Univerzita Pardubice
Fakulta elektrotechniky a informatiky

# Mobilní aplikace pro základní školy

Jan Čapek

Bakalářská práce
2018

**Univerzita Pardubice**
**Fakulta elektrotechniky a informatiky**
Akademický rok: **2017/2018**

# ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

| | |
|---|---|
| Jméno a příjmení: | **Jan Čapek** |
| Osobní číslo: | **I15063** |
| Studijní program: | **B2646 Informační technologie** |
| Studijní obor: | **Informační technologie** |
| Název tématu: | **Mobilní aplikace pro základní školy** |
| Zadávající katedra: | **Katedra informačních technologií** |

Zásady pro vypracování:

Cílem práce je vytvořit mobilní aplikaci pro základní školy, jejíž obsah bude řízen webovou aplikací.
V teoretické části bude provedena rešerše podobných existujících aplikací a popsány vhodné technologie pro realizaci zadané aplikace.
V praktické části bude proveden návrh, analýza a popis implementace vlastní systému, který se bude skládat z mobilní aplikace pro OS Android a z webové aplikace realizované vybranou technologií. Mobilní aplikace, určená pro žáky, bude zobrazovat údaje o akcích, plánovaných písemkách a známkách, které si bude moci do systému také zadávat. Webová aplikace, určená pro učitele, bude umožňovat tvorbu informací, zpracovávat jednotlivé známky a vše ukládat do vybrané databáze.

Rozsah grafických prací:

Rozsah pracovní zprávy: **min. 30 stran**

Forma zpracování bakalářské práce: **tištěná**

Seznam odborné literatury:

**LACKO, L'uboslav. Mistrovství - Android. Přeložil Martin HERODEK. Brno: Computer Press, 2017. Mistrovství. ISBN 978-80-251-4875-4.**
**LECKY-THOMPSON, Ed a Steven D. NOWICKI. PHP 6: programujeme profesionálně. Brno: Computer Press, 2010. Programujeme profesionálně. ISBN 978-80-251-3127-5.**
**CASTRO, Elizabeth a Bruce HYSLOP. HTML5 a CSS3: názorný průvodce tvorbou WWW stránek. Brno: Computer Press, 2012. ISBN 978-80-251-3733-8.**

Vedoucí bakalářské práce: **Ing. Zdeněk Šilar, Ph.D.**
Katedra informačních technologií

Datum zadání bakalářské práce: **31. října 2017**

Termín odevzdání bakalářské práce: **12. května 2018**

Ing. Zdeněk Němec, Ph.D.
děkan

L.S.

Ing. Lukáš Čegan, Ph.D.
pověřený vedením katedry

V Pardubicích dne 20. března 2018

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 9/2012, bude práce zveřejněna v Univerzitní knihovně a prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 6. 5. 2018

Jan Čapek

# PODĚKOVÁNÍ

Chtěl bych poděkovat, Ing. Zdeňku Šilarovi, Ph. D., za odborné rady a vedení při zpracování této práce.

**ANOTACE**

Bakalářská práce se zabývá návrhem a implementací mobilní aplikace pro žáky základních škol. Součástí práce je webová aplikace, která slouží učitelům pro předávání dat aplikaci. Teoretická část práce se zabývá porovnáním již existujících aplikací a popsáním vhodné technologie pro realizaci. V praktické části je vyvinuta aplikace v jazyce Java pro operační systém Android a webová aplikace.

**KLÍČOVÁ SLOVA**

Android, Java, Mobilní aplikace, Webová aplikace, Základní škola

**TITLE**

Mobile application for primary schools.

**ANNOTATION**

The Bachelors thesis is focused on developing mobile application for students of primary schools. Thesis includes web application which allows to teachers put data into application. Theoretical part is based on comparison similar application and describing appropriate technology for realization. The practical part includes developing application in Java programming language and developing web application.

**KEYWORDS**

Android, Java, Mobile application, Web application, Primary school

# OBSAH

**Seznam obrázků** ..... **9**

**Seznam tabulek** ..... **10**

**Seznam zkratek** ..... **11**

**Úvod** ..... **12**

**1 Výzkum již existujících řešení** ..... **13**

1.1 Systém Bakaláři ..... 13

1.2 Škola OnLine ..... 13

1.3 School Manager ..... 14

**2 Operační systém android** ..... **16**

2.1 Android Studio ..... 16

2.2 Základní součásti aplikace pro Android ..... 16

2.3 Databáze SQLite ..... 17

2.4 Kontejnery na rozmístěný prvku ..... 18

**3 Technologie pro tvorbu webové aplikace** ..... **20**

3.1 Jazyk PHP ..... 20

3.1.1 Framework Nette ..... 20

3.2 Jazyk JavaScript ..... 22

3.2.1 Knihovna JQuery ..... 22

3.3 Relační databáze MySQL ..... 23

3.3.1 Dotazovací jazyk SQL ..... 23

3.3.2 Výhody relační databáze ..... 23

3.4 Nette a MySQL ..... 24

3.4.1 Připojení ..... 24

3.4.2 Získávání záznamů ..... 24

3.4.3 Vkládání, modifikace a mazání záznamů ..... 24

3.5 Nástroj phpMyAdmin ..... 25

3.6 Komunikace klient server ..... 25

3.6.1 HTTP ..... 26

3.6.2 JSON ..... 26

**4 Tvorba mobilní aplikace pro žáky** ..... **28**

4.1 Požadavky a návrh ..... 28

4.2 Implementace ..... 28

4.2.1 Aktivita Přihlášení ..... 28

4.2.2 Aktivita Menu ..... 29

4.2.3 Aktivita detail předmětu ..... 29

4.2.4 Aktivita Známky ..... 30

4.2.5 Aktivita Absence ..... 30

4.2.6 Procvičování ..... 30

4.3 Návrh databáze ..... 31

4.4 Získání dat ze serveru ..... 32

4.5 Popis aplikace ..... 34
**5 Tvorba mobilní aplikace pro učitele ..... 38**
5.1 Požadavky a návrh ..... 38
5.2 Implementace dialogových oken ..... 38
5.2.1 Dialogové okno pro přidání známky ..... 38
5.2.2 Dialogové okno pro přidání absence ..... 39
5.2.3 Dialogové okno pro přidání aktuality ..... 39
5.3 Popis aplikace ..... 39
**6 Tvorba webové aplikace ..... 44**
6.1 Role ..... 44
6.2 Návrh databáze ..... 45
6.3 Návrh rozložení stránky ..... 46
6.4 Administrátorská sekce ..... 47
6.4.1 Správa tříd ..... 47
6.4.2 Správa učitelů ..... 48
6.4.3 Správa žáků ..... 48
6.4.4 Správa předmětů ..... 48
6.5 Učitelská sekce ..... 48
6.5.1 Moje Předměty ..... 48
6.5.2 Moje třída ..... 51
6.6 Žákova sekce ..... 52
6.7 Odesílání dat mobilním aplikacím ..... 52
**Závěr ..... 54**
**Použitá literatura ..... 55**
**Přílohy ..... 56**

# SEZNAM OBRÁZKŮ

Obr. 1: Mobilní aplikace Bakaláři[1] ..... 13
Obr. 2: Aplikace Škola OnLine ..... 14
Obr. 3: Aplikace School Manager ..... 15
Obr. 4: MVC architektura [5] ..... 20
Obr. 5: Životní cyklus presenteru[6] ..... 22
Obr. 6: Nástroj phpMyAdmin ..... 25
Obr. 7: Ukázka kódu JSON ..... 27
Obr. 8: Diagram aktivit ..... 28
Obr. 9: Ukázka kódu ..... 29
Obr. 10: Karta Aktuality ..... 30
Obr. 11: Ukázka kódu pro výběr otázky. ..... 31
Obr. 12: Databáze pro mobilní aplikace ..... 32
Obr. 13: Ukázka kódu pro odeslání žádosti na server. ..... 33
Obr. 14: Ukázka kódu pro parsování JSON souboru. ..... 34
Obr. 15: Přihlašovací aktivita ..... 35
Obr. 16: Detail předmětu ..... 36
Obr. 17: Ukázka testování ..... 37
Obr. 18: Diagram aktivit ..... 38
Obr. 19: Aktivita Menu ..... 40
Obr. 20: Aktivita Předmět ..... 41
Obr. 21: Aktivita Aktuality ..... 42
Obr. 22: Aktivita Absence ..... 43
Obr. 23: Use Case Diagram pro role ..... 44
Obr. 24: Logický model databáze ..... 45
Obr. 25 Webová stránka ..... 46
Obr. 26: Ověření oprávnění pro přístup do admin. sekce ..... 47
Obr. 27: Správa tříd ..... 47
Obr. 28: Správa učitelů ..... 48
Obr. 29: Správa žáků ..... 49
Obr. 30: Správa předmětů ..... 50
Obr. 31: Přidávání známek ..... 50
Obr. 32: Dialogové okno známky ..... 51
Obr. 33: Přidání procvičování ..... 51
Obr. 34: Žákova sekce ..... 52
Obr. 35: metoda getznamkydata() ..... 53

# SEZNAM TABULEK

Tabulka 1: Zastoupení operačních systémů na trhu.....16

# SEZNAM ZKRATEK

| | |
|---|---|
| PDF | Portable Document Format |
| PHP | HyperText Preprocessor |
| JSON | JavaScript Object Notation |
| AJAX | Asynchronous JavaScript and XML |
| HTML | HyperText Markup Language |
| XSS | Cross-site scripting |

# ÚVOD

V dnešní době, kdy většina lidí vlastní chytrý telefon, existuje tisíce aplikací. Od jednoduchých aplikací nebo her až po aplikace složité. Aplikace nám v mnoha případech usnadňují život. Pro cestování již nepotřebujeme papírové mapy. Pro plánování používáme kalendář v telefonu. Dokonce je možné sehnat pomocí aplikace ubytování či odvoz.

Tato práce popisuje tvorbu dvou aplikací pro operační systém Android. Jedná se o aplikaci určenou pro žáky a pro aplikaci určenou pro učitele. Operační systém Android byl vybrán z jeho výrazného zastoupení na trhu a také z důvodu programovacího jazyka, kterým je Java. Aplikace pro žáky by měla být nápomocná žákům základních škol, ať se již jedná o zapomínání domácích úkolů či přípravy na hodinu. Součástí práce je i webová stránka díky níž bude možné vkládat data do databáze se kterou bude mobilní aplikace propojena. Aplikace pro učitelům umožňuje přidávat známky jednotlivým žákům. Dále umožňuje přidávat absence a aktuality.

V teoretické části se práce zabývá porovnáním podobných aplikací, které jsou na trhu a jsou popsány vhodné technologie pro vývoj mobilní a webové aplikace.

# 1 VÝZKUM JIŽ EXISTUJÍCÍCH ŘEŠENÍ

V době, kdy je aplikací převeliké množství, je zbytečné psát mobilní aplikace stejnou, nebo velice podobnou již existující aplikaci. Proto je provedena rešerše podobných aplikací. Zjištění nedostatků těchto aplikací může vést k lepšímu vytvoření aplikace vlastní.

## 1.1 Systém Bakaláři

Jedná se o nejrozšířenější online systém pro zjednodušení správy školní agendy. V současnosti jej používá více než 3500 českých škol. Tento systém se skládá z programu pro evidenci žáků a zaměstnanců. Webová aplikace slouží pro komunikaci mezi školou a rodinou. Umožnuje zobrazovat informace o prospěchu docházce či úkolu, a i možnost omlouvání žáků. Mobilní aplikace je určená pro žáky, rodiče a učitele. Mobilní aplikace umožňuje podobnou funkcionalitu jako aplikace webová, přičemž umožňuje i režim offline, tedy stáhnout data a následně je prohlížet bez připojení k internetu[1]. Aplikace Bakaláři je zobrazena na Obr. 1.

Obr. 1: Mobilní aplikace Bakaláři[1]

## 1.2 Škola OnLine

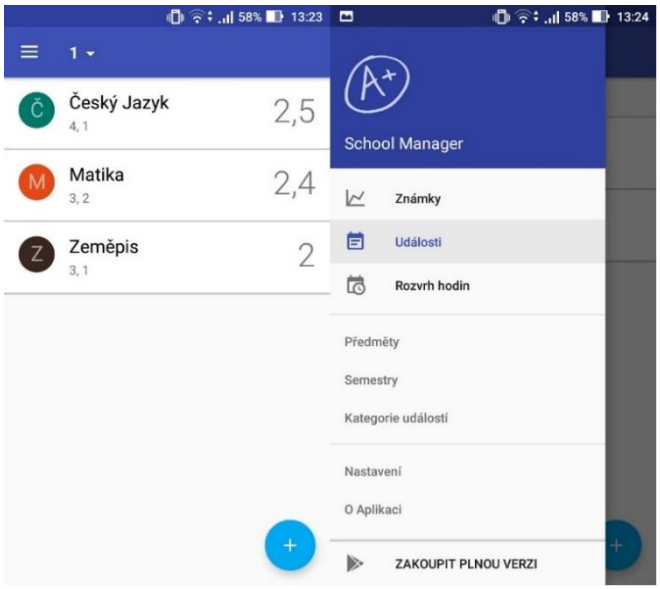
Mobilní aplikace Škola OnLine od stejnojmenné firmy nabízí komplexní řešení pro školy. Mobilní aplikace slouží pro žáky, učitelé a pro rodiče. Učitelé mohou například sledovat změny v rozvrhu, zadávat známky a zapisovat do třídnice, číst, posílat zprávy, a zadávat hodnocení chování žáka a další. Pro žáky a rodiče je pak možnost sledovat aktuální rozvrh i se suplováním a školními akcemi. Dále sledovat přehled plánovaných písemek a domácích

úkolů. Rodiče pak mohou omlouvat absenci či komunikovat se školou prostřednictvím zpráv[2]. Aplikace Škola OnLine je znázorněna na Obr. 2.

Obr. 2: Aplikace Škola OnLine

## 1.3 School Manager

School Manager je aplikace pro vytváření známek, rozvrhu a přehledu o zkouškách a úkolech. Autorem aplikace School Manager je Jan Bína. Pro zadávání známek je možno si vybrat, zda uživatel chce zadávat známky systémem bodů (zlomek) případně číslem (desetinné číslo). Při zadávání známky je nabídnuta uživateli možnost výběru předmětu, případně jeho vytvoření a také možnost vytvoření či přidání semestru. Samostatná známka pak patří do určitého předmětu a semestru včetně data, kdy byla známka zadána. Aplikace umožňuje sestavit si rozvrh. Bezplatná verze aplikace ale neumožňuje při vytvoření předmětu v určitý čas nastavit opakování předmětu. Tím se stává bezplatná verze nevhodná pro vytvoření školního rozvrhu. Dále pro bezplatnou aplikaci je omezení množství semestrů, kategorie událostí a možnost nastavit si upozornění na události. Aplikace také umožňuje přidávat události a kategorie události, do kterých lze události vkládat. Události umožňují nastavit si upozornění. Aplikace nabízí vytvoření tří widgetu, a to pro rozvrh, známky a nadcházející události. Aplikace School Manager je zobrazena na Obr. 3.




Obr. 3: Aplikace School Manager

# 2 OPERAČNÍ SYSTÉM ANDROID

Tato kapitola popisuje platformu Android a její základní komponenty. Mobilní aplikace je napsána pro operační systém Android. Důvodem vyběru operačního systému Android je jeho dominantní pozice na trhu. Proto vývoj aplikace pro Android představuje velký tržní potenciál. Výhodou pro vytváření aplikací na Android je v tom, že je možné, použít počítač s libovolným operačním systémem, tedy Windows, Linux i OS X. Až do letošního roku byl jediným oficiálním programovacím jazykem pro vývoj Android aplikací jazyk Java. V květnu roku 2018 Google oznámil, že přidává další oficiální jazyk, kterým je Kotlin. Kotlin je programovací jazyk, který běží nad JVM (Java Virtual Machine), přičemž je plně interoperabilní s jazykem Java. Pro vytvoření aplikace jsou zapotřebí dva balíky softwaru[3]:

- Java Development Kit (JDK),
- Android Studio.

Tabulka 1: Zastoupení operačních systémů na trhu

| Operační systém | Procentuální zastoupení v % |
|---|---|
| Android | 73,11 |
| IOS | 20,34 |
| Windows OS | 0,71 |
| Ostatní | 5,84 |

## 2.1 Android Studio

Android studio je volně stažitelný oficiální nástroj pro vývoj aplikací pro Android. Dříve společnost Google doporučovala pro vývoj aplikací používat vývojové prostředí Eclipse [3].

## 2.2 Základní součásti aplikace pro Android

Mobilní aplikace pro operační systém Android jsou složeny ze čtyř základních komponentů. Tyto komponenty jsou realizovány jako třídy v Javě [3].

**Aktivity (Activity)**

Aktivita je třída, která slouží pro zobrazování grafického rozhraní uživateli. Aplikace může obsahovat více aktivit, které si mezi sebou mohou odevzdávat údaje. Typicky k tomuto účelu slouží třída `Intent`. Aktivity umožňují ovládání aplikace a zobrazení informací uživatelům, přičemž by měla být navržena tak, aby umožnila uživateli soustředit se na konkrétní věc, kterou uživatel potřebuje realizovat.

**Služby (Services)**

Služby se používají pro déle trvající operace a pro operace běžící na pozadí. Služby umožňují provádět operace paralelně s hlavním vláknem, jako je například přehrávání služby na pozadí či získávání dat ze serveru. Oproti aktivitě služby nepotřebují uživatelské rozhraní. Služba se vytvoří zděděním třídy `Service` nebo `IntentService`.

**Broadcast recievers**

Tyto objekty poslouchají (pracují) na pozadí a reagují na události, které se odehrávají na zařízení. Typickým příkladem je například přijetí SMS či e-mailu.

Poskytovatelé obsahu (Content providers)

Umožňují ukládání nebo sdílení dat mezi více aplikacemi a procesy. Aplikace tedy může přistupovat k jiným aplikacím, které vystupují jako poskytovatelé obsahu.

## 2.3 Databáze SQLite

Android disponuje vestavěnou databází SQLite. Jedná se o malou, avšak výkonnou relační databázi. Databáze pracuje bez serveru a nevyžaduje instalaci a nemá žádné konfigurační soubory. Na rozdíl od jiných databází jako je MySQL či Oracle používá jen klientskou část, zastává funkce klienta i serveru zároveň. Aplikace pak přistupuje k souboru, kde jsou uložena data. SQLite podporuje funkcionalitu klasických databázových serveru. Umožňuje podporu více tabulek, indexování, možnost vytváření spouští a pohledů. Soubory obsahující data, lze nalézt ve složce */data/data/ <název balíčku> /databases/<název databáze>.db.* Pro přístup do databáze se používá třída **SQLiteDatabase** nejčastěji se pak využívají následující metody [3]:

- `insert()` – vloží záznam
- `delete()` – vymaže záznam
- `query()` – provede dotaz a vrátí výsledku jako objekt typu `Cursor`
- `execSQL()` – provede SQL příkaz, ale nevrací žádné výsledky
- `rawQuery()` – provede SQL příkaz, ale vrací výsledky jako objekt typu `Cursor`

Pro komunikaci aplikace s databází se používá třída `SQLiteOpenHelper`. Třída obsahuje následující metody:

- `onCreate(SQLiteDatabase)` – Metoda se volá pokud databáze neexistuje a je potřeba její vytvoření.
- `onUpgrade(SQLiteDatabase, int, int)` – Metoda je zavolána při aktualizaci aplikace.

## 2.4 Kontejnery na rozmístěný prvku

K rozmístění prvků slouží takzvané vizuální kontejnery. Tyto kontejnery jsou odvozeny od třídy `ViewGroup`. Při otočení zařízení „na výšku“ nebo „na šířku“ je zapotřebí geometrická transformace a také změna rozmístění prvků [3].

**LinearLayout**

Rozložení typu LinearLayout umožnuje zobrazovat jednotlivé sloupce do řádku případně do sloupce. Důležité parametry jsou:

- `orientation` – směr upořádání prvků (vertikální/horizontální)
- `layout_gravity` – takzvané těžiště, které definuje centrální bod
- `Layout_wight` – priorita vnořeného pohledu vzhledem k layoutu. Čím vyšší priorita tím zabere prvek více prostoru.

**RelativeLayout**

Relativní rozvržení umožňují definovat poziční vztahy jednak vůči ostatním prvkům nebo podle rodičovského rozvržení. Prvky jsou implicitně umísťovány do levého horního rohu.

Příklady definování polohy:

- `layout_above`
- `layout_alignLeft`
- `layout_alignParentLeft`
- `layout_centerHorizontal`
- `layout_toLeftOf`

**FrameLayout**

Tento layout se využívá hlavně k zobrazování dynamicky vytvořených fragmentů. Layout umožňuje zobrazovat pouze jeden prvek. Umístění prvků je možné měnit pomocí atributu `android:layout_gravity`.

**TableLayout**

Layout se používal do verze API 14. Layout slouží pro vykreslování prvků do formy řádků a sloupců. Pro konstrukci řádků se pak používá komponenta `TableRow`. Nyní se spíše využívá kontejner `GridLayou`t.

### GridLayout

Layout byl přidán do verze od API 14. `GridLayout` představuje matici prvků, přitom poskytuje plnou kontrolu nad tím, jak budou prvky rozvrženy. Pozice prvků se nastavují pomocí atributu `layout_width` a `layout_height`. Jednotlivým buňkám pak lze nastavovat, přes kolik buněk má být prvek roztažen.

### CoordinatorLayout

Kontejner CoordinatorLayout byl zaveden od verze Android 5.0. Je navržen ke koordinaci vzhledu a fungování aplikační lišty se zbytkem rozhraní aplikace.

# 3 TECHNOLOGIE PRO TVORBU WEBOVÉ APLIKACE

Tato kapitola se zabývá jednotlivými technologiemi pro tvorbu webových stránek. Popisuje jazyk PHP a architekturu MVC a jazyk JavaScript včetně knihovny JQuery.

## 3.1 Jazyk PHP

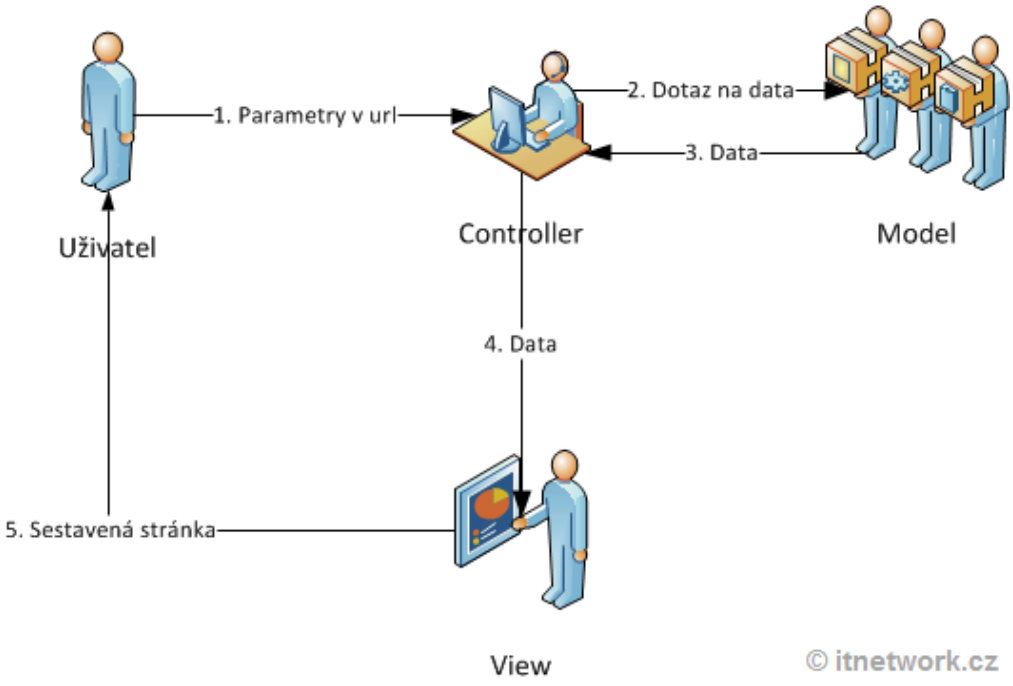
PHP (Hypertext Preprocessor) je skriptovací jazyk, běžící na serveru. Za pomoci PHP je možno vytvářet dynamické a interaktivní webové stránky. PHP skript se nejdříve provede na serveru a potom odešle výsledek prohlížeči.

### 3.1.1 Framework Nette

Nette je český framework napsaný v PHP s plným využitím objektů. V roce 2008 byl uvolněn jako open source a zpřístupněn veřejnosti. Nette spadá do frameworku typu MVC [4].

**MVC**

Architektura se skládá ze tří částí, které se nazývají Model, View a Controller. Architektura vznikla z důvodu potřeby oddělit u aplikací kód obsluhy od kódu aplikační logiky a od kódu zobrazujícího data. Výhodou architektury MVC je zpřehlednění aplikace a zároveň možnosti testovat každou část samostatně [5]. MVC architektura je znázorněna na Obr. 4.




Obr. 4: MVC architektura [5]

**Model**

Model je datový základ aplikace, který obsahuje logiku aplikace, jako je např. práce s databází či výpočty. Model spravuje svůj vnitřní nabízí pevně dané rozhraní, přičemž nemá zdání o existenci pohledu (View) nebo kontroleru (Controller).

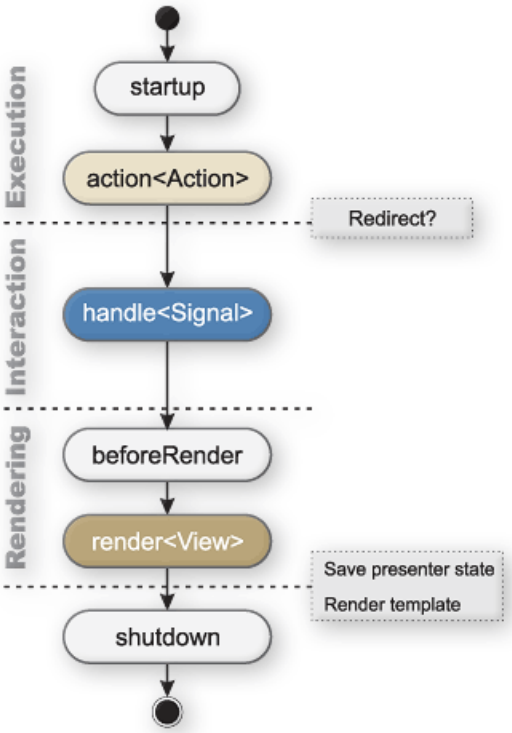
**View**

Pohled má na starosti zobrazit výsledek požadavku. Obvykle se používá šablonovací systém. Pro zobrazení se používá šablonovací systém Latte. Latte zabezpečí vstup před zranitelnostmi jako je například XSS

**Controller (Presenter)**

Controller je řadič, který zpracovává požadavky, v jejichž důsledku následně volá aplikační logiku (Model) a poté požádá pohled (View) o vykreslení. Ve frameworku Nette je obdobou kontroleru komponenta zvaná **presenter**.

Životní cyklus presenteru zobrazen na Obr. 5 se skládá z několika metod. Ihned po inicializaci presenteru se zavolá metoda `startup`. Uvnitř této metody se inicializují proměnné nebo ověřují uživatelské ověření. Metoda `action` se volá před vykreslováním. Používá se v případech, kdy presenter provádí určitý úkon jako přihlášení uživatele či zápis do databáze a může rozhodnout o změně vykreslovací metody `render`. Pro zpracování signálů jako je například požadavek typu AJAX, slouží metoda handle. Těsně před voláním metody `render` se volá metoda `beforeRender`, která obsahuje nastavení šablony nebo předání proměnných společných pro více pohledů. Samotná metoda `render` pak typicky slouží pro naplnění šablony potřebnými daty. Jako poslední volaná metoda, tedy při ukončení životního cyklu je metoda `shutdown` [6].




Obr. 5: Životní cyklus presenteru[6]

## 3.2 Jazyk JavaScript

Javascript je programovací jazyk, jež nachází využití na internetových stránkách. JavaScript má výhodu, že lze zapsat přímo do HTML kódu. JavaScript je klientský skript, to znamená, že běží A je vykonáván v prohlížeči uživatele, oproti tomu serverové skripty jsou vykonávány na serveru[7].

**Omezeni jazyka:**

- Funguje pouze v prohlížeči.
- Může být zakázán uživatelem.
- Odlišné verze prohlížečů můžou vést k chybám.
- Neumí přistupovat k souborům.

### 3.2.1 Knihovna JQuery

Knihovna umožňuje provádět následující úlohy[8]:

- Přistupovat k jednotlivým elementům
- Upravovat vzhled stránky
- Měnit obsah dokumentu
- Reagovat na akce uživatele
- Animovat změny dokumentu
- Pomocí technologie AJAX načítat data ze serveru bez nutnosti obnovy stránky

## 3.3 Relační databáze MySQL

Relační databáze ukládá data do tabulek, které mezi sebou mají vztahy. Relační model určují tři typy vztahů:

- 1:1
- 1:N
- M:N

Vztah je pak zřízen pomocí dvou odpovídacích hodnot. Každá tabulka musí obsahovat primární klíč, díky čemuž lze jednoznačně identifikovat daný řádek tabulky, a pokud má vztah s jinou tabulkou musí obsahovat i cizí klíč [9].

### 3.3.1 Dotazovací jazyk SQL

S daty se pracuje pomocí jazyka SQL (Structured Query Language). Příkazy jazyka SQL lze rozdělit na příkazy typu DDL (Data Definition Language), DML (Data Manipulation Language) a DCL (Data Control Language). Příkazy typu DDL slouží změnu databázové struktury a patří do něj následující příkazy:

- CREATE – Vytváří tabulku či pohled případně jiný objekt.
- ALTER – Upravuje již existující objekt.
- DROP – Odstraňuje objekt.

Příkazy typu DML slouží pro manipulaci s daty v databázi. Do skupiny DML patří následující příkazy:

- SELECT – Získání záznamu z tabulky.
- INSERT – Vložení záznamu do tabulky.
- UPDATE – Modifikace záznamu.
- DELETE – Odstranění záznamu.

DCL slouží pro administraci přístupu do databáze, obsahuje příkazy:

- GRANT – Slouží pro přidělení privilegií uživateli.
- REVOKE – Slouží pro odebrání privilegií uživateli.

### 3.3.2 Výhody relační databáze

Mezi výhody relační databáze patří [9]:

- Zabudovaná víceúrovňová integrita.
- Logická a fyzická nezávislost dat na databázové aplikaci.
- Garantovaná konzistence a přesnost dat.
- Snadné získání dat.

## 3.4 Nette a MySQL

Framework Nette poskytuje rozhraní pro snadnější práci s databází. Umožňuje snadněji získávat data a ulehčuje skládání dotazů [10].

### 3.4.1 Připojení

Pro vytvoření připojení lze použít dvě metody. První metodou lze připojení získat za pomoci instance třídy `Nette\Database\Connection`.

```
$database = new Nette \ Database \ Connection ( $dsn , $user , password )
```

Parametr `$dsn` (data source name) je stejný jako používá PDO. Druhou metodou získáme připojení tak, že do aplikační konfigurace přidáme sekci database, tím se vytvoří potřebné objekty. Sekce pak vypadá následovně:

```
database:
    dsn: 'mysql:host=127.0.0.1;dbname=test'
    user: root
    password: password
```

### 3.4.2 Získávání záznamů

Pro operace s databází se používá Nette Database Explorer, ten výrazným způsobem ulehčuje získávání dat z tabulky, a to bez nutnosti psát SQL dotazy. Tabulka lze získat zavoláním metody `table()` Tato metoda nám vrátí objekt třídy `Selection`.

```
$zaci = $context->table('zaci') // vrátí celou tabulku žáků
```

Nad tímto objektem je možno iterovat z čehož se získají instance typu `ActiveRow` což je již reprezentace řádku ze kterého je možno získávat hodnoty. Získání hodnoty řádku podle jeho primárního klíče se provádí metodou `get()`.

```
$zaci = $context->table('zaci')->get(12)// vrátí žáka s id=12
```

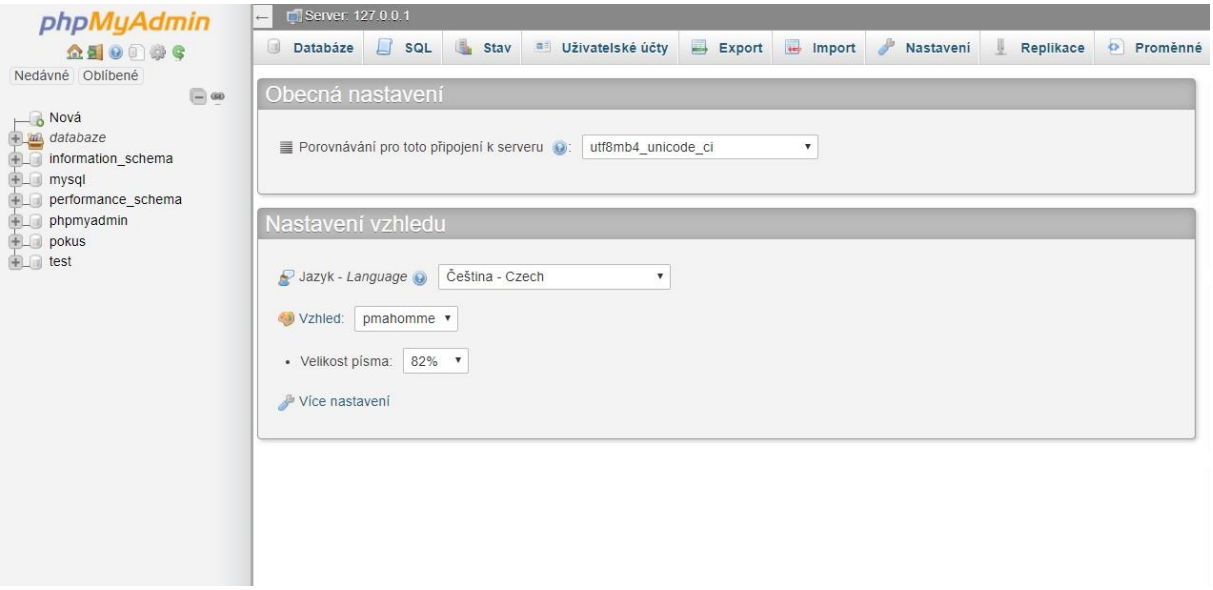
### 3.4.3 Vkládání, modifikace a mazání záznamů

Pro vkládání záznamů se používá metoda `insert()` jako argument bere pole. či Traversable objekty (např. `ArrayHash`). Modifikace probíhá pomocí metody `update()`, která přijímá stejné argumenty jako metoda `insert()`, ale předchází jí metoda `where()` pro specifikaci záznamu, který má být upraven. Pro mazání se používá metoda `delete()` této metodě

stejně jako u metody update musí předcházet upřesnění, o který záznam se jednota a to za pomoci metody `where()`.

## 3.5 Nástroj phpMyAdmin

phpMyAdmin je volně dostupný nástroj napsaný v jazyce PHP, který slouží pro správu MySQL databáze přes webové rozhraní. Rozhraní umožňuje jednoduché vykonávání většinu databázových operací. phpMyAdmin umožňuje export tabulek do několika formátů jako je CSV, XML, SQL či PDF. Součástí nástroje je rovněž grafický návrhář. phpMyAdmin je zobrazen na Obr. 6.




Obr. 6: Nástroj phpMyAdmin

## 3.6 Komunikace klient server

Pro komunikaci klientské aplikace se serverem lze využít dvě strategie, a to strategií **polling** nebo strategií **pushing**. Při strategií polling klientská aplikace posílá zprávy serveru za účelem zjištění změny dat. Největší nevýhodou této strategie je, že při zvýšení četností dotazů na server se výrazně spotřebovává baterie mobilního zařízení. Typickým zastupitel aplikace s využitím strategie polling je aplikace pro zobrazení předpovědi počasí. Při strategií typu pushing místo, aby se aplikace dotazovala serveru, zdali jsou k dispozici nějaké aktualizace, tak server je zodpovědný sdělit mobilnímu klientovi že má aktualizovaná data.

### 3.6.1 HTTP

HTTP (Hypertext Transfer Protocol) je protokol pracující na aplikační vrstvě a je založen na principu požadavek/odpověď. Klient pošle požadavek (typicky adresu stránky) a server mu pošle nazpět odpověď [11].

**Dotazovací metody**

Dotazovací metoda indikuje metodu, která se má provést nad zdrojem identifikovaným v adrese. Protokol HTTP/1.1 definuje následující metody:

- GET – metoda slouží pro přijetí informací ze serveru a neměla by data nijak ovlivnit
- HEAD – stejné jako metoda GET ale nepřenáší se tělo.
- POST – slouží pro odeslání dat na server.
- PUT/DELETE – vytvoří/smaže objekt ze serveru.

**Výsledkové kódy**

Výsledkové kódy lze rozdělit do pěti skupin, a to podle počáteční číslice následovně:

- Informační (1xx) – požadavek byl obdržen
- Úspěch (2xx) – dotaz proběhl v pořádku
- Přesměrování (3xx) – je potřeba dalších akcí pro získání dokumentu
- Chyba na straně klienta (4xx) – Byl zadán chybný dotaz nebo klient nedisponuje dostatečným oprávněním
- Chyba na straně serveru (5xx) – Server není schopen obsloužit požadavek

### 3.6.2 JSON

JSON (JavaScript Object Notation) je textový formát pro výměnu dat přes internetové připojení. Data jsou reprezentovány jako dvojice jméno a hodnota (odděleny dvojtečkou). Složené závorky slouží pro držení objektu a hranaté závorky pro držení pole kde hodnoty jsou odděleny čárkou. Ukázka kódu JSON je zobrazena na Obr. 7.

Obr. 7: Ukázka kódu JSON

**Porovnání JSON a XML**

JSON má oproti XML následující vlastnosti [12]:

- Nepoužívá ukončující značky
- Je kratší
- Je rychlejší pro čtení a zápis
- Může používat pole.

Další výhodou formátu JSON je rychlost a jednoduchost parsování. JSON může být parsován standardní JavaScriptovou funkcí.

# 4 TVORBA MOBILNÍ APLIKACE PRO ŽÁKY

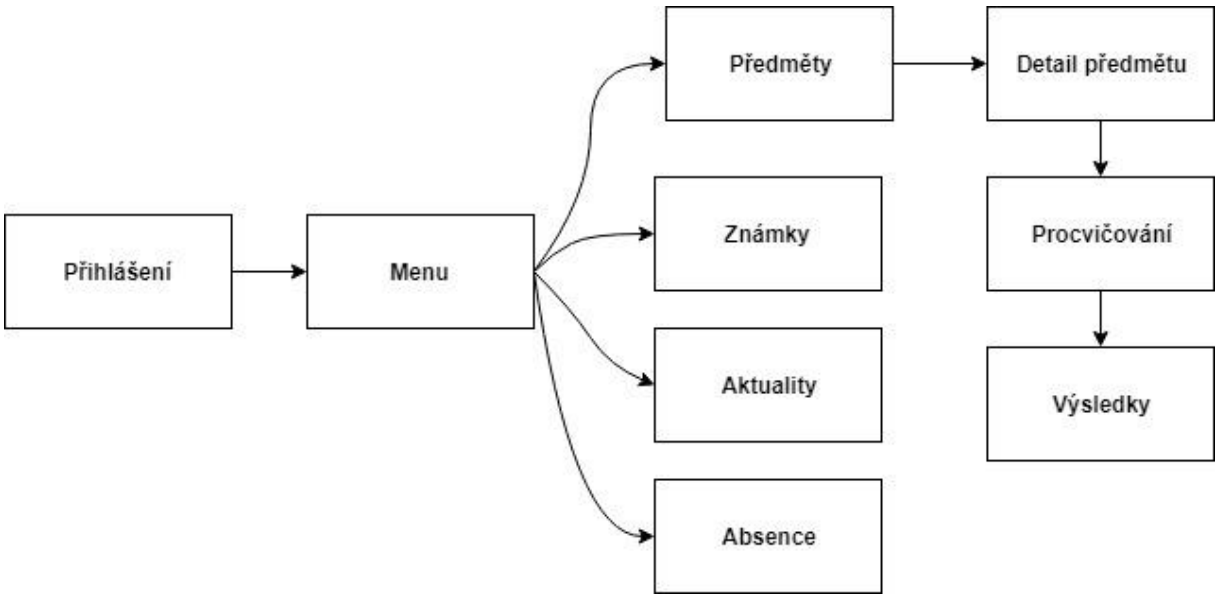
Mobilní aplikace pro žáky má za úkol zobrazovat známky, aktuality, absenci a možnost procvičování látky.

## 4.1 Požadavky a návrh

Aplikace pro žáky má za úkol zobrazovat následující entity:

- Známky,
- Předměty,
- Absence,
- Aktuality.

Aplikace by také měla mít možnost procvičování, tedy otestování otázek, které učitel zadal prostřednictvím webové aplikace. Pro zobrazování každé entity byla navržena jedna aktivita. Pro lepší přehled v rámci jednoho předmětu byla navržena aktivita, která představuje daný předmět se známkami i společně s aktualitami a procvičováním. Diagram aktivit je na Obr. 8.




Obr. 8: Diagram aktivit

## 4.2 Implementace

Tato kapitola se zabývá implementací daných třídy a aktivit včetně ukázek kódu.

### 4.2.1 Aktivita Přihlášení

Aktivita přihlášení se skládá ze dvou komponent typu `EditText` pro přihlašovací údaje a pro přihlašovací údaje a komponenty `CheckBox` zapamatování údajů. Po stisknutí tlačítka a ověření, zda byla pole vyplněná, se předají údaje do třídy `BackgroundReciever`, která

tyto údaje ověří na serveru. Pokud údaje souhlasí tak aplikace spustí aktivitu Menu a začne stahovat data (viz kapitola 6.5), pokud zadané údaje nesouhlasí tak aplikace zobrazí uživateli zprávu o neúspěšném přihlášení.

### 4.2.2 Aktivita Menu

Aktivita je postavená na rozvržení typu `GridLayout`. Uvnitř tohoto rozvržení se nachází čtyři komponenty typu `CardView`. Na tyto komponenty byl vytvořen posluchač `setOnClickListener`. Ukázka kódu je vidět na Obr. 9.

```
for (int i = 0; i < mainGrid.getChildCount(); i++) {
    CardView cardView = (CardView) mainGrid.getChildAt(i);
    final int finalI = i;
    cardView.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            switch (finalI){
                case 0:
                    Intent intent = new Intent( packageContext: MainActivity.this,PredmetyActivity.class);
                    startActivity(intent);
                    break;
                case 1:
                    startActivity(new Intent( packageContext: MainActivity.this,AktualityActivity.class));
                    break;
                case 2:
```

Obr. 9: Ukázka kódu

### 4.2.3 Aktivita detail předmětu

Aktivita detail předmětu je založena na aktivitě typu Tabbed Activity. Třída obsahuje vnitřní třídu `SectionsPagerAdapter`, která dědí ze třídy `FragmentStatePagerAdapter`. Tato třída slouží pro práci s fragmenty a překrývá následující metody:

- `getItem()` – vrací fragment dle pozice v dané záložce
- `instantiateItem()`
- `destroyItem()`
- `getCount()`
- `getPageTitle()`

Jako fragmenty byly vytvořeny tři třídy, které dědí ze třídy `Fragments`. Každá třída reprezentuje jednu záložku. Záložky byly vytvořeny pro aktuality, známky, a procvičování.

**Záložka aktuality**

Rozvržení ve třídě `TabAktuality` která reprezentuje záložku aktualit obsahuje pouze kontejner typu `ListView`. Pro reprezentaci jednotlivé aktuality bylo použité rozvržení s kořenovým prvkem typu `CardView` uvnitř se pak nachází textové pole a tlačítko.

Rozvržení je na Obr. 10. Pro vložení prvku do seznamu typu `ListView` bylo potřeba přetížit třídu `ArrayAdapter`.

Obr. 10: Karta Aktuality

**Záložka známky**

Tato záložka obsahuje stejně jako záložka aktualit `ListView` nad kterým se nachází `TextView`, který slouží pro vypsání průměru známek. Pro reprezentaci jednotlivé známky je vytvořeno rozvržení, které obsahuje komponentu typu `ImageView` pro zobrazení známky a dva komponenty typu `TextView` pro zobrazení popisu a datumu.

**Záložka procvičovaní**

Záložka je postavena téměř shodně jako záložka aktualit s výjimkou toho, že karta pro reprezentaci jednotlivé položky obsahuje o jeden textový prvek méně.

### 4.2.4 Aktivita Známky

Aktivita pro zobrazení známek používá stejné rozvržení pro zobrazení známky, jako tomu je v aktivitě Detail předmětu. Je zde přidán textový prvek pro znázornění, z jakého předmětu je známka. Aktivita dále obsahuje komponentu `Spinner` podle, kterého lze vybrat jeden předmět.

### 4.2.5 Aktivita Absence

Aktivita absence obsahuje komponentu `ListView`. Komponenta obsahuje základní rozložení typu `simple_list_item_1`

### 4.2.6 Procvičování

Aktivita obsahuje čtyři tlačítka pro možnost odpovědi a text pro zadání otázky. Pro otázku byla vytvořená třída, která obsahuje dvojici hodnot pro odpověď a pro zadání otázky. Aktivita nejdříve nahraje otázky z databáze do objektu typu `ArrayList` a následně vytvoří mapu,

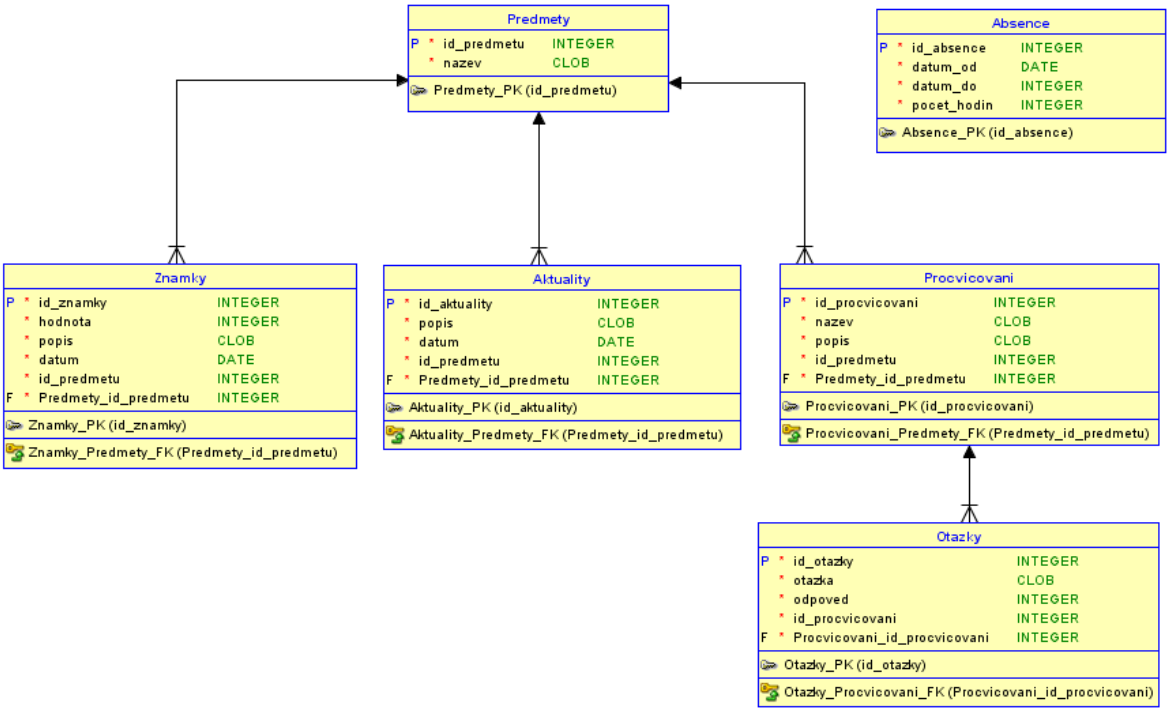
která jako klíč obsahuje indexy, které představují danou otázku v seznamu otázek a logickou hodnotu boolean. Hodnota slouží pro zjištění, zdali již otázka byla zobrazena či nikoliv. Generátorem náhodných čísel se pak vybere náhodné číslo v rozmezí 0 – počet otázek. Toto číslo se pak použije pro zjištění hodnoty v mapě. Pokud je hodnota typu false tak se proces opakuje. Pokud je hodnota typu true zjistí se daná otázka z listu otázek. Následně se zadání otázky nahraje do textview a odpověď se vypíše do jednoho náhodně vybraného tlačítka. Do ostatních tlačítek se nahraje odpověď z jiných otázek. Po zvolení odpovědi se tento proces zopakuje, pokud uživatel zadá špatnou odpověď tak se uloží do seznamu špatných odpovědí. Po skončení procvičování se pak tento seznam předá aktivitě s názvem Výsledky. Ukázka kódu je na Obr. 11.

```
    boolean vybrano = false;
    int cislo;
    do{
        cislo = (int) (Math.random() * pocetOtazek-1);
        boolean nebylaOtazka = mapa.get(cislo);
        if(nebylaOtazka){
            vybrano = true;
            mapa.put(cislo,false);
        }
    }while (!vybrano);

    return cislo;
}
```

Obr. 11: Ukázka kódu pro výběr otázky.

## 4.3 Návrh databáze

Aplikace i přes to, že se stahují informace z webového serveru, používá svou SQLite databázi. Po spuštění aplikace pak tuto databázi obnoví tak, aby se shodovala z databází na serveru, respektive z části, ke které má žák přístup. Hlavní tabulka je tabulka předmětu, jejíž primární klíč je cizím klíčem v ostatních tabulkách (vyjma obecné aktuality, které jsou vázané na třídu). Databáze je znázorněna na Obr. 12.




Obr. 12: Databáze pro mobilní aplikace

Pro práci s databází byla vytvořena třída `DataModel`, která dědí ze třídy `SQLiteOpenHelper`. Třída obsahuje metody pro práci s databází.

## 4.4 Získání dat ze serveru

Pro získání dat ze serveru byla vytvořena třída `BackgroundReciever`, která dědí ze třídy `AsyncTask`. Třída `AsyncTask` se používá pro krátce trvající operace (doporučuje se pár sekund) a umožňuje provádět tyto operace v pozadí. Výhoda provádění operací na pozadí je ta, že neblokují hlavní vlákno. Pokud by se používalo hlavní vlákno a nastala nějaká událost, kterou by nebylo možné zpracovat, aplikace by se zasekla a stala neovladatelnou. Android by v tomto případě použil dialogové okno tázající uživatele, zdali chce počkat nebo aplikaci vypnout. Třída `AsyncTask` má tři generické typy `Params`, `Progress` a `Result`. Dědící třída přetíží minimálně jednu metodu a tou je `doInBackground(Params ...)`. Vykonání úlohy se provede zavoláním metody `execute`.

**Příklad načtení známek**

Načtení známek probíhá tak, že aplikace odešle id žáka (které získalo od serveru po přihlášení) a poslední id známky na server. Ukázka kódu je znázorněna na Obr. 13.

```
String url_adresa = "http://www.aplikace-zs.cz/?do=getznamkydata";
URL url = new URL(url_adresa);
HttpURLConnection httpURLConnection;
httpURLConnection = (HttpURLConnection) url.openConnection();
httpURLConnection.setRequestMethod("POST");
httpURLConnection.setDoOutput(true);
httpURLConnection.setDoInput(true);

OutputStream outputStream = httpURLConnection.getOutputStream();
BufferedWriter bufferedWriter =
        new BufferedWriter(
                new OutputStreamWriter(outputStream, charsetName: "UTF-8"));
String post_data =  URLEncoder.encode( s: "id_zaka", enc: "UTF-8")
            + "=" + URLEncoder.encode(String.valueOf(idZaka), enc: "UTF-8");

bufferedWriter.write(post_data);
bufferedWriter.flush();
bufferedWriter.close();
outputStream.close();
```

Obr. 13: Ukázka kódu pro odeslání žádosti na server.

Metoda `getZnamkyData()` na web serveru přijme id žáka a id poslední známky, provede dotaz do databáze kde požaduje známky které patří žákovi a které mají větší hodnotu id než přijaté id. Následně odešle data přes typ JSON do aplikace. Více viz kapitola 6.7. Aplikace přijatý soubor parsuje, vytvoří objekty typu známka, a ty uloží do databáze. Ukázka kódu pro parsování JSON souboru je zobrazena na Obr. 14: .

```
InputStream inputStream = httpURLConnection.getInputStream();
BufferedReader bufferedReader =
        new BufferedReader(
        new InputStreamReader(inputStream,  charsetName: "UTF-8"));

while ((line = bufferedReader.readLine()) != null) {
    responseStrBuilder.append(line);
}

JSONArray jsonArray = new JSONArray(responseStrBuilder.toString());

for (int i = 0;i < jsonArray.length();i++){
    JSONObject c = jsonArray.getJSONObject(i);
    int znamkaID= c.getInt( name: "id");
    int hodnota_znamky = c.getInt( name: "Znamka");
    int id_predmetu = c.getInt( name: "id_predmetu");
    String popis_znamky = c.getString( name: "Popis");
    String datum = c.getString( name: "Datum");
    dm.pridejZnamku(new Znamka
            (hodnota_znamky,popis_znamky,znamkaID,datum),id_predmetu);
}
```

Obr. 14: Ukázka kódu pro parsování JSON souboru.

## 4.5 Popis aplikace

Tato podkapitola obsahuje základní popis a funkcionalitu aplikace z pohledu uživatele.

**Přihlášení**

Při spuštění aplikace se jako první zobrazí aktivita sloužící pro přihlášení do systému. Žák zadá svůj email a své heslo, které mu bylo přiděleno. Žák má také možnost zvolit možnost zapamatování přihlašovacích údajů, nadále se pak nebude zobrazovat přihlašovací obrazovka (pokud nenastane chyba například v důsledku nepřipojení zařízení k internetu), ale zobrazí se menu aktivita. Přihlašovací aktivita je zobrazena na Obr. 15.

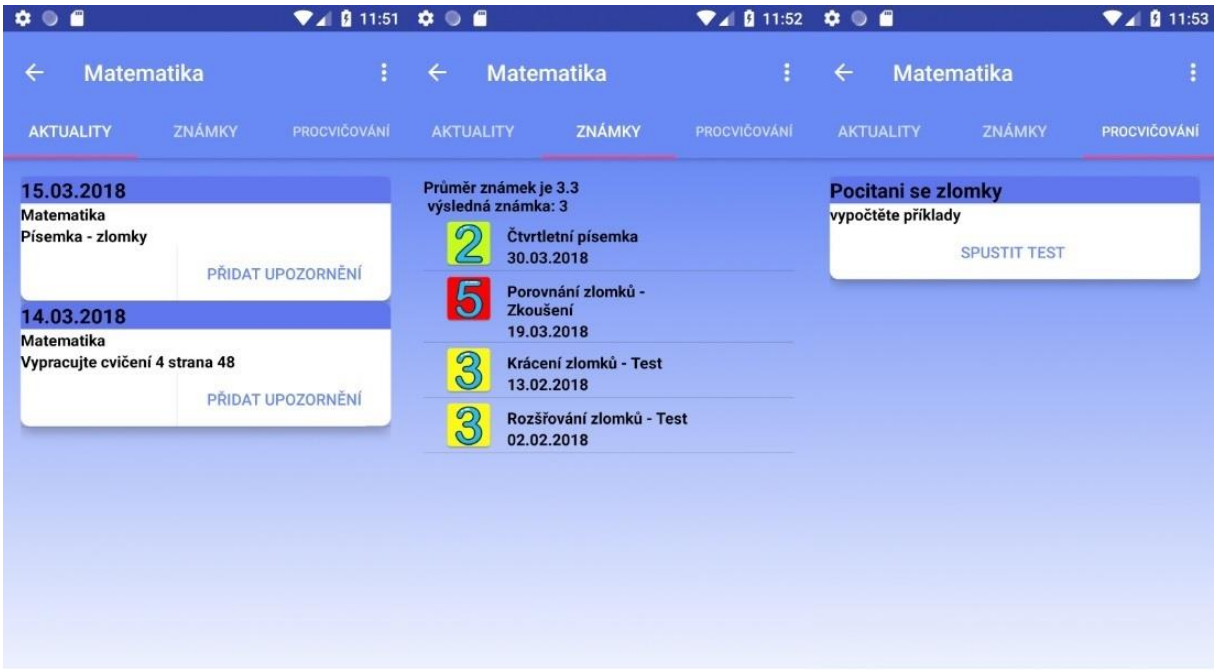
Obr. 15: Přihlašovací aktivita

**Menu**

Po přihlášení se uživateli zobrazí nabídka menu. Menu obsahuje možnost přepnutí do jednoho z modulu, kterými jsou: předměty, aktuality, známky a absence.

**Detail předmětu**

Po přepnutí do modulu předmětu a vybrání určitého předmětu se zobrazí aktivita s názvem Detail předmětu. Tato aktivita obsahuje tři záložky, a to pro aktuality, známky a procvičování. Aktivita Detail předmětu je znázorněna na Obr. 16.

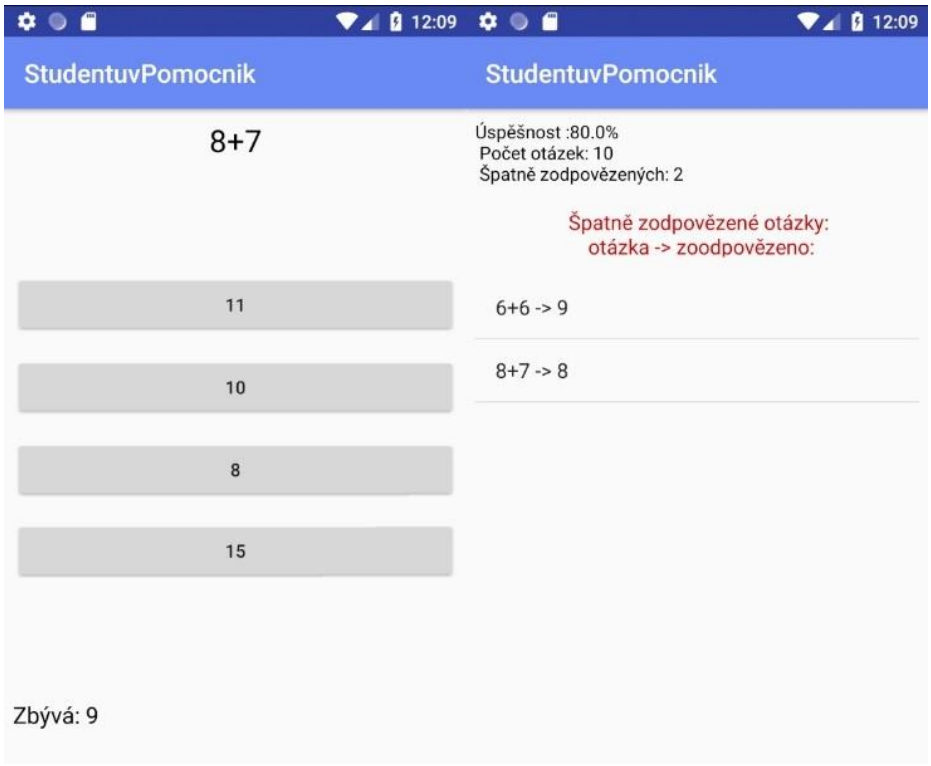



Obr. 16: Detail předmětu

Záložka aktuality zobrazuje nadcházející aktuality předmětu a umožňuje k nim přidat upozornění na určitý čas a datum. Upozornění se pak zobrazí jako notifikace. Záložka známky zobrazuje žákovy známky z předmětu a jejich průměr. Poslední záložka pak obsahuje procvičování, které učitel daného předmětu vytvořil prostřednictvím webové aplikace. Testování pak funguje na principu vybrání jedné za čtyř možností. Po dokončení testu se žáku zobrazí otázky, na které špatně odpověděl společně s jeho odpověďmi. Zobrazí se také procentuální úspěšnost, která je následně odeslána na server, kde jí má možnost shlédnout daný učitel předmětu. Testování je zobrazeno na Obr. 17.

**Modul Aktuality, Známky, Absence**

Moduly aktuality a známky fungují na stejném principu jako jednotlivé záložky v detailu předmětu, ale zobrazují vše, tedy všechny aktuality a všechny žákovy známky, přičemž je možno filtrovat podle jednotlivých předmětů. U aktualit navíc zobrazuje i obecné aktuality, které vytváří třídní učitel. Modul absence zobrazuje žákovu absenci tedy datum, kdy chyběl a kolik hodin zameškal.




Obr. 17: Ukázka testování

# 5 TVORBA MOBILNÍ APLIKACE PRO UČITELE

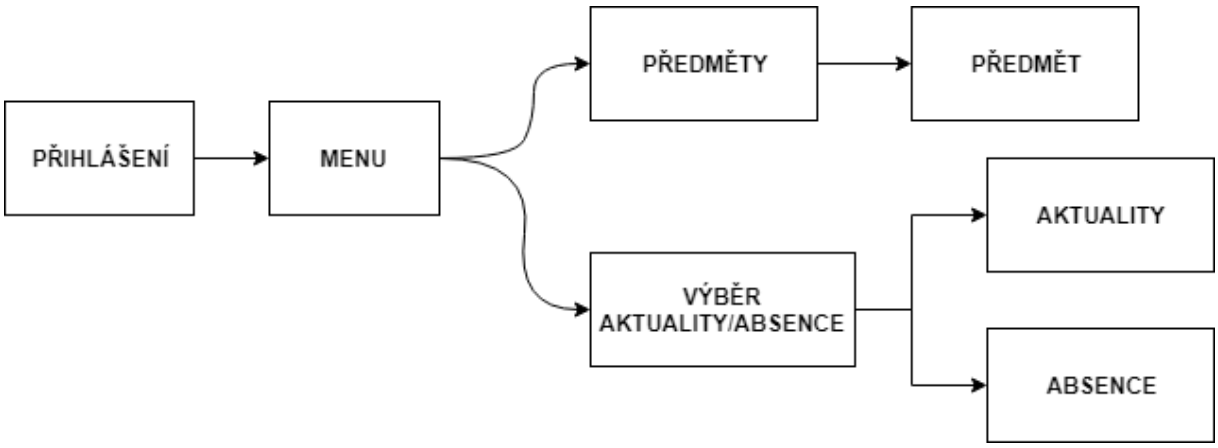
Tato kapitola se zabývá vývojem a popisem mobilní aplikace pro učitele. Aplikace obsahuje některé komponenty jako aplikace pro žáky.

## 5.1 Požadavky a návrh

Aplikace pro učitelé disponuje možností přidat:

- známky,
- absence,
- aktuality.

Návrh diagramu aktivit je zobrazen na Obr. 18.




Obr. 18: Diagram aktivit

## 5.2 Implementace dialogových oken

Pro přidání entit byla vytvořena dialogová okna. Tato kapitola popisuje jejich implementaci. Pro dialogové okno byla použita třída `Dialog`. Instanci této třídy byli přiděleno rozvržení pomocí metody `setContentView()`.Vybrané přiřazení prvkům rozhraní s instancemi podtříd `View` byla na instanci třídy `Dialog` zavolána metoda `findViewByID`.

### 5.2.1 Dialogové okno pro přidání známky

Pro dialogové okno byly přidány následující komponenty:

- `RadioGroup` – Pro vybrání známky
- `EditText` – Pro popis známky
- `NumberPicker` – Pro jednoduché vybrání data
- `Button` – Pro potvrzení případně zrušení dialogového okna

### 5.2.2 Dialogové okno pro přidání absence

Pro dialogové okno byly přidány následující komponenty:

- `TextView` – pro zadání počtu zameškaných hodin
- `Button` – pro vybrání data či potvrzeni/ukončení dialogového okna

Pro vybrání dat byla použita třída `DatePickerDialog`. Po zavolání metody `show()` a vybrání určitého data se datum předá do metody `onDateSet()`. Jako parametr metoda přijímá rok, měsíc a datum. Uvnitř metody pak dojde přiřazení datumu do proměnných pro následující zpracování.

### 5.2.3 Dialogové okno pro přidání aktuality

Pro dialogové okno byly přidány následující komponenty:

- `EditText` – pro název a pro popis
- `Button` – tlačítka pro vybrání data a případné potvrzeni/ukončení dialogového okna

Výběr data probíhá stejně jako v dialogovém okně pro přidání absence.

## 5.3 Popis aplikace

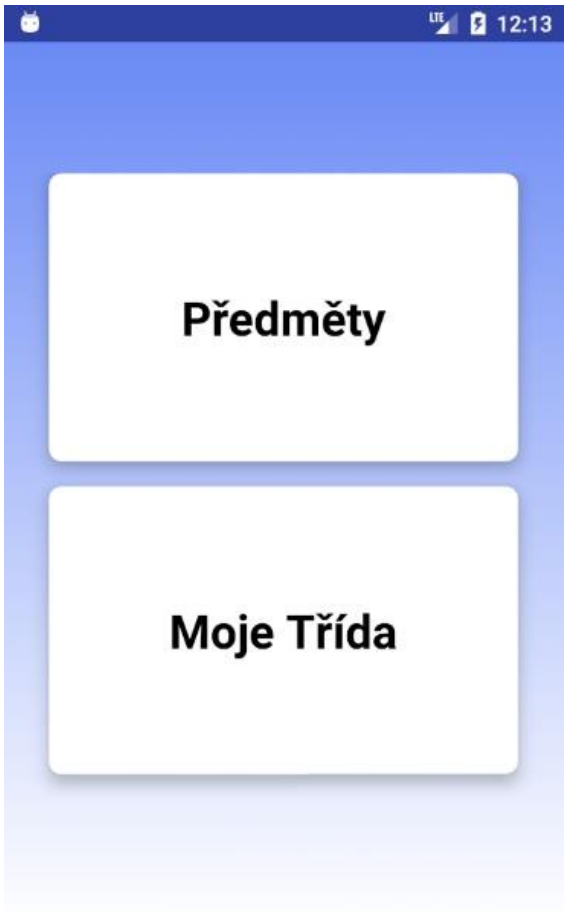
Tato podkapitola obsahuje základní popis a funkcionalitu aplikace z pohledu uživatele

**Přihlášení**

Přihlášení funguje stejně jako u aplikace pro žáky. Rozdíl je akorát v tom, že není možné zapamatovat si přihlašovací údaje a je tak nutno pokaždé údaje zadat.

**Menu**

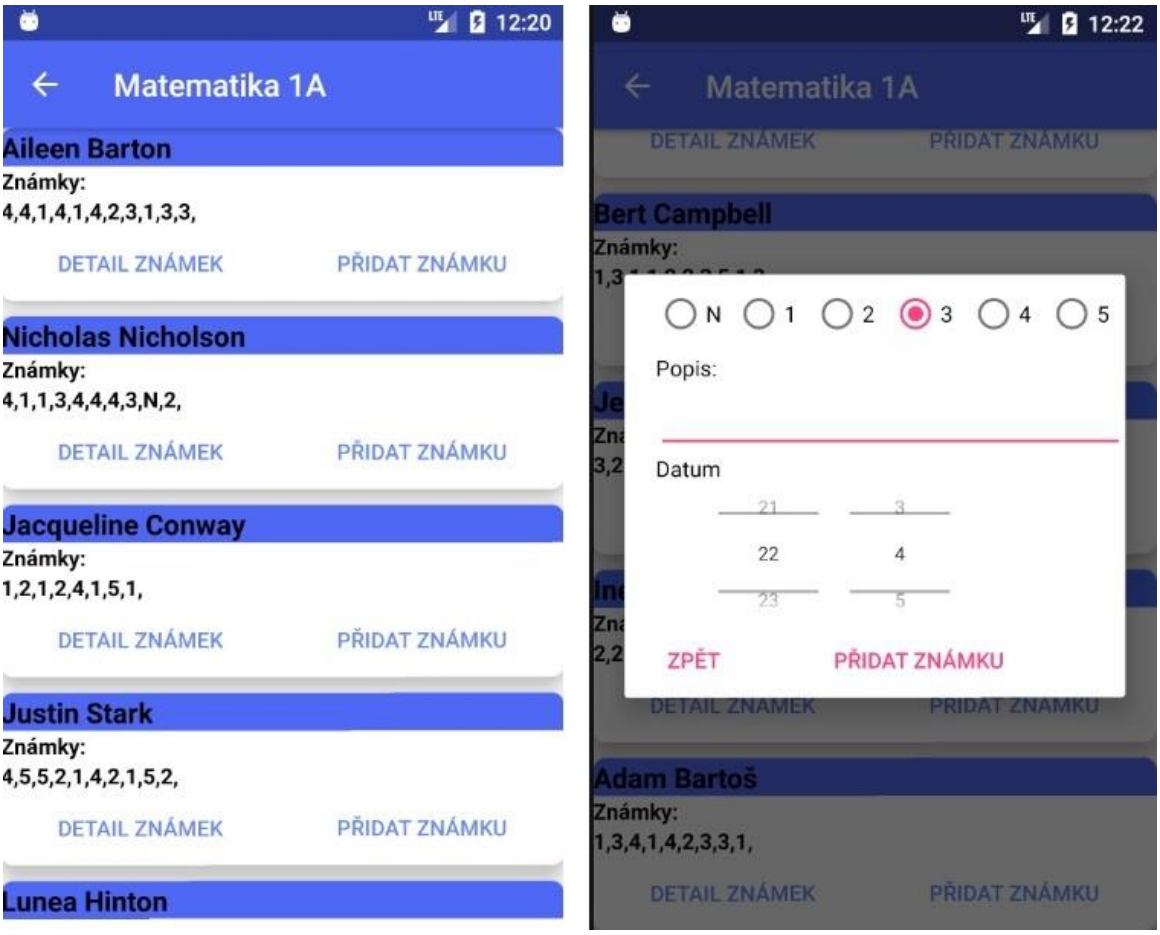
Menu obsahuje možnost zvolit si buď seznam předmětů a spravovat známky případně vybrat možnost „Moje třída“ kde je možnost spravovat aktuality a absence. Aktivita Menu je zobrazeno na Obr. 19




Obr. 19: Aktivita Menu

**Předmět**

Po vybrání předmětu ze seznamu se objeví aktivita, která obsahuje seznam žáků včetně známek z daného předmětu. U každého žáka je možnost zobrazit detailní přehled známek a také možnost přidat známku. Pro přidání známky se otevře dialogové okno, kde lze zadat známku, popis známky a datum. Aktivita a dialogové okno je znázorněno na Obr. 20.



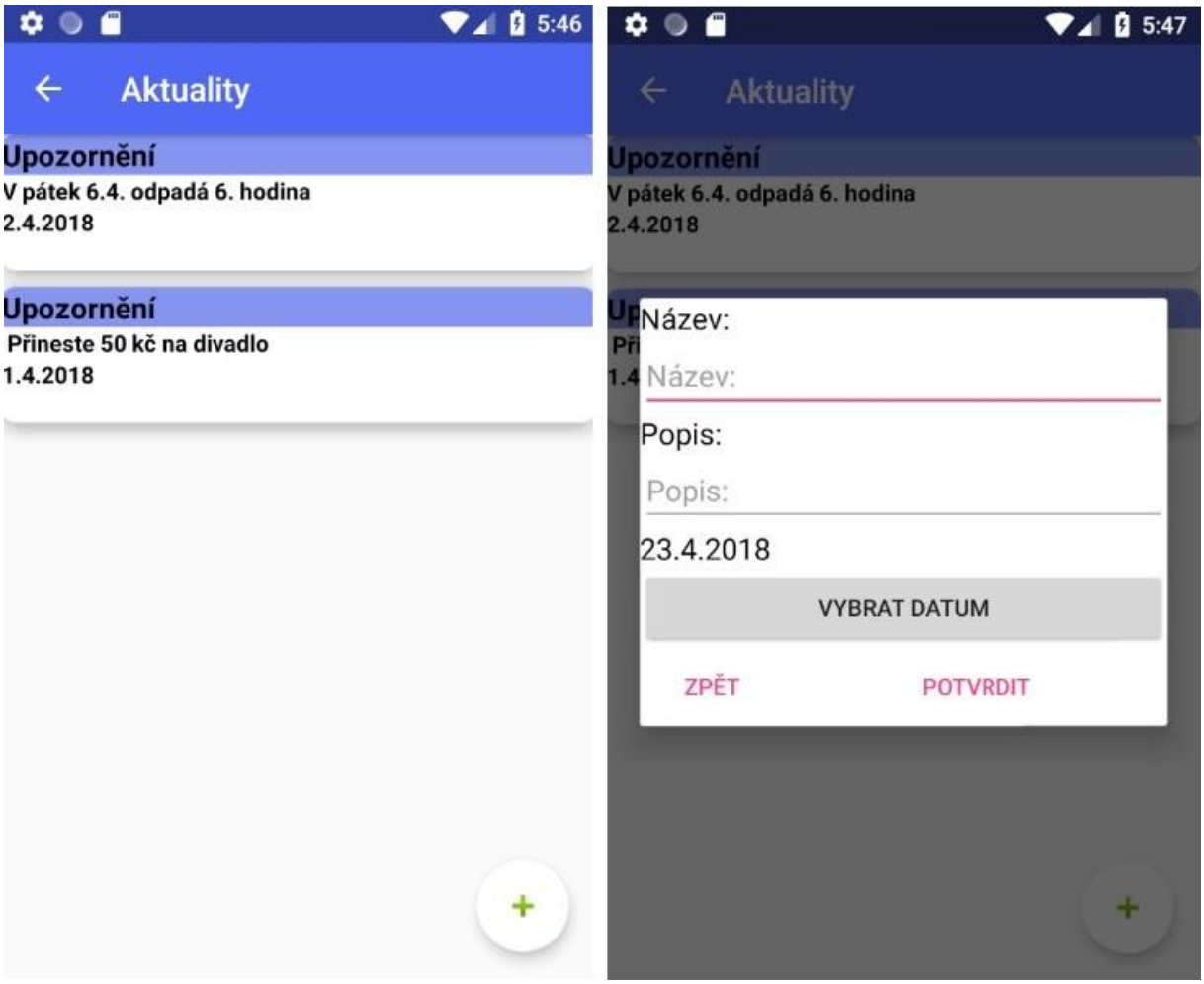

Obr. 20: Aktivita Předmět

**Detail známek**

Pokud uživatel zvolí možnost zobrazit detail známek, otevře se mu seznam známek včetně popisu a data. V této aktivitě lze také známky upravit či odstranit.

**Aktuality**

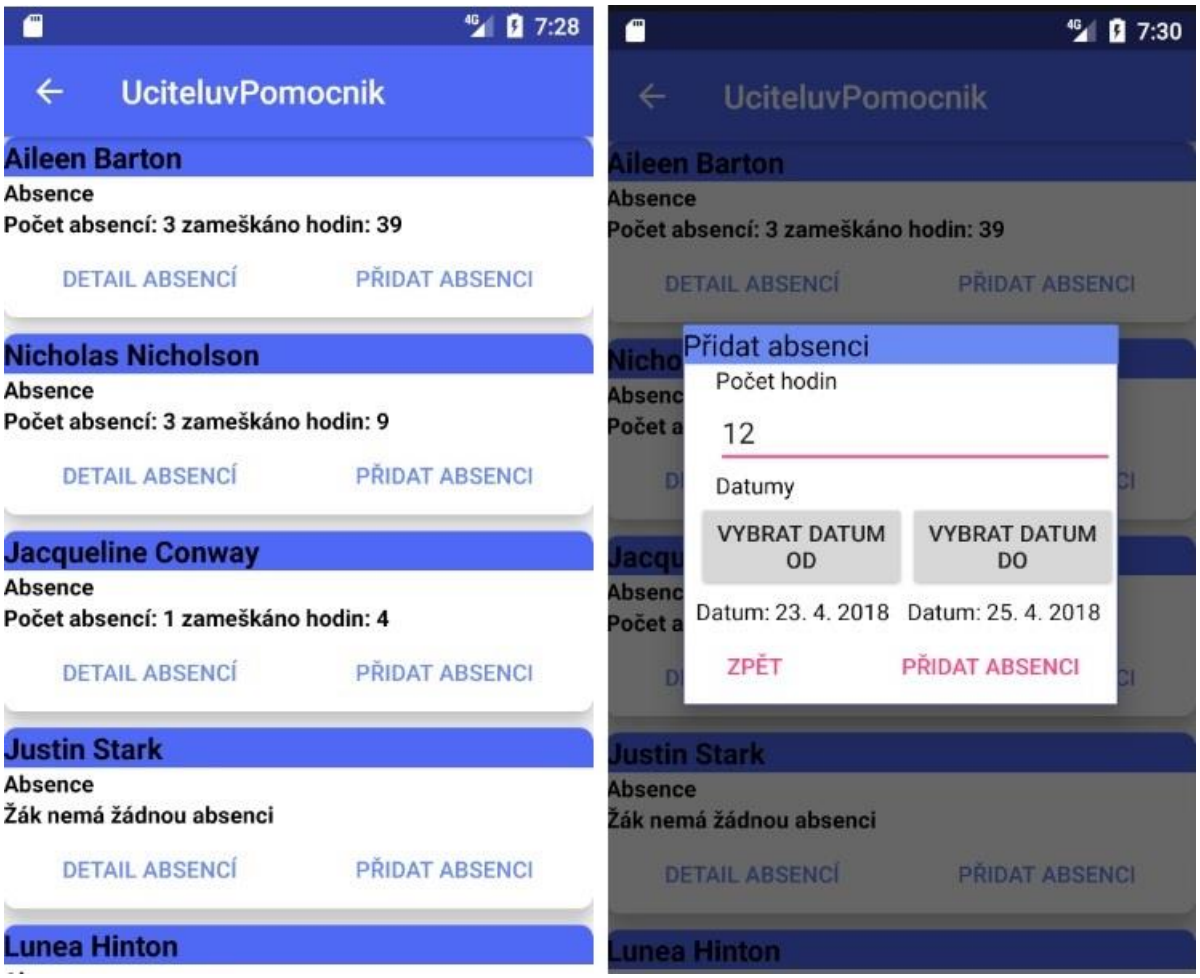
Tato aktivita obsahuje seznam aktualit třídy včetně možnosti přidání aktuality. Aktivita Aktuality je zobrazena na Obr. 21.




Obr. 21: Aktivita Aktuality

### Absence

Tato aktivita obsahuje seznam žáků společně s jejich absencemi a možností absenci přidat. Aktivita Absence je zobrazena na Obr. 22.




Obr. 22: Aktivita Absence

# 6 TVORBA WEBOVÉ APLIKACE

Webová aplikace slouží pro učitelé pro zadávání známek, aktualit, procvičování a absencí a pro žáky, kteří si mohou prostřednictvím aplikace prohlédnout své známky a absence.

## 6.1 Role

Do webové stránky se mohou přihlásit jak žáci, tak učitelé. Učitel může mít navíc administrátorské oprávnění, což mu rozšiřuje funkcionalitu o správu učitelů, správu žáku, správu předmětů a správu tříd. Po přihlášení se přihlašujícímu nastaví jedna ze tří rolí. Role popisuje následující use case diagram. V databázi pak jednotlivé role představují číslo pro roli admin číslo 1 pro roli učitel číslo 0 a pro žáka pak číslo 2. Role jsou znázorněny za pomoci Use Case diagramu na Obr. 23.

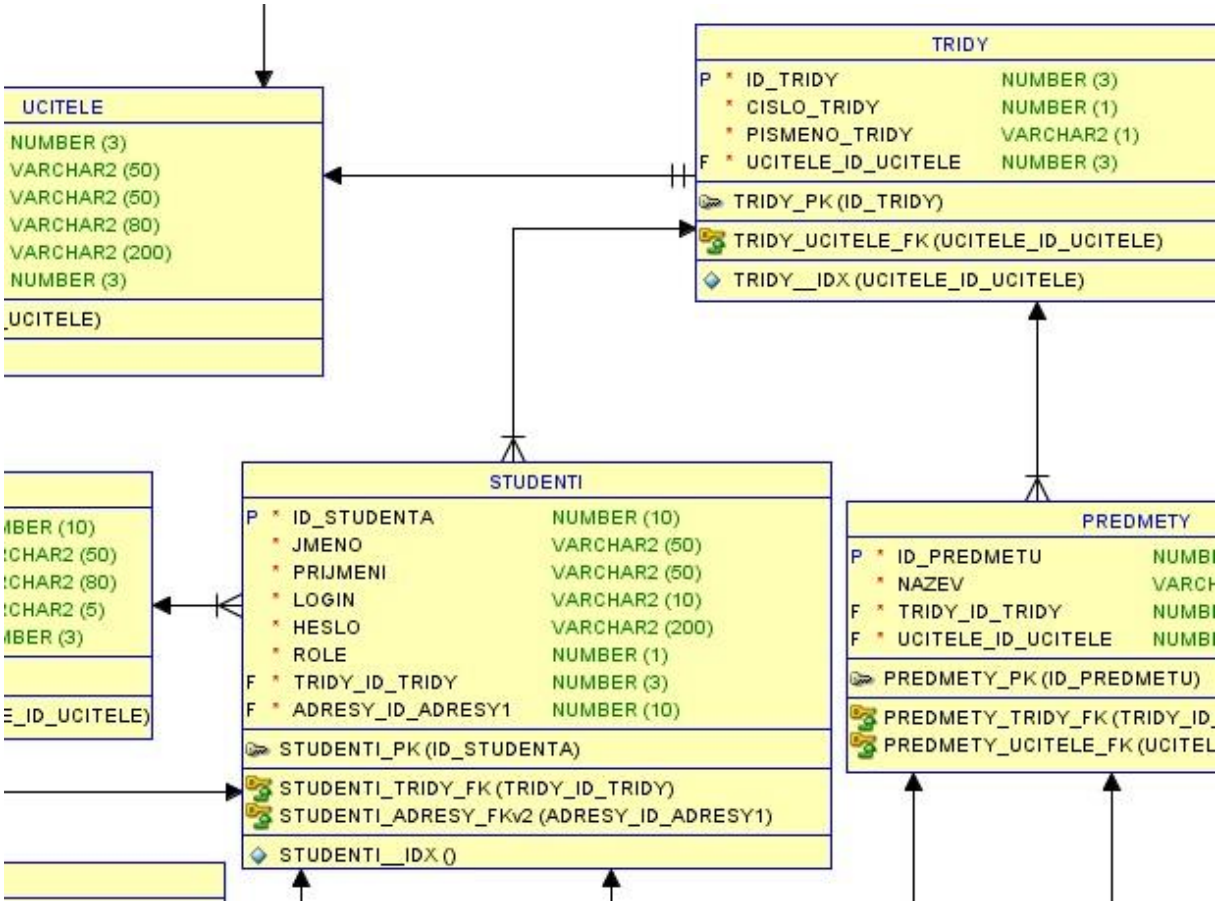
Obr. 23: Use Case Diagram pro role

## 6.2 Návrh databáze

Pro uchování dat byla vytvořena databáze, která obsahuje následující tabulky:

- Třídy – Tabulka reprezentující třídu obsahuje číslo a písmeno pro označení třídy.
- Učitelé – Tabulka obsahuje informace u učiteli včetně přihlašovacích údajů a rolí.
- Žáci – Podobně jako tabulka Učitelé namísto emailu obsahuje přihlašovací zkratku.
- Předměty – Reprezentuje daný předmět je vázán na třídu a učitele, který předmět vyučuje.
- Známky – V tabulce jsou uchovány známky žáků. Tabulka je vázaná na předmět a na žáka.
- Absence – Tabulka obsahuje absence žáka.
- Aktuality – Tabulka obsahuje aktuality obecné nebo ty, které patří jednotlivému předmětu. Je tedy vázaná jak na předmět, tak na třídu.
- Adresy – Obsahuje adresy žáků a učitelů.
- Procvičování – Obsahuje název a popis procvičování.
- Otázky – Vázány na procvičování.
- Výsledky – Výsledky žáků, které získali z procvičování.

Fragment datového modelu je zobrazen na Obr. 24 celý model je pak v příloze A




Obr. 24: Logický model databáze

**Spouště**

Pro změny, ať již úpravu či smazání jednotlivé položky, jsou vytvořeny spouště. Pokud dojde ke smazání či upravení, spoušť zajistí uložení entity do dané tabulky. Příklad spouště:

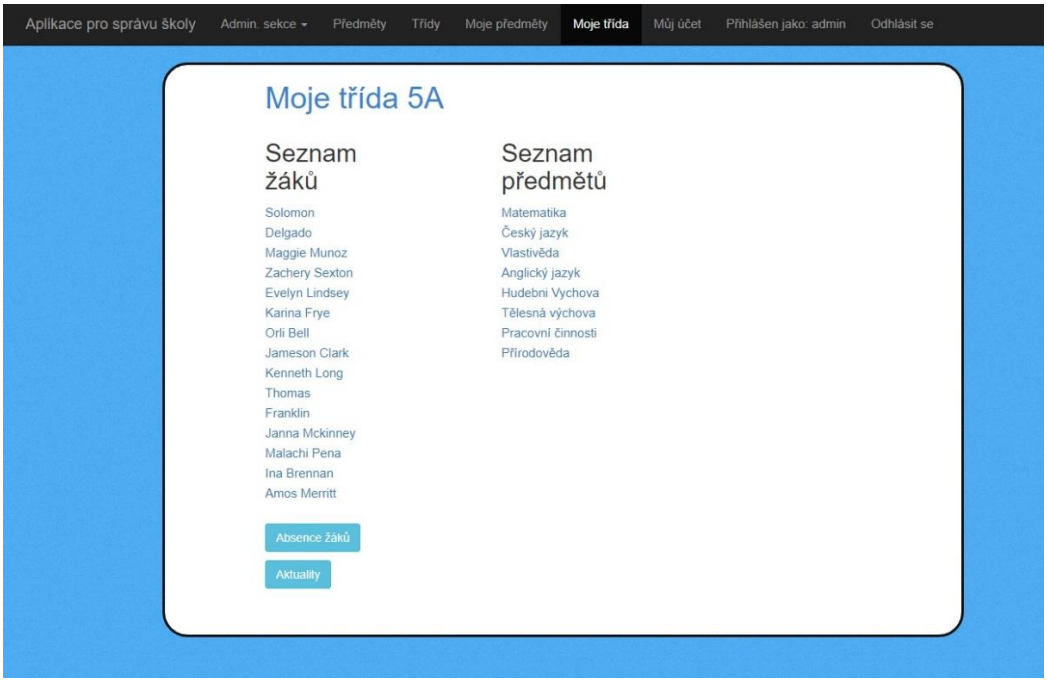
```
CREATE TRIGGER trig_znamka_delete
AFTER DELETE ON ZNAMKY FOR EACH ROW
BEGIN
  INSERT INTO znamky_del
  ( id_znamky,znamka,popis_znamky,datum)
  VALUES
   ( OLD.id_znamky,OLD.znamka,OLD.popis_znamky,OLD.datum);
END
```

## 6.3 Návrh rozložení stránky

Stránka se skládá z navigačního menu, které bylo umístěno na záhlaví stránky. Menu obsahuje následující položky:

- Admin. sekce
- Předměty
- Třídy
- Moje předměty
- Můj učet
- Odhlášení

Rozložení stránky je zobrazeno na Obr. 25 Webová stránka




Obr. 25 Webová stránka

## 6.4 Administrátorská sekce

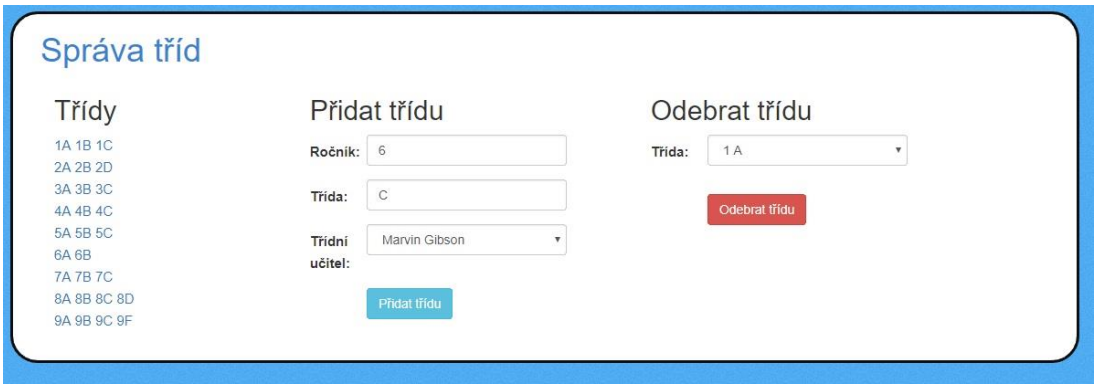
Administrátorská sekce slouží pro vytvoření žáků, učitelů, tříd a předmětů. Prostřednictvím sekce je také možné spravovat jednotlivá hesla. Ověření oprávnění se provádí v presenteru, který spravuje administrátorskou sekci. V presenteru v metodě startup (tedy v metodě která se volá jako první) se ověří, zdali je uživatel dané role. Pokud nemá, přesměruje se na domovskou stránku, kde uživateli oznámí, že nemá dostatečná oprávnění, případně na přihlašovací stránku, pokud uživatel není přihlášen. Funkce je znázorněna Obr. 26.

```
protected function startup()
{
parent::startup();

if(!$this->getUser()->isLoggedIn())
    {
        $this->flashMessage( message: 'Nejste přihlášen');
        $this->redirect( code: 'Sign:in');
    }

if(!$this->getUser()->isInRole( role: 1))
    {
        $this->flashMessage( message: 'Nemáte administrátorská oprávnění');
        $this->redirect( code: 'Homepage:default');
    }
}
```

Obr. 26: Ověření oprávnění pro přístup do admin. sekce

### 6.4.1 Správa tříd

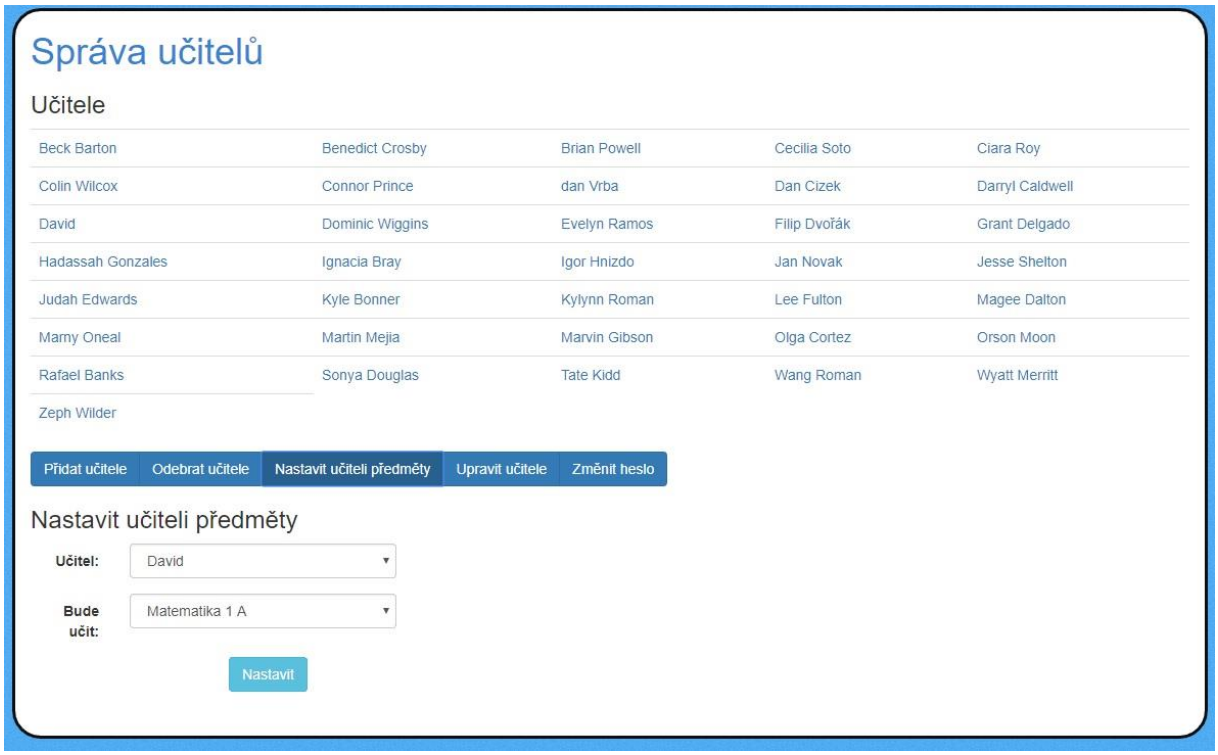
Správa tříd slouží pro vytvoření či odebrání třídy. Při vytvoření je také nutné vybrat učitele pro danou třídu. Správa tříd je znázorněna na Obr. 27.




Obr. 27: Správa tříd

### 6.4.2 Správa učitelů

Správa učitelů umožňuje administrátorovi přidat, odebrat či upravit učitele. Nastavit učiteli předměty a změnit jeho heslo. Správa učitelů je znázorněna na Obr. 28.




Obr. 28: Správa učitelů

### 6.4.3 Správa žáků

Správa žáků slouží podobně jako správa učitelů pro přidání, odebrání či úpravu žáků. Správa žáků je znázorněna na Obr. 29: Správa žáků

### 6.4.4 Správa předmětů

Správa předmětu slouží pro přidání či odebrání předmětu. Správa předmětů je zobrazena na Obr. 30.

## 6.5 Učitelská sekce

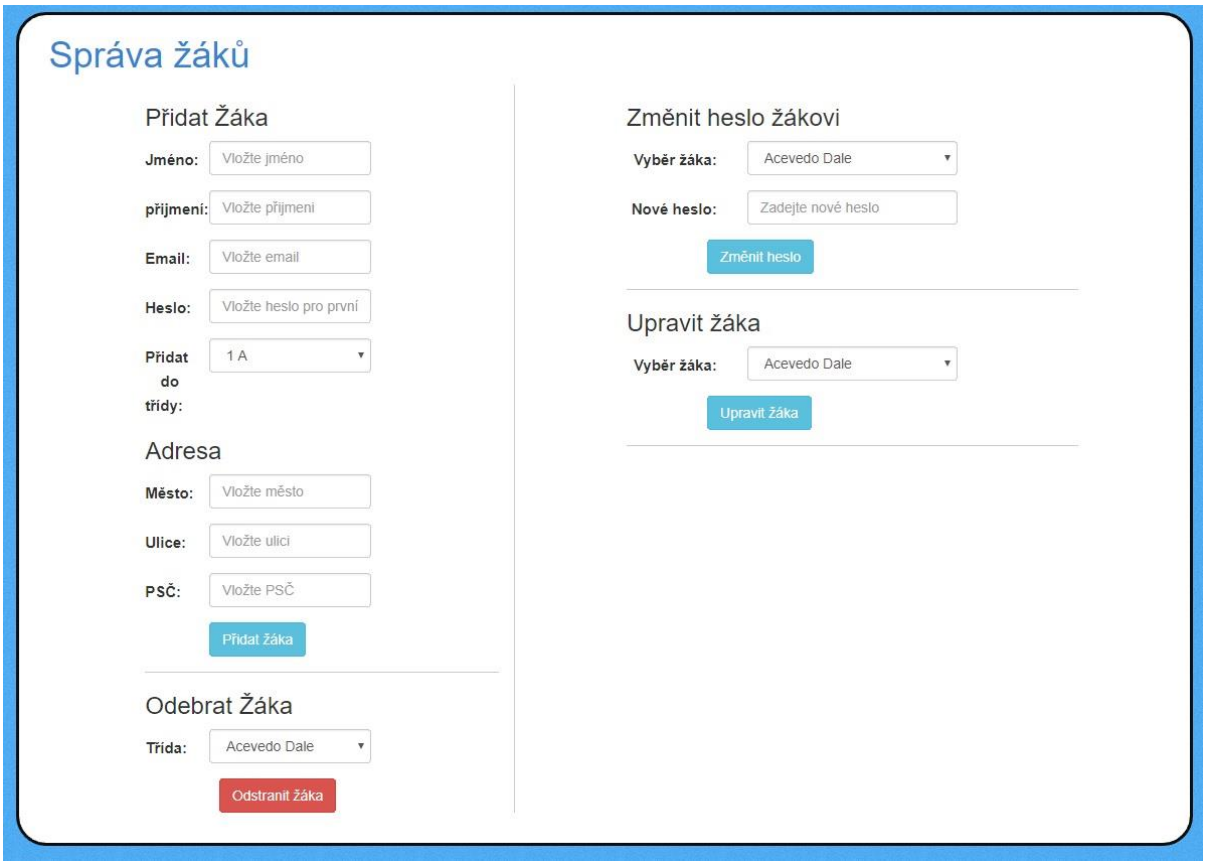
Učitelská sekce představuje nejvyužívanější sekci ve webové aplikaci. Učitel zde nalezne předměty, které učí, případně pokud je třídní tak i svou třídu.

### 6.5.1 Moje Předměty

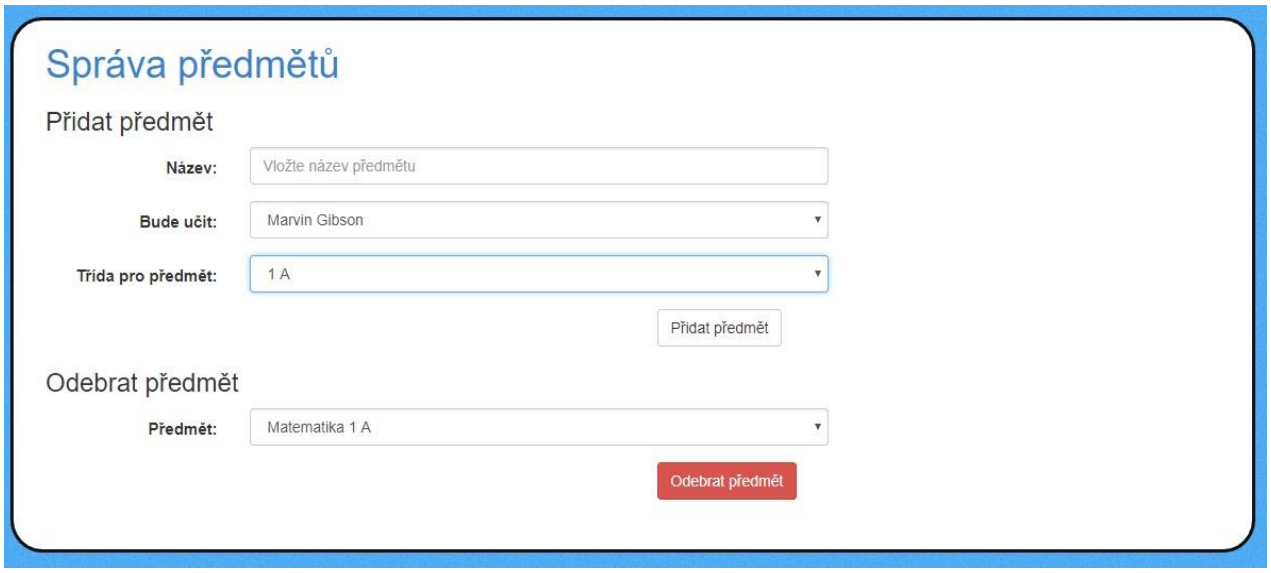
Po přístup do této sekce se učiteli zobrazí seznam předmětů, které učí. Po vybrání předmětu se zobrazí stránka se seznamem žáků včetně s jejich známky. Učitel zde má možnost přidat známku. Pro zjednodušení přidávání více známek ze stejné látky zde bylo umístěno pole

s titulkem popis všem. Po vyplnění tohoto pole se vyplní popisky u všech jednotlivých žáků a po přidání známky (odeslání formuláře) je tento popis zachován. Přidávání známek je zobrazeno na Obr. 31. Po přidání se zobrazí zpráva, která informuje učitele o úspěšném přidání známky. Pokud by chtěl učitel zjistit informace o známce případně tuto známku upravit či odstranit provede tak kliknutím na známku, načež se mu otevře dialogové okno s informacemi a možnostmi úpravy. Dialogové okno známky je zobrazeno na

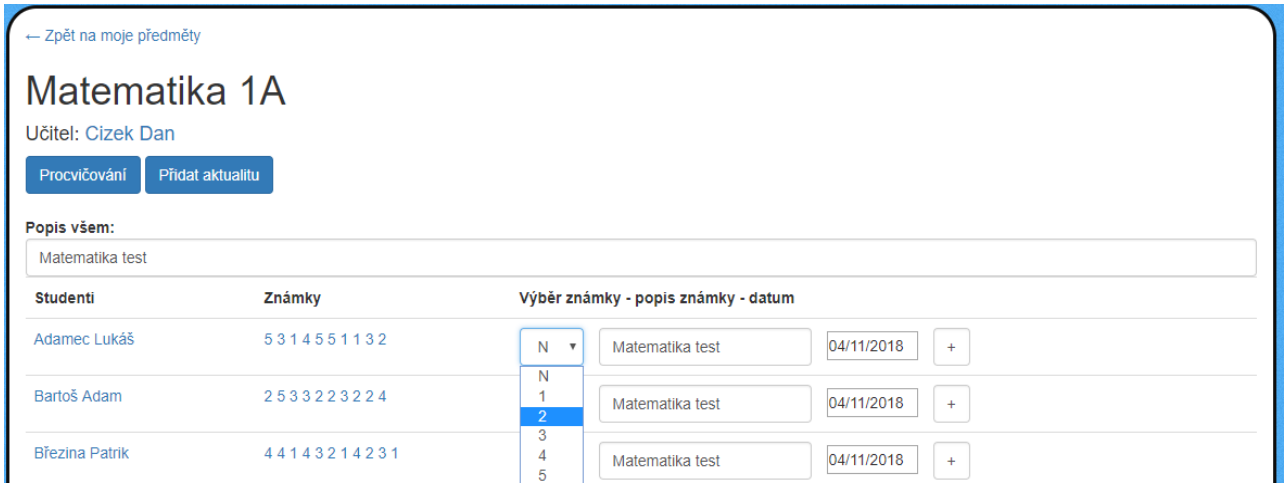
Obr. 32.



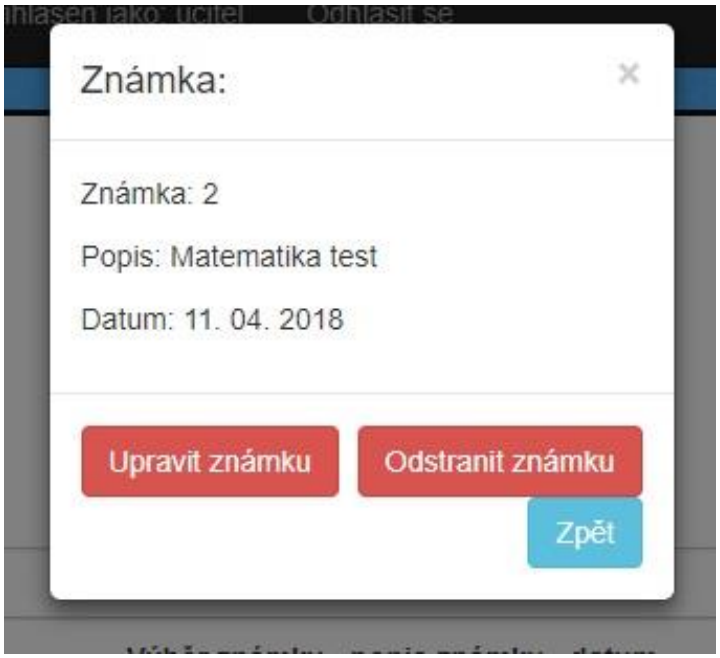

Obr. 29: Správa žáků




Obr. 30: Správa předmětů

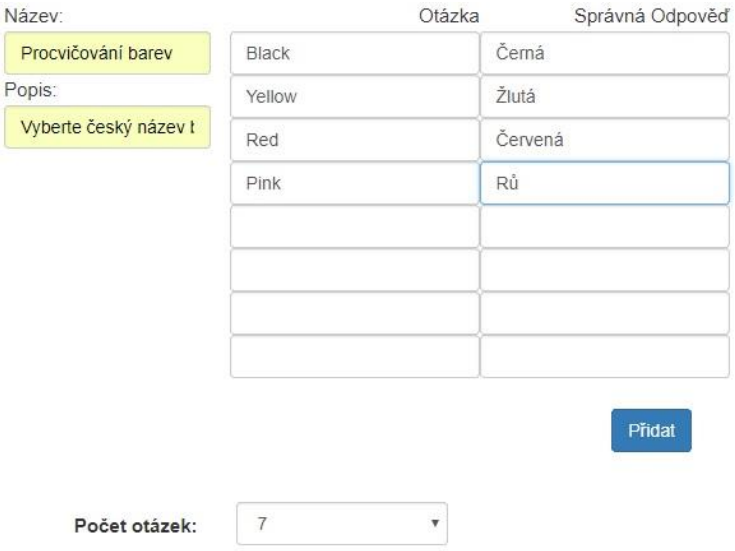



Obr. 31: Přidávání známek




Obr. 32: Dialogové okno známky

Sekce moje předměty také umožňuje přidat procvičování do aplikace pro studenty, či přidat aktualitu. Pokud učitel chce přidat procvičování tak vybere počet otázek (minimum je 5) a do dynamicky vygenerovaných polí doplní otázka a správnou odpověď. Přidání procvičování je zobrazeno na Obr. 33.




Obr. 33: Přidání procvičování

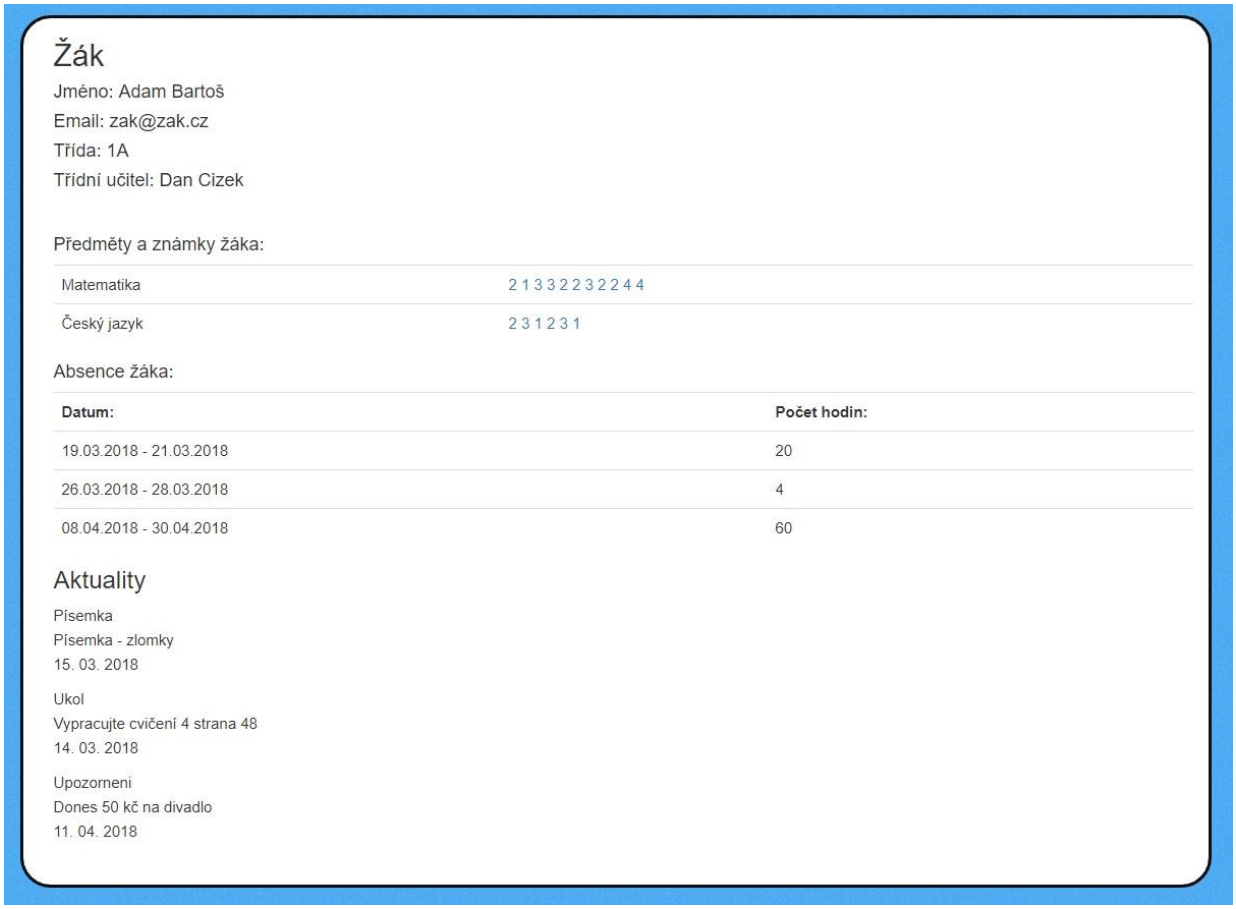
### 6.5.2 Moje třída

Jak již bylo zmíněno, záložka moje třída je pro učitele, jež jsou učiteli třídními. Tato sekce obsahuje seznam žáků s možností zobrazit detail žáka a také obsahuje seznam předmětů.

Sekce moje třída také umožňuje přidávat žákům absence. Přidání absence probíhá tak, že se u vybraného žáka vybere datum od kdy chyběl a datum do kdy chyběl včetně počtu hodin které zameškal. Stejně jako v předchozí sekci je možno přidat aktualitu.

## 6.6 Žákova sekce

Žák má možnost přihlásit se do webové aplikace, přičemž má přístup k datu stejný jako v mobilní aplikaci. Ukázka žákovy sekce je na Obr. 34.




Obr. 34: Žákova sekce

## 6.7 Odesílání dat mobilním aplikacím

Odesílání dat probíhá pomocí metod, na které se aplikace odkazují. Typicky daná metoda přijme od uživatele identifikátor, pomocí kterého se rozhodne, jaká data mobilní aplikace odeslat. Například pro odeslání známek žáka slouží metoda getZnamkyData(). Průběh metody vypadá následovně:

1. Provede se načtení žádosti metodou `getHttpRequest()`.
2. Z žádosti se zjistí parametr id (id žáka) metodou `getPost()`.
3. Zjištěný parametr id se použije na dotaz pro databázi, který vybere všechny známky, které náleží danému žáku.

4. Získá se objekt typu `Selection` ze kterého se pomocí cyklu získají objekty typu `ActiveRow`.
5. Objekt `ActiveRow` pak představuje řádek tabulky tedy jednu známku.
6. Známky se uloží do pole a odešlou se jako odpověd typu JSON za pomoci metody `sendJSON()`.

Metoda handleGetznamkydata() je znázorněna na Obr. 35

```
public function handleGetznamkydata()
{
    $httpRequest = $this->getHttpRequest();

    $id_studenta = $httpRequest->getPost( key: 'id_zaka');
    $id_posledni_znamky = $httpRequest->getPost( key: 'id_znamky');

    $znamky = $this->database->table( table: 'znamky')
        ->where( condition: 'id_studenta', $id_studenta)
        ->where( condition: 'id_znamky >', $id_posledni_znamky);

    $znamkyJSONarray = array();

    $znamka = array();
    foreach ($znamky as $row) {
        $znamka["id"] = $row->id_znamky;
        $znamka["Znamka"] = $row->znamka;
        $znamka["Popis"] = $row->popis_znamky;
        $znamka["Datum"] = $row->datum->format('d.m.Y');
        $znamka["id_predmetu"] = $row->id_predmetu;
        $znamkyJSONarray[] = $znamka;
    }
    $this->sendJson($znamkyJSONarray);
}
```

Obr. 35: metoda getznamkydata()

# ZÁVĚR

Hlavním cílem práce bylo vytvořit mobilní aplikaci pro žáky, kde by měli možnost sledovat své známky, absence a aktuality k předmětu či třídě. Současně bylo cílem vytvořit i webovou stránku pro učitele, která by umožňovala přidávat známky, aktuality a absence do systému. V průběhu byla práce doplněna o mobilní aplikaci pro učitele, která nabízí část funkcionality jako webová stránka.

Praktická část se zabývala tvorbou jednotlivých aplikací. Nejprve byla popsána aplikace pro žáky, která začínala návrhem aplikace a specifikací požadavků. Následně byla popsána implementace jednotlivých aktivit a jejich popis. Mobilní aplikace pro učitele převzala značnou část kódu od aplikace pro žáky, a to zejména grafické rozhraní. Přidanou hodnotou této aplikace je možnost přidávání známek, absence a aktualit. Webová aplikace byla postavena na frameworku Nette. Framework Nette, díky své funkcionalitě, jako je snadné tvoření formulářů, či dotazování do databáze, práci na webové stránce velmi ulehčil.

Pro získávání dat ze serveru byl nejdříve vyzkoušen přístup, že aplikace ihned po spuštění žádala po serveru všechna data, která měla zobrazovat. Tento přístup měl však za následek dlouhé načítání, čímž docházelo ke zhoršení uživatelské zkušenosti s aplikací. Tato strategie byla změněna tak, že mobilní aplikace stále uchovává data a při přihlášení požádá server jen o data, která se změnila od posledního přihlášení.

# POUŽITÁ LITERATURA


[1] Soukromá střední škola výpočetní techniky. Bakaláři – mobilní aplikace pro studenty a rodiče [online]. Praha, 2018 [cit. 2018-05-03]. Dostupné z: https://www.sssvt.cz/IS/mobilni-bakalari/

[2] Vyzkoušejte novou verzi mobilní aplikace Školy OnLine. *Škola OnLine* [online]. Praha, 2017 [cit. 2018-05-03]. Dostupné z: https://www.skolaonline.cz/Aktuality.aspx

[3] LACKO, Ľuboslav. *Mistrovství – Android: Kompletní průvodce vývojáře*. Přeložil Martin HERODEK. Brno: Computer Press, 2017. Mistrovství. ISBN 978-80-251-4875-4.

[4] GRUDL, David. Nette Framework: zvyšte svoji produktivitu. In: Zdroják [online]. 2009 [cit. 2018-04-25]. Dostupné z: https://www.zdrojak.cz/clanky/nette-framework-zvyste-svoji-produktivitu/

[5] ČÁPKA, David. Popis MVC architektury. In: *ITnetwork.cz* [online]. 2012 [cit. 2018-05-03]. Dostupné z: https://www.itnetwork.cz/php/mvc/objektovy-mvc-redakcni-system-v-php-popis-architektury

[6] MVC aplikace & presentery. Nette [online]. [cit. 2018-05-03]. Dostupné z: https://doc.nette.org/cs/2.4/presenters

[7] JANOVSKÝ, Dušen. Úvod do JavaScriptu. *Jak psát web* [online]. [cit. 2018-05-03]. Dostupné z: https://www.jakpsatweb.cz/javascript/javascript-uvod.html

[8] CHAFFER, Jonathan a Karl SWEDBERG. *Mistrovství v jQuery: [kompletní průvodce vývojáře]*. Brno: Computer Press, 2013. Mistrovství. ISBN 978-80-251-4103-8.

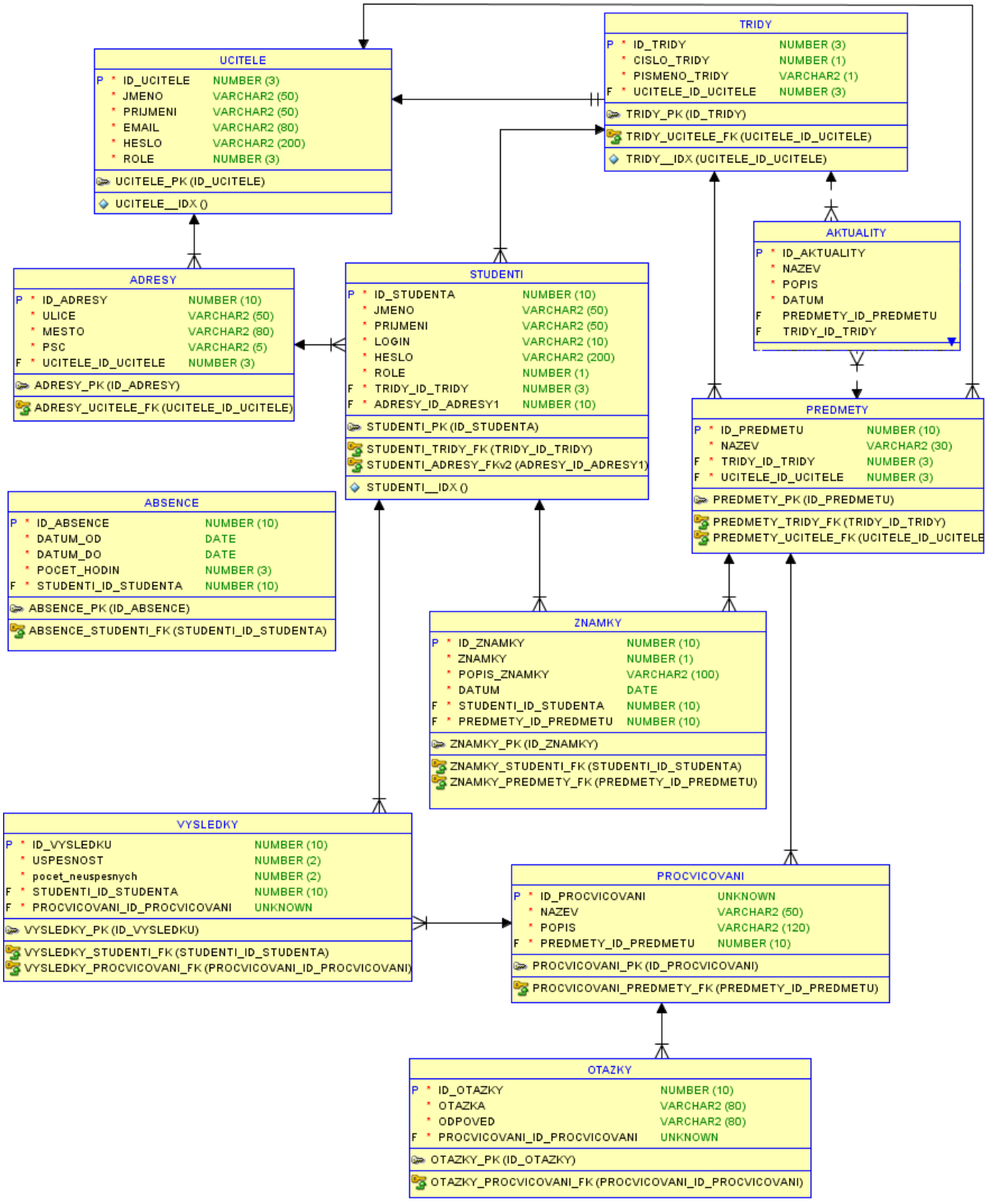
[9] HERNANDEZ, Michael J. *Návrh databází*. Praha: Grada, 2006. Profesionál. ISBN 80-247-0900-7.

[10] Database Explorer. *Nette* [online]. 2018 [cit. 2018-04-27]. Dostupné z: https://doc.nette.org/cs/2.4/database-explorer

[11] JANOVSKÝ, Dušen. HTTP protokol. *Jak psát web* [online]. [cit. 2018-05-03]. Dostupné z: https://www.jakpsatweb.cz/server/http-protokol.html

[12] JSON vs XML. *W3schools.com* [online]. [cit. 2018-05-03]. Dostupné z: https://www.w3schools.com/js/js_json_xml.asp

# PŘÍLOHY

Příloha A – Datový model .....57

# PŘÍLOHA A – DATOVÝ MODEL