        $enrollment->setValidFrom($from);
        $enrollment->setValidTo($to);
        $enrollment->setIsFinished(true);

        $this->assertEquals(true, $enrollment->isActive());
        $this->assertEquals(false, $enrollment->isPassed());
    }
    //...
```

Ukázka kódu 4.1: Příklad jednoduchého jednotkového testu

4.2 Integrační testy

Pro testování funkčnosti operací, které využívají více entit nebo více tříd či například přistupují do databáze, využíváme integrační testy. [42] Pro jejich realizaci byl v této práci použit framework PHPUnit, který nabízí potřebnou funkcionalitu. Testování probíhá v zásadě obdobně jako u jednotkových testů, avšak je třeba testovacím třídám kromě entit zpřístupnit také další důležité služby aplikace. Mohou jimi být například Doctrine, jednotlivé třídy *operation* nebo formulářové třídy. Často se k tomu využívá třída *kernel* Symfony aplikace.

V rámci práce autor vytvořil například testy formulářových tříd 4.2. Je otestována funkčnost předání dat po odeslání, předvyplnění hodnot, vykreslení všech formulářových polí a také validace vstupů, které byly v rámci testu vytvořeny. Validní formát daného pole vždy vychází z anotace u jednotlivých proměnných příslusných entit, pokud není přidána dodatečná validace.

```php
class SubjectsTypeTest extends TypeTestCase
{
    //...
    public function testSubmitValidData()
    {

        $validator = $this->get('validator');

        $formData = array(
            'title' => ''
        );

        $objectToCompare = new Subject();
        $form = $this->factory
            ->create(SubjectType::class, $objectToCompare);

        $form->submit($formData);
        $this->assertTrue($form->isSynchronized());

        $errors = $validator->validate($objectToCompare);
        $this->assertEquals(1, count($errors));

        $view = $form->createView();
        $children = $view->children;

        foreach (array_keys($formData) as $key) {
            $this->assertArrayHasKey($key, $children);
        }
    }
    //...
```

Ukázka kódu 4.2: Příklad testu formulářové třídy

Dále byly otestovány jednotlivé třídy *operation*, které pracují s mnoha entitami a ostaními třídami *operation* či *functionality*. Při testování těchto operací je často využíván tzv. *mock* různých objektů. Jedná se o objekty, které se tváří jako dané třídy, ale neovlivňují zbytek aplikace, například je tento princip využit v aplikaci u repozitářů.

4.3 Heuristická analýza

Následující kapitola popíše první úroveň testování uživatelského rozhraní vytvořeného modulu. Heuristická analýza spočívá v průchodu jednotlivých stránek aplikace a jejich ohodnocení z hlediska použitelnosti. [43]

Při analýze autor práce postupoval dle desatera principů použitelnosti vytvořených Jakobem Nielsnem. [43]

1. **Viditelnost stavu systému**
   Systém by měl vždy dát uživateli vědět, co se právě odehrává.

2. **Spojení mezi systémem a reálným světem**
   Komunikace systému s uživatelem by se měla odehrávat uživatelsky příjemným způsobem (srozumitelný jazyk bez odborných termínů).

3. **Uživatelská kontrola a svoboda**
   Uživatelé při práci se systémem dělají chyby a potřebují proto únikový východ pro návrat do předchozího stavu.

4. **Konzistence a standardizace**
   Uživatelé by neměli být nuceni přemýšlet, jesti různé termíny znamenají to stejné, proto se doporučuje dodržovat obecné zásady.

5. **Prevence chyb**
   Vyvarovat se chybových hlášení bezpečným designem, který bude preventivně působit proti problémům.

6. **Rozpoznání místo vzpomínání**
   Uživatel by neměl být nucen vzpomínat si na provádění operací v systému, instrukce by měly být v systému vždy viditelně umístěny.

7. **Flexibilní a efektivní použití**
   Umožnění zrychlení práce se systém pro pokročilé uživatele.

8. **Estetický a minimalistický design**
   Design má být bez nepotřebných informací.

9. **Pomoc uživatelů poznat, pochopit a vzpamatovat se z chyb**
   Chybové hlášky by měly být uváděny v přirozeném jazyce a měly by navrhovat řešení.

10. **Nápověda a návody**
    Všechny informace se musí dát lehce vyhledat, nápověda by měla obsahovat postupy v krocích.

Po provedení analýzy byly zjištěny následující nedostatky, které byly vyřešeny a opraveny.

Uživatelská kontrola a svoboda
Bylo zjištěno, že se na některých obrazovkách nenachází odkaz na předchozí obrazovku. Tedy například při editaci předmětu se nelze pohodlě odkazem v drobečkové navigaci vrátit na detail předmětu, ale jen na seznam všech předmětů. Dále pokud se uživatel při vytváření validační skupiny u zápisu předmětů rozhodne vrátit zpět bez odeslání formuláře, skupina je již vytvořená a zobrazuje se v seznamu validačních skupin.

Nápověda a návody
Byly zjištěny drobné nedostatky v popisu různých akcí, formulářových polí a odkazů, u kterých chyběla nápověda nebo popis.

Tyto nedostatky, které vyplynuly z analýzy, byly autorem práce opraveny. V prvním případě byly přidány odkazy do drobečkové navigace a ve druhém případě doplněny popisky odkazů nebo symbol otazníku s nápovědou.

4.4 Uživatelské testování

Vzhledem k povaze modulu byly pro uživatelské testování vybrány dvě části modulu. První částí je výběr volitelných předmětů studenty a druhou části je vyplnění požadavků na výuku učitelů školy. Pro testování byla v systému vytvořena modelová data. Během testování se uživatelé budou přihlašovat do informačního systému, tuto funkcionalitu poskytuje informační systém, nikoliv modul, proto tato funkcionalita není testována.

Protože vývoj modulu probíhal přímo pro potřeby školy Michael a tvorbu zápisů, jejich vyhodnocení a tvorbu úvazků bude provádět jedna osoba, byl vývoj modulu průběžně konzultován a jednotlivé nedostatky průběžně opravovány.

4.4.1 Výběr volitelných předmětů - studenti

Důležitou součástí systému je výběr volitelných předmětů studenty školy, proto bylo pro toto testování osloveno celkem pět studentů. Vzhledem k tomu, že se jedná pouze o výběr předmětů v prostředí internetového prohlížeče, není potřeba tuto funkcionalitu testovat přímo se studenty školy Michael. Pro testování byli vybráni dva studenti gymnázia, jedna studentka střední školy se zaměřením na ekonomiku a dva studenti vysoké školy. Všichni bez zaměření na studium informačních technologí a s běžnými uživatelskými schopnostmi práce s počítačem.

Scénář: Vyplnění výběru preferovaných předmětů

Přihlaste se do systému přístupovým kódem, který vám byl poskytnut. Najděte seznam aktuálních zápisů, vyberte si zápis a vyplňte preferované volitelné předměty. Přečtěte si popis zápisu a pokuste se vybrat si volitelné předměty. Formulář odešlete. Pokud si nebudete s něčím vědět rady, nalezněte v systému nápovědu a pokuste se najít odpovědi tam.

Scénář: Změna výběru preferovaných předmětů

Rozmyslíte si svůj výběr volitelných předmětů. Přihlaste se proto do systému a svůj výběr libovolně změňte.

Všichni studenti si s testovacími scénáři poradili velmi dobře a nenastaly žádné významnější komplikace. Studenti ale byli zmateni z nemožnosti vybrat si jeden konkrétní předmět z nabídky. Jednalo se o předmět tzv. navazující, který si mohli zapsat pouze studenti, kteří v minulosti absolvovali jiný předmět.

Dalším nedostatkem, který vyplynul z testování, je tlačítko *Odpovědět*, které by dle návrhů mělo mít spíše popisek *Vyplnit zápis*. Dále se text tohoto tlačítka nelišil, pokud student dosud zápis nevyplnil vůbec a v případě, když student zápis již vyplnil.

V modulu tedy autor provedl potřebné změny. Byl přidán popis k navazujícím předmětům a také byly změněny popisky tlačítka pro vyplnění a úpravy odpovědí zápisu.

4.4.2 Vyplnění preferencí na úvazky - učitelé

Na střední škole Michael se vyučuje široké spektrum předmětů včetně informačních technologí. Většina učitelů má však běžné uživatelské schopnosti práce s počítačem. Jelikož funkcionality modulu pro vyplnění požadavků učitelů nejsou nikterak unikátní právě pro školu Michael, byly pro účely testování osloveni dva učitelé a jeden student pedagogické falkulty na VŠ.

Scénář: Vyplnění požadavků na úvazky

Přihlaste se do systému přístupovým kódem, který vám byl poskytnut. Najděte formulář pro vyplnění požadavků na úvazky, vyplňte a následně formulář odešlete.

Během testování byl odhalen jeden chybějící překlad popisku vybraných tříd a jeden učitel chybně porozuměl počítadlu vybraných preferovaných úvazků a celkového požadovanému úvazku. V modulu byl tedy opraven chybějící překlad a přidána nápověda ve formě ikony otazníku k počítadlu úvazků.

Celkově tedy uživatelské testování proběhlo úspěšně a modul je připraven pro plnou integraci do informačního systému, reálné nasazení a následné testování v ostrém provozu.

Závěr

Cílem práce bylo vytvoření modulu pro tvorbu úvazků a výběr volitelných předmětů pro připravovaný informační systém Střední školy a Vyšší odborné školy reklamní a umělecké tvorby Michael a jeho integrace do systému, otestování a následné nasazení.

Z počátku bylo nutné provést analýzu současného stavu a způsobu realizace procesů ve škole. Po rozhovorech s vedením školy a zpracování analýzy současných procesů byly specifikovány jednotlivé požadavky kladené na modul. Na základě analýzy byl připraven návrh obrazovek a architektury aplikace tak, aby pokrývala všechny funkční požadavky a zároveň, aby architektura modulu umožnila jeho následnou integraci do systému.

V části návrhu aplikace byly vymodelovány všechny případy užití systému a na jejich základě poté došlo k vytvoření datového modelu a návrhu hlavních obrazovek, tzv. wireframů. Vytvořený návrh byl následně ověřen implementací a uživatelským testováním. Po nezbytných úpravách, které vyplynuly z uživatelského testování, byl modul integrován do informačního systému školy.

Hlavním přínosem modulu je zjednodušení procesu výběru volitelných předmětů studenty školy, shromáždění výsledků a jejich následné vyhodnocení. Eliminace papírové podoby dotazníků je velkým přínosem a zamezuje vzniku chyb. Efektivnější je i tvorba úvazků učitelů školy. V nové podobě vedení školy získává od učitelů jejich požadavky a lépe je pak může při tvorbě rozvrhu zohlednit. Modul disponuje interaktivními formuláři s okamžitou validací dat a upozorněním na chyby.

Případné rozšíření modulu o další funkcionality by měla usnadnit jeho architektura, která dodržuje standardy programování a doporučení pro vytváření modulů publikované přímo v dokumentaci frameworku Symfony.

Bibliografie

1. TRBOLA, T. *Webová administrace informačního systému pro střední školu*. Praha, 2018. Bakalářská práce. České vysoké učení technické v Praze, Fakulta informačních technologií.
2. NOVOTNÝ, O. Školní informační systémy. *Metodický portál RVP.CZ* [online]. 2012 [cit. 2018-04-12]. Dostupné z: https://clanky.rvp.cz/clanek/s/Z/8019/SKOLNI-INFORMACNI-SYSTEMY.html/.
3. ČESKÁ ŠKOLNÍ INSPEKCE. *Kvalita a efektivita vzdělávání a vzdělávací soustavy ve školním roce 2016/2017 - výroční zpráva* [online]. 2017 [cit. 2018-04-12]. Dostupné z: http://www.csicr.cz/getattachment/cz/Dokumenty/Vyrocni-zpravy/Kvalita-a-efektivita-vzdelavani-a-vzdelavaci-soust/VZ_CSI_2017_web_new.pdf.
4. BAKALÁŘI S.R.O. *Bakaláři – mezi školou a rodinou* [online]. 2017 [cit. 2018-05-03]. Dostupné z: https://www.bakalari.cz/.
5. ŠKOLA ONLINE A.S. *Školní informační systém Škola OnLine* [online]. 2018 [cit. 2018-04-12]. Dostupné z: https://www.skolaonline.cz/Skolni_informacni_system.aspx.
6. MENCÁK, T. Wireframe - drátěný model - skica webu. *CS Technologies s.r.o* [online]. 2018 [cit. 2018-04-20]. Dostupné z: https://www.cstechnologies.cz/slovnik-pojmu-wireframe-drateny-model-skica-webu-detail-3325.
7. POTENCIER, F. What is Symfony2? *Fabien Potencier* [online]. 2018 [cit. 2018-04-20]. Dostupné z: http://fabien.potencier.org/what-is-symfony2.html.
8. SYMFONY.COM. *Best Practices for Reusable Bundles* [online]. 2018 [cit. 2018-04-20]. Dostupné z: https://symfony.com/doc/current/bundles/best_practices.html.

9. ČÁPKA, D. 1. díl - Popis MVC architektury. *itnetwork.cz* [online]. 2018 [cit. 2018-04-22]. Dostupné z: https://www.itnetwork.cz/php/mvc/objektovy-mvc-redakcni-system-v-php-popis-architektury.

10. SYMFONY.COM. *doctrine: Console Commands* [online]. 2018 [cit. 2018-04-20]. Dostupné z: https://symfony.com/doc/3.3/cookbook/doctrine/console.html.

11. SYMFONY.COM. *Console Commands* [online]. 2018 [cit. 2018-04-20]. Dostupné z: https://symfony.com/doc/current/console.html.

12. SYMFONY.COM. *The Bundle System* [online]. 2018 [cit. 2018-04-20]. Dostupné z: https://symfony.com/doc/3.3/bundles.html.

13. SYMFONY.COM. *Symfony and HTTP Fundamentals* [online]. 2018 [cit. 2018-04-23]. Dostupné z: https://symfony.com/doc/3.3/introduction/http_fundamentals.html.

14. VÍTEK, R. Symfony po krůčkách – Routing. *Zdroják - o tvorbě webových stránek a aplikací* [online]. 2016 [cit. 2018-04-23]. Dostupné z: https://www.zdrojak.cz/clanky/symfony-po-kruckach-routing/.

15. SYMFONY.COM. *HTTP Cache* [online]. 2018 [cit. 2018-04-23]. Dostupné z: https://symfony.com/doc/3.3/http_cache.html.

16. W3SCHOOLS.COM. *HTML Introduction* [online]. 2018 [cit. 2018-04-23]. Dostupné z: https://www.w3schools.com/html/html_intro.asp.

17. W3SCHOOLS.COM. *HTML5 Browser Support* [online]. 2018 [cit. 2018-04-23]. Dostupné z: https://www.w3schools.com/html/html5_browsers.asp.

18. STAUFFER, T. *Tvorba webových stránek pro úplné začátečníky*. SoftPress s.r.o, 2003. ISBN 80-86497-38-0.

19. W3SCHOOLS.COM. *CSS Reference With Browser Support* [online]. 2018 [cit. 2018-04-23]. Dostupné z: https://www.w3schools.com/cssref/css3_browsersupport.asp.

20. PATEL, V. What's new in CSS3? *W3Schools Online Web Tutorials* [online]. 2014 [cit. 2018-04-29]. Dostupné z: https://www.htmlgoodies.com/html5/css/whats-new-in-css3.html.

21. LAVIN, P. *PHP – objektově orientované*. Grada Publishing a.s., 2009. ISBN 978-80-247-2137-8.

22. ZAJÍC, P. PHP (1) - Historie a budoucnost. *Linuxsoft.cz* [online]. 2004 [cit. 2018-04-29]. Dostupné z: http://www.linuxsoft.cz/article.php?id_article=171.

23. W3TECHS. *Usage statistics and market share of PHP for websites* [online]. 2018 [cit. 2018-04-29]. Dostupné z: https://w3techs.com/technologies/details/pl-php/all/all.

24. JANSSON, J. Comparison of summary statistics of Node.js and Ruby On Rails. *James Jansson* [online]. 2013 [cit. 2018-04-29]. Dostupné z: https://jamesjansson.wordpress.com/2013/12/24/comparison-of-summary-statistics-of-node-js-and-ruby-on-rails/.

25. BUCKLER, C. Front-End Tooling Trends for 2017. *sitepoint.com* [online]. 2017 [cit. 2018-04-29]. Dostupné z: https://www.sitepoint.com/front-end-tooling-trends-2017/.

26. ARSENAULT, C. Top 10 Front-End Frameworks of 2016. *keycdn.com* [online]. 2018 [cit. 2018-04-29]. Dostupné z: https://www.keycdn.com/blog/front-end-frameworks/.

27. SYMFONY.COM. *Top 10 Front-End Frameworks of 2016* [online]. 2018 [cit. 2018-04-29]. Dostupné z: https://twig.symfony.com/doc/2.x/intro.html.

28. TICHÝ, J. Seriál: Doctrine 2. *Zdroják.cz* [online]. 2010 [cit. 2018-04-29]. Dostupné z: https://www.zdrojak.cz/serialy/doctrine-2/.

29. DB-ENGINES. *DB-Engines Ranking - Trend Popularity* [online]. 2017 [cit. 2018-04-29]. Dostupné z: https://db-engines.com/en/ranking_trend.

30. KUTÁČ, P. Composer - řešení závislostí a balíčků v PHP. *kutac.cz* [online]. 2016 [cit. 2018-04-29]. Dostupné z: http://www.kutac.cz/blog/weby-a-vse-okolo/composer-reseni-zavislosti-a-balicku-v-php/.

31. SENSIOLABS. *Databases and Doctrine ("The Model")* [online]. 2012 [cit. 2018-04-29]. Dostupné z: http://symfony2-document.readthedocs.io/en/latest/book/doctrine.html.

32. SYMFONY.COM. *Databases and the Doctrine ORM* [online]. 2018 [cit. 2018-04-20]. Dostupné z: https://symfony.com/doc/current/doctrine.html.

33. SYMFONY.COM. *Validation* [online]. 2018 [cit. 2018-04-29]. Dostupné z: https://symfony.com/doc/current/validation.html.

34. DOCTRINE. *Association Mapping* [online]. 2018 [cit. 2018-04-29]. Dostupné z: https://www.doctrine-project.org/projects/doctrine-orm/en/2.6/reference/association-mapping.html.

35. SYMFONY.COM. *Symfony Best Practices* [online]. 2018 [cit. 2018-04-29]. Dostupné z: https://symfony.com/doc/current/best_practices/index.html.

36. OWASP. *OWASP Top 10 - 2017; The Ten Most Critical Web Application Security Risks* [online]. 2018 [cit. 2018-04-29]. Dostupné z: https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf.

37. JAHODA, B. Využití XSS chyby. *JE ČAS* [online]. 2014 [cit. 2018-04-30]. Dostupné z: http://jecas.cz/xss.

38. DOCTRINE. *Security* [online]. 2018 [cit. 2018-04-30]. Dostupné z: https://www.doctrine-project.org/projects/doctrine-orm/en/2.6/reference/security.html.

39. BALOGUN, U. Symfony2 Application Security Guidelines. *University of Pennsylvania – Penn Arts and Sciences* [online]. 2018 [cit. 2018-04-30]. Dostupné z: https://www.sas.upenn.edu/computing/infosec_symfony2.

40. HACKERLAB. *Cross-site Request Forgery (CSRF)* [online]. 2018 [cit. 2018-04-30]. Dostupné z: https://www.hackingkurzy.cz/blog/zranitelnost-cross-site-request-forgery/.

41. ČIŽMAR, L. SQL Injection (Full Paper). *Security-Portal.cz* [online]. 2009 [cit. 2018-04-30]. Dostupné z: http://www.security-portal.cz/clanky/sql-injection-full-paper.

42. ZAMRZLA, J. Testování a tvorba testovatelného kódu v PHP. *Zdroják.cz* [online]. 2012 [cit. 2018-05-01]. Dostupné z: https://www.zdrojak.cz/clanky/testovani-a-tvorba-testovatelneho-kodu-v-php/.

43. LICHNOVSKÁ, P. Heuristická analýza. *Testování a hodnocení rozhraní* [online]. 2009 [cit. 2018-05-05]. Dostupné z: https://human-computer-interaction.webnode.cz/testovani-a-hodnoceni-rozhrani/metody-testovani/heuristicka-analyza/.

Příloha **A**

Seznam použitých zkratek

CSRF Cross-site request forgery

CSS Cascading Style Sheets

DQL Doctrine Query Language

GUI Graphical user interface

HTML Hypertext Markup Language

HTTP Hypertext Transfer Protocol

JS JavaScript

JSON JavaScript Object Notation

MVC Model–view–controller

ORM Object-relational mapping

PHP PHP: Hypertext Preprocessor

Sass Syntactically awesome style sheets

SQL Structured Query Language

URI Uniform Resource Identifier

URL Uniform Resource Locator

XML Extensible markup language

XSS Cross-site scripting

Příloha **B**

Obsah přiloženého CD

```
readme.txt ..... stručný popis obsahu CD
src
    impl ..... zdrojové kódy implementace
    thesis ..... zdrojová forma práce ve formátu LaTeX
text ..... text práce
    thesis.pdf ..... text práce ve formátu PDF
```