ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Kožár** Jméno: **Slavomír** Osobní číslo: **406511**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačů**

Studijní program: **Otevřená informatika**

Studijní obor: **Softwarové inženýrství**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Systém pro podporu výuky programování

Název diplomové práce anglicky:

System to support teaching of programming

Pokyny pro vypracování:

Cílem práce je vytvoření systému pro podporu výuky programování na základních a středních školách. Součástí návrhu systému bude i analýza existujících systémů pro odevzdávání a vyhodnocovaní řešení algoritmických úloh pro účely výuky a soutěží.
Systém bude sestávat z těchto hlavních částí:
- správa uživatelů a přiřazování uživatelů do skupin,
- tvorba a úprava zadání úloh včetně vstupních a výstupních dat,
- odevzdávání a vyhodnocování vyřešených úloh.

Specifikujte a analyzujte konkrétní funkční a nefunkční požadavky na systém. Na základě těchto požadavků navrhněte aplikaci. Pro analýzu a návrh použijte vhodné prostředky softwarového inženýrství. Navržený program implementujte a výslednou aplikaci otestujte včetně testů použitelnosti.

Seznam doporučené literatury:

Ian Sommerville: Software Engineering, Global Edition, Pearson Higher Ed, 2016, ISBN1292096144
Arlow, J., Neustat, I.: UML 2 a unifikovaný proces vývoje aplikací. Computer Press, 2007.

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Božena Mannová, Ph.D., Software Engineering and Networking FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **16.02.2018** Termín odevzdání diplomové práce: _____

Platnost zadání diplomové práce: **30.09.2019**

Ing. Božena Mannová, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Ing. Pavel Ripka, CSc. podpis děkana(ky)

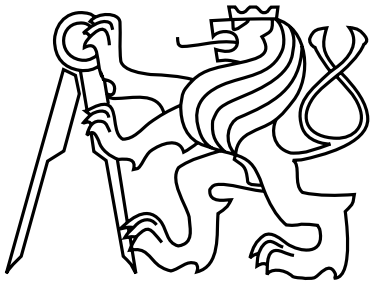
III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

20.4.2018
Datum převzetí zadání

Podpis studenta

České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra počítačov



Diplomová práca

Systém pre podporu výuky programovania

Bc. Slavomír Kožár

Vedúci práce: Ing. Božena Mannová, Ph.D.

Študijný program: Otevřená informatika, Magisterský

Odbor: Softwarové inženýrství

25. mája 2018

Poďakovanie

Na tomto mieste by som chcel poďakovať ľuďom, bez ktorých by táto práca nemohla vzniknúť.

V prvom rade, sú to moji rodičia, ktorí mi umožnili štúdium na vysokej škole. Najbližšej rodine (predovšetkým rodičom, sestre a jej rodine) za podporu pri zvládaní nástrah tohto štúdia.

Chcem sa poďakovať svojej priateľke za trpezlivosť a zhovievavosť počas ťažkých chvíľ pri štúdiu aj realizácii tejto práce.

Veľká vďaka patrí vedúcej mojej práce, Ing. Božene Mannovej, Ph.D, za odvahu pustiť sa so mnou do tejto práce.

V neposlednom rade sa chcem poďakovať mojim kolegom a všetkým mojím známym, za trpezlivosť, ktorú so mnou mali počas realizácie tejto práce.

Prehlásenie

Prehlasujem, že som predloženú prácu vypracoval samostatne a že som uviedol všetky použité informačné zdroje v súlade s Metodickým pokynom o dodržiavaní etických princípov pri príprave vysokoškolských záverečných prác.

V Prahe 24. 5. 2018 .....

Abstract

The diploma thesis contains an application for the support of teaching at primary and secondary schools. The application is designed to manage users, organize them into groups, share student materials with this application, and create programming assignments for students„ uploading and evaluating these solutions.

In the framework of the solution, the author developed an analysis of such an application, designed the architecture and the data model. The proposed application was implemented by the Software as a Service - Web Application. The described application should provide schools with a tool to increase the efficiency of teaching programming and, last but not least, make programming lessons more attractive to school pupils.

Keywords: support, programming, education, notifications, teacher, student, groups, assignment, submitting, testing, evaluation

Abstrakt

Diplomová práca obsahuje aplikáciu pre podporu vyučovania programovania na základných a stredných školách. Aplikácia je určená pre správu užívateľov, ich organizáciu do skupín, zdieľanie študijných materiálov žiakom pomocou tejto aplikácie a vytváranie programátorskych zadaní pre žiakov, ich odovzdávanie a hodnotenie týchto riešení.

V rámci riešenia autor vypracoval analýzu takejto aplikácie, navrhol architektúru a dátový model. Navrhnutá aplikácia bola implementovaná modelom Softvér ako služba - formou webovej aplikácie. Popisovaná aplikácia by mala priniesť školám nástroj pre zvýšenie efektivity vyučovania programovania a v neposlednom rade učiniť výučbu programovania viac atraktívnou pre žiakov na školách.

Kľúčové slová: podpora, programovanie, výuka, oznámenia, učiteľ, žiak, skupiny, zadanie, odovzdávanie, testovanie, hodnotenie

Zoznam obrázkov

2.1 Logo aplikácie Moodle . . . 5
2.2 Ukážka práce s aplikáciou Moodle . . . 6
2.3 Ukážka práce s aplikáciou DOMJudge . . . 7
2.4 Sociálne siete . . . 8
2.5 Sociálne siete . . . 10

3.1 MVC Architektúra . . . 25
3.2 Viacvrstvá Architektúra . . . 26
3.3 Väzba typu 1 : N . . . 27
3.4 Väzba typu N : M . . . 28
3.5 Dátový model aplikácie . . . 29
3.6 Model užívateľov aplikácie . . . 30
3.7 Model entity Odkaz . . . 32
3.8 Model entity Súbor . . . 33
3.9 Model entity Článok . . . 34
3.10 Model entity Obrázok . . . 35
3.11 Model entity Príloha . . . 36
3.12 Model entity Zadanie . . . 37
3.13 Model entity Riešenie . . . 38
3.14 Model entity Príloha . . . 40
3.15 Model entity Príloha . . . 41
3.16 Farebná schéma aplikácie . . . 42
3.17 Rozloženie aplikácie . . . 43
3.18 Hlavný obsah aplikácie . . . 44

4.1 Migrácie vytvorené pri impelemntácii aplikácie . . . 49
4.2 Model použitia Eloquent ORM . . . 49
4.3 Model middleware v rámci Laravel . . . 50
4.4 Zobrazenie formulára bez chybových hlásení . . . 52
4.5 Zobrazenie formulára s chybových hláseniami . . . 52
4.6 Zobrazenie hlavnej stránky na displeji mobilného zariadenia . . . 54
4.7 Zobrazenie hlavnej stránky na monitore počítača . . . 55
4.8 Zobrazenie článku . . . 56
4.9 Úprava článku . . . 56
4.10 Simplemde editor . . . 57

4.11 Bootstrap datepicker 58
4.12 jQuery File Upload 58

Kapitola 1

Úvod

Programovanie by v 21. storočí malo mať v školách podobné postavenie ako matematika v tom minulom. Toto je jeden z motívov svetovej kampane, pod ktorú sa podpísali osobnosti ako zakladateľ Microsoftu Bill Gates, zakladateľ Facebooku Mark Zuckerberg a ďalší [13].

Každý by mal vedieť programovať počítač, pretože Vás to naučí rozmýšlať (Steve Jobs). Technológie a špeciálne programovanie sa rozvíja najrýchlejšie zo všetkých odvetví. Bohužiaľ školstvo na tieto trendy nestíha reagovať. [18].

1.1 Motivácia a popis problému

Autor práce vidí hlavne nedostatky vo vyučovaní programovania na školách v niekoľkých oblastiach. Veľkým problémom vyučovania programovania, alebo obecne informačných technológii je samotný obsah tohto vyučovania. To sa odklonilo od vyučovania algoritmizácie a programovania k vyučovaniu práce s komerčnými aplikáciami [12].

Autor práce sa domnieva, že jeden zo spôsobov, ako vrátiť programovanie a algoritmizáciu do vyučovacieho procesu, je poskytnúť školám podporu a nástroje, podobne ako učinili autori vyššie spomínaných komerčných aplikácii.

Obsah vyučovania, je však len jedným z problémov.

Časová dotácia pre vyučovanie programovania nie je dostatočná. Na Väčšine stredných škôl je vyučovaniu programovania venovaná jedna dvojhodinová lekcia týždenne. Autor práce z vlastnej skúsenosti vie, že 90 minút je málo. Zvlášť ak sú zaradené v bloku vyučovania s humanitnými, alebo spoločenskými predmetmi. V takom prípade je nevyhnutný čas pre zmenu spôsobu myslenia žiakov na spôsob potrebný pre riešenie algoritmických problémov.

Z časovej dotácie, ktorá je pridelená programovaniu na školách vyplýva aj ďalší negatívny jav. Sú ním veľké prestoje medzi jednotlivými hodinami venovanými programovaniu. Bežnou praxou je lekcia programovania raz za sedem dní. Kôli týmto prestojom nastáva u žiakov problém s nadväzovaním učiva z predchádzajúcich lekcií. Tento problém je často riešený venovaním veľkej časovej porcie z hodiny opakovaniu minulého učiva, čo ukrajuje ďalší čas z už tak malej časovej dotácie.

Tento problém by mohol byť riešený zmenou koncepcie vyučovania programovania, kde by dôležitejšiu rolu prebrala samostatná práca žiakov na komplexnejších projektoch a pravidelných úlohach na precvičenie preberaného učiva.

Nemenej veľkým problémom vyučovania je využívanie nesprávnych nástrojov - programovacích jazykov, integrovaných vývojových rozhraní a podobne. V dnešnej dobe je u stredoškolákov bežná vysoká úroveň počítačovej zručnosti. Títo žiaci zároveň už kladú vysoké nároky na aplikácie a nástroje, ktoré využívajú a je náročne zaujať ich nástrojmi, ktoré nespĺňajú nároky kladené na rýchlosť, ergonómiu a kvalitné užívateľské rozhranie.

Na základe dotazníka, ktorý autor realizoval na vzorke učiteľov programovania na stredných školách vyplýva, že učitelia často nevyužívajú najmodernejšie nástroje v samotnom vyučovacom procese ani pri komunikácií so študentami.

Myšlienka vytvorenia systému ako je CodeTutor vznikla u autora tejto práce už v roku 2014. Na Strednej Priemyselnej Škole Elektrotechnickej v Prešove, ktorú autor absolvoval vznikla z iniciatívy autora spolu s jeho bývalými spolužiakmi súťaž CodeLeague. Cieľom tejto súťaže bolo dlhodobo pripravovať žiakov školy na súťaže v programovaní - súťaž ZENIT v Programovaní, Olympiáda v Informatike a ďalšie.

Autor tejto práce sa na týchto súťažiach pravidelne zúčastňoval. Po učastiach na týchto súťažiach autor práce zaznamenal značný rozdiel medzi zadaniami programátorských úloh na súťažiach a úloh, ktoré sa precvičujú a realizujú vrámci vyučovania programovania na školách. Kým zadania na súťažiach boli komplexné príklady vyžadujúce zručnosti v riešení praktických problémov, zatiaľ čo na vyučovaní sa riešia len konkrétne programátorské úlohy, väčšinou absolútne nesúvisiace s praktickými problémami. Tieto úlohy slúžia len na precvičenie jednotlivých programátorských techník, ale žiaci si pomocou nich neosvoja myslenie potrebné pre riešenie problémov.

Autor práce sa podieľal hlavne na vytvorení zadaní pre túto súťaž. Pri ich tvorbe sa značne inšpiroval zadaniami vytvorenými pre predmet Algoritmizace, ktorý absolvoval v bakalárskej etape štúdia na Fakulte Elektrotechnickej Českého Vysokého Učení Technického. Pri tvorbe tejto súťaže autor spolupracoval s Bc. Kamilom Triščíkom z Mendelovej Univerzity v Brne a Bc. Pavlom Vargovčíkom z VUT v Brne.

Počas tejto súťaže a ďalších aktivít spojených s programovaním na stredných školách prišiel autor tejto práce aj s myšlienkou na toto zadanie.

1.2 Cieľ práce

Cieľom práce je navrhnutie a vytvorenie systému pre podporu výuky programovania na základných a stredných školách. Tento systém bude realizovaný formou responzívnej webovej aplikácie. Aplikácia bude navrhnutá a implmenetovaná využitím štandardných nástrojov softwarového inžinierstva.

1.3 Obsah základných kapitol

V následujúcej kapitole analýza budú čitateľovi predstavené informácie na základe ktorých vznikalo riešenie tejto diplomovej prace.

V úvode to bude prehľad existujúcich riešení, ktoré sa zaoberajú podobnou tématikou - podporou vyučovanie, alebo programátorských súťaží.

Ďalej v tejto kapitole autor zadefinuje požiadavky, na základe ktorých bude navrhnutá a implementovaná webová aplikácia.

Doplnením požiadaviek nevyhnutným pre úspešné navrhnutie a implementáciu aplikáciue sú prípady použitia systému, vrátane popisu jednotlivých užívateľov aplikácie. [16]

Kapitola analýza je završená vykonanou SWOT analýzou, v ktorej autor popisuje silné, ale aj potencionálne slabé stránky navrhovanej aplikácie.

V kapitole návrh systému autor predstavuje navrhnutý systém pomocou vhodných prostriedkov softwarového inžinierstva.

Autor predstavuje celkovú architektúru systému. Prezentovaný je tiež koncept funkčných blokov ktorými je tvorený celý systém. Kapitola návrh systému pokračuje ilustráciou dátového modelu aplikácie. V tejto časti sú popísané jednotlivé objekty tvoriace dátový model, dáta, ktoré sú reprezentované týmito objektami a ich význam v navrhovanej aplikácii. Súčasťou tejto časti je aj popis väzieb medzi jednotlivými objektami dátového modelu.

Po predstavení dátového modelu autor pokračuje návrhom grafického rozhrania. V tejto časti sú vysvetlené základné princípy, ktoré boli využité pri návrhu grafického rozhrania, farebná schéma aplikácie a ďalšie pravidlá podľa ktorých bolo navrhnuté grafické rozhranie aplikácie.

V nasledujúcej kapitole Implementácia autor čitateľovi predstavuje samotný proces implementácie popisovanej aplikácie. V úvode tejto kapitoly autor predstavuje použité technológie a odôvodňuje ich použitie. Autor popisuje implementáciu jednotlivých funkčných blokov aplikácie a pri každom predstavuje použité technológie a samotné riešenie funkčného bloku.

Kapitola testovanie je venovaná popisu testovania, ktoré bolo súčasťou samotného procesu vývoja aplikácie. Ďalej sú v tejto kapitole popísane akceptačné testy, ktoré boli použité pre overenie funkčnosti aplikácie a taktiež testy použiteľnosti.

Kapitola 2

Analýza

2.1 Prehľad existujúcich riešení

Existuje mnoho rôznych systémov pre podporu vyučovania na školách. Tieto systémy sa vyznačujú rôznym zameraním na druh školy, prípadne na konkrétny proces vo vyučovaní. Táto práca je zameraná na podporu vyučovania programovania na základných a stredných školách.

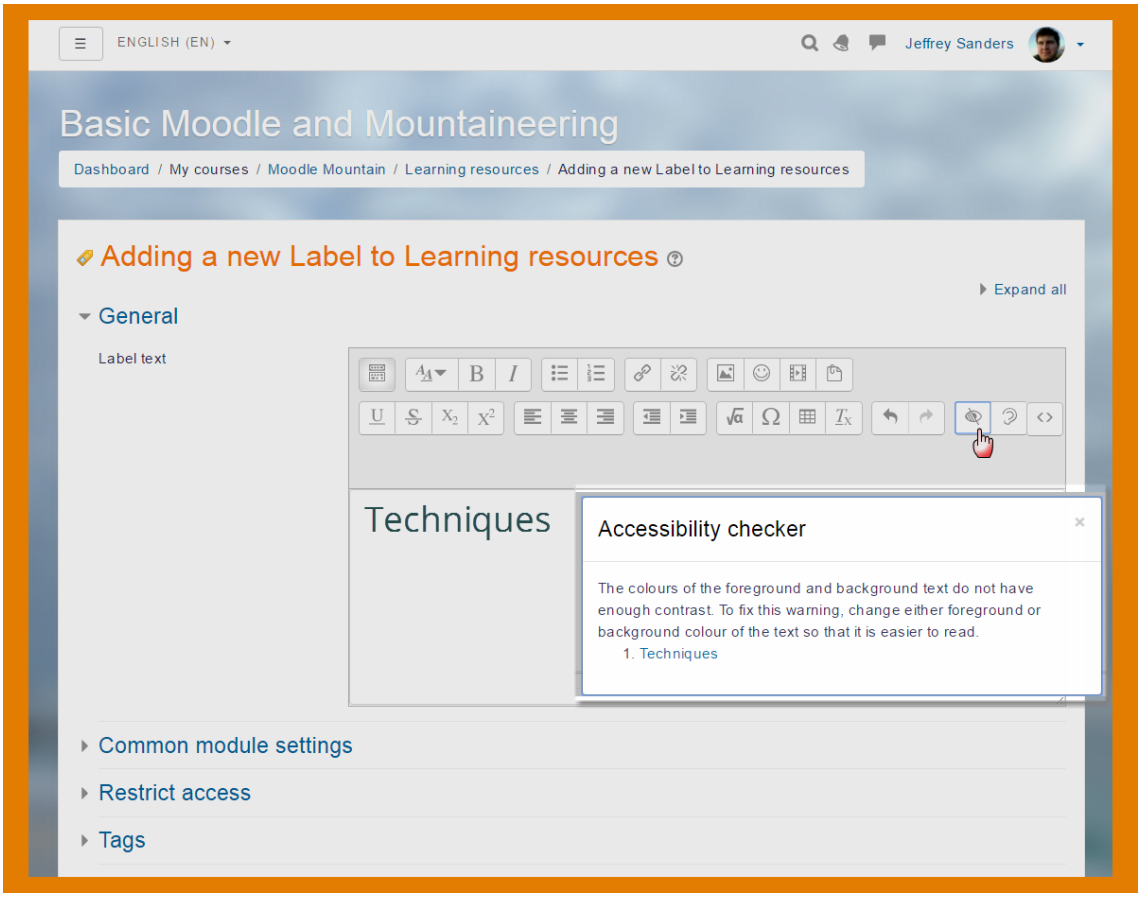
Podporu procesu programovania na základných a stredných školách je možné rozdeliť na tri základné piliere. Komunikácia učiteľov so žiakmi, zdieľanie informácií a materiálov pre žiakov a zdieľanie zadaní a odovzdávanie s automatickým vyhodnotením žiackych riešení. Z existujúcich riešení autor práce vybral následovné, kvôli ich rozšírenosti. Aplikácie, ktoré budú spomenuté, riešia jeden z vyššie uvedených pilierov.

2.1.1 Moodle

Obr. 2.1: Logo aplikácie Moodle

Moodle je najväčšie riešenie s otvoreným kódom vyvíjané jednou z najväčších komunít na svete. Táto platforma je jednou z najpoužívanejších riešení pre e-learning, teda zdieľanie

materiálov a informácii žiakom. Okrem iných funkcií poskytuje Moodle možnosti pre spravovanie užívateľov v skupinách, vytváranie a správu kurzov, zdieľanie súborov v kurzoch a testy pre užívateľov v kurze.



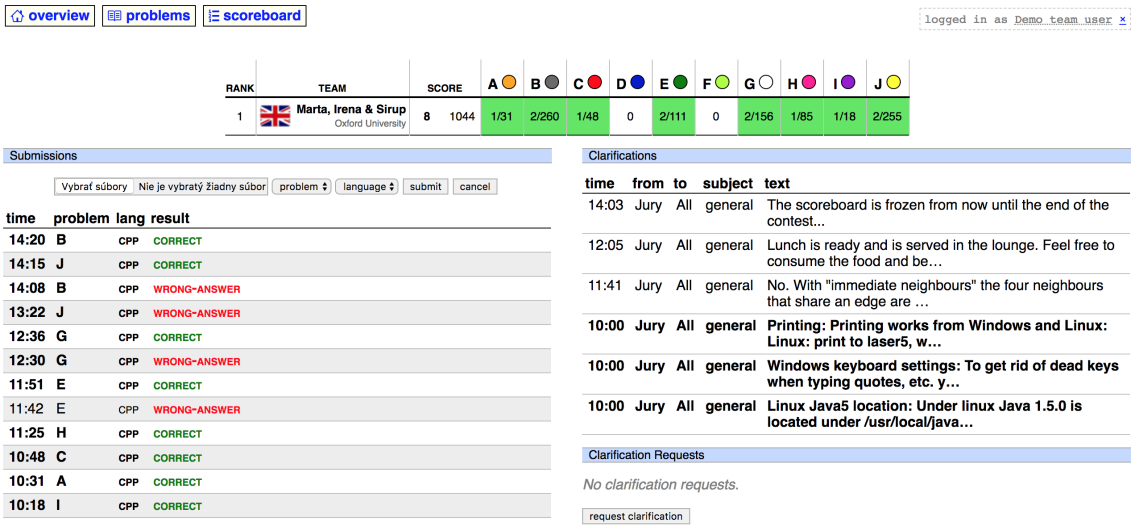
Obr. 2.2: Ukážka práce s aplikáciou Moodle

Z pohľadu vyššie uvedených pilierov, Moodle zabezpečuje zdieľanie informácií a materiálov pre žiakov.

Hlavná nevýhoda Moodlu je rovnaká ako pri všetkých projektoch s veľkou komunitou a otvoreným kódom. Sú obrovské a príliš komplexné. Je náročné si Moodle nainštalovať a nakonfigurovať, a rväčšina škôl bez hlavného zamerania na programovanie na nasadenie a spustenie takéhoto systému nemajú personálne ani technické kapacity. Moodle je preto často využívaný na vysokých školách a univerzitách, ale jeho nasadenie pre základnú a strednú školu nie je úplne vhodné.

2.1.2 DOMJudge

DOMJudge je automatizovaný testovací systém pre programátorské súťaže. Obsahuje mechanizmus pre odovzdávanie riešení, automatické otestovanie a webové rozhranie pre tímy riešiteľov a porotu.



Obr. 2.3: Ukážka práce s aplikáciou DOMJudge

Toto riešenie je používané pri mnohých programátorských súťažiach ako napríklad:

- ACM ICPC Sub-regionals: BAPC (od roku 2004), GCPC (od roku 2010)
- ACM ICPC Regionals: NWERC (od roku 2007) , SWERC (od roku 2008), SER USA (od roku 2010), SOCAL (od roku 2013), RMRC (od roku 2014), South Pacific Regional
- ACM ICPC World Finals (od roku 2012, v roku 2018 ako hlavný odovzdávací systém)

[18].

Z popisu aplikácie DOMJudge vyplýva, že táto aplikácia zabezpečuje zdieľanie zadaní a odovzdávanie s automatickým vyhodnotením žiackych riešení.

Pre spustenie DOMJudge je potrebný linuxový server s root prístupom, apache web serverom s PHP 5 a vyšším MySQL, alebo MariaDB databázu.

DOMJudge je fungujúce riešenie vhodné pre technicky dobre vybavené školy alebo súťaže. Bežné stredné školy, ktoré nie sú zamerané na programovanie, nedisponujú personálnymi ani technickými kapacitami pre nasadenie systému ako je DOMJudge.

2.1.3 Cloudové úložiská a sociálne siete

Z výskumu, ktorý autor realizoval na stredných školách vyplýva, že nezanedbateľné množstvo učiteľov využíva pre komunikáciu so žiakmi aj rôzne cloudové úložiská a sociálne siete.

Tieto aplikácie poskytujú riešenie zabezpečujúce dva z vyššie uvedených pilierov. Konkrétne komunikáciu učiteľov so žiakmi a zdieľanie informácií a materiálov pre žiakov.

Výhodou pri použití týchto aplikácii je v tom, že žiaci ich pravidelne využívajú aj na súkromné účely, nie je preto problém získať ich pozornosť kontaktovaním v takejto apliácii.

Obr. 2.4: Sociálne siete
Zdroj: https://www.worldatlas.com/r/w728-h425-c728x425/upload/8f/89/03/socialmedia-pm.png

Problém týchto aplikácii je že nie sú vhodné pre vyučovací proces, keďže (predovšetkým sociálne siete) poskytujú žiakom aj mnoho rozptyľovacích faktorov. Ďalším problémom je, že obsahujú veľké množstvo informácii, ktoré nie je jednoduché archivovať a spravovať.

2.1.4 Proprietárne riešenia

Mnohé väčšie vzdelávacie inštitúcie zamerané na vyučovanie informatiky a programovania vlastnia svoje vlastné proprietárne riešenia. (screen z brute)

Tieto riešenia poslúžili autorovi práce ako inšpirácia, nie sú ale použiteľné pre základné a stredné školy, nakoľko nemajú technické, ani ľudské kapacity pre vytvorenie takéhoto systému na mieru.

2.2 Navrhované riešenie

Predmetom tejto práce je návrh a realizácia systému pre podporu vyučovania programovania na základných a stredných školách.

Navrhované riešenie kombinuje prvky existujúcich riešení pre komunikáciu učiteľa so žiakom, ako aj odovzdávacích systémov a systémov pre zdieľanie materiálov pre žiakov.

Takmer všetky existujúce riešenia vyžadujú inštaláciu na lokálny server, ktorým musí škola disponovať. Keďže väčšina škôl takýmto technickým vybavením nedisponuje, autor sa rozhodol realizovať systém pre podporu programovania na školách modelom softvér ako služba.

Softvér ako služba (v angličtine SaaS - Software as a Service) je model, pri ktorom užívatelia využívaju službu poskytovanú prevádzkovateľom služby. Služba je zákazníkom poskytovaná cez internet bez nutnosti čokoľvek inštalovať na lokálne počítače. Jediná požiadavka pre prevádzkovanie systému využívajúceho model softvér ako služba je internetový prehliadač.

Aplikácia poskytne učiteľom komunikačný kanál so žiakmi. Poskytne im nástroj pre cielené zdieľanie študijných materiálov, zdrojových kódov a zadaní svojim študentom.

V dnešnej dobe su najnavštevovanejšími a najpoužívanejšími webovými aplikáciami sociálne siete [9]. Preto boli aj do popisovanej aplikácie navrhnuté prvky sociálnych sietí ako je zdieľanie, komentovanie a združovanie užívateľov do skupín. Ďalšou funkcionalitou inšpirovanou sociálnymi sieťami sú oznámenia o udalostiach v aplikácii. Tie sú dôležitým nástrojom, ako vylepšiť zážitok užívateľa z používania aplikácie a značne zvyšujú použiteľnosť aplikácie .

Oznámenia distribuované užívateľom správnymi kanálmi poskytujú prehľad o aktivite v aplikácii bez nutnosti mať túto aplikáciu nepretržite spustenú. To autor práce považuje za nevyhnutné pre popularizáciu aplikácie medzi žiakmi základných a stredných škôl.

Oznámenie formou emailu alebo správy v niektorom z mobilných komunikátorov aplikácia upozorní žiaka na nový materiál zdieľaný od učiteľa, alebo na blížiaci sa termín odovzdania domácej úlohy, čím sa zníži aj podiel žiakov, ktorí z nedbalosti zabúdajú na domácu prípravu.

Dôležitou súčasťou procesu vyučovania je aj spätná väzba, bez ktorej sa žiaci nemôžu zlepšovať. Dôležitá nie je len forma a obsah spätnej väzby, ale aj jej promptnosť. Spätná väzba je efektívna a má výsledky, len ak je čo najaktuálnejšia [19].

Navrhovaná aplikácia preto poskytne žiakom nástroj, pomocou ktorého môžu nahrať svoje riešenie zadania do aplikácie, kde bude spustené a otestované. Žiaci tak budú mať okamžitý prehľad o správnosti a úspešnosti ich riešení. Automatická spätná väzba je dôležitá, ale samozrejme si autor uvedomuje aj nevyhnutnosť ľudského faktora pri hodnotení odovzdaných riešení. Pre aplikáciu je preto dôležitá aj funkcia zobrazovania odovzdaných zdrojových kódov a komentovanie týchto riešení učiteľmi.

2.3 SWOT analýza

SWOT analýza je nástroj plánovane používaný na hodnotenie vnútorných aj vonkajších vplyvov a z nich vyplývajúcich silných a slabých stránok strategického cieľa. V prípade tejto práce sa jedná o úspešné používanie aplikácie CodeTutor pre zvýšenie kvality vyučovania programovania na základných a stredných školách.

2.3.1 Silné stránky (Strengths)

Aplikácia bude disponovať príťažlivým uživateľským rozhraním a bude poskytovať vhodnú platformu pre komunikáciu učiteľov so žiakmi. Aplikácia bude realizovaná modelom softvér ako služba a preto bude jednoduché nasadiť ju do použitia na školách, kurzoch a súťažiach.

2.3.2 Slabé stránky (Weaknesses)

Slabá stránka aplikácie môže spočívať v nepochopení potrieb učiteľov alebo žiakov zo strany autora aplikácie. Autor aplikácie sa pokúsi túto slabú stránku kompenzovať konzultáciami s užívateľmi aplikácie a pravidelným zbieraním spätnej väzby od užívateľov.

Obr. 2.5: SWOT analýza Zdroj: http://excel-navod.fotopulos.net/swot-analyza/1.png

2.3.3 Príležitosti (Opportunities)

Stretnutia s učiteľmi, ktorí by mali záujem o využívania aplikácie pri ich vyučovacom procese, a mimoštátnymi organizáciami vyučujúcimi programovanie tzv. bootcampy, pomaturitné kurzy a podobne. Organizácia rôznych programátorskych udalostí - workshopov, hackathonov a iných obdorných, ale aj networkingových aktivít.

2.3.4 Ohrozenia (Threats)

Nízky záujem zo strany učiteľov o zmenu ich zaužívaného vyučovacieho procesu. Ohrozenie od náhodných užívateľov, ktorý by nejakým spôsobom mohli z vonku poškodiť, alebo na čas znefunkčniť systém, alebo dáta v aplikácii.

2.4 Funkčné požiadavky

2.4.1 Autentifikácia a autorizácia

Aplikácia obsahuje jednoduchú prezentačnú stránku, ktorá je verejne dostupná. Okrem tejto prezentačnej stránky všetky akcie v aplikácii vyžadujú autorizáciu užívateľa, ktorý aplikáciu používa. Pre účely autentifikácie sa každý užívateľ identifikuje emailom a heslom.

2.4.1.1 Spôsob uloženia hesla

Heslo bude uložené v zahashovanej forme. Hash bude vypočítaný tzv. pomalou hashovacou funkciou. Heslá budú uložené vo formáte s jedinečným prefixom - tzv. soľou (z anglického salt) [4].

2.4.1.2 Reset hesla

Pre prípad, kedy užívateľ zabudne svoje heslo, bude aplikácia obsahovať mechanizmus pre reset hesla. Pre reset hesla bude vyžadované zadanie prihlasovacieho emailu. Na ten bude užívateľovi zaslaný odkaz s unikátnou URL, na ktorej si nastaví nové heslo

2.4.1.3 Užívateľský profil

Užíveteľský profil pozostáva z mena, priezviska, titulu pred menom a prihlasovacieho emailu. Tieto parametre môže zmeniť len správca skupiny, ktorej je užívateľ členom. Profil ale obsahuje aj kontaktné informácie - sekundárny email a Facebook profil pre účely oznámení.

2.5 Užívatelia a užívateľské skupiny

Užívatelia budú na základe príslušnosti k rôznym školám, triedam a študijným skupinám rozdelení do užívateľských skupín. Tieto skupiny sú nevyhnutné pre potreby zadefinovania užívateľskych práv, zdieľanie obsahu týmto skupinám a podobne. V aplikácii CodeTutor bude možné užívateľov zadeliť do nasledujúcich druhov skupín.

2.5.1 Škola

Škola je skupina užívateľov združujúca všetkých užívateľov jednej školy. Každá škola, ktorá začne používať CodeTutor, tvorí jednu skupinu užívateľov.

2.5.2 Skupina

Skupina užívateľov je menšia skupina užívateľov zdieľajúcich spoločné dáta v systéme CodeTutor. Skupina môže byť podmnožinou školy. V tom prípade sa jedna typicky o jednu triedu alebo jednu študijnú skupinu, ktorá má hodinu programovania v jednom termíne u jedného učiteľa. Skupina, ktorá je podmnožinou školy, je tvorená len užívateľom patriacim do školy, ktorej podmnožinou je daná skupina.

2.5.2.1 Globálna skupina

Skupina môže byť vytvorená aj mimo školy. V takom prípade sa môže stať členom skupiny akýkoľvek užívateľ CodeTutor bez ohľadu na to, či je členom školy, alebo nie.

Význam takýchto skupín je pri organizovaní súťaží, ktorých sa budú zúčastňovať užívatelia z rôznych škôl, alebo v prípade rôznych školení, sústredení a podobne, kedy nemá význam vytvárať v systéme CodeTutor novú užívateľskú skupinu typu škola.

2.5.3 Roly používateľov

Užívateľ má priradenú vždy jednu globálnu rolu a jednu rolu vrámci každej skupiny, ktorej je členom.

2.5.3.1 Globálne roly

Každý užívateľ systému ma predvolene priradenú globálnu rolu užívateľ. Globálna rola správca je určená pre systémových správcov, ktorí majú možnosť vytvárať nové školy a skupiny, ktoré nie sú priradené žiadnej škole.

2.5.3.2 Roly v skupinách

Užívateľ, ktorý je členom skupiny užívateľov, či už školy alebo skupiny, má vrámci tejto skupiny užívateľov priradenú jednu z troch nasledujúcich rolí.

2.5.3.3 Žiak

Žiak je užívateľ, ktorý má právo len zobrazovať si obsah v skupine, nemá právo spravovať užívateľov v skupine, alebo pridávať do skupiny nový obsah.

2.5.3.4 Učiteľ

Učiteľ má právo vykonávať všetky činnosti, ktoré vyplývajú z roly žiaka. Má navyše právo vytvárať nové skupiny v rámci školy, v ktorej má rolu učiteľa. Do týchto skupín môže pridávať užívateľov, ktorí sú tiež členmi školy, v ktorej má rolu učiteľa. Učiteľ má právo vytvárať nový obsah v skupinách, v ktorých má rolu učiteľa. Má právo vytvárať články, zadania a zdieľať ich študentom.

2.5.3.5 Správca

Správca má vrámci skupiny všetky práva vyplývajúce z roly učiteľa. Navyše má právo priamo spravovať užívateľov. Správca môže nie len pridať užívateľa do skupiny, ale môže aj vytvoriť nového užívateľa vrámci školy alebo skupiny, v ktorej má rolu správcu. Môže upraviť, vymazať existujúceho užívateľa vrámci školy alebo skupiny, v ktorej má rolu správcu.

2.5.4 Materiály pre študentov

Užívateľ s rolou správcu alebo učiteľa skupiny má možnosť zdieľať študentom rôzne študijné materiály. Tie môžu mať štyri rôzne formy.

2.5.4.1 Odkaz

Odkaz je najjednoduchším spôsobom, ako môže užívateľ s rolou správcu alebo učiteľa zdieľať obsah pre užívateľov v skupinách, v ktorých majú tieto roly.

Odkaz môže užívateľ zdieľať priamo v zobrazení aktivity v skupine, v ktorej má rolu správcu, alebo užívateľa, alebo vytvoriť nový odkaz vo svojom profile a následne nastaviť zdieľanie tohto odkazu do jednej zo skupín, v ktorej má užívateľ rolu správcu, alebo učiteľa.

K takémuto odkazu môže jeho autor priradiť komentár. Tento komentár má obmedzenú dĺžku a môže byť formátovaný pomocou jazyka Markdown, alebo HTML.

2.5.4.2 Súbor

Užívateľ s rolou správcu alebo učiteľa skupiny môže tejto skupine zdieľať súbor. Tento súbor nesmie byť spustiteľný a jeho veľkosť nesmie presiahnuť 15MB.

Súbor môže užívateľ nahrať v zobrazení aktivity v skupine, v ktorej má rolu správcu, alebo užívateľa, alebo nahrať nový súbor vo svojom profile a nastaviť zdieľanie tohto súboru do jednej, alebo viacero zo skupín, v ktorej má rolu učiteľa, alebo správcu.

K súboru zdieľanému v skupine môže jeho autor priradiť komentár, rovnako ako v prípade odkazu zdieľaného do skupiny.

2.5.4.3 Článok

Užívateľ s rolou správcu alebo učiteľa skupiny môže priamo v systéme vytvoriť obsah pre študentov formou článku. Tento článok môže užívateľ vytvoriť vo výpise aktivity skupiny, alebo vo svojom profile a nastaviť zdieľanie skupine, alebo skupinám, v ktorej má tento užívateľ rolu správcu, alebo užívateľa. Článok môže byť formátovaný pomocou jazyka Markdown, alebo HTML. Články budú upravované pomocou vstavaného editora, ktorý umožní formátovať články aj užívateľom neznalým štruktúry jazykov Markdown, alebo HTML.

Každý článok bude môcť samozrejme okrem formátovaného textu obsahovať aj obrázky, ktoré bude užívateľ nahrávať priamo pomocou editora.

Každý článok môže obsahovať okrem obrázkov aj odkazy na iné stránky, alebo odkazy na stiahnutie súborov - príloh, ktoré bude môcť autor článku nahrávať pomocou editora článku.

Priamo pri zobrazení článku bude zobrazený aj výpis všetkých príloh a obrázkov, ktoré sú obsahom článku.

2.5.5 Zadania a odovzdávanie riešení

Užívateľ s rolou správcu alebo užívateľa bude mať právo vytvoriť zadanie, ktoré bude pozostávať z častí popísaných nižšie v tejto kapitole.

Zadanie môže byť vytvorené vo výpise aktivity v konkrétnej skupine, v ktorej má užívateľ práva učiteľa, alebo správcu, alebo v profile užívateľa, ktorý môže nastaviť zdieľanie do jednej, alebo niekoľkých skupín, v ktorých má užívateľ rolu učiteľa, alebo správcu.

Pri zadaniach budú dôležitú rolu hrať vstupy a výstupy. Práve tie budú definovať korektné riešenie zadania. Riešenie zadania bude vrámci testovania spustené s množinou zadefinovaných vstupných dát a následne sa výstup riešenia bude porovnávať so zadefinovanými výstupnými dátami.

2.5.5.1 Úlohy zadania

Každé zadanie bude pozostávať z jednej alebo viacerých úloh.

2.5.5.2 Text zadania

Text zadania je určený pre popis úloh zadaných žiakom. Môže byť formátovaný pomocou jazyka Markdown, alebo HTML. Text zadanie bude možné upravovať pomocou vstavaného editora, ktorý umožní formátovať články aj užívateľom neznalým štruktúry jazykov Markdown, alebo HTML.

Text zadania bude môcť rovnako ako text článku obsahovať aj obrázky, odkazy na internetové stránky, alebo prílohy na stiahnutie.

2.5.5.3 Parametre zadania

Autor bude mať právo nastavovať rôzne parametre zadania. Termíny, odkedy bude užívateľom umožnené odovzdávať riešenia zadania a dokedy bude umožnené odovzdávať riešenia tohto zadania. Auto zadania bude môcť vybrať programovacie jazyky, ktoré budú akceptované pri odovzdávaní riešenia od užívateľov.

2.5.5.4 Vzorové vstupy a výstupy

Autor zadania má možnosť definovať vzorové vstupy a výstupy zadania. Tieto vstupy a výstupy sú dôležité pre riešiteľov zadania. Pomocou nich si môžu predbežne skontrolovať validnosť svojho riešenia bez nutnosti odovzdávať toto riešenie do aplikácie.

Zatiaľ čo vstupy autor definuje pre celé zadanie, výstupy sa definujú pre jednotlivé úlohy, z ktorých zadanie pozostáva. Autorovi zadania to dá možnosť vytvoriť zadanie, kde riešiteľ musí realizovať viacero úloh nad rovnakou sadou dát, ale aj zadanie, kde každá úloha má unikátnu sadu dát.

Ku každému riadku výstupu bude autor zadania definovať, akým spôsobom sa tento riadok výstupu bude porovnávať s prislúchajucim riadkom výstupu užívateľského riešenia. Štandardné porovnanie je porovnanie dvoch reťazcov, kde sa vyžaduje úplná zhoda obsahu riadka. Autor zadania ale bude môcť zvoliť aj celočíselné porovnanie, alebo porovnanie čísel s plávajúcou desatinnou čiarkou. V prípade číselného porovnania, si autor zadania bude môcť zvoliť presnosť, s akou bude číslo porovnávané. V prípade číselného porovnávania, bude riadok uznaný za korektný v prípade, že bude platiť, že absolútna hodnota rozdielu autorom definovaného výstupu a výstupu riešenia je menšia ako zadefinovaná presnosť.

Tieto vzorové vstupy a výstupy sú priamo súčasťou zobrazenia textu zadania. Riešitelia si ich zároveň budú môcť stiahnuť vo forme textových súborov.

2.5.5.5 Testovacie vstupy a výstupy

Autor zadania bude mať v aplikácii možnosť vytvovriť testovacie vstupy a výstupy zadania.

Tieto vstupy a výstupy sa budú definovať v rovnakej forme ako vzorové vstupy a výstupy. V prípade výstupov zadania bude navyše autor zadania zadávať aj bodové ohodnotenie jednotlivých riadkov výstupu. V prípade, že výstup riešenia sa zhoduje (alebo spĺňuje zadefinovanú presnosť), bude k bodovému ohodnoteniu riešenia pripočítaný prislúchajúci počet bodov. Odovzdávanie riešenia Užívateľ - člen skupiny, do ktorej autor zadania zdieľal zadanie, bude môcť odovzdávať riešenie zadania nahraním jedného súbora na server. V prípade, že riešenie bude pozostávať z viacerých súborov bude môcť odovzdať tieto súbory zbalených v ZIP archív. Súbory iných typov ako zip, alebo typov, ktoré neodpovedajú typom súborov príslušiacim zdrojovým kódom jedného z povolených programovacích jazykov pre dané zadanie budú vyhodontené ako nevalídne. Odovzdávaný nebude väčší ako 10MB, v prípade, že odovzdávané riešenie túto veľkosť presiahne bude riešenie označené za nevalídne.

V prípade, že zadanie bolo úspešne nahrané na server (súbor s riešením bude označený za validný) bude spustený automatický test. Užívateľ bude na stránke odovzdávania riešenia plne informovaný o aktuálnom stave procesu testovania jeho riešenia. Po ukončení testu si užívateľ priamo na stránke odovzdávania zobrazí výsledok automatického testovania a pridelené body. História odovzdávania Užívateľ bude mať možnosť zobraziť si históriu odovzdaní, vrátane zobrazenia odovzdaných súborov a výsledkov automatického hodnotenia týchto riešení.

2.5.5.6 Manuálne hodnotenie riešenia

Aplikácia poskytne autorom riešenia možnosť manuálne preskúmať zdrojový kód odovzdaného riešenia. Zdrojový kód bude zobrazený so zvýrazneným syntaxom daného jazyka, v ktorom je riešenie vytvorené. Autor zadania bude mať možnosť umiestnit komentár ku každému riadku riešenia. Autor zadania bude mať možnosť udeliť riešeniu jeden komplexný komentár, ktorého súčasťou je aj manuálne subjektívne udelenie bodov, ktoré sa pričítajú k bodom získaným z automatického hodnotenia. Bodové ohodnotenie, ktoré môže autor zadania priradiť užívateľskému riešeniu je obmedzený maximálnym manuálnym bodovým ohodnotením, ktoré je nastavené pri vytvorení zadania.

2.5.6 Zobrazenie aktivity

2.5.6.1 Oznámenia

Oznámenia sú nevyhnutnou súčasťou každej modernej aplikácie [6]. Každý užívateľ bude informovaný o aktivitách v aplikácií, ktoré sa ho týkajú. Jedná sa predovšetkým o pridaní, alebo odobraní užívateľa z / do skupiny užívateľov, aktualizácie obsahu v skupine, ktorej je užívateľ členom. Predovšetkým je to zdieľanie nového obsahu do skupiny užívateľov, ktorej je užívateľ členom, odpoveď na komentár, ktorý užívateľ udelil a podobne. Oznámenia budú užívateľovi zobrazované štyrmi spôsobmi.

2.5.6.2 Emailové oznámenia

Oznámenia zaslané emailom sú základný formát používaný takmer každou webovou aplikáciou. Code tutor pošle každé oznámenie formou emailu na prihlasovaciu adresu, alebo na sekundárnu emailovú adresu, ak ju má užívateľ zadanú vo svojom profile. Zaslané emailové

oznámenie bude obsahovať základný popis udalosti, ktorú oznamuje a akčné tlačidlo, po kliknutí na ktoré si užívateľ otvorí obsah, ktorý je predmetom oznámenia. Po kliknutí na akčné tlačidlo a otvorenia obsahu je oznámenie označené za prečítané. Oznámenie správou na Facebook-u Facebook je bezo sporu jednou z najčastejšie používaných webových aplikácii. Služba Messenger, ktorá je súčasťou Facebook-u je jednou z najčastejšie používaných komunikačných aplikácií. CodeTutor bude posielať oznámenie formou správy v službe Messenger z Facebook-ovej stránky CodeTutor. Toto oznámenie bude tiež obsahovať akčné tlačidlo, ktoré sa bude správať rovnako ako akčné tlačidlo v emailovom oznámení. Oznámenie z webového prehliadača Notifikácia z webového prehliadača je nová technológia, ktorá poskytuje webovým prehliadačom využívať vstavané notifikácie v operačnom systéme [22].

Oznámenie z webového prehliadača bude rovnako ako oznámenie na Facebook-u, alebo emailom obsahovať základný popis akcie, ktorú oznamuje, a akčné tlačidlo. Zobrazenie oznámení v systéme Priamo v systéme si užívateľ môže zobraziť všetky oznámenia, ktoré mu systém vytvoril. Tieto oznámenia nadobúdajú stav zobrazené, alebo nezobrazené. Pri vytvorení sa oznámenie označí ako zobrazené a pri otvorení oznámenia sa toto oznámenie zmení na nezobrazené. Oznámenie v systéme tiež obsahuje akčné tlačidlo, ktoré užívateľa nasmeruje priamo na obsah, ktorého sa oznámenie týka.

2.5.6.3 Komentáre

Jednou z kľúčových funkcií systému je možnosť konzultovať materiály a zadania učiteľov so žiakmi bez nutnosti osobného kontaktu. CodeTutor bude obsahovať možnosť komentovať všetky typy obsahu, ktoré môže učiteľ svojim žiakom zdieľať v užívateľskej skupine. Pri každom takomto obsahu vznikne jednoduché diskusné fórum, kde môžu všetci užívatelia, ktorí majú prístup k tomuto obsahu, vyjadriť svoj názor, alebo sa dopýtať na ďalšie informácie. Autor tohto obsahu bude o každom udelenom komentári informovaný formou oznámenia. Odpoveď na komentáre Komentáre obsahu budú obsahovať aj možnosť odpovedať na komentár konkrétneho užívateľa, čo dáva autorovi obsahu výbornú možnosť ako odpovedať na konkrétnu otázku žiaka k zdieľanému obsahu. Pri takejto odpovedi bude vytvorené oznámenie pre autora obsahu (v prípade, že autor odpovede nie je aj autor obsahu) a autorovi komentára, ku ktorému bola vytvorená odpoveď.

2.6 Nefunkčné požiadavky

2.6.1 Výkon a stabilita

Výsledná aplikácia musí spĺňovať základné podmienky škálovania kvôli rozširovaniu aplikácie na ďalšie školy. Aplikácia musí tiež splňať požiadavky na stabilitu, keďže sa dá predpokladať špička v používaní systému v období pred uzávierkou zadania, alebo v prípade súťaže pre viacero užívateľov.

2.6.2 Užívateľské rozhranie a užívateľská skúsenosť

Systém musí spĺňať požiadavky na moderné a priamočiare užívateľské rozhranie, aby zabezpečil čo najvyššiu použiteľnosť pre užívateľov systému.

Obecne platí, že pre systém bude navrhnuté jednoduché užívateľské rozhranie, ktoré sa bude zobrazovať korektne na rôznych typoch zariadení. Rozhranie bude minimalistické, zamerané hlavne na obsah s jednoduchou farebnou schémou obsahujúcou maximálne štyri rôzne farby.

Pre úspešný návrh a následné otestovanie užívateľského rozhrania je nevyhnutné stanoviť si profily užívateľov, ktorí budú aplikáciu využívať. Cieľovú skupinu aplikácie budú tvoriť tri základné skupiny užívateľov.

2.6.2.1 Učitelia a správcovia na školách

Túto skupinu užívateľov budú tvoriť pedagogickí zamestnanci vyučujúci technické predmety so zameraním na programovanie a administratívni zamestnanci školy. Užívatelia v tejto skupine sú vo veku 25 až 60 rokov, autor predpokladá vysokoškolské vzdelanie aspoň čiastočne technicky zamerané a pokročilé znalosti v práci s počítačom. Vzhľadom na skladbu a zameranie užívateľov a tiež činnosti, na ktoré budú užívatelia aplikáciu používať (tvorba programovacích zadaní, komplexnejšieho vzdelávacieho obsahu) autor predpokladá, že táto skupina užívateľov bude k aplikácii pristupovať z konvenčného počítača so štandardnými metódami vstupu (klávesnica a myš, prípadne klávesnica a touchpad).

2.6.2.2 Žiaci stredných škôl

Pre žiakov stredných škôl je typický vek medzi 15 a 20 rokov, značné skúseností s používaním počítačov, sociálnych sietí a ďalších webových aplikácií. Vzhľadom na súčasné trendy prehliadania internetu, podľa ktorých si užívatelia už častejšie prehliadajú internetové stránky a používajú internetové aplikácie na mobilných dotykových zariadeniach ako na konvenčných počítačoch so štandardnými vstupnými zariadeniami, autor predpokladá, že táto skupina užívateľov bude používať aplikáciu prevažne z mobilných zariadení s dotykovým displejom. Napriek tomu autor predpokladá, že časti systému spojené s odovzdávaním riešení zadaní budú žiaci realizovať z počítačov so štandardnými zobrazovacími aj vstupnými zariadeniami. Tento predpoklad autor odôvodňuje tým, že je vysoko pravdepodobné, že žiaci budú riešenie odovzdávať z rovnakého zariadenia, na akom toto riešenie vyprodukujú, čo bude z najväčšou pravdepodobnosťou zariadenie s konvečnými vstupnými zariadeniami. Napriek tomu, že už sa objavujú pokusy o plnohodnotné programovanie na mobilnom zariadení s dotykovým vstupným zariadením, (https://maymay.net/blog/2014/08/01/turn-your-android-phone-into-a-full-fledged-programming-environment/) konvenčná klávesnica pri písaní programov stále prevláda [2].

2.6.2.3 Žiaci základných škôl

Žiaci základných škôl tvoria svojim správaním veľmi podobnú skupinu ako žiaci stredných škôl. Tvoria ju užívatelia vo veku od 10 do 15 rokov (autor nepredpokladá využívanie aplikácie žiakmi prvého stupňa základných škôl, keďže na prvom stupni sa ešte nevyučujú predmety zamerané na IT a programovanie). Autor obecne u žiakov základných škôl nepredpokladá, že budú odovzdávať komplexné programátorské riešenia, preto predpokladá ešte vyššie využívanie aplikácie na mobilných zariadeniach.

2.6.3 Kompatibilita v prehliadačoch

Keďže predmetná aplikácia bude realizovaná ako webová aplikácia, autor deklaruje minimálne verzie štyroch najbežnejších webových prehliadačov, pri ktorých autor zaručuje funkčnosť aplikácie. Ide o Internet Explorer vo verzii 9 a novšej, Google Chrome vo verzii 10 a novšej, Mozilla Firefox vo verzii 10 a novšej a Opera vo verzii 12 a novšej. V týchto prehliadačoch autor zaručuje funkčnost aplikácie, ale pre maximálny užívateľský zážitok z aplikácie autor odporúča použitie najnovších verzií jedného z týchto webových prehliadačov.

2.7 Prípady použitia

Prípady použitia opisujú aktivitu uívateľa v systéme, aby dosiahol požadovaný cieľ. Nasledujúce prípady použitia sú preto definované ako zoznam akcií, ktoré užívateľ musí vykonať pre dosiahnutie cieľa prípadu použitia.

V následujúcich prípadoch použitia sa vyskytujú prípady použitia, ktoré sú zhodné pre všetky druhy obsahu, ktoré je možné zdieľať do skupiny, ktorej je daný užívateľ členom. Jednotlivé prípady použitia teda autor generalizoval a miesto jednotlivých druhov obsahu používa jednotný termín obsah.

Všetky nasledujúce prípady použitia (ak nebude priamo pri prípade použitia vyznačené inak) začínajú zobrazením domovskej stránky prihláseného užívateľa.

2.7.1 Aktéri

Na základe požiadaviek na užívateľské roly v Prípadoch použitia rozlišujeme troch následujúcich aktérov. Keďže jeden užívateľ aplikácie môže byť členom viacerých skupín a v nich mať pridelené rôzne roly, je možné, že jeden užívateľ môže v rôznych skupinách vystupovať ako rôzny aktér.

Všetci aktéri prípadov použitia sú užívatelia aplikácie, ktorí sú členmi aspoň jednej užívateľskej skupiny a majú v tejto skupine rolu, ktorej názov je rovnaký ako označenie tohto aktéra. V popisovanej aplikácii budeme hovoriť teda o troch druhoch aktérov a to žiak, učiteľ, správca.

2.7.1.1 Žiak

Zobrazenie obsahu

- Žiak klikne na jeden z obsahov v zobrazení aktivity

Vytvorenie komentára

- Žiak klikne na jeden z obsahov v zobrazení aktivity
- Žiak klikne na tlačidlo pridať komentár pod obsahom zadania, alebo článku
- Žiak odošle komentár

Odpoveď na komentár

- Žiak klikne na jeden z obsahov v zobrazení aktivity
- Žiak si vyberie konkrétny komentár pod obsahom, alebo si klikne na tlačilo zobraziť všetky komentáre
- Žiak klikne na tlačilo odpovedať pod komentárom, na ktorý chce odpovedať
- Žiak odošle odpoveď na komentár

Odovzdanie riešenia

- Žiak klikne na jedno zo zadaní
- Žiak klikne na tlačilo odovzdanie
- Žiak klikne na tlačidlo vybrať súbor
- Žiak klikne na tlačidlo nahrať
- Žiakovi bude zobrazovaný aktuálny stav automatického testovania

História odovzdávania

- Žiak klikne na jedno zo zadaní
- Žiak klikne na tlačidlo odovzdanie
- Žiak klikne na tlačidlo história odovzdávania

2.7.1.2 Učiteľ

Zdieľanie obsahu

- Učiteľ klikne na tlačidlo pridať obsah
- Učiteľ klikne na tlačidlo zdieľať existujúci obsah
- Učiteľ pridá cieľovú skupinu do nastavenia obsahu
- Učiteľ uloží zmenu obsahu kliknutím na tlačidlo uložiť

Učiteľ si zobrazí výpis aktivity (domovská stránka prihláseného užívateľa)

- Učiteľ klikne na tlačidlo pridať obsah
- Učiteľ klikne na tlačidlo vytvoriť nový obsah
- Učiteľ pridá cieľovú skupinu do nastavenia obsahu
- Učiteľ vytvorí obsah kliknutím na tlačidlo uložiť

- Učiteľ klikne na svoje meno v pravom hornom rohu obrazovky
- Učiteľ klikne na tlačilo obsah v rolovacom menu pod menom užívateľa
- Učiteľ klikne na obsah, ktorý chce zdieľať v skupine
- Učiteľ klikne na tlačidlo upraviť
- Učiteľ pridá cieľovú skupinu do nastavenia obsahu
- Učiteľ uloží zmenu obsahu kliknutím na tlačidlo uložiť

- Učiteľ klikne na svoje meno v pravom hornom rohu obrazovky
- Učiteľ klikne na tlačilo obsah v rolovacom menu pod menom užívateľa
- Učiteľ klikne na tlačidlo vytvoriť obsah
- Učiteľ pridá cieľovú skupinu do nastavenia obsahu
- Učiteľ vytvorí obsah kliknutím na tlačidlo uložiť

Parametre zadania

- Učiteľ klikne na jedno zo zadaní vo výpise aktivít
- Učiteľ klikne na tlačidlo upraviť
- Učiteľ upraví hodnoty parametrov zadania
- Učiteľ uloží zmenu zadania kliknutím na tlačidlo uložiť

- Učiteľ klikne na svoje meno v pravom hornom rohu obrazovky
- Učiteľ klikne na tlačilo obsah v rolovacom menu pod menom užívateľa
- Učiteľ klikne na jedno zo zadaní
- Učiteľ klikne na tlačidlo upraviť
- Učiteľ upraví hodnoty parametrov zadania
- Učiteľ uloží zmenu zadania kliknutím na tlačidlo uložiť

Vzorové vstupy a výstupy

- Učiteľ klikne na jedno zo zadaní vo výpise aktivít

- Učiteľ klikne na tlačidlo vzorové dáta
- Učiteľ klikne na tlačidlo upraviť
- Učiteľ upraví hodnoty vzorových vstupov a výstupov
- Učiteľ uloží zmenu hodnôt kliknutím na tlačidlo uložiť

- Učiteľ klikne na svoje meno v pravom hornom rohu obrazovky
- Učiteľ klikne na tlačilo obsah v rolovacom menu pod menom užívateľa
- Učiteľ klikne na jedno zo zadaní
- Učiteľ klikne na tlačidlo vzorové dáta
- Učiteľ klikne na tlačidlo upraviť
- Učiteľ upraví hodnoty vzorových vstupov a výstupov
- Učiteľ uloží zmenu hodnôt kliknutím na tlačidlo uložiť

Testovacie vstupy a výstupy

- Učiteľ klikne na jedno zo zadaní vo výpise aktivít
- Učiteľ klikne na tlačidlo testovacie dáta
- Učiteľ klikne na tlačidlo upraviť
- Učiteľ upraví hodnoty vzorových vstupov a výstupov
- Učiteľ uloží zmenu hodnôt kliknutím na tlačidlo uložiť

- Učiteľ klikne na svoje meno v pravom hornom rohu obrazovky
- Učiteľ klikne na tlačilo obsah v rolovacom menu pod menom užívateľa
- Učiteľ klikne na jedno zo zadaní
- Učiteľ klikne na tlačidlo testovacie dáta
- Učiteľ klikne na tlačidlo upraviť
- Učiteľ upraví hodnoty vzorových vstupov a výstupov
- Učiteľ uloží zmenu hodnôt kliknutím na tlačidlo uložiť

Manuálne hodnotenie riešenia

- Učiteľ klikne na jedno zo zadaní vo výpise aktivít
- Učiteľ klikne na tlačidlo riešenia
- Učiteľ klikne na jedno z riešení
- Učiteľ upraví komentár k riešeniu a zadá počet bodov
- Učiteľ uloží zmenu hodnôt kliknutím na tlačidlo uložiť

- Učiteľ klikne na svoje meno v pravom hornom rohu obrazovky
- Učiteľ klikne na tlačilo obsah v rolovacom menu pod menom užívateľa
- Učiteľ klikne na jedno zo zadaní
- Učiteľ klikne na tlačidlo riešenia
- Učiteľ klikne na jedno z riešení
- Učiteľ upraví komentár k riešeniu a zadá počet bodov
- Učiteľ uloží zmenu hodnôt kliknutím na tlačidlo uložiť

2.7.1.3 Správca

Vytvorenie užívateľa
V prípade, že Správca má globálnu rolu správcu

- Správca klikne na tlačidlo Užívatelia
- Správca klikne na tlačidlo Vytvoriť
- Správca vyplní informácie o novom užívateľovi
- Správca vytvorí užívateľa kliknutím na tlačidlo Uložiť

V prípade, že Správca má rolu správcu v škole, ktorej je členom

- Správca klikne na tlačidlo Užívatelia
- Správca klikne na tlačidlo prislúchajúce role, ktorú chce novému užívateľovi prideliť (Správca, Učitelia, Žiaci)
- Správca klikne na tlačidlo Vytvoriť
- Správca vyplní informácie o novom užívateľovi

- Správca vytvorí užívateľa kliknutím na tlačidlo Uložiť

V prípade, že Správca má rolu správcu v skupine, ktorej je členom

- Správca klikne na tlačidlo Užívatelia
- Správca klikne na tlačidlo prislúchajúcej role, ktorú chce novému užívateľovi prideliť (Správca, Učitelia, Žiaci)
- Správca klikne na tlačidlo Pridať
- Správca vyberie užívateľa, ktorého chce pridať do skupiny
- Správca pridá užívateľa kliknutím na tlačidlo Pridať

Vytvorenie školy
Tento prípad použitia sa týka len užívateľa, ktorý má priradenú globálnu rolu správcu

- Správca klikne na tlačidlo Užívatelia
- Správca klikne na tlačidlo Školy
- Správca klikne na tlačidlo Vytvoriť
- Správca vyplní informácie o novej škole
- Správca vytvorí školu kliknutím na tlačidlo Uložiť

Vytvorenie skupiny

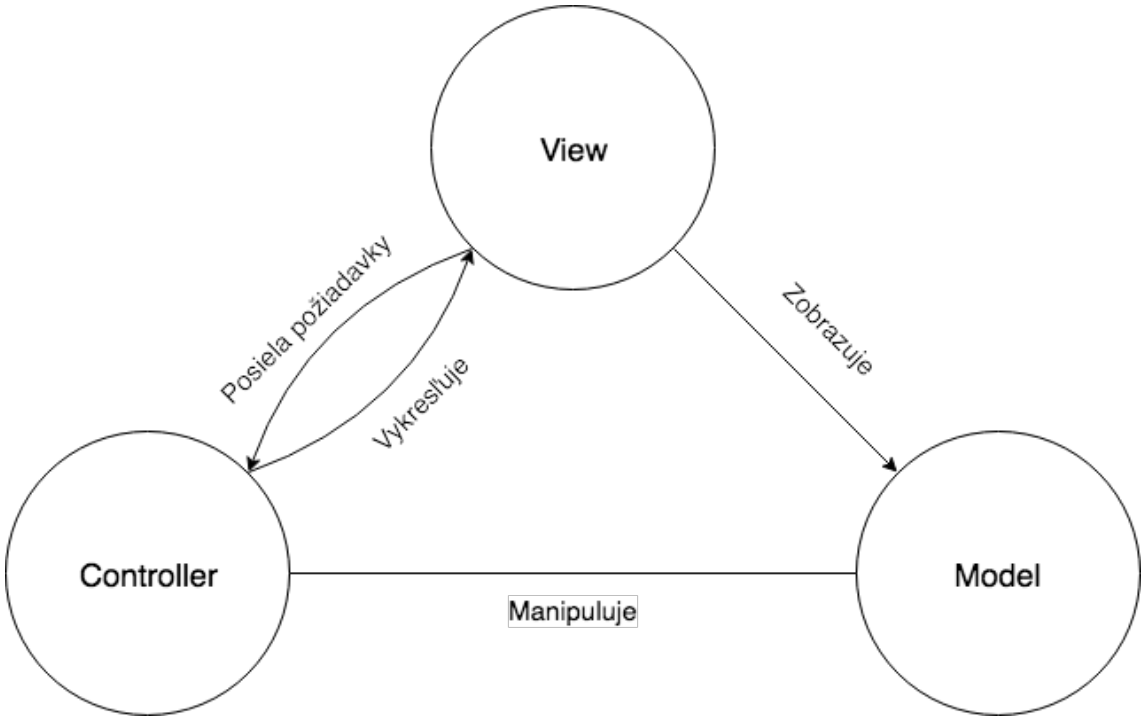
- Správca klikne na tlačidlo Užívatelia
- Správca klikne na tlačidlo Skupiny
- Správca klikne na tlačidlo Vytvoriť
- Správca vyplní informácie o novej skupine
- Správca vytvorí skupinu kliknutím na tlačidlo Uložiť

Kapitola 3

Návrh systému

3.1 Architektúra systému

Pre aplikáciu bola navrhnutá viacvrstvová Model View Controller (ďalej len MVC) architektúra. MVC pozostáva z troch typov objektov. Model je aplikačný objekt, v ktorom je uchovaný stav aplikácie. View - v slovenskom preklade obrazovka, definuje prezentáciu dát pomocou užívateľského rozhrania a Controller definuje interakciu aplikácie na užívateľské vstupy a požiadavky.



Obr. 3.1: MVC Architektúra

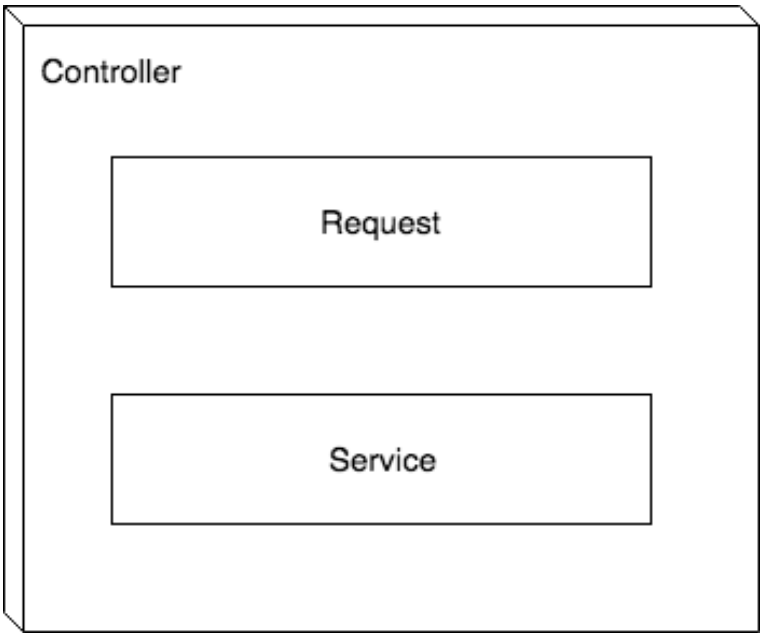
MVC rozdeľuje obrazovky a modely vytvorením komunikačného rozhrania medzi týmito

dvoma objektmi. Obrazovky zabezpečujú, aby ich zobrazenie odzrkadľovalo aktuálny stav modelu. Kedykoľvek sa informácie v modeli zmenia, obrazovky, ktoré závisia na tomto modeli, sa zmenia. Tento prístup umožňuje pripojiť viacero obrazoviek k jednému dátovému modelu. Umožňuje tiež vytvoriť rôzne obrazovky pre zobrazovanie dát bez nutnosti meniť model.

Výhodou implementácie projektu navrhnutého s architektúrou MVC je jednoduchá údržba a čistota kódu, pretože užívateľské rozhranie a obchodná logika sa nachádzajú v samostatných moduloch projektu. V prípade práce v tíme je jednoduché rozdeliť prácu v tíme pre jednotlivých vývojárov.

Nevýhodou MVC architektúry je práve komplexnosť tejto architektúry. Vyžaduje komplexné pochopenie toku informácií medzi obrazovkami, obchodnou logikou a ďalšími časťami projektu. Bez neho by bolo náročné odstraňovať chyby v aplikácii.

Do navrhnutej architektúry aplikácie boli pridané ďalšie vrstvy. Pridaním ďalších vrstiev bolo dosiahnuté, že každá vrstva realizuje práve jednu funkciu obchodnej logiky. Aplikácia sa tým stane prehľadnejšia a v neposlednom rade aj bezpečnejšia.



Obr. 3.2: Viacvrstvá Architektúra

Samozrejmosťou je zvýšená prehľadnosť aplikácie, čo umožní jednoduchšie objavovanie a opravovanie chýb. Jednoúčelové bloky kódu sú aj oveľa jednoduchšie testovateľné. Pre takéto časti aplikácie je podstatne jednoduchšie napísať jednotkové testy a dosiahnuť tak vyššie pokrytie projektu testami.

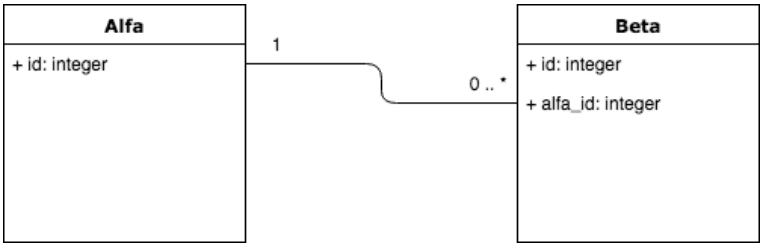
3.2 Dátový model

Dátový model slúži na návrh dátovej štruktúry, ktorá reprezentuje informácie spracovávané a prezentované v aplikácii. Ide o konceptuálny model, ktorý je veľmi podobný klasickým ER diagramom používaným pre návrh štruktúry relačných databáz, alebo UL Class diagramom používaných v objektovom návrhu [11].

Model je tvorený entitami. Entita reprezentuje objekty reálneho sveta, ktoré potom označujeme ako inštancie entity. Inštancie entity sa vyznačujú rovnakou vnútornou dátovou štruktúrou, ktorá sa vyjadruje množinou atribútov. Všetky entity v dátovom modeli aplikácie budú zdieľať atribúty o čase vytvorenia (created_at), o čase poslednej úpravy inštancie entity (updated_at) a čase vymazania entity (deleted_at). Zavedenie času vymazania ako atribútu jednotlivých entít je nevyhnutné pre uchovávanie historických informácii v databáze aplikácie aj po tom, čo tieto údaje už budú vymazané. V momente, kedy v aplikácii vznikne požiadavka na vymazanie jednej z inštancií entity, bude nastavený čas vymazania na aktuálny čas, kedy táto požiadavka vznikla. Aplikácia bude následne všetky entity, ktoré majú nastavený čas vymazania, ignorovať, akoby v databáze neexistovali. Zároveň ale tieto inštancie budú v databáze stále existovať a bude existovať možnosť tieto inštancie obnoviť alebo v prípade potreby definitívne vymazať. Každá entita užívateľa zároveň obsahuje jeden alebo dva atribúty tvoriace unikátne identifikátory. Automatický unikátny identifikátor (id) je unikátne číslo, ktoré bude interne využívané pre identifkáciu inštancie entity vrámci databázového systému. Druhý unikátny identifiátor je unikátny kód (code). Tento unikátny kód bude používaný pre potreby unikátnej identifikácie inštancie entity pre potreby interakcie s užívateľom - bude napríklad súčasťou URL, alebo názvu súboru a tak ďalej. Tento unikátny kód sa nachádza len v entitách, s ktorými bude užívateľ priamo interagovať.

Väzby medzi entitami budú v aplikácii realizované technikou takzvaných cudzích kľúčov (z anglického foreign keys). Tieto cudzie kľúče implementujú vzťah medzi entitami na databázovej úrovni [23].

V prípade väzby typu 1 : N (N je celé kladné číslo vrátane nuly) medzi entitou alfa a entitou beta (jedna inštancia entity alfa je asociovaná s N inštanciami entity beta) obsahuje entita beta atribút cudzí kľúč, ktorý odkazuje na primárny identifikátor (kľúč) entity alfa. Autor práce zavádza konvenciu, kedy cudzí kľúč odkazujúci na primárny kľúč druhej entity bude mať názov vo formáte názov-entity_id (v hore popisovanom prípade alfa_id).



Obr. 3.3: Väzba typu 1 : N

V prípade väzieb typu N : M (N aj M sú celé kladné čísla vrátane nuly) medzi entitou alfa a entitou beta (jedna inštancia entity alfa môže byť asociovaná s M inštanciami entity

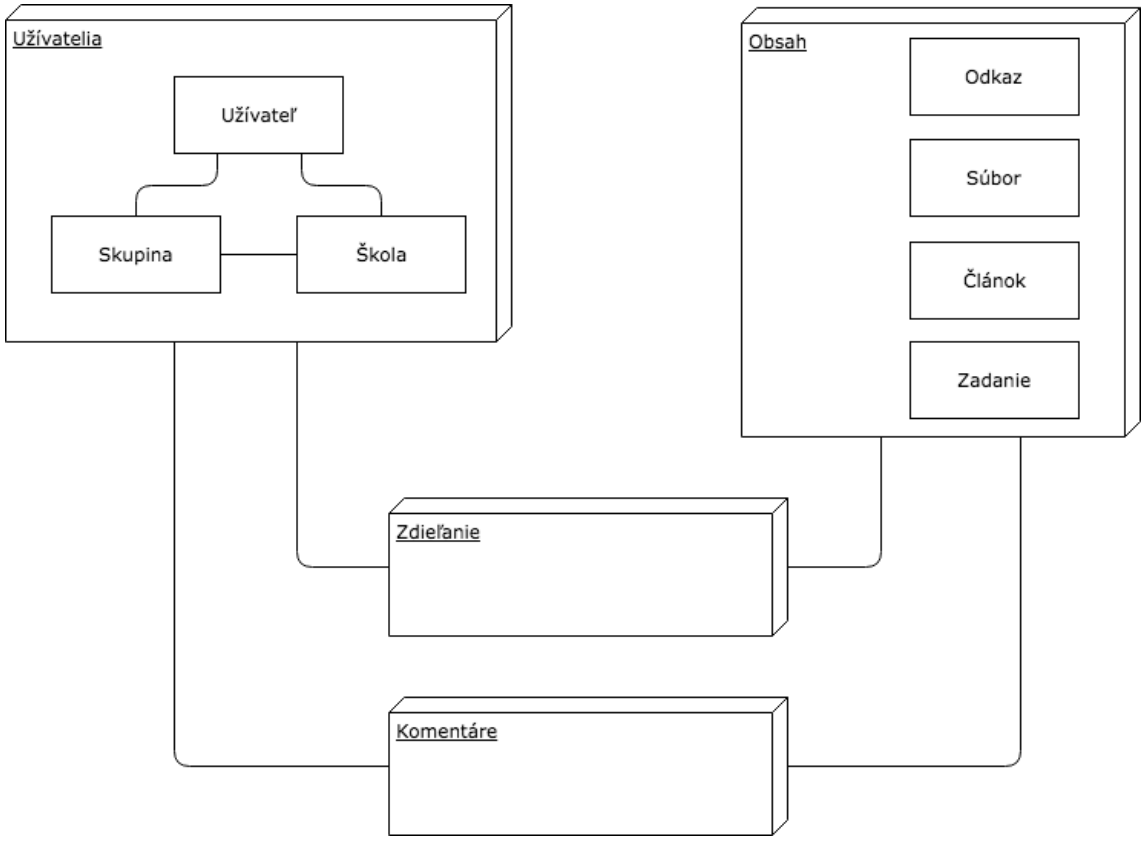
beta, ale zároveň aj každá inštancia entity beta môže byť asociovaná s N inštanciami entity alfa) bude použitá technika spojovacích entít (z anglického pivot entity, alebo association entity) [5].

Pri tejto technike sa cudzie kľúče nachádzajú v samostatnej entite, ktorá reprezentuje asociáciu medzi dvoma entitami.

Obr. 3.4: Väzba typu N : M

Každá asociácia medzi inštanciou entity alfa a inštanciou entity beta je reprezentovaná jednou inštanciou asociačnej entity. Autor práce zavádza konvenciu pre pomenovanie asociačnej entity, kde každá asociačná entita je pomenovaná spojením názvov entít, medzi ktorými definuje asociáciu v abecednom poradí (vo vyššie popisovanom prípade pre asociáciu medzi entitami Alfa a Beta bude asociačná entita nazvaná AlfaBeta)

Entity obsahujú aj metódy, poskytujú jednoduchý spôsob prepojovania inštancií entít v aplikácii. Každá metóda reprezentuje jednu väzbu v databázovom modeli medzi entitou, ktorej súčasťou sú tieto metódy a ďalšími entitami, s ktorými má daná entita definovanú väzbu v dátovom modeli.



Obr. 3.5: Dátový model aplikácie

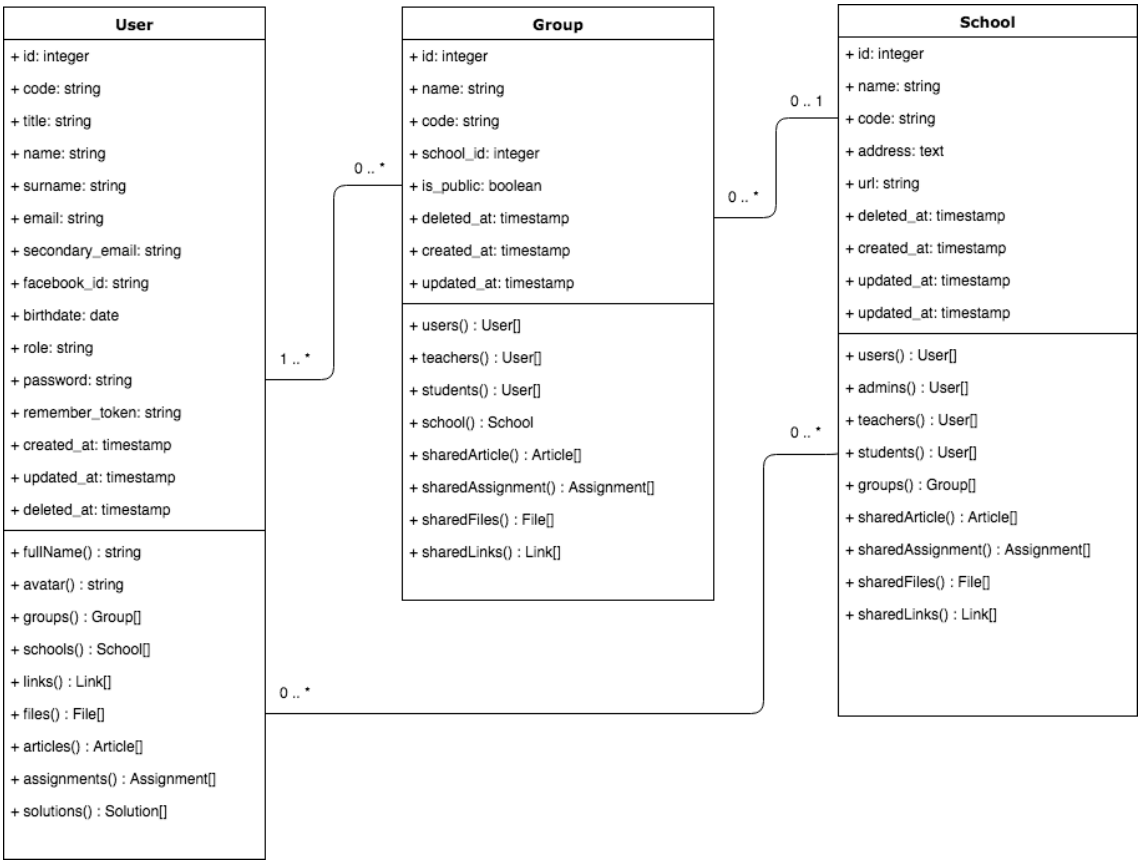
Dátový model aplikácie sa skladá z entít pokrývajúcich tri základné oblasti aplikácie. Prvú z týchto častí tvoria entity reprezentujúce objekty užívateľov a všetkých väzieb medzi nimi. Táto skupina entít bude v ďalšom texte označovaná názvom Užívatelia. Druhú skupinu entít tvoria entity reprezentujúce obsah, ktorý učitelia v aplikácii vytvorili, aby ho mohli zdieľať svojim študentom. Tieto entity zdieľajú spoločné vlastnosti a možnosť byť zdieľaný alebo komentovaný užívateľmi.

Tretiu skupinu tvoria entity reprezentujúce objekty, ktoré tvoria väzby a prepojenia medzi entitami zo skupiny užívatelia a entitami zo skupiny obsah.

3.3 Užívatelia

Skupinu entít užívatelia tvoria tri entity. Tieto entity reprezentujú samotných užívateľov a dva druhy skupín, v ktorých sa užívatelia v aplikácii môžu spájať. Vrámci tejto skupiny entít sú zadefinované aj väzby medzi jednotlivými entitami.

Na základe funkčných požiadaviek je medzi entitou Užívateľ (*User*) a entitou Skupina (*Group*) väzba N : M. Táto väzba znamená, že každý z užívateľov môže byť členom M (M je kladné číslo, vrátane nuly) skupín, zatiaľ čo každá skupina môže obsahovať N (N je kladné číslo vyššie ako nula) užívateľov.



Obr. 3.6: Model užívateľov aplikácie

Medzi entitou Skupina (*Group*) a entitou Škola (*School*) je na základe funkčných požiadaviek väzba N : (0 - 1). Táto väzba reprezentuje prepojenie, kedy každá skupina patrí do jednej zo škôl, alebo do žiadnej. Každá škola zároveň môže obsahovať N (N je kladné číslo, vrátane nuly) skupín.

Medzi entitou Užívateľ (*User*) a entitou Škola (*School*) je väzba typu N : M, ktorá vyplýva z funkčných požiadaviek. Táto väzba definuje vlastnosť užívateľa byť členom M (M je kladné celé číslo, vrátane nuly) škôl. Zároveň táto väzba popisuje, že každá škola môže mať N (N je kladné celé číslo vrátane nuly) užívateľov.

3.3.1 Užívateľ (*User*)

Užívateľ je základná entita reprezentujúca užívateľov systému. Táto entita je využívaná pre autentifikáciu a autorizáciu užívateľov systému, obsahuje teda atribúty email, heslo (*password*), token pre zapamätanie prihlásenia (*remember_ token*) a atribút rola (*role*) pre definíciu roly užívateľa. Táto entita zároveň slúži pre uchovanie všetkých profilových informácii o užívateľovi. Jeho akademický titul, meno, priezvisko a taktiež sekundárny email a identifikátor v aplikácii messenger pre účely oznámení zo systému.

Entita užívateľ obsahuje aj metódy, ktoré poskytujú ďalšie funkcie spojené s inštanciou užívateľa - celé meno (*fullName*) používateľa pozostávajúce z titulu, mena a priezviska. Druhou funkciou je funkcia poskytujúca webovú adresu profilového obrázku (*avatar*) užívateľa reprezentovaného danou inštanciou.

3.3.2 Skupina (*Group*)

Entita skupina reprezentuje skupinu užívateľov, ktorí budú v aplikácii zdieľať obsah. Typicky takúto skupinu tvoria žiaci jednej školskej triedy, alebo vyučovacej skupiny a ich učiteľ, alebo jedna kategória súťažiacich a porota.

Táto entita okrem unikátnych identifikátorov má aj atribút názov (*name*) a atribút verejnosti (*is_public*). Atribút verejnosti slúži na odlíšenie verejných skupín, do ktorých sa môže užívateľ zapojiť z vlastnej iniciatívy, alebo súkromných skupín, do ktorých môže užívateľov pridávať len správca skupiny. Entita skupiny tiež obsahuje atribút indentifikujúci školu (*school_id*), ktorej je skupina súčasťou.

3.3.3 Škola (*School*)

Entita škola reprezentuje školu využívajúcu popisovanú aplikáciu. Okrem unikátnych identifikátorov táto entita obsahuje atribúty pre popis školy. Tieto atribúty sú názov školy (*name*), adresa školy (*address*) a taktiež url adresa internetového sídla školy (*url*).

3.4 Entity skupiny obsah

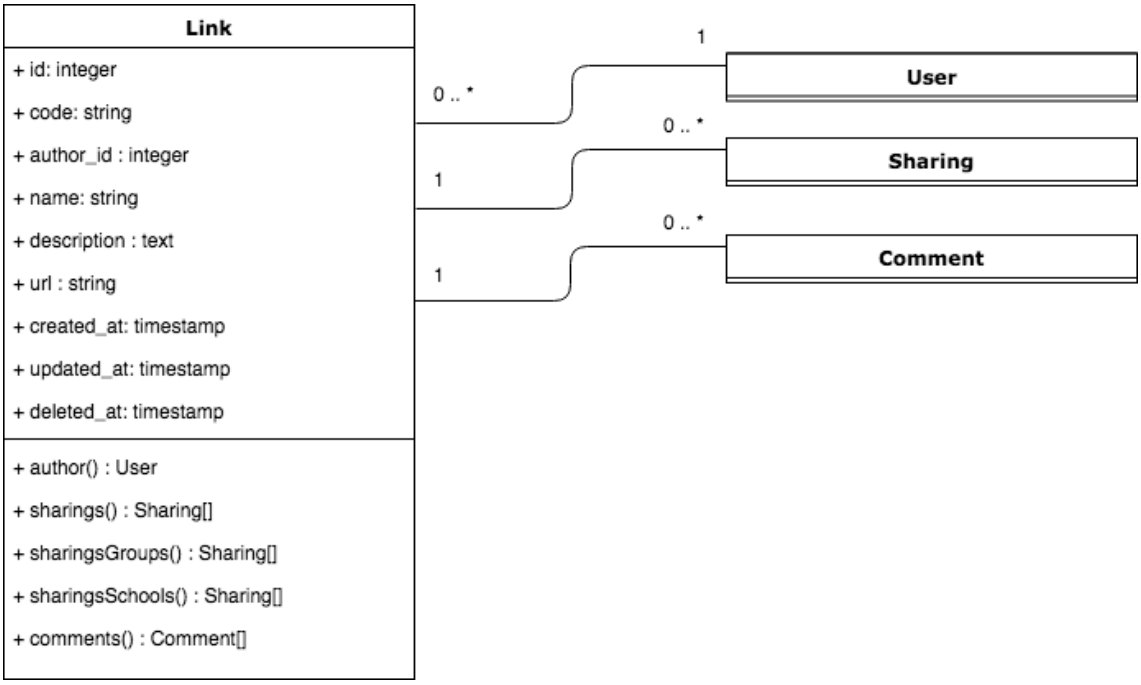
Skupinu entít obsah tvoria štyri entity reprezentujúce štyri typy obsahu, ktorý môže byť v aplikácii zdieľaný medzi užívateľmi. Táto skupina entít sa s vysokou pravdepodobnosťou bude v budúcnosti rozširovať o ďalší druh obsahu, ktorý budú môcť užívatelia pomocou aplikácie zdieľať.

Všetky entity v tejto skupine obsahujú okrem unikátnych identifikátorov (*id, code*) aj názov (*name*) a krátky popis (*description*) obsahu, ktorý reprezentuje daná entita. Tento názov a popis bude použitý pre výpis obsahu zdieľaného s užívateľom, preto je potrebné, aby tieto atribúty mali všetky entity reprezentujúce obsah zdieľaný v aplikácii.

Všetky entity reprezentujúce obsah majú zadefinovanú väzbu N : 1 s entitou užívateľ. Táto väzba popisuje autorstvo obsahu vytvoreného v aplikácii. Na základe tejto väzby môže byť jeden užívateľ autorom N (N je kladné číslo vrátane nuly) inštancií obsahu a každá inštancia obsahu musí mať práve jedného autora.

Ďalšie dve väzby, ktoré majú zadefinované entity reprezentujúce obsah zdieľaný v aplikácii sú typu 1 : N s entitami zdieľanie (*Sharing*) a komentár (*Comment*). Tieto väzby znamenajú, že ku každému obsahu v aplikácii môže existovať N (N je kladné číslo, vrátane nuly) zdieľaní a komentárov. Taktiež z tejto väzby vyplýva, že pre každý komentár a zdieľanie existuje práve jedna inštancia obsahu, ktorej sa daný komentár, alebo zdieľanie týka.

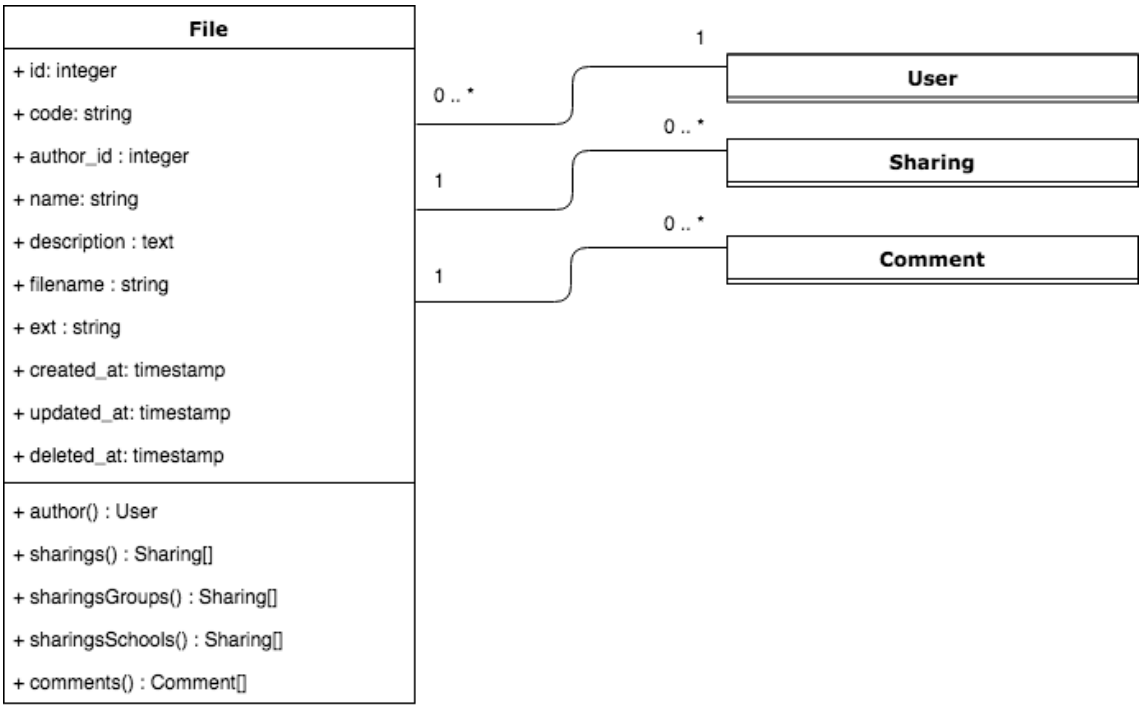
3.4.1 Odkaz (*Link*)



Obr. 3.7: Model entity Odkaz

Odkaz je entita reprezentujúca najjednoduchšiu formu obsahu, ktorú pomocou popisovanej aplikácie môžu užívatelia zdieľať. Jediný atribút entity, ktorý je špecifický pre túto entitu vrámci skupiny Obsah, je adresa (*url*) zdieľaného odkazu.

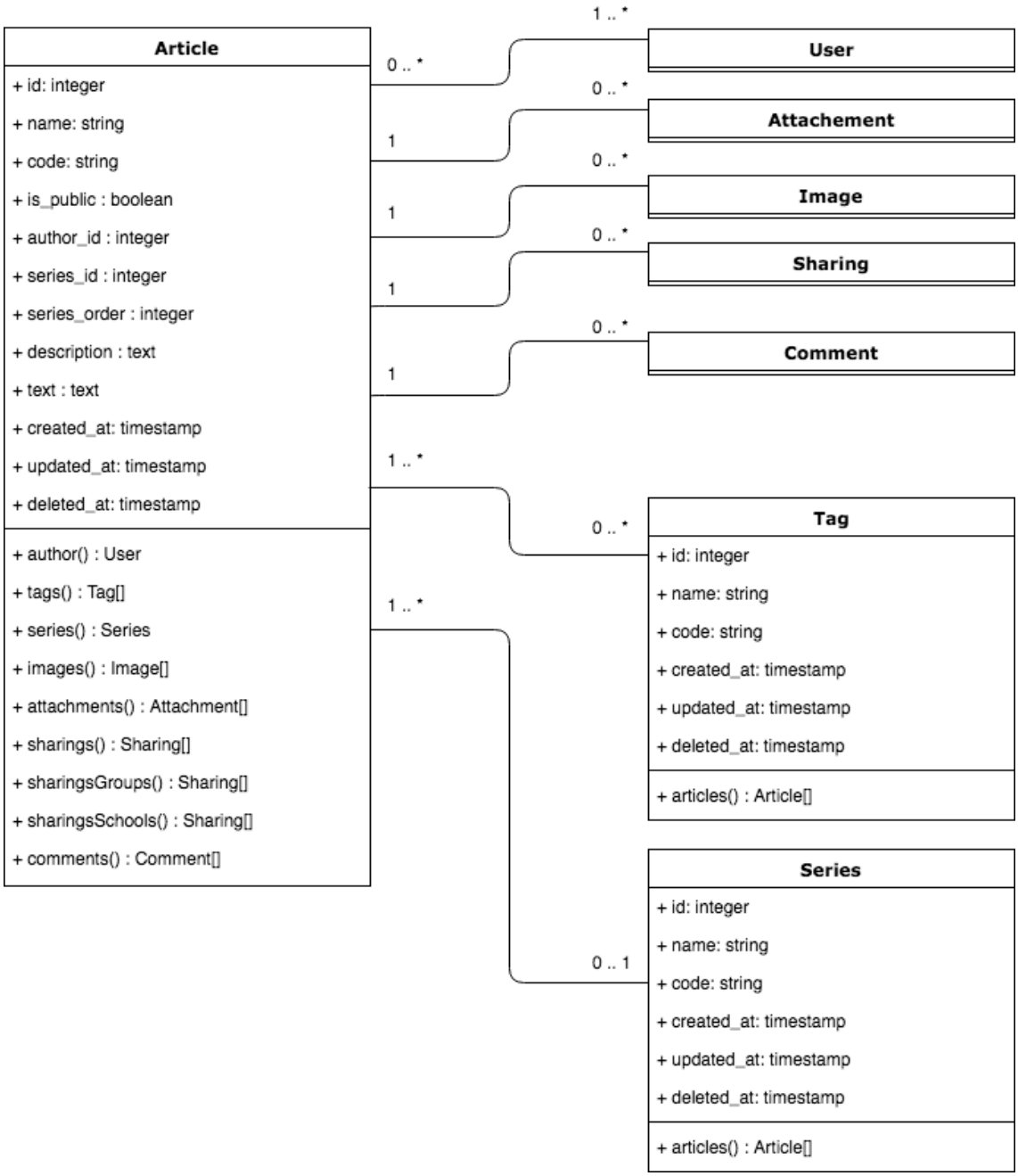
3.4.2 Súbor (*File*)



Obr. 3.8: Model entity Súbor

Entita súbor reprezentuje súbor, ktorý môžu užívatelia v aplikácii zdieľať. Táto entita má len dva atribúty, ktoré sú špecifické pre túto entitu vrámci skupiny Obsah. Tieto atribúty sú názov súboru (*filename*) a prípona (*ext*) zdieľaného odkazu.

3.4.3 Článok (*Article*)



Obr. 3.9: Model entity Článok

Článok je entita reprezentujúca komplexný obsah, ktorý môže byť zdieľaný v aplikácii. Táto entita má okrem štandardných atribútov spoločných pre všetky entity reprezentujúce obsah v aplikácii atribúty pre text (*text*) obsahu a aj označenie, či sa jedná o verejný

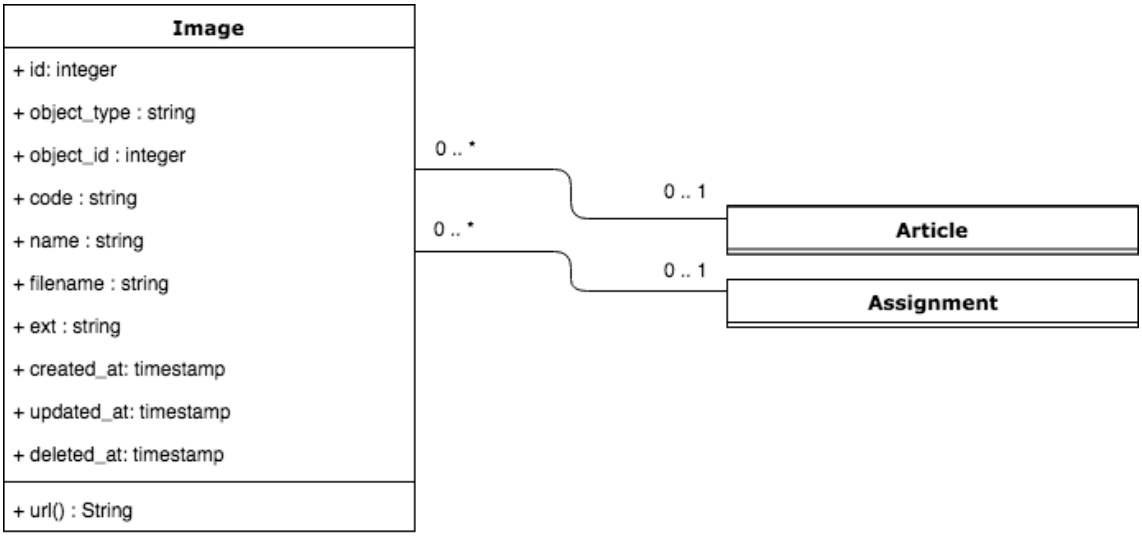
(*is_public*) obsah, ktorý bude zobrazený na hlavnej stránke aplikácie, alebo sa jedná o súkromný obsah, ktorý na to, aby si ho iný užívatelia mohli prehliadať, musí byť tento obsah zdieľaný do skupiny, ktorej sú títo užívatelia členmi. Entita článok má zadefinovanú väzbu N : M s entitou Značka (*Tag*). Táto väzba znamená, že každá inštancia entity Článok môže byť označená M (M je kladné číslo, vrátane nuly) značkami a každou značkou môže byť označené N (N je kladné číslo, vrátane nuly) článkov.

Entita značka (*Tag*) reprezentuje značku, ktorou môže byť označený článok. Tieto značky v prípade, že budú článkom správne pridelené, pomôžu užívateľom na prvý pohľad spoznať, čo sa nachádza v texte článku.

Entita článok má väzbu tyou N : (0 - 1) s entitou séria (*Series*) článkov. Z tejto väzby vyplýva, že každý článok môže mať väzbu s jednou sériou článkov a každá séria článkov môže obsahovať N (N je kladné číslo, vyššie ako nula) článkov. V súvislosti s touto väzbou medzi článkami a sériami článkov existujú v entite článok dva atribúty. Sú to identifikátor série (*series_id*) a poradie v sérii (*series_order*), podľa ktorého budú články zoradené vrámci série.

Entita séria (*Series*) reprezentuje sériu článkov. Napríklad v prípade viacerých článkov týkajúcich sa spoločnej témy.

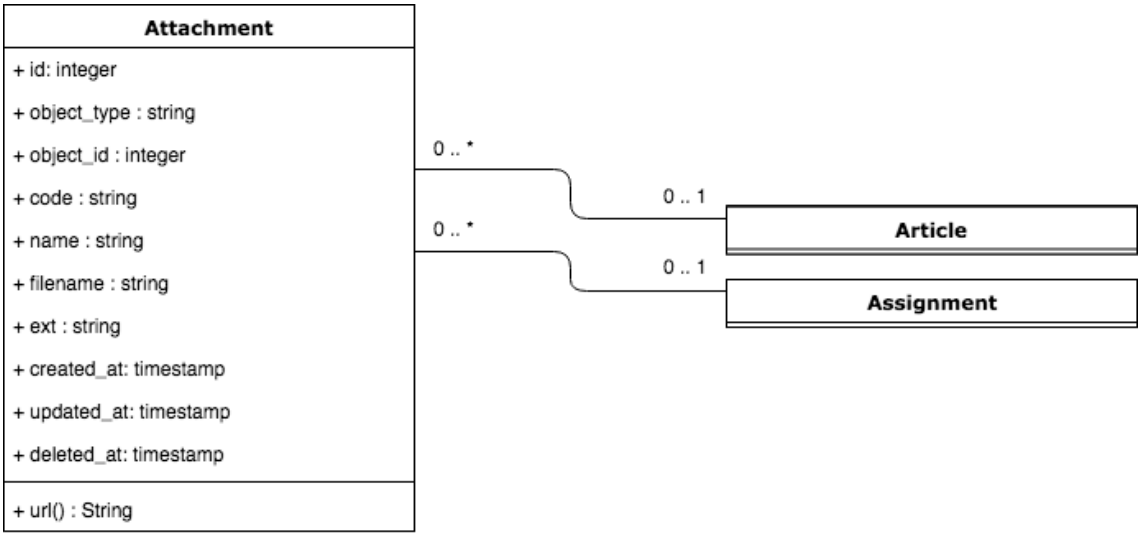
Entita článok má taktiež definované väzby 1 : N (kde N je celé kladné číslo, vrátane nuly) s entitou obrázok (*Image*) a entitou príloha (*Attachment*). Tieto entity dopĺňajú obsah článku.



Obr. 3.10: Model entity Obrázok

Entita obrázok (*Image*) reprezentuje obrázok, ktorý je súčasťou obsahu článku alebo zadania. Táto entita obsahuje atribúty určujúce článok alebo zadanie, s ktorým je prepojená daná inštancia obrázka. Tieto atribúty sú identifikátor objektu (*object_id*) a druh objektu (*object_type*) určujúce, či je tento objekt článok, alebo zadanie.

Obrázok obsahuje atribúty nevyhnutné pre identifikáciu obrázka ako súboru. Sú to názov súboru (*filename*) a prípona súboru (*ext*).



Obr. 3.11: Model entity Príloha

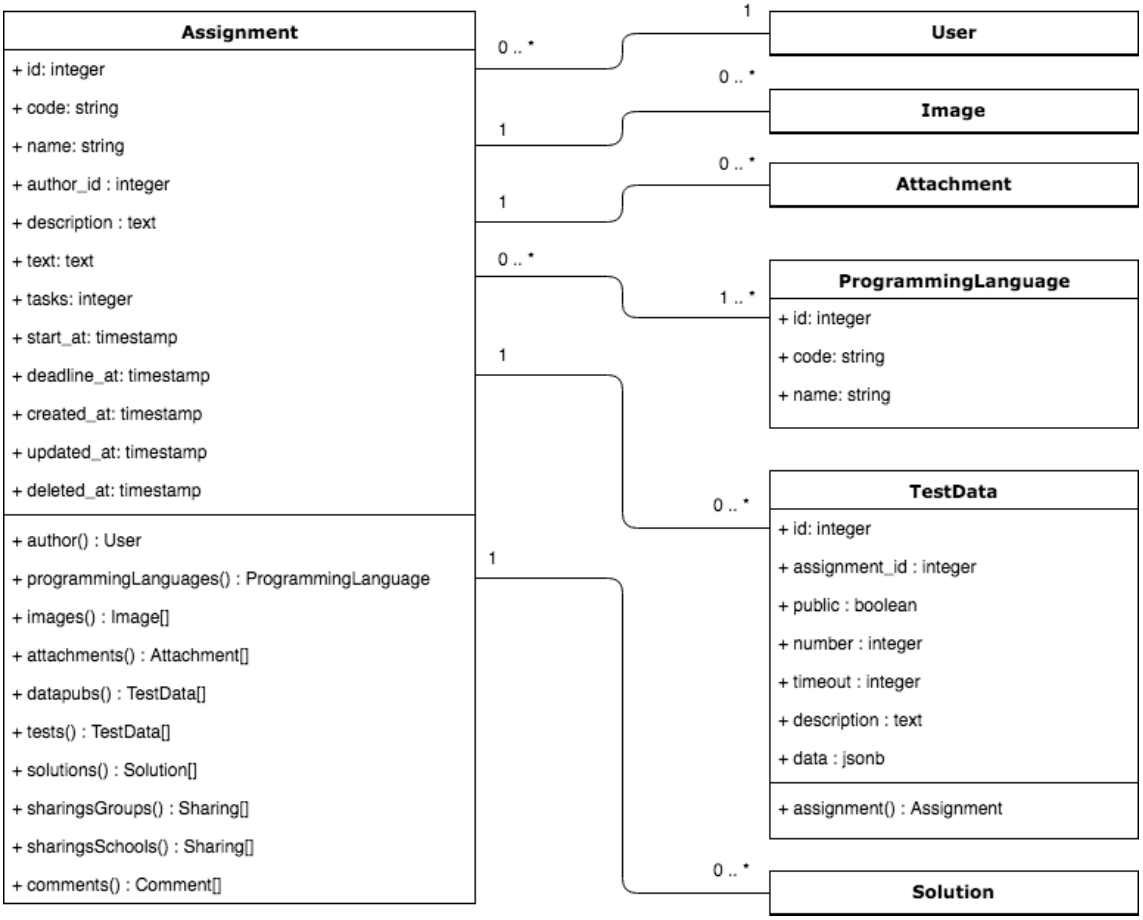
Entita príloha (*Attachment*) reprezentuje súbor, ktorý je súčasťou obsahu článku alebo zadania. Táto entita, rovnako ako entita obrázok, obsahuje atribúty určujúce článok alebo zadanie, s ktorým je prepojená daná inštancia prílohy.

Príloha obsahuje aj atribúty identifikujúce prílohu ako súbor, názov súboru (*filename*) a prípona súboru (*ext*).

3.4.4 Zadanie (*Assignment*)

Entita zadanie (*Assignment*) reprezentuje zadania, ktoré budú pomocou aplikácie učitelia zdieľať svojim žiakom, porota súťažiacim a podobne. Zadanie je najkomplexnejším obsahom, aký môže byť zdieľaný v popisovanej aplikácii, preto aj entita zadanie je najzložitejšou entitou v navrhovanom dátovom modeli. Okrem štandardných atribútov, ktoré majú všetky entity v skupine entít obsah má entita zadanie atribúty o počte úloh v zadaní (*tasks*) a informácie o termíne, kedy je možné začať odovzdávať riešenia tohto zadania (*start_ at*) a aj termíne, do kedy je možné odovzdávať riešenie (*deadline_ at*). Rovnako ako entita článok má aj entita zadanie atribút text, ktorý bude obsahovať textový popis zadania.

Entita zadanie má rovnako ako entita článok definované väzby 1 : N (kde N je celé kladné číslo, vrátane nuly) s entitami obrázok (*Image*) a entitou príloha (*Attachment*). Tieto entity dopĺňajú obsah textu zadania rovnako ako v prípade entity článok. Detailne boli entity obrázok a príloha popísané pri popise entity článok. Zadanie má zadefinovanú väzbu typou N : M (N je kladné číslo vrátane nuly, M je kladné číslo väčšie ako nula) s entitou typu programovací jazyk (*Programming language*). Táto väzba znamená, že každé zadanie je asociované s M programovacími jazykmi a každý programovací jazyk je asociovaný s N zadaniami. Táto väzba v praxi reprezentuje programovacie jazyky, ktoré sú dostupné pre odovzdávanie riešení danej inštancie zadania. Entita programovací jazyk reprezentuje jeden programovací jazyk, v ktorom je možné realizovať a následne odovzdávať riešenia zadaní



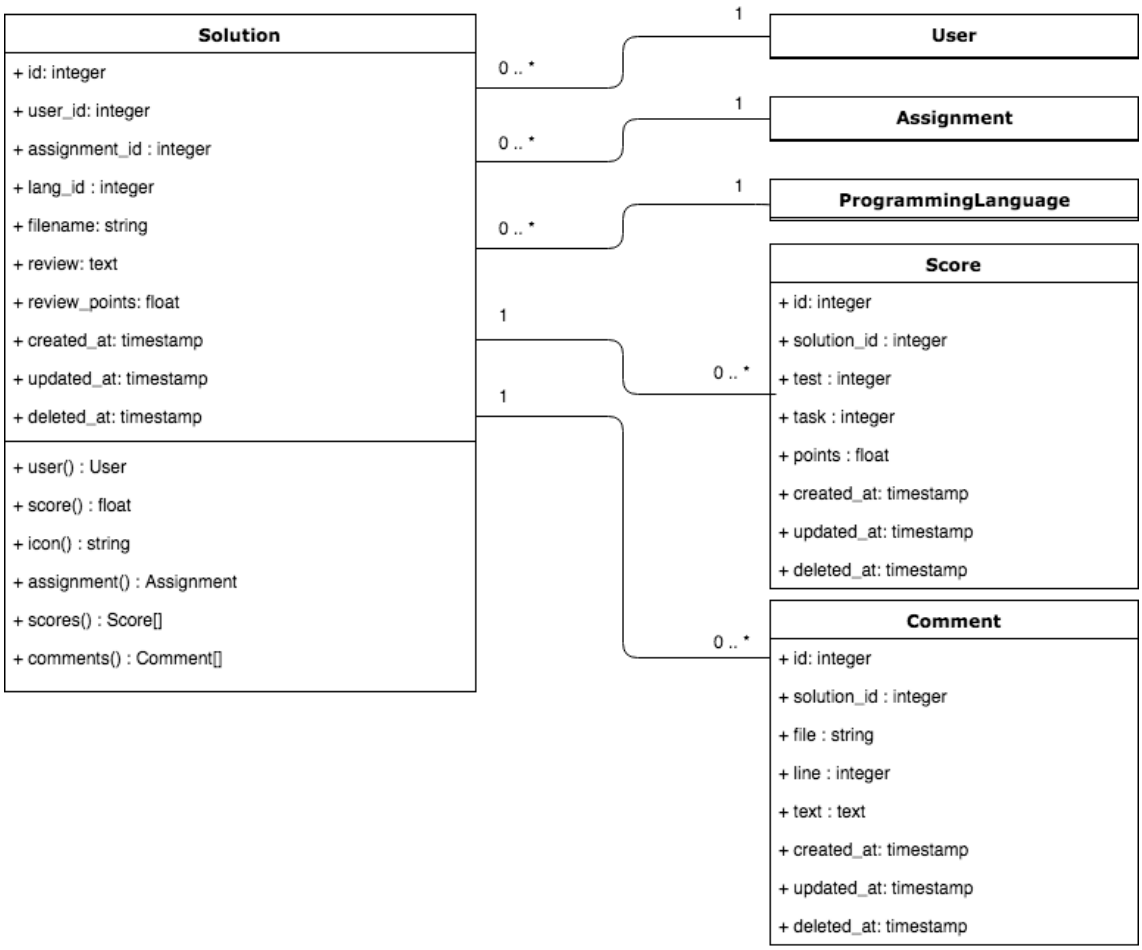
Obr. 3.12: Model entity Zadanie

v aplikácii. Táto entita sa vyznačuje svojimi atribútmi názov (*name*) a kód (*code*), ktoré identifikujú daný programovací jazyk.

Zadanie má tiež väzbu typu 1 : N (N je celé kladné číslo, vrátane nuly) s entitou testovacie dáta (*TestData*). Táto väzba znamená, že každá inštancia entity zadanie je asociovaná s N inštanciami testovacie dáta a každá inštancia entity testovacie dáta musí byť asociovaná s práve jedným zadaním. Entita testovacie dáta (*TestData*) reprezentuje vstupné a výstupné dáta určené pre testovanie riešení zadaní. Táto entita obsahuje atribúty časový limit (*timeout*), ktorý definuje, ako dlho môže užívateľské riešenie bežať s danými vstupnými a výstupnými dátami. Obsahuje aj atribúty číslo (*number*) a popis (*description*) určené pre identifikáciu sady vstupných a výstupných dát. Inštancie entity testovacie obsahujú aj binárny (áno - nie) atribút definujúci dostupnosť inštancie (*public*), ktorý rozhoduje o účele tejto inštancie. Inštancia môže tvoriť takzvané vzorové, alebo verejné dáta, v tomto prípade je atribút verejnosti nastavený na kladnú hodnotu (áno - True). Tieto dáta sú užívateľom aplikácie dostupné a slúžia pre manuálne otestovanie aplikácie bez nutnosti odovzdávania riešenia nahraním do aplikácie. Druhým prípadom je inštancia, ktorej účelom sú testovacie dáta, ktoré nie sú dostupné užívateľom a využívajú sa pre automatické testovanie riešení odovzdaných

užívateľmi aplikácie v takom prípade je atribút verejnosti nastavený na hodnotu (nie - False). Samotné vstupné a výstupné dáta pre testovanie riešení obsahujú hodnotu atribútu data. Hodnotu tohto atribútu tvoria štrukturované dáta, preto bol tento atribút navrhnutý ako typ JSONB. Tento dátový typ je vhodnou voľbou pre uchovávanie komplexne formátovaných dát [17].

Entita zadanie má definovanú aj väzbu typu 1:N (N je celé číslo vrátane nuly) s entitou riešenie (*solution*). Z tejto väzby vyplýva, že ku každej inštancii entity zadanie prislúcha N inštancií entity riešenie. Zároveň každá inštancia entity riešenie má väzbu s práve jednou inštanciou entity zadanie.



Obr. 3.13: Model entity Riešenie

Entita riešenie (*Solution*) reprezentuje riešenie inštancie zadania užívateľom aplikácie.

Táto entita má okrem atribútov nevyhnutných pre definície väzieb tejto entity a atribútov, ktoré sú spoločné pre všetky entity v skupine entít obsah atribúty názov súboru (*filename*), ktorý bol odovzdaný do odovzdávacieho systému. Entita obsahuje aj atribúty obsahujúce informácie o manuálnom ohodnotení riešenia. Sú to manuálne ohodnotenie (*review*) typu text a manuálne bodové ohodnotenie (*review_points*).

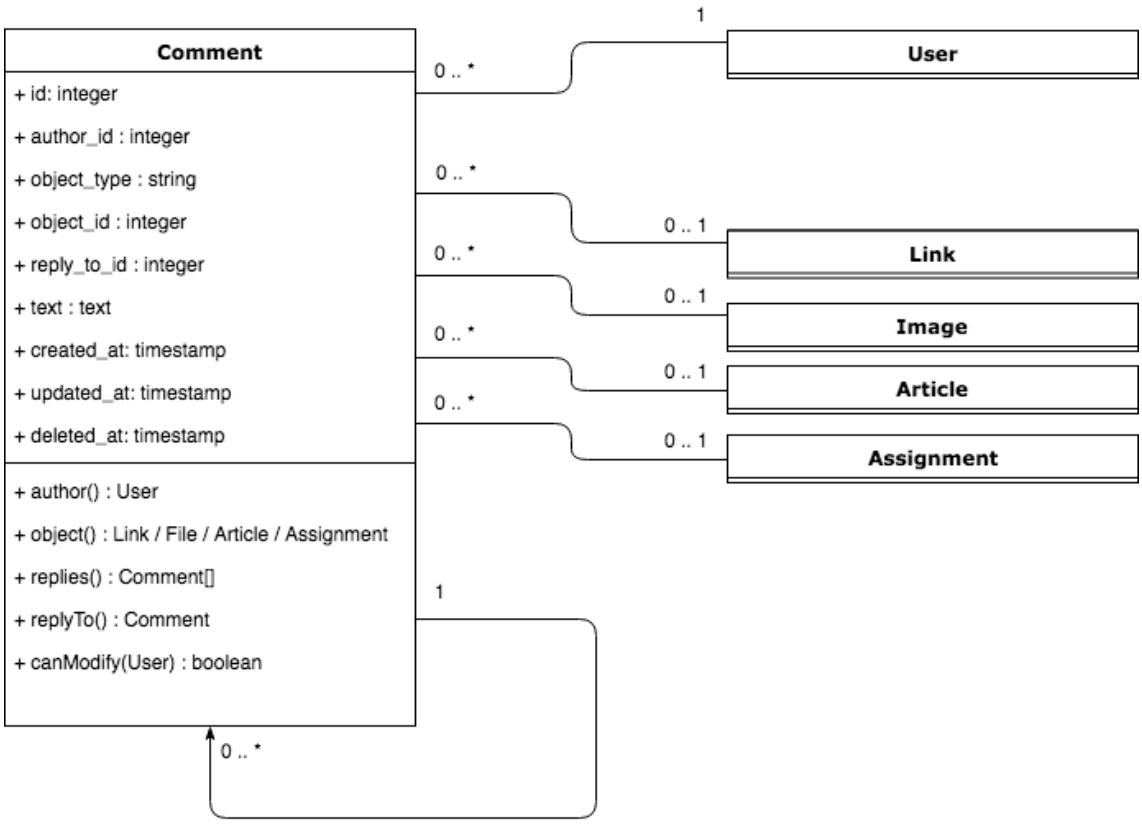
Riešenie má väzbu typu N : 1 (N je kladné celé číslo vrátane nuly) s entitou užívateľ a entitou zadanie. Každá inštancia riešenia prislúcha práve jednej inštancii entity zadania a práve jednej inštancii entity užívateľ. Jedna inštancia užívateľa a taktiež jedna inštancia zadania môže mať väzbu s N inštanciami riešenia.

Entita riešenie zároveň obsahuje väzbu N : 1 (N je kladné celé číslo väčšie ako nula) s entitou programovací jazyk. Táto väzba slúži na identifikáciu programovacieho jazyka, v ktorom bolo vytvorené dané riešenie. Platí, že každá inštancia entity riešenie môže mať väzbu len s jednou inštanciou entity programovací jazyk a inštancia entity programovací jazyk môže mať väzbu s N inštanciami entity riešenie.

Entita riešenie má taktiež zadefinovaný vzťah typu 1 : N (N je celé kladné číslo vrátane nuly) s entitou typu hodnotenie (*Score*). Jedna inštancia entity riešenie má väzbu s N inštanciami entity hodnotenie. Entita hodnotenie reprezentuje výsledok automatického hodnotenia jednej úlohy zadania pomocou jednej sady dát. Táto entita obsahuje atribúty nevyhnutné pre identifikovanie úlohy a sady dát, ktoré výsledok obsahuje a atribút body (points) pre uloženie bodového ohodnotenia úlohy riešenia.

S entitou riešenie má väzbu zadefinovanú aj entita komentár (*Comment*) reprezentujúca komentár udelený riešeniu autorom zadania. Entita komentár obsahuje atribúty nevyhnutné pre identifikáciu riešenia (*solution_id*), súbora (*file*) a riadku (*line*) a taktiež atribút text pre samotný text komentára.

3.4.5 Komentár (*Comment*)



Obr. 3.14: Model entity Príloha

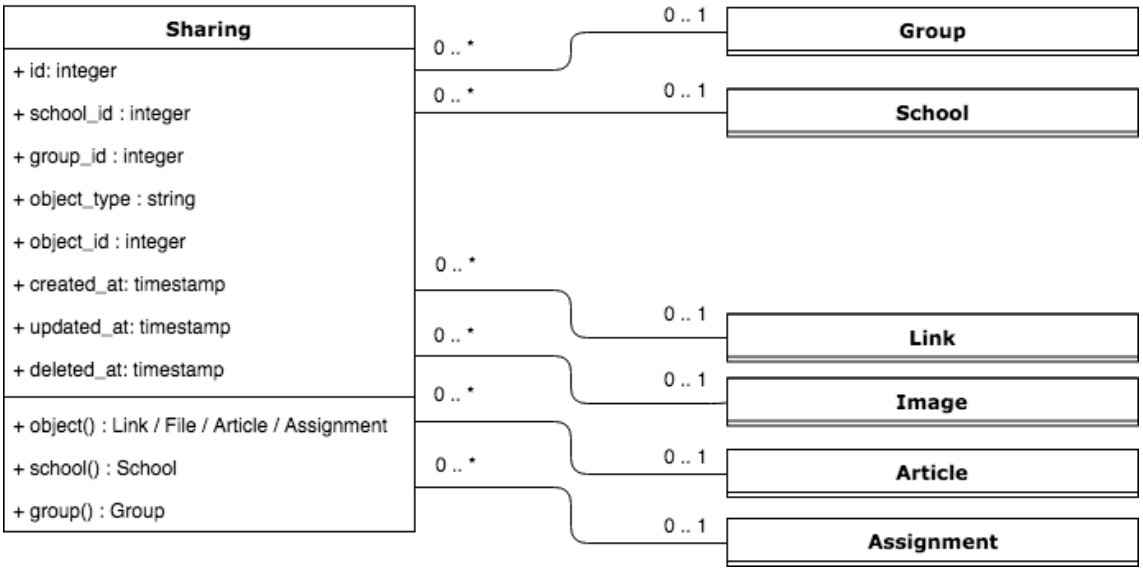
Entita komentár reprezentuje komentár, ktorým užívateľ okomentoval jednu z inštancii entity zo skupiny entít Obsah.

Zároveň má entita definovanú väzbu N : (0 - 1) (N je kladné celé číslo vrátane nuly) s entitami tvoriacimi skupinu entít obsah. Definuje, že jedna inštancia zdieľania má vždy väzbu s jednou inštanciou entít zo skupiny obsah. Každá inštancia obsahu môže byť prepojená s N inštanciami entity zdieľanie. Táto väzba je realizovaná cudzím kľúčom pre entitu obsahu (*object_ id*) a atribútom, ktorý definuje s akým typom obsahu je inštancia komentára asociovaná (*object_ type*). Každá entita obsahu má definovaný unikátny kód, ktorým je daná entita identifikovateľná pri asociácii s inštanciou entity komentár.

Entita komentár obsahuje väzbu typu N : 1 (N je kladné číslo, vrátane nuly) s entitou užívateľ. Táto väzba v praxi znamená, že každý užívateľ môže vytvoriť N komentárov, a každý komentár musí mať práve jedného autora.

Inštancia entity komentár má zároveň väzbu s ďalšími inštanciami typu 1 : N (N je kladné číslo, vrátane nuly). Táto väzba určuje vzťah medzi komentárom a ďalšími komentármi, ktoré tvoria odpoveď na daný komentár.

3.4.6 Zdieľanie (*Sharing*)



Obr. 3.15: Model entity Príloha

Entita zdieľanie reprezentuje zdieľanie inštancie jednej z obsahových entít užívateľskej skupine, alebo škole.

Táto entita má definovanú väzbu N : (0 - 1) (N je kladné celé číslo vrátane nuly) s entitami skupina a škola. Táto väzba definuje skupinu alebo školu, v ktorej bude zdieľaný obsah. Jedna inštancia má na základe tejto väzby prepojenie s jednou školou, alebo skupinou zatiaľ čo každá inštancia školy a skupiny môže mať prepojenie s N inštanciami zdieľania.

Zároveň má entita definovanú väzbu N : (0 - 1) (N je kladné celé číslo vrátane nuly) s entitami tvoriacimi skupinu entít obsah. Definuje, že jedna inštancia zdieľania má vždy väzbu s jednou inštanciou entít zo skupiny obsah. Každá inštancia obsahu môže byť prepojená s N inštanciami entity zdieľanie. Táto väzba je realizovaná rovnakým spôsobom ako väzba medzi entitou komentár a entitami obsahu.

3.5 Grafické rozhranie

Pri návrhu aplikácie bol zvolený minimalistický princíp. Aplikácia bola navrhnutá tak, aby užívateľ nikdy neostal na pochybách, ako vykonať nasledujúcu akciu.

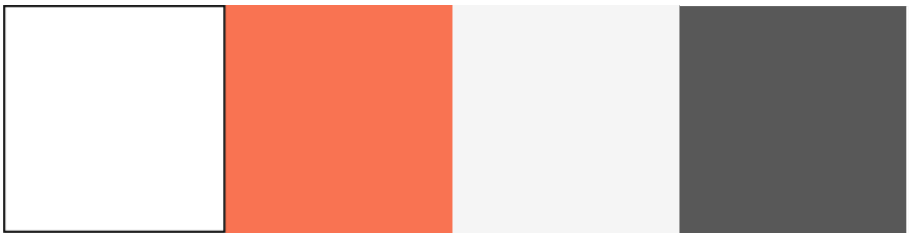
Rozhranie bolo navrhované princípom Mobile-First. Tento princíp kladie do popredia vývoj pre menšie displeje, ktorými väčšinou disponujú mobilné zariadenia. Zásadou tohto princípu je poskytnúť plnohodnotný užívateľský zážitok užívateľom používajucim aplikáciu na zobrazovačoch s malou uhlopriečkou a nízkym rozlíšením. Pre zobrazovače s vyšším rozlíšením sa môže zobrazenie grafického rozhrania zmeniť, alebo obsahovať prvky navyše.

Týmto prístupom sa zaručí maximálny uživateľský zážitok bez ohľadu na to, na akom zariadení bude užívateľ aplikáciu používať [21].

Pri návrhu užívateľského rozhrania aplikácie sa autor inšpiroval takzvaným flat designom. Tento štýl bol predstavený pri uvedení Material Design štýlu, ktorý je základným štýlom všetkých aplikácii od špoločnosti Google a taktiež mobilného operačného systému Android. Druhý predstaviteľ flat designu bol uvedený, keď spoločnosť Microsoft uviedla svoje Language Metro [3].

Hlavnou črtou tohto štýlu užívateľského rozhrania sú jednoduché jednofarebné plochy. Aplikácia je preto navrhnutá s jednoduchou farebnou schémou pozostávajúcou zo štyroch farieb.

3.5.1 Farebná schéma aplikácie



Obr. 3.16: Farebná schéma aplikácie

Hlavné pozadie stránky bude tvoriť biela farba (#FFFFFF). Pozadie niektorých zvýraznených prvkov (aktívne prvky, tlačidlá, orientačná navigácia a podobne) bude tvoriť svetlo sivá farba (#F5F5F5). Hlavnou farbou popredia - textu je tmavo sivá farba (#585858). Sekundárnou farbou popredia je tehlová farba (#F97352). Touto farbou budu zobrazené nadpisy, dôležité akčné prvky a podobne.

3.5.2 Rozloženie aplikácie

Grafické rozhranie aplikácie je rozložené do troch hlavných elementov so šírkou vypĺňajúcou celú šírku prehliadača.

Obr. 3.17: Rozloženie aplikácie

V smere zhora dole tieto elementy tvoria hlavičku aplikácie, hlavný obsah aplikácie a pätičku aplikácie.

3.5.2.1 Hlavička

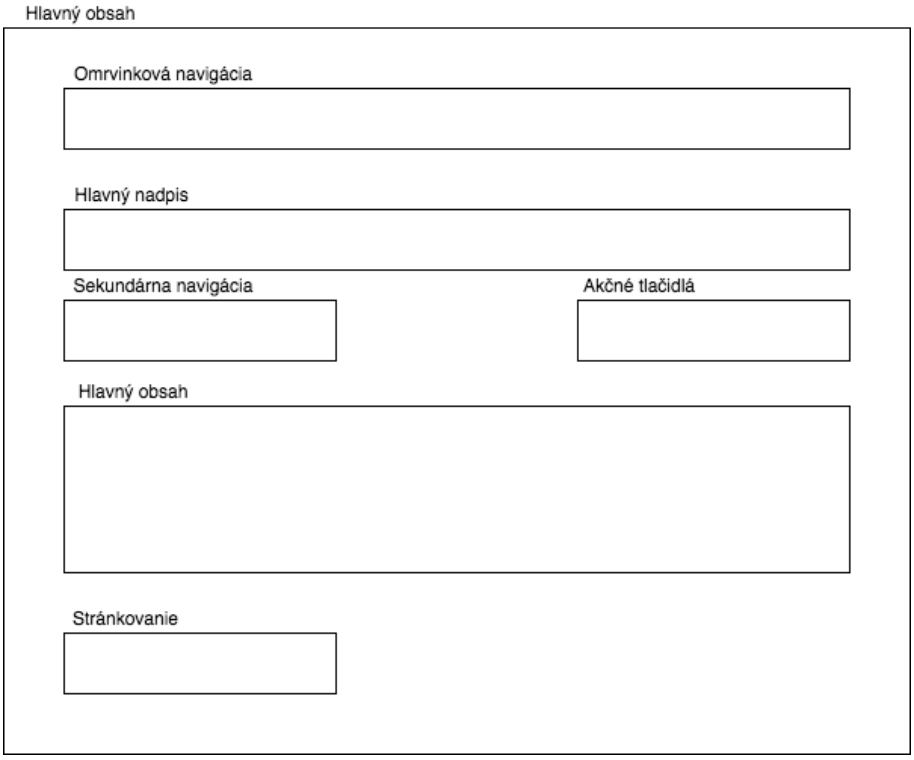
Hlavička aplikácie bude blok obsahu vypĺňajúci celú šírku prehliadača. Bez ohľadu na to, ako bude obsah stránky posunutý vertikálnym posuvníkom obsahu, bude hlavička vždy zobrazená v hornej časti prehliadača. Blok hlavičky je horizontálne rozdelený na ľavú a pravú časť hlavičky. Ľavú časť hlavičky tvorí logo aplikácie. Pravú časť hlavičky bude tvoriť hlavná navigácia aplikácie. Táto hlavná navigácia obsahuje odkazy do základných častí aplikácie dostupných aj neprihlásenému užívateľovi. Tieto odkazy budú odkazy pre zobrazenie prezentačnej stránky aplikácie, ktorá bude tvoriť zároveň domovskú stránku aplikácie. Ďalšie odkazy budú určené pre zobrazenie verejných článkov a verejných zadaní. Jedná sa o články a zadania, ktoré nie sú určené pre konkrétnych žiakov na školách, ale ktoré sa ich autori rozhodli zverejniť a pomôcť tým celej komunite programátorov. Ďalej sa v hlavnej navigácii bude nachádzať odkaz na podstránku, na ktorej sa existujúci aj potencionálny užívatelia dozvedia detailné informácie o funkcii aplikácie. Posledný odkaz vľavo bude v prípade neprihláseného užívateľa smerovať na prihlasovací a registračný formulár.

V prípade prihláseného užívateľa bude posledný odkaz smerujúci na prihlasovací a registračný formulár nahradený tlačidlom, ktorého popis bude tvoriť meno prihláseného užívateľa. Kliknutím na toto tlačidlo sa otvorí kontextová ponuka, ktorá bude obsahovať odkazy spojené s profilom a aktivitou prihláseného užívateľa v aplikácii. Tieto odkazy budú rozdelené do troch skupín. Prvá skupina bude spojená s profilom užívateľa. Budú ju tvoriť odkazy na zmenu profilových informácii, zmenu hesla a výpis oznámení užívateľa. Druhá skupina odkazov bude tvorená odkazmi, ktoré smerujú na stránky, kde budú užívateľovi zobrazené ním vytvorené inštancie entít obsahu, teda odkazy, súbory, články a zadania. Táto skupina

odkazov bude zobrazovaná len užívateľom, ktorí majú prislúchajúcu rolu na to, aby mohli tvoriť obsah v aplikácii (majú inú rolu ako rolu študent). Tretiu skupinu odkazov bude tvoriť odkaz na odhlásenie z aplikácie.

3.5.2.2 Hlavný obsah

Hlavný obsah je blok obsahujúci samotný obsah, ktorý aplikácia prezentuje užívateľovi. Tento blok je tvorený niekoľkými elementmi obsahu, ktoré sú umiestnené pod sebou.



Obr. 3.18: Hlavný obsah aplikácie

Prvým elementom umiestneným v hlavnom obsahu najvyššie je omrvinková navigácia. Omrvinková navigácia používateľovi umožňuje navigovať sa po prejdených stránkach až po aktuálnu, resp. na druhej strane po počiatočnú, hlavnú stránku. Táto navigácia poskytuje pre užívateľa prejdenú cestu začínajúcu od východiskového bodu. Napríklad pri úprave článku s názvom Prvý Článok by omrvinková navigácia vyzerala následovne: Domov - Články - Prvý článok - Úprava

Pod omrvinkovou navigáciou sa bude nachádzať hlavný nadpis obsahu.

Pod hlavným nadpisom sa bude nachádzať navigačná oblasť, ktorá bude horizontálne rozdelená na sekundárnu navigáciu vľavo a akčné tlačidlá vpravo.

Sekundárna navigácia bude obsahovať odkazy na obsah súvisiaci s práve zobrazovaným obsahom. Napríklad v prípade zobrazenia zoznamu užívateľov pre užívateľa s globálnou

rolou správcu systému sa tam zobrazia odkazy pre zobrazenie zoznamu užívateľských skupín a škôl.

Akčné tlačidlá v pravej časti navigačnej oblasti budú tlačidlá vedúce k aktivite súvisiacej so správou práve zobrazovaného obsahu. Napríklad v prípade zobrazenia zoznamu užívateľov bude v tejto oblasti akčné tlačidlo pre vytvorenie nového užívateľa alebo pri zobrazení konkrétneho užívateľa budú v tejto oblasti tlačidlá pre úpravu alebo vymazanie užívateľa.

Hlavný obsah bude blok obsahujúci samotný obsah prezentovaný užívateľovi. V prípade zoznamu inštancií niektorej z dátových entít bude mať formu zoznamu alebo tabuľky.

V prípade, že prezentovaný obsah bude zoznam inštancií entity a týchto inštancií bude viac, ako by bolo možné efektívne a efektne prezentovať v jednej tabuľke alebo zozname, budú tieto inštancie rozdelené do stránok s menším počtom inštancií. Pri takomto rozdelení bude pod hlavným obsahom umiestnená navigácia medzi jednotlivými stránkami tabuľky alebo zoznamu. Táto navigácia je v návrhu rozloženia označená ako stránkovanie.

3.5.2.3 Pätička

Pätička je posledný blok rozloženia stránky zaberajúci celú šírku prehliadača. Pätička stránky na rozdiel od hlavičky nemusí byť zobrazovaná vždy. V prípade príliš dlhého hlavného obsahu, ktorý je dlhší ako maximálny obsah, ktorý je možné umiestniť do okna prehliadača bez použitia vertikálneho posuvníka, sa pätička zobrazí len pri posunutí posuvníka do dolnej polohy. Pätička bude obsahovať informácie o autorstve a licencii aplikácie a bude obsahovať kontaktný email, ktorým je možné kontaktovať autora aplikácie.

Kapitola 4

Implementácia

4.1 Použité nástroje a technológie

4.1.1 Vývojové nástroje

Pri návrhu aplikácie bola využitá online aplikácia draw.io pre kreslenie diagramov a drôtových modelov. Samotná aplikácia bola vyvíjaná v integrovanom vývojovom prostredí PHP Storm od spoločnosti IntelliJ, ktorá poskytuje študentskú licenciu pre tento nástroj.

Pri implementácii aplikácie bol využívaný verzovací nástroj Git. Repozitár obsahujúci zdrojové kódy aplikácie je umiestnený v aplikácii GitHub [**?**].

4.1.2 Jazyk PHP

Autor práce sa rozhodol implementovať aplikáciu v jazyku PHP verzie 7.

Tento programovací jazyk si autor zvolil kvôli širokej komunite, ktorá vyvíja aplikácie v tomto jazyku. Len v jednej zo skupín na sociálnej sieti Twitter je 95 tisíc členov sledujúcich novinky zo sveta PHP [**?**].

Takáto veľká komunita je veľkou výhodou pri realizácii projektov s otvoreným kódom, nakoľko mnoho problémov, s ktorými sa autor aplikácie v jazyku PHP stretne pri realizácii, už boli riešené komunitou.

Ďalšou výhodou takto rozšíreného jazyka, ktorá pomohla autorovi pri výbere sú takzvané softvérové balíčky a správca balíčkov. Tieto balíčky sú fragmenty aplikácii, ktoré riešia konkrétny problém. (https://packagist.org/) Používanie týchto balíčkov je dnes veľmi moderný spôsob ako tvoriť aplikácie, pretože odoberá nutnosť vývojára riešiť problémy, ktoré už boli vyriešené.

Pre balíčky v jazyku PHP existuje správca balíčkov Composer. Táto aplikácia prepája veľké množstvo repozitárov s týmito balíčkami a poskytuje automatický nástroj ako tieto balíčky inštalovať do vyvíjaných aplikácii.

4.1.3 Laravel

Ďalšou veľkou výhodou značne rozšíreného jazyka, akým je PHP, je aj veľké množstvo aplikačných rámcov (z anglického framework). Aplikačný rámec tvorí obecnú základnú štruktúru aplikácie, ktorá vývojárovi poskytuje kostru, do ktorej dopĺňa špecifickú funkcionalitu aplikácie. Výhodou využívania aplikačných rámcov pri tvorbe aplikácii spočíva vo veľkom množstve preddefinovaných funkcií, ktoré už vývojár nemusí znova vytvárať.

Pre vytvorenie aplikácie si autor vybral aplikačný rámec Laravel. Laravel je v súčasnosti najobľúbenejší PHP aplikačný rámec. (https://www.dailyrazor.com/blog/best-php-frameworks/)

Tento aplikačný rámec je vystavaný na základoch aplikačného rámca Symfony, ktorý je najväčším aplikačným rámcom v jazyku PHP.

Autor aplikácie využíva množstvo vstavaných súčastí aplikačného rámca Laravel pre vývoj aplikácie. V neposlednom rade výhodou využívania preverených aplikačných rámcov, ktoré sú využívané vo veľkom množstve aplikácii a podporované veľkou komunitou je vyššie zabezpečenie aplikácie vystavanej okolo takéhoto aplikačného rámca. Aplikácie, ktoré využíva veľká množina užívateľov z rôznych vstiev, majú vyššiu pravdepodobnosť prieniku zabezpečenia. Pri internetových aplikáciách, ktoré bežia v internetovom prehliadači, ktorý je sám o sebe náchylný na chyby jeho autorov, je riziko zraniteľnosti ešte vyššie. Práve pri takýchto aplikáciách je výhodné využívať prvky zabezpečenia integrované v aplikačnom rámci. Pri veľkej komunite vývojárov používajúcich jeden rámec je vyššia pravdepodobnosť objavenia zraniteľnosti jedným z týchto vývojárov. Objavenie tejto chyby sa z pravidla do krátkeho času prejaví vydaním novej verzie aplikačného rámca, alebo len jednej jeho súčasti.

Aplikačný rámec Laravel využíva aplikáciu Composer pre správu jednotlivých súčastí rámca. Používanie správcu balíčkov umožňuje autorovi aplikácie postavenej okolo rámca Laravel jednoduché zabudovanie ďalších balíčkov.

4.1.4 Databáza

Implementovaná aplikácia využíva databázu typu PostgreSQL. Využitie tohto typu databázy má oproti najbežnejšie využívanej databáze MySQL niekoľko výhod. Hlavný rozdiel medzi PostgreSQL a MySQL je v dodržiavaní štandardu jazyka SQL, ktoré je pri databáze MySQL benevolentnejšie. PostgreSQL naproti tomu dodržiava štandardy SQL a podporuje aj pokročilé funkcie [14].

4.1.5 Migrácie

Migrácie sú spôsob, akým sa v aplikácii postavenej na rámci Laravel upravuje štruktúra databázy. Táto technika oproti klasickému spôsobu upravovania štruktúry priamo v opytovacom jazyku danej databázy umožňujú správu verzií tejto štruktúry. Každá zmena je v tejto technike reprezentovaná jednou migráciou - jedným súborom v zdrojových kódoch aplikácie. Tento prístup umožňuje jednoduché zdieľanie zmien v databáze medzi členmi vývojárskeho tímu štandardnými technikami zdieľania zdrojových kódov, ako sú Git, alebo SVN.

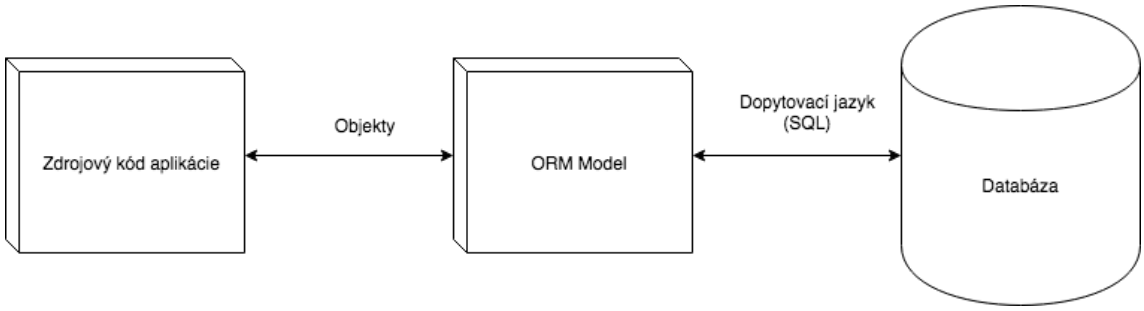
```
Migrating: 2014_10_12_000000_create_users_table
Migrated:  2014_10_12_000000_create_users_table
Migrating: 2014_10_12_100000_create_password_resets_table
Migrated:  2014_10_12_100000_create_password_resets_table
Migrating: 2018_01_01_000001_create_schools_table
Migrated:  2018_01_01_000001_create_schools_table
Migrating: 2018_01_01_000002_create_school_user_table
Migrated:  2018_01_01_000002_create_school_user_table
Migrating: 2018_01_01_000003_create_user_groups_table
Migrated:  2018_01_01_000003_create_user_groups_table
Migrating: 2018_01_01_000004_create_user_group_user_table
Migrated:  2018_01_01_000004_create_user_group_user_table
Migrating: 2018_01_03_000001_create_articles_table
Migrated:  2018_01_03_000001_create_articles_table
Migrating: 2018_01_03_000002_create_article_tags
Migrated:  2018_01_03_000002_create_article_tags
Migrating: 2018_01_03_000003_create_article_tag_article
Migrated:  2018_01_03_000003_create_article_tag_article
Migrating: 2018_01_03_000004_create_article_series_table
Migrated:  2018_01_03_000004_create_article_series_table
Migrating: 2018_04_05_011704_create_images_table
Migrated:  2018_04_05_011704_create_images_table
Migrating: 2018_04_17_234927_create_feed_sharings_table
Migrated:  2018_04_17_234927_create_feed_sharings_table
```

Obr. 4.1: Migrácie vytvorené pri impelemntácii aplikácie

Pre každú tabuľku nevyhnutnú pre aplikáciu bola vytvorená jedna migrácia, ktorá vytvára túto tabuľku.

4.1.6 Eloquent

Eloquent ORM je súčasť rámca Laravel, ktorá poskytuje implementáciu vzoru ActiveRecord pre prácu s databázou.



Obr. 4.2: Model použitia Eloquent ORM

Každá entita, ktorá je v databáze reprezentovaná tabuľkou, je vrámci Eloquent ORM reprezentovaná takzvaným modelom, ktorý sa následne v kóde aplikácie využíva pre interakciu s databázou. V Eloquent ORM sú jednotlivé modely implementované ako trieda, zatiaľ čo jednotlivé inštancie týchto modelov sú objekty vytvorené ako inštancie týchto tried.

ORM je technika ktorá sa stará o konverziu medzi relačnou databázou a objektami, ktoré vývojári využívajú v samotnej aplikácii. Táto technika dáva vývojárovi unifikovaný prístup k ľubovoľnému objektu, respektíve entite s ktorou v aplikácii pracujeme. Vďaka tejto technike

je vývojár do istej miery oslobodený od práce s dopytovacím jazykom konkrétnej databázi, ktorú aplikácia využíva. Vývojárovi dáva táto technika možnosť pracovať v rámci jednej aplikácie aj s viacerými databázami rôznych typov. Použitie ORM uľahčuje predovšetkým najčastejšie operácie s databázou. Tieto operácie sú predovšetkým zápis, čítanie, úprava a mazanie dát [15].

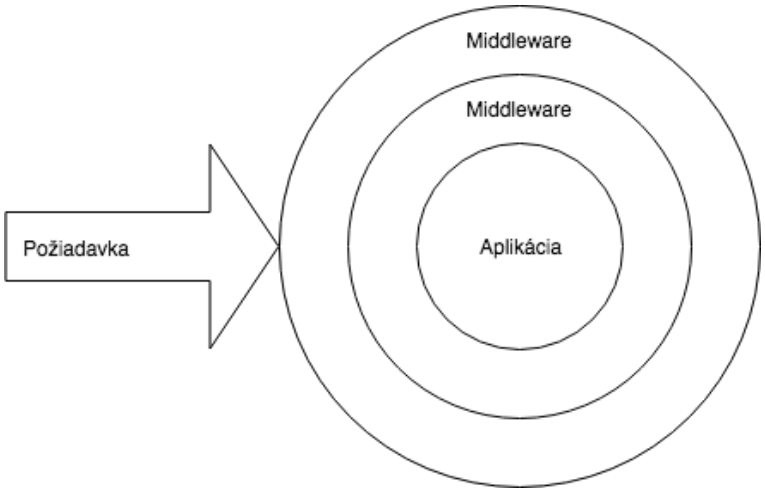
Pre využívanie Eloquent ORM je nevyhnutné správne nakonfigurovať pripojenie k databáze vo frameworku Laravel. V aplikácii boli vytvorené modely pre všetky entity popisované v časti návrh aplikácie.

Tieto modely sú následne využívané pri všetkých operáciách s dátami v aplikácii. Autentifikácia Laravel poskytuje takmer kompletnú implementáciu autentifikácie užívateľa ako jednu zo zabudovaných funkcií (v dokumentácii pod názvom Laravel Auth). Laravel Auth je možné nakonfigurovať pre autorizáciu voči ľubovoľnému úložisku dát.

V aplikácii bol modul autentifikácie nakonfigurovaný tak, aby využíval model reprezentujúci entitu užívateľ (user).

4.1.7 Middleware

Middleware je mechanizmus na filtrovanie požiadaviek vstupujúcich do vašej aplikácie.



Obr. 4.3: Model middleware v rámci Laravel

Laravel napríklad obsahuje middleware, ktorý overuje, či je užívateľ prihlásený, alebo overuje prítomnosť unikátneho tokenu v requestoch smerujúcich do aplikácie. Tento token je spôsob ochrany pred takzvaným CSRF útokom. Pri tomto útoku ide o odosielanie požiadaviek z tretích aplikácii do napadnutej aplikácie. Laravel poskytuje ochranu, formou pridávania unikátneho tokenu do tela každej požiadavky. CSRF middleware následne do aplikácie vpustí len požiadavky s validným tokenom.

V popisovanej aplikácii bolo vytvorených niekoľko middleware-ov, ktoré sú využívané pre účely autorizácie jednotlivých požiadaviek na základe príslušnosti prihláseného užívateľa do konkrétnej skupiny, alebo roly tohto užívateľa v jednej z užívateľských skupín.

4.1.8 Kontroléry

Biznis logika aplikácie je v implementácii MVC architektúry realizovaná v kontrolóroch (z anglického Controller). Títo kontrolóri sú v rámci Laravel implementovaní ako trieda s množinou verejných metód. Každá táto metóda (nazývaná v terminológii Laravelu aj ako akcia - z anglického Action) spracováva jeden druh HTTP požiadavku. Akcie (metódy kontroléra) jedného kontrolóra spracovávajú požiadavky týkajúce sa jednej entity. Napríklad v triede UserController nájdeme metódy spracovávajúce požiadavky pre správu inštancií entity užívateľ. Tieto metódy v sú v Laraveli volané s parametrami, ktoré obsahujú informácie o prichádzajúcom požiadavku. Sú to premenné, ktoré sa nachádzajú v URL požiadavku. Napríklad v prípade požiadavku na zobrazenie obsahu inštancie zadania s unikátnym kódom prve-zadanie bude URL tohto požiadavku /zadania/prve-zadanie Na základe tohto identifikátora, predaného metóde kontrolóra ako parameter, je možné z databázy vybrať konkrétny záznam a vrátiť ho pomocou odpovede na požiadavok do prehliadača, kde sa zobrazí užívateľovi aplikácie. Okrem premenných z URL požiadavku je do metód kontrolóra ako parameter poslaný aj objekt reprezentujúci požiadavok. Tento objekt obsahuje všetky informácie o požiadavku, vrátane dát odoslaných z formulárov, informácii o súboroch cookies a podobne. Väčšina kontrolérov, ktoré spravujú jednu z entít obsahuje niekoľko metód, ktoré obsluhujú požiadavky podľa štandardu architektúr webových služieb REST [20].

Tieto metódy sú:

- **index** - zobrazí výpis inštancií danej entity
- **create** - zobrazí formulár pre zadanie hodnôt atribútov novej inštancie entity
- **store** - uloží inštanciu entity do databází
- **show** - zobrazí jednu inštanciu entity
- **edit** - zobrazí formulár pre úpravu hodnôt atribútov existujúcej inštancie entity
- **update** - upraví hodnoty atribútov existujúcej inštancie entity v databáze
- **deleteModal** - zobrazí modálové potvrdzovacie okno potvrdzujúce vymazanie inštancie entity z databáze
- **destroy** - odstráni existujúcu entitu z databáze

4.1.9 Validácia vstupov

Validácia vstupov od užívateľa je nevyhnutnou funkcionalitou každej webovej aplikácie. https://ieeexplore.ieee.org/abstract/document/5999632/ Implementovaná aplikácia ošetruje všetky vstupy zadané užívateľom v niektorom z formulárov aplikácie. O validáciu každého z formulárov sa stará samostatná trieda na to učená. Tieto triedy tvoria vrstvu Requests spomínanú v podkapitole návrh systému. Jedna trieda existuje pre každý formulár, ktorý si vyžaduje validáciu užívateľských vstupov. V každej z týchto tried je zadefinovaná metóda rules, ktorá popisuje pravidlá validácie pre jednotlivé vstupy.

Prípadné chyby v užívateľských vstupoch sú zobrazené priamo vo formulári, ktorý obsahuje nesprávne vyplnené, alebo nevyplnené pole.

Obr. 4.4: Zobrazenie formulára bez chybových hlásení

Obr. 4.5: Zobrazenie formulára s chybových hláseniami

4.1.10 Servisné triedy

Servisné triedy tvoria vrstvu Services popisovanú v podkapitole Architekúra. Tieto triedy obsahujú jednoúčelové metódy, ktoré vykonávajú operácie nad dátovým modelom aplikácie. V kóde aplikácie mimo servisných tried sa nikde nevykonávajú žiadne operácie nad dátovým modelom. Všade kde je takáto operácia nutná je použitá jedna z metód servisných tried.

Vrstva servisných tried zvyšuje bezpečnosť a prehľadnosť aplikácie. Metódy servisných tried rozdeľujú biznis logiku aplikácie na jednoúčelové bloky, ktoré sú teda oveľa lepšie otestovateľné jednotkovými testami.

4.1.11 Upozornenia

Upozornenia sú dôležitou súčasťou implementovanej aplikácie. Laravel obsahuje podporu pre vytváranie oznámení a ich doručovanie viacerými kanálmi vrátane emailu, SMS pomocou služby Nexmo a Slack. Tieto upozornenia sa uložia do databázy a budú taktiež zobrazované priamo v aplikácii.

Pre odosielanie oznámeni prostredníctvom Facebook Messengeru bol do aplikácie doplnený balíček laravel-notification-channels/facebook [**?**].

4.1.12 Jazykové mutácie

Aplikácia bola implementovaná v slovenskom jazyku. Bola ale implementovaná s použitím mechanizmu pre jazykové mutácie zabudovanom v Laraveli. Použitím tohto mechanizmu vznikla aplikácia, ktorú je možné jednoducho preložiť do nového jazyka bez nutnosti zásahu do zdrojových kódov aplikácie.

4.1.13 Odozdávanie riešení

Odovzdávanie riešení zadaní je realizované pomocou knižnice pre odosielanie súborov na server na pozadí aplikácie. Takéto odosielanie súborov je dnes bežným štandardom a poskytuje možnosť odovzdávať ľubovoľne veľké súbory. Keďže HTTP protokol, ktorý sa používa pre odosielanie súborov má obmedzenie na maximálnu veľkosť odoslaného súboru sú všetky súbory odosielané na server rozdelené na kúsky veľké 10MB a následne na serveri znova spojené do jedného veľkého súboru.

V aplikácii je možné odovzdať vždy jeden súbor. V prípade, že zadanie vyžaduje viacero súborov užívateľ má možnosť odovzdať riešenie v zip archíve. Takýto archív sa automaticky extrahuje a ďalej sa pracuje so všetkými súbormi z odovzdaného archívu.

Odovzdávací systém uchováva aj všetky riešenia zadania odovzdané v minulosti. Autor zadania si tak vie jednoducho porovnať progres žiaka, pri jednotlivých odovzdaniach.

4.1.14 Automatické testovanie

Aplikácia využíva externú službu pre automatické spúšťanie a otestovanie odovzdaného užívateľského riešenia, ktorá bola vyvinutá autorom práce a jeho spolupracovníkmi zo súťaže, ktorá je spomínana v kapitole úvod.

Táto služba bola vyvinutá v čase pred vznikom riešeného zadania diplomovej práce a jej implementácia nebola súčasťou diplomovej práce.

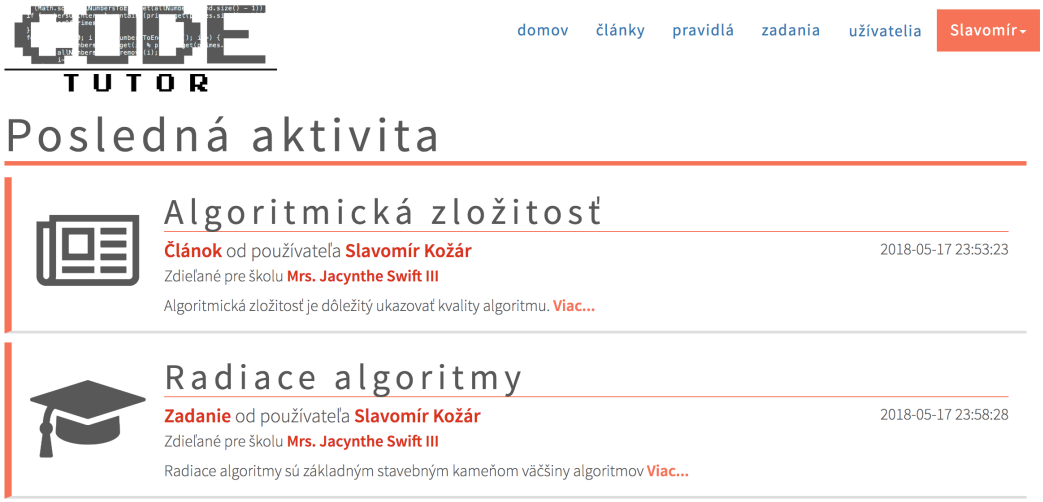
Aplikácia implementovaná vrámci diplomovej práce komunikuje s touto službou pomocou REST API, ktoré poskytuje jednak testovacia služba aj implementovaná aplikácia.

Po odovzdaní a overení validnosti riešenia (overenie programovacieho jazyka zadania) je odoslaný požiadavok do testovacej služby, ktorá toto zadanie spustí a otestuje. Súčasťou tohto požiadavku sú testovacie dáta, parametre s ktorými sa riešenie má spustiť, časové obmedzenie riešenia a podobne... Po ukončení testu testovacia služba požiadavkou na API, ktoré poskytuje implementovaná aplikácia odovzdá výsledky testov aplikácii, ktorý ich následne spracuje a bude ich prezentovať užívateľovi.

4.2 Užívateľské rozhranie

Užívateľské rozhranie aplikácie bolo implementované tak, aby sa korektne zobrazovalo aj na zariadeniach s rôznym rozlíšením displeja.

Obr. 4.6: Zobrazenie hlavnej stránky na displeji mobilného zariadenia



Obr. 4.7: Zobrazenie hlavnej stránky na monitore počítača

Keďže užívateľské rozhranie implementovanej aplikácie sa zobrazuje vo webovom prehliadači, platia pre toto rozhranie špecifiká spojené s vývojom webových stránok.

Implementované užívateľské rozhranie sa rovnako ako webové stránky dá rozdeliť do troch oblastí, ktoré sú špecifické svojim obsahom, použitými technológiami aj zásadami.

4.2.1 Obsah

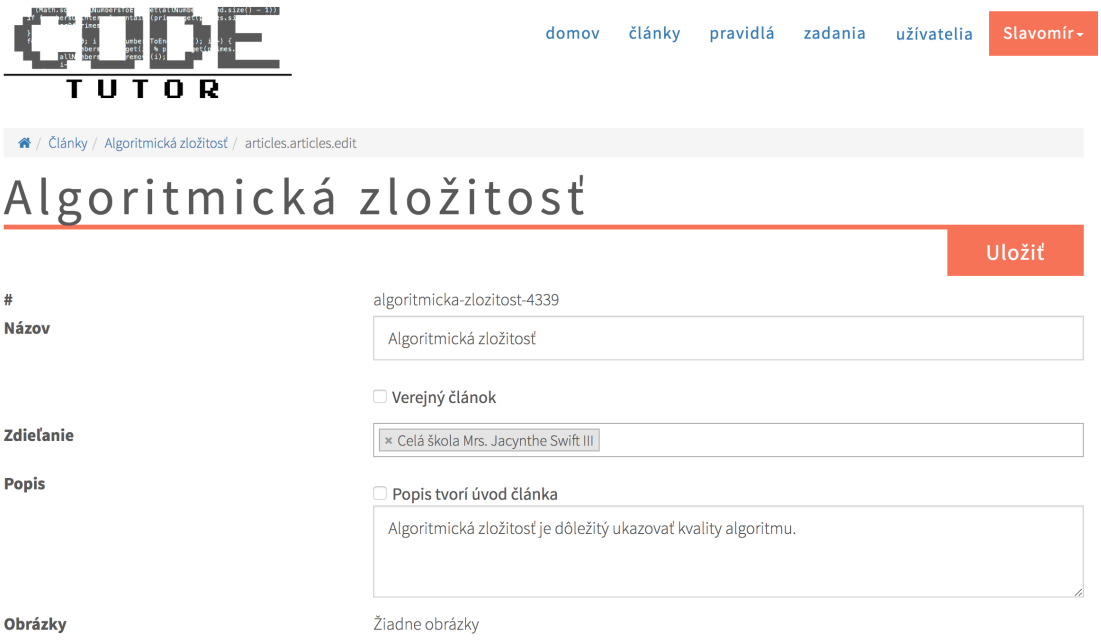
Obsah tvorí najdôležitejšiu časť webovej stránky. Pre implementáciu obsahu webovej stránky je určený jazyk HTML [24] (HyperText Markup Language), ktorý je interpretovaný webovým prehliadačom.

Pre generovanie HTML obsahu stránky bol pri implementácii užívateľského rozhrania aplikácie použitý jazyk blade, ktorý je súčasťou Laravel-u. Tento jazyk je určený pre tvorbu šablón webových aplikácií. Obsahuje mnoho preddefinovaných funkcií, ktoré zjednodušujú implementáciu užívateľského rozhrania.

4.2.2 Prezentácia

Prezentačná časť webovej stránky definuje vzhľad obsahu stránky. Pre túto časť webového obsahu sa používa jazyk CSS (Cascade Style Sheets). Pri implementácii užívateľského rozhrania aplikácie bol použitý preprocesor jazyka CSS s názvom Less. Tento preprocesor pridáva viaceré pokročilé možnosti jazyku css a poskytuje nástroje pre efektívnejšie vytváranie css súborov. Štýly vytvorené pomocou preprocesora Less boli kompilované do jazyka CSS pomocou nástroja Gulp.

Ako základ štýlu užívateľského rozhrania bol použitý framework bootstrap verzie 3. Tento framework bol oproti štandardnému vzhľadu upravený tak, aby spĺňal požiadavky na takzvaný flat material dizajn popisovaný v kapitole analýza.



Obr. 4.8: Zobrazenie článku

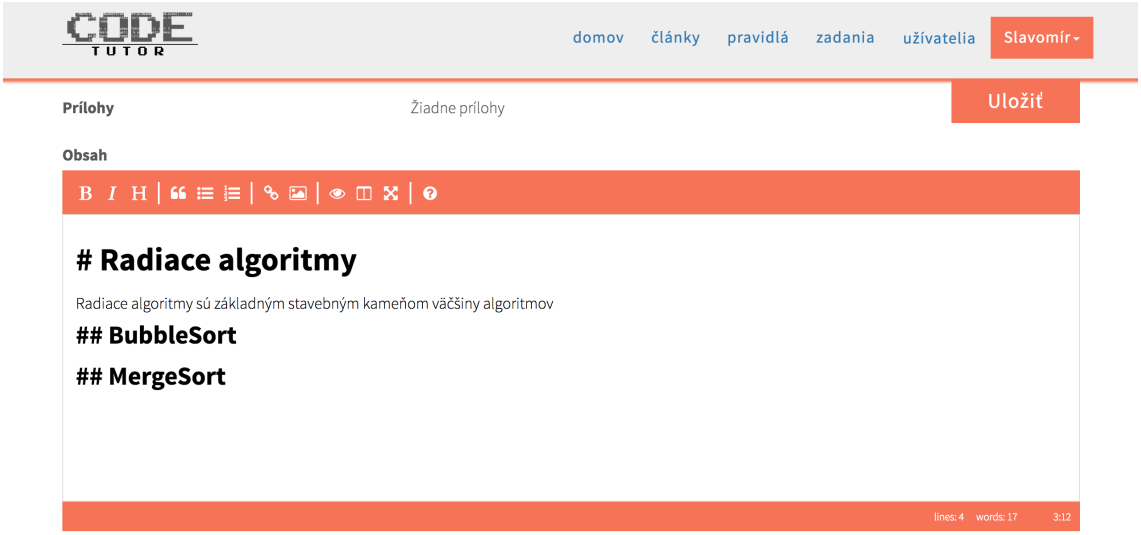
Obr. 4.9: Úprava článku

Pri kompilácii zdrojových súborov v jazyku Less do CSS súborov prebieha aj minifikácia týchto súborov. Z týchto súborov sú v tomto procese odstránené všetky znaky, ktoré nie sú nutné pre správnu interpretáciu týchto súborov webovým prehliadačom. V prípade css súborov je to odstránenie všetkých takzvaných whitespace súborov - medzier, koncov riadkov a podobne. Touto minifikáciou sa zníži veľkosť súboru, ktorý sa načítava pri otvo-

rení webovej stránky a tým sa toto načítanie skráti. Správanie Správanie je dôležitá súčasť predovšetkým webových aplikácii. Jedná sa o správanie priamo v prehliadači užívateľa. Pre definovanie tohto správania sa využíva programovací jazyk JavaScript. Pri implementácii aplikácie bola použitá knižnica jQuery a ďalšie knižnice pre vytvorenie špecifických komponent užívateľského rozhrania.

4.2.2.1 Simplemde

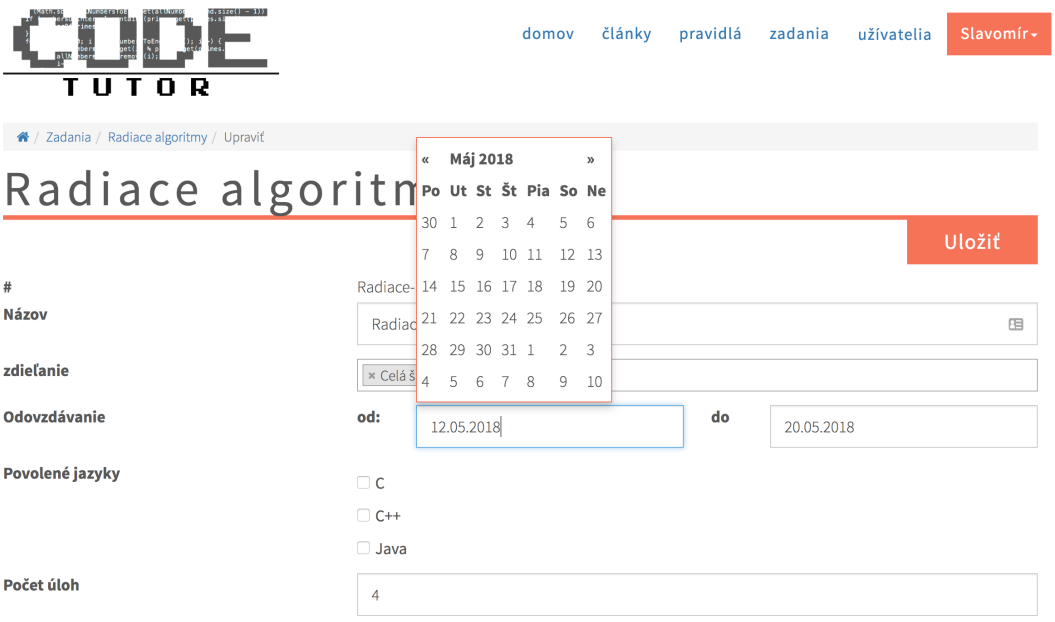
Simplemde je jednoduchý editor jazyka markdown [8].



Obr. 4.10: Simplemde editor

4.2.2.2 Bootstrap datepicker

Bootstrap datepicker je knižnica pre vytvorenie užívateľsky prívetivého vstupného prvka pre výber dátumu [1].



Obr. 4.11: Bootstrap datepicker

4.2.2.3 jQuery File Upload

jQuery File Upload je knižnica zabezpečujúca nahrávanie súborov na server na pozadí aplikácie používaním asynchrónnych HTTP požiadavkov [10].

Obr. 4.12: jQuery File Upload

Kapitola 5

Testovanie

Testovanie je súčasťou procesu overovania kvality aplikácie. Testovanie aplikácie prebiha v niekoľkých iteráciách a na rôznych úrovniach aplikácie. Aplikácia implementovaná pri realizácii tejto diplomovej práce bola testovaná dvoma spôsobmi.

5.1 Jednotkové testy

Jednotkové testovanie označuje automatické testovanie a overovanie funkčnosti implementácie aplikácie. Tieto testy sú tvorené priamo počas implementácie aplikácie a poskytujú okamžitú spätnú vazbu o funkčnosti aplikácie počas implementácie tejto aplikácie.

Jednotkový test testuje jednu izolovanú jednotku aplikácie. Pre dosiahnutie tejto izolovanosti sa v procese jednotkového testovania vytvárajú objekty, ktoré simulujú kontext v ktorom testovaná jednotka pracuje.

Za jednotku je považovaná samostatne testovateľná časť aplikácie. Z pohľadu kódu môže byť jednotkou trieda, metóda, funkcia a tak podobne.

V implementovanej aplikácii za jednotky považujeme metódy tried servisnej vrstvy aplikácie. Keďže tieto metódy vykonávajú izolovateľné operácie nad dátovým modelom plnia jednotkové testy zároveň funkciu integračných testov. Tieto testy overujú funkčnosť prepojenia dvoch blokov aplikácie. V tomto prípade servisnej vrstvy aplikácie a databázy.

Pre jednotkové testovanie aplikácie bola využitá knižnica PHPUnit. Táto knižnica je implementácia architektúry xUnit pre jednotkové testovanie v jazyku PHP [7].

Každý PHPUnit test je implementovaný ako jedna metóda. V rámci jednej metódy je možné vykonať niekoľko porovnaní (takzvanými assert funkciami). Test sa považuje za splnený, ak sú všetky porovnania v rámci tohto testu splnené. Testy, ktoré zdieľajú jeden kontext v ktorom pracujú sú združené do tried. Pri vytvorení inštancie triedy sa vždy najprv vytvoria objekty, ktoré budú simulujú tento kontext. Pri použití PHPUnit je na tieto účely vyhradená funkcia setUp, v ktorej sú vytvorené všetky objekty potrebné pre vykonanie testu.

Pretože je dôležité aby testy pracovali so simulovaným kontextom a neovplyvňovali samotnú aplikáciu sú všetky testy spúšťané v simulovanom prostredí, ktoré tvorí databáza rovnakého typu s rovnakou schémou, ako databáza využívaná testovanou aplikáciou.

5.2 Akceptačné testy

Akceptačné testy sú testy ktoré overujú funkčnosť implementovanej aplikácie porovnávaním s požiadavkami definovanými v analytickej časti návrhu realizácie aplikácie.

Akceptačné testy boli pri realizácii aplikácie realizované formou takzvaných smoke testov (niekedy prekladané ako zahorovacie testy). Tieto testy sa používajú v čase, kedy je implementácia dokončená a aplikáciu je možné spustiť. Jedná sa o testy, pri ktorých sa overuje, že implementovaná aplikácia funguje a je pripravená na ďalšie fázy testovania.

Tieto testy sa zameriavajú na hlavné funkcie programu, ktoré sú užívateľmi najčastejšie používané a nebývajú často upravované.

Impelenovaná aplikácia bola otestovaná scenármi, ktoré vyplývajú z príkladov použitia aplikácie uvedených v kapitole analýza. V prípade aplikácie implementovanej v rámci realizácie tejto diplomovej práce neboli tieto testy automatizované. Boli vykonané manuálne po dokončení implementácie. Pri tomto manuálnom testovaní boli zistené drobné nedostatky spôsobené hlavne nedostatočnou validáciou vstupných polí formulárov (autor implementácie zabudol implementovať validáciu pre vstupné pole počet úloh vo formulári, pre nastavenie vlastností zadania)

5.3 Užívateľské testovanie použiteľnosti

Užívateľským testovaním alebo niekedy aj testovaním použiteľnosi sa označuje testovanie, pri ktorom sa zisťuje, aký je užívateľský zážitok užívateľov, ktorí budú aplikáciu využívať v ostrej prevádzke. Toto testovanie sa vykonáva za účasti koncových užívateľov testovanej aplikácie. Je dôležité, aby testujúci užívatelia zapadali do profilu daného cieľovou skupinou používateľov. Pre konkrétnu aplikáciu, ktorá vznikla ako realizácia tejto práce túto skupinu tvoria učitelia druhého stupňa základných škôl, učitelia stredných škôl a ich žiaci. Táto cieľová skupina je detailnejšie popísaná v kapitole analýza.

Pri takomto testovaní je užívateľovi predstavený scenár vychádzajúci z príkladov použitia. Používanie aplikácii týmito testujúcimi užívateľmi je zaznamenávané a autor testov z týchto záznamov určí najväčšie problémy, ktoré znižujú kvalitu užívateľského zážitku aplikácie. Pre účely takéhoto užívateľského testovania autor práce kontaktoval stredné školy - gymnázia v Prahe a dve stredné priemyselné školy - v Prahe a v Prešove.

Pre toto testovanie boli pripravené nevyhnutné podklady. Konkrétne úvodný dotazník pre učiteľov, ktorý bol použitý pre overenie konceptu tejto aplikácie. Druhý dotazník pre užívateľov zisťuje ich skúsenosti a návyky z používania aplikácii a počítačov obecne. Po týchto dotazníkoch nasleduje testovanie z užívateľom, podľa scenárov vypracovaných na základe funkčných požiadaviek a prípadov použitia definovaných v kapitole Analýza.

Bohužiaľ, v čase dokončenia tejto práce a aj implementovanej aplikácie sa stredné školy pripravovali na maturitnú skúšku a toto užívateľské testovanie bolo po dohode s vedúcou práce presunuté na čas po odovzdaní práce.

Kapitola 6

Záver

6.1 Zhodnotenie

Cieľom práce bolo analyzovať, navrhnúť a implementovať aplikáciu pre podporu vyučovania programovania na základných a stredných školách. Navrhovaný systém mal umožňovať organizáciu užívateľov do skupín, správu a zdieľanie zadaní a odovzdávanie a hodnotenie riešení od žiakov. V úvode aplikácie autor analyzuje požiadavky na základe vlastných skúseností, ale aj skúseností žiakov a učiteľov dotknutých škôl. Z tejto analýzy vyplynuli funkčné a nefunkčné požiadavky. Autor ďalej zadefinoval parametre aplikácie nevyhnutné pre správny návrh a následne testovanie tejto aplikácie. Ide predovšetkým o zadefinovanie užívateľských aktérov, ktorí budú s aplikáciou pracovať a aj prípadov použitia, ktoré pokrývajú najdôležitejšie funkcie v aplikácii.

Pre túto aplikáciu bol navrhnutý komplexný dátový model, ktorý je rozšíriteľný o ďalšie funkcie. Navrhnuté bolo tiež užívateľské rozhranie, ktoré spĺňa moderné štandardy a trendy užívateľských rozhraní aplikácií.

Navrhnutá aplikácia bola implementovaná modelom Software as a Service, teda aplikácia poskytovaná ako služba. Táto aplikácia je užívateľom dostupná prostredníctvom webového prehliadača. Ako základ implementácie bol využitý aplikačný rámec Laravel vo verzii 5.5. Tento rámec je rámcom jazyka PHP s najrýchlejšie rastúcou komunitou. Implementácia použitím takto rozšíriteľných aplikácii zabezpečí bezproblémové rozšírenie aplikácie. Jednak jej funkčnosti a zároveň aj vývojárskeho tímu.

Aplikácia bola otestovaná voči zadefinovaným požiadavkám a prípadom použitia. V rámci riešenia tejto práce bolo navrhnuté aj používateľské testovanie, ktoré sa neuskutočnilo kôli prekrývaniu termínov maturitných skúšok na stredných školách, ktorých učitelia a žiaci tvoria cieľovú skupinu a termínu dokončenia funkčného prototypu aplikácie.

Pri realizácii tejto práce autor vychádzal z pevných základov vzdelania v oblasti softvérového vývoja, ktoré mu bolo poskytnuté počas bakalárskej aj magisterskej etapy štúdia na Fakulte Elektrotechnicke ČVUT v Prahe. Samotná realizácia obohatila autorove skúsenosti v oblasti softvérového vývoja, predovšetkým vo fázach analýzy a navrhovania systému takejto veľkosti.

6.2 Ďalšie pokračovanie práce

Autor práce už v minulosti pracoval na projektoch s podobnou tématikou ako aplikácia, ktorá vznikla vrámci riešenia tejto práce, preto predpokladá, že sa bude ďalšiemu rozvoju aplikácie aj naďalej venovať až po úspešné nasadenie a prevádzku.

Bezprostredne ďalším krokom pri rozvoji aplikácie budú užívateľské testy, ktoré nebolo možné z časových dôvodov realizovať počas realizácie práce. Z výstupov týchto testov vyplynú nedostatky v oblasti užívateľskej skúsenosti s aplikáciou a taktiež nové podnety pre rozšírenie funkčných požiadavkov na aplikáciu.

Vzhľadom na navrhutý dátový model jednotlivých inštancií obsahových entít je aplikácia ľahko rozšíriteľná o ďalší druh obsahu. Autor práce predpokladá rožšírenie aplikácie práva týmto smerom.

Ďalšie možnosti rozšírenia sú v samotnej práci s odovzdávaním žiackych riešení. Autor vidí priestor napríklad v prepojení odovzdávacieho systému s verzovacími nástrojmi Git, alebo SVN.

Literatúra

[1] Bootstrap datepicker, 2018.
http://bootstrap-datepicker.readthedocs.io/en/latest/, stav z 24. 5. 2018.

[2] Desktop vs Mobile vs Tablet Market Share Worldwide, 2018.
http://gs.statcounter.com/platform-market-share/desktop-mobile-tablet|, stav z 30. 4. 2018.

[3] Flat Design, 2018.
https://www.interaction-design.org/literature/topics/flat-design, stav z 24. 5. 2018.

[4] Differences Between Fast Hashes and Slow Hashes, 2012.
http://openwall.info/wiki/john/essays/fast-and-slow-hashes, stav z 7. 7. 2012.

[5] Design pattern: many-to-many (order entry, 2004.
http://www.tomjewett.com/dbdesign/dbdesign.php?page=manymany.php, stav z 24. 5. 2018.

[6] Why the Notification is the Most Important Email You Can Send for Growth, 2014.
goo.gl/tTmMj1, stav z 19. 11. 2014.

[7] PHPUnit, 2018.
https://phpunit.de/, stav z 24. 5. 2018.

[8] Simplemde, 2018.
http://simplemde.com/, stav z 24. 5. 2018.

[9] The top 500 sites on the web, 2018.
https://www.alexa.com/topsites, stav z 24. 5. 2018.

[10] jQuery File Upload, 2018.
http://blueimp.github.io/jQuery-File-Upload/, stav z 24. 5. 2018.

[11] WebML – datové modelování, 2014.
https://www.interval.cz/clanky/webml-datove-modelovani/, stav z 8. 1. 2014.

[12] BLAHA, M. Co učit tředoškoláky o počítačích a internetu?, 2017.
https://www.michalblaha.cz/2017/01/co-ucit-stredoskolaky-o-pocitacich-a-internetu/|, stav z 12. 1. 2017.

[13] BRANDON GRIGGS, C. Gates, Zuckerberg: Kids, learn to code, 2013. `https://edition.cnn.com/2013/02/27/tech/innovation/`code-video-gates-zuckerberg/index.html|stav z 24. 5. 2018.

[14] CONRAD, T. Postgresql vs. mysql vs. commercial databases: It's all about what you need, 2006.

[15] FOWLER, M. *Patterns of Enterprise Application Architecture.* : Addison-Wesley Professional, 2002. Dostupné z: <https://www.amazon.com/Patterns-Enterprise-Application-Architecture-Martin/dp/0321127420?SubscriptionId=0JYN1NVW651KCA56C102&tag=techkie-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=0321127420,>. ISBN 0321127420.

[16] IAN, S. *Software Engineering, Global Edition.* : Pearson Educacion, 2015. Dostupné z: <https://www.amazon.com/Software-Engineering-Global-Sommerville-Ian/dp/1292096136?SubscriptionId=0JYN1NVW651KCA56C102&tag=techkie-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=1292096136>. ISBN 1292096136.

[17] LERNER, R. M. At the Forge: PostgreSQL, the NoSQL Database. *Linux J.* November 2014, 2014, 247. ISSN 1075-3583. Dostupné z: <http://dl.acm.org/citation.cfm?id=2713595.2713600>.

[18] MANNOVA, B. – PRESTON, C. COMPUTER SCIENCE INCLUSION IN THE SCHOOL CURRICULUM. In *EDULEARN17 Proceedings*, 9th International Conference on Education and New Learning Technologies, s. 2682–2684. IATED, 3-5 July, 2017 2017. doi: 10.21125/edulearn.2017.1554. Dostupné z: <http://dx.doi.org/10.21125/edulearn.2017.1554>. ISBN 978-84-697-3777-4.

[19] MARCIANO, P. *Cukr a bič nefungují.* : Motiv Press, 1st edition, 2013. In Czech. ISBN 978-80-904133-9-9.

[20] MASSE, M. *REST API Design Rulebook: Designing Consistent RESTful Web Service Interfaces.* : Ö'Reilly Media, Inc."„ 2011.

[21] MICHáLEK, M. Co je to „Mobile First"? Ale doopravdy, 2017. `https://www.vzhurudolu.cz/prirucka/mobile-first`, stav z 4. 5. 2017.

[22] MILLS, C. Using the Notifications API, 2018. `goo.gl/25Vjzx`, stav z 17. 5. 2018.

[23] TEOREY, T. J. – YANG, D. – FRY, J. P. A Logical Design Methodology for Relational Databases Using the Extended Entity-relationship Model. *ACM Comput. Surv.* June 1986, 18, 2, s. 197–222. ISSN 0360-0300. doi: 10.1145/7474.7475. Dostupné z: <http://doi.acm.org/10.1145/7474.7475>.

[24] VAUGHAN-NICHOLS, S. J. Will HTML 5 Restandardize the Web? *Computer.* April 2010, 43, 4, s. 13–15. ISSN 0018-9162. doi: 10.1109/MC.2010.119.

[25] web:infodp. K336 Info — pokyny pro psaní diplomových prací, 2013. `https://info336.felk.cvut.cz/clanek.php?id=400`, stav ze 4. 5. 2009.

Dodatok A

Obsah priloženého CD

```
/implementacia
         /database
         /composer.lock
         /server.php
         /.DS_Store
         /phpunit.xml
         /bootstrap
         /app
         /Icon
         /config
         /resources
         /tests
         /storage
         /readme.md
         /artisan
         /public
         /.gitignore
         /package-lock.json
         /package.json
         /gulpfile.js
         /rules.md
         /.env.example
         /routes
         /routes/console.php
         /routes/web.php
         /routes/api.php
         /composer.json

/text
         /kozarsla-2018.lof
         /kozarsla-2018.blg
         /kozarsla-2018.log
```

```
/kozarsla-2018.aux
/hyphen.tex
/chapters
         /1_uvod.tex
         /1_uvod.aux
         /2_analyza.tex
         /2_analyza.aux
         /3_navrh_systemu.tex
         /3_navrh_systemu.aux
         /4_implementacia.tex
         /4_implementacia.aux
         /5_testovanie.tex
         /5_testovanie.aux
         /6_zaver.tex
         /6_zaver.aux

/Makefile
/code
/code/hello.c
/README.md
/csplainnat.bst
/kozarsla-2018.nlo
/.gitignore
/kozarsla-2018.nls
/kozarsla-2018.out
/k336_thesis_macros.sty
/figures
         .. 43 figures
/reference.bib
/kozarsla-2018.tex
/kozarsla-2018.toc
/kozarsla-2018.pdf
/obrazky.tex
/kozarsla-2018.ilg
/kozarsla-2018.bbl
```