

Slovenská technická univerzita v Bratislave

Fakulta informatiky a informačných technológií

FIIT-5212-73878

Tomáš Gábrš

TVORBA OTÁZOK Z UČEBNÉHO TEXTU

Bakalárska práca

Študijný program:	Informatika
Študijný odbor:	9.2.1 Informatika
Miesto vypracovania:	Ústav informatiky, informačných systémov a softvérového inžinierstva, FIIT STU, Bratislava
Vedúci bakalárskej práce:	Ing. Miroslav Blšták

Máj 2017

Čestné prehlásenie

Čestne vyhlasujem, že som túto prácu vypracoval samostatne, na základe konzultácií a s použitím uvedenej literatúry.

ANOTÁCIA

Slovenská technická univerzita v Bratislave

FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLOGIÍ

Študijný program: INFORMATIKA

Autor: Tomáš Gábrš

Bakalárska práca: Tvorba otázok z učebného textu

Vedúci bakalárskej práce: Ing. Miroslav Blšták

máj, 2017

V dnešnej dobe sa kladie vysoký dôraz na štúdium a na skvalitnenie procesu výučby. Jednou z kľúčových oblastí v rámci štúdia je testovanie vedomostí študentov. Najčastejšie sa toto testovanie vyskytuje vo forme písomného testu. S rastúcim počtom elektronických materiálov sa zvyšuje záujem učiteľov o možnosť automatického generovania testov z týchto materiálov. Táto práca sa zaoberá možnosťami automatickej tvorby otázok z textu a následného použitia týchto metód na vytvorenie aplikácie pre učiteľov, ktorí budú môcť využívať počas prípravy testov. Cieľom vyvíjanej aplikácie je zvýšiť efektivitu práce učiteľov, rovnako ako zvýšiť kvalitu vytvorených testov.

ANNOTATION

Slovak University of Technology in Bratislava

FACULTY OF INFORMATICS AND INFORMATION TECHNOLOGIES

Degree Course: INFORMATICS

Author: Tomáš Gábrš

Bachelor Thesis: Question generation from educational text

Supervisor: Ing. Miroslav Blšták

2017, May

Nowadays, there is a high emphasis on studies and enhancement of the educational process. One of the key factors of studies is testing the student's knowledge. Knowledge tests are the most common form of student testing. With the rising number of studying materials in electronic form, the interest in automatically generated tests from these materials also rises. This bachelor thesis focuses on creation of an application for teachers, which they can use during test preparation to automatically generate test questions from text. The goal of the application is to make the teacher's work more efficient, while simultaneously increasing the quality of the tests.

Obsah

1	Úvod	1
1.1	Motivácia	1
1.2	Ciele práce	2
2	Analýza tvorby otázok	3
2.1	Typy otázok	3
2.1.1	Rozdelenie podľa účelu otázky	3
2.1.2	Rozdelenie podľa typu informácie, ktorú chceme získať	3
2.1.3	Rozdelenie podľa zložitosti vyžadovanej odpovede	4
2.2	Porovnanie tvorby otázok z vety alebo z odseku	4
2.3	Učebné materiály	5
2.4	Úrovne poznávacích cieľov	6
2.5	Proces tvorby otázky	7
2.5.1	Výber obsahu otázky	7
2.5.2	Výber typu otázky	7
2.5.3	Konštrukcia otázky	7
2.6	Problémy a výzvy pri tvorbe otázok	8
2.6.1	Lexikálne výzvy	8
2.6.2	Syntaktické výzvy	9
2.6.3	Výzvy diskurzu	9
2.7	Zhrnutie prístupov pri tvorbe otázok	10
3	Spracovanie prirodzeného jazyka	11
3.1	Úlohy spracovania prirodzeného jazyka	11
3.1.1	Značkovanie slovných druhov	12

3.1.2	Rozpoznanie názvoslovných entít	12
3.1.3	Lematizácia	12
4	Návrh systému na tvorbu testov	13
4.1	Architektúra systému	13
4.2	Spracovanie vstupného textu	15
4.2.1	Token	15
4.2.2	OpenIE trojica	15
4.2.3	Veta	16
4.2.4	Článok	16
4.2.5	Štruktúra spracovaného textu	17
4.3	Tvorba otázok	17
4.4	Navrhnuté typy otázok	18
4.4.1	Otázka so zameraním na rok	18
4.4.2	Otázka so zameraním na mesiac	18
4.4.3	Otázka so zameraním na dátum v úplnom tvare	18
4.4.4	Otázka so zameraním na názvoslovné entity	19
4.5	Tvorba testu a štruktúra informácií v teste	19
4.5.1	Otázka	20
4.5.2	Test	20
4.6	Návrh klientskej aplikácie	21
4.6.1	Uchovanie dát v databáze	21
4.6.2	Vytvorenie testu	23
4.6.3	Zobrazenie prehľadu testov	24
4.6.4	Zobrazenie detailov testu	24
4.6.5	Úpravy informácií o teste	25
4.6.6	Zobrazenie odpovedových hárkov k testu	25
4.6.7	Vytvorenie otázky	25
4.6.8	Úpravy existujúcej otázky	26
4.6.9	Vytvorenie odpovedového hárku	26
4.6.10	Vyplnenie a odoslanie odpovedového hárku	27
4.6.11	Zobrazenie detailov a vyhodnotenie odpovedového hárku	28
4.6.12	Zobrazenie prehľadu odoslaných odpovedových hárkov	28

4.6.13	Zobrazenie prehľadu prijatých odpovedových hárkov	28
4.7	Zhrnutie	28
5	Systém na tvorbu testov	29
5.1	Použité technológie	29
5.2	Spracovanie vstupného textu	30
5.3	Uchovanie informácií o texte	30
5.4	Tvorba testu	31
5.5	API	33
5.5.1	Požiadavka na vytvorenie testu	33
5.5.2	Požiadavka na spracovanie článku	33
5.6	Zhrnutie	33
6	Klientská aplikácia	35
6.1	Použité technológie	35
6.2	Spracovanie dát a ich zobrazenie v prehliadači	35
6.3	Vytvorenie testu	36
6.4	Úpravy vytvoreného testu	36
6.5	Odoslanie odpovedového hárku	36
6.6	Vyhodnotenie a zobrazenie odpovedového hárku	37
6.7	Zhrnutie	38
7	Evaluácia	39
7.1	Vstupná množina viet	39
7.2	Pokrytie viet	40
7.3	Ohodnotenie vytvorených otázok	40
7.4	Zhrnutie	44
8	Zhodnotenie	45
A	Technická príručka	A-1
A.1	Použité technológie	A-1
A.1.1	Java Development Kit 8	A-1
A.1.2	PostgreSQL	A-1
A.1.3	Stanford CoreNLP	A-1

A.1.4	Laravel	A-1
A.1.5	SQLite	A-2
A.1.6	Bootstrap	A-2
A.1.7	jQuery	A-2
A.1.8	Apache Tomcat	A-2
A.1.9	PHP	A-2
A.1.10	Composer	A-3
A.2	Inštalácia a spustenie aplikácie	A-3
A.3	Dátový model klientskej aplikácie	A-4
A.4	Diagram komponentov	A-6
A.5	Prehľad HTTP požiadaviek pre klientskú aplikáciu	A-7
B	Obsah elektronického média	B-1
C	Používateľská príručka	C-1
C.1	Registrácia a prihlásenie	C-1
C.2	Vytvorenie testu	C-1
C.3	Správa testov	C-4
C.3.1	Vytvorenie otázky	C-4
C.3.2	Úprava otázky	C-4
C.3.3	Úprava informácií o teste	C-5
C.3.4	Zobrazenie odpovedňových hárkov k testu	C-6
C.4	Správa odpovedňových hárkov	C-7
C.4.1	Vytvorenie odpovedňového hárku	C-7
C.4.2	Vyplnenie a odoslanie odpovedňového hárku	C-7
D	Príklady otázok vytvorených počas testovania	D-1
D.1	Otázky zamerané na osobu	D-1
D.2	Otázky zamerané na dátum	D-2
D.3	Otázky zamerané na organizáciu	D-2
D.4	Otázky zamerané na miesto	D-3
D.5	Ostatné otázky	D-4
E	Plán na letný semester	E-1

F	Článok z IIT.SRC
----------	-------------------------

F-1

Kapitola 1

Úvod

Cieľom tejto bakalárskej práce je vytvorenie prototypu systému na tvorbu otázok z učebného textu. Za týmto účelom vykonáme analýzu existujúcich riešení a vytvoríme návrh vlastného riešenia, ktoré budeme implementovať a testovať.

1.1 Motivácia

Automatické generovanie otázok z textu je relatívne nová disciplína, ktorá sa súčasne týka rôznych odvetví. Ľudia sa v podstate neustále pýtajú otázky v snahe získať nové informácie a vedomosti. Napríklad otázky od študentov pre učiteľov počas výučby, otázky od učiteľov pre študentov v rámci testu, rovnako ako otázky v rámci každodenných konverzácií v spoločnosti, sú ukázkovým príkladom toho, ako často ľudia otázky používajú (Ali, 2012). Ľudia sa pýtajú otázky z rôznych dôvodov (kapitola 2.1.1), no keďže naším cieľom je tvorba otázok z učebného textu, budeme sa primárne venovať otázkam položeným s cieľom overenia vedomostí alebo ujasnenia si nejasností.

Pri otázkach so zámerom ujasnenia si nejasností je však základným problémom odhalenie vlastných nedostatkov, ktoré musí predchádzať samotnej otázke. Ľudia nie sú schopní vytvárať dostatočne dobré otázky a preto by bolo výhodné, ak by im pri tejto činnosti asistoval systém (Rus et al., 2009). V prípade, že by ľudia boli schopní odhaliť svoje vlastné nedostatky, mohli by si následne danú problematiku doštudovať.

V prípade otázok položených so zámerom overiť, či iná osoba má určité znalosti, ktoré sú od nej požadované, je potrebné otázku konštruovať tak, aby na ňu bolo možné odpovedať na základe poskytnutých informácií. Týmto otázkami je možné overiť napríklad študentovu znalosť faktov, jeho schopnosť chápať informáciám v poskytnutých materiáloch alebo schopnosť nájsť súvislosti medzi informáciami (Heilman, 2011).

Systém schopný generovať otázky z učebného textu by bol nápomocný obom zúčastneným stranám v procese výučby. Učitelia by ho mohli využívať ako pomocný nástroj pri zostavovaní testov, keďže pokryť požadovaný obsah manuálne vytvorenými otázkami je náročná a zdĺhavá úloha. Rovnako by systém mohol byť využívaný študentami, ktorým by pomáhal pri odhaľovaní vlastných nedostatkov počas štúdia a umožnil by efektívnejšie využitie času.

1.2 Ciele práce

Naším cieľom pri písaní tejto práce je preskúmať možnosti automatického generovania otázok z učebného textu. Po tomto prieskume sa následne zameriame na porovnanie dostupných riešení, na základe ktorého sa pokúsime vybrať a skombinovať prístupy s čo najväčším potenciálom automaticky generovať korektné otázky z textu. Finálnym produktom bude aplikácia určená pre učiteľov, ktorej cieľom bude asistovať učiteľom pri tvorbe testov a teda navrhnúť im možné otázky vytvorené z poskytnutého vstupu. Učitelia budú môcť tieto otázky dodatočne upravovať a následne z nich vytvoriť test pre študentov.

Kapitola 2

Analýza tvorby otázok

V tejto časti sa venujeme typom otázok a možných odpovedí (kapitola 2.1), porovnaniu vstupov, z ktorých môže byť otázka generovaná (kapitola 2.2), rovnako ako problémom, ktoré komplikujú tvorbu otázok (kapitola 2.6).

2.1 Typy otázok

Prirodzený jazyk ponúka obrovské spektrum otázok, ktoré je možné rozdeliť do kategórií podľa rôznych kritérií. Tomuto rozdeleniu sa venujú autori v prácach (Graesser et al., 2008) a (Heilman, 2011), podľa ktorých je možné otázky rozdeľovať na základe nasledujúcich kritérií.

2.1.1 Rozdelenie podľa účelu otázky

Otázky môžeme rozdeliť do skupín napríklad podľa cieľu, s akým bola otázka položená.

- ujasnenie nejasností (*Ako sa dostanem do Mlynskej Doliny?*)
- overenie vedomostí (*Aké je hlavné mesto Švédska?*)
- udržanie konverzácie a pozornosti (*Ako sa máš?*)
- zachovať sa slušne a podľa etikety (*Mohol by si mi podať pero?*)

2.1.2 Rozdelenie podľa typu informácie, ktorú chceme získať

Ďalším delením je delenie na základe typu informácie, ktorú chceme získať.

- overenie pravdivosti informácie (*Je Madrid hlavným mestom Španielska?*)
- doplnenie čiastočnej informácie (*Za aké futbalové kluby nastupoval Marek Hamšík počas jeho pôsobenia na Slovensku?*)
- uvedenie príkladu (*Uveď príklad filmu, v ktorom hral Leonardo di Caprio.*)
- ohodnotenie informácie (*Je Peter Sagan úspešným športovcom?*)

2.1.3 Rozdelenie podľa zložitosti vyžadovanej odpovede

Delenie na základe zložitosti odpovede rozdeľuje otázky na základe toho, akú zložitú odpoveď vyžadujú.

- jednoduché otázky, na ktoré je možné odpovedať jednoduchou vetou (*Z akého typu látok je zložený Saturn?*)
- komplexné otázky, ktoré vyžadujú podrobnejšie vysvetlenie (*Aký je vplyv sociálnych sietí na mládež?*)

Podobným delením je delenie otázok na uzavreté a otvorené.

- uzavreté otázky vyžadujú odpoveď v tvare áno alebo nie (*Boli Košice hlavným mestom Československa?*)
- otvorené otázky, ktoré pri odpovedi poskytujú voľnosť a odpoveď môže byť v rôznych podobách (*Prečo je dôležité recyklovať odpadky?*)

2.2 Porovnanie tvorby otázok z vety alebo z odseku

Otázka môže byť generovaná z jednotlivých viet, ale aj z kompletných odsekov. Pri tvorbe otázky zo samostatnej vety môžeme vychádzať z faktu, že odpoveď na otázku je obsiahnutá v zadanej vete. Toto je samozrejmé, keďže vstup neposkytuje žiadne ďalšie informácie o oblasti, ktorou sa veta zaoberá. Samotný proces tvorby otázky podrobnejšie rozoberáme v kapitole 2.5.

Pri generovaní otázky z odseku je nevyhnutné si uvedomiť, že odpoveď na otázku nemusí byť obsiahnutá vo vete, z ktorej otázka priamo vznikla. Veľké množstvo viet by po ich osamostatnení stratilo svoj význam, nakoľko sa viazali na ďalšie informácie uvedené vo zvyšku odseku. Pre naprogramovaný systém je veľmi náročné tieto vzťahy medzi vetami rozpoznať a správne interpretovať. Napriek tomu je možné určitými technikami vytvoriť otázky z celého odseku.

V práci (Agarwal – Mannem, 2011) sa autori zameriavajú na tvorbu otázok, ktoré sú tvorené formou vynechaného miesta v texte, na ktoré musí študent zareagovať doplnením správnej odpovede z ponúknutých možností. Základnou úlohou pri tvorbe takéhoto typu otázky je identifikácia kľúčových slov a slovných spojení, na ktoré sa bude vytvorená otázka pýtať. Pri tejto úlohe použili tri metódy, ktorými ohodnotili dôležitosť jednotlivých slov. Pred samotným ohodnotením vybrali z textu len podstatné mená, prídavné mená a slová určujúce nejaké množstvo. Každé z týchto slov najskôr ohodnotili na základe počtu jeho výskytov v dokumente, následne zvýšili hodnotenie slovám, ktoré sa nachádzali priamo v nadpise a na záver pridali do hodnotenia informáciu o pozícii slova v syntaktickom strome vety, ktorá indikuje dôležitosť slova v danej vete. Po ohodnotení dôležitosti jednotlivých slov je možné zamerať sa na tvorbu otázok týkajúcich sa kľúčových informácií. Odpovede na otázky sú teda priamo obsiahnuté v texte. Avšak tento prístup často môže viesť k výsledku, že študenti budú na otázky odpovedať na základe zapamätaných informácií, bez potreby hlbšieho porozumenia (Lin et al., 2007).

Na tento problém sa zameriavajú autori v práci (Lin et al., 2007), kde študentovi neponúkajú ako správnu možnosť slovo, ktoré sa pôvodne nachádzalo vo vete. Ako svoju doménu si zvolili prídavné mená a tvorbu otázok zameraných na správne pochopenie významu prídavného mena v texte. Ich cieľom je teda motivovať študentov k učeniu s porozumením.

Tvorba otázok z odsekov so sebou prináša viaceré problémy a obmedzenia (kapitola 2.6), avšak systém, ktorý dokáže generovať otázky z odsekov môže byť pre učiteľov podstatne väčším prínosom pri tvorbe testov ako systém, ktorý je zameraný na tvorbu otázok zo samostatných viet.

2.3 Učebné materiály

Učebné materiály publikované formou písaného textu môžeme rozdeliť do dvoch základných kategórií.

Prvou z nich sú učebnice napísané odborníkmi priamo za účelom naučiť čitateľa nové informácie. Tieto knihy sú často kvalitne pripravené a okrem informácií a textov obsahujú rôzne ďalšie pomocné časti, zamerané na zvýšenie efektivity výučby. Môže ísť napríklad o cvičenia z vyučovanej látky alebo prehľad informácií po určitej ucelenej časti (Heilman, 2011). Ide teda o kvalitne pripravené zdroje informácií s premysleným plánom výuky a rôznymi sekciami podporujúcimi zapamätanie požadovaných informácií. Jedným z najpodstatnejších problémov učebníc je však ich veľkosť a cena (Heilman, 2011). Učebnice je potrebné tlačiť a distribuovať, z čoho vyplýva, že produkt, ktorý sa dostane k čitateľovi je podstatne drahší, než bola jeho hodnota ihneď po vypracovaní. Túto nevýhodu je však v dnešnej dobe možné zmierňovať elektronickými verziami, pri ktorých odpadajú náklady na tlač a distribúciu. Ďalším problémom učebníc je ich aktuálnosť. Je veľmi náročné napísať učebnicu, ktorej obsah bude aktuálny aj po uplynutí určitého času. Svet sa neustále mení a preto sa často stáva, že informácie v učebnici sú nekorektné už po veľmi krátkej dobe od vydania. Pri tlačенých verziách je teda udržiavanie aktuálnosti informácií veľmi náročné a preto aj často v konečnom dôsledku zanedbávané.

Ako protiklad k učebniciam by sme chceli spomenúť fenomén dnešnej doby, internet. Na internete je možné v dnešnej dobe nájsť informácie takmer o čomkoľvek a vyhľadávanie je vďaka moderným vyhľadávacím nástrojom veľmi rýchle a efektívne. Obsah je aktualizovaný a rozširovaný v podstate neustále a preto je možné nájsť na internete aj tie najnovšie informácie. Väčšina z týchto výhod vyplýva z toho, že internet umožňuje prispievať akémukoľvek používateľovi. Z toho však paradoxne vyplýva aj najpodstatnejšie negatívum internetu ako prostriedku využívaného pri štúdiu. Drvivá väčšina textov zdieľaných na internete nebola vytvorená za účelom stať sa učebným materiálom a taktiež nemuseli byť napísané overenými odborníkmi. Preto je nevyhnutné tieto informácie rozlišovať a filtrovať na základe ich dôveryhodnosti (Heilman, 2011). Ďalšou nevýhodou v porovnaní s učebnicami je častá absencia pomocných sekcií, ako napríklad test na overenie získaných vedomostí.

2.4 Úrovne poznávacích cieľov

Kategorizácia poznávacích cieľov ich rozdeľuje do viacerých úrovní, z ktorých každá predstavuje určitú mieru poznania. Najznámejším modelom je Bloomova taxonómia, publikovaná v roku 1956, ktorá tieto ciele rozdeľuje do šiestich úrovní. Taxonómia je hierarchická, z čoho vyplýva, že ak študent dosiahol istú úroveň poznania, automaticky musel dosiahnuť aj všetky nižšie úrovne. Úrovne Bloomovej taxonómie (hierarchicky od najnižšej po najvyššiu) sú (Forehand, 2010):

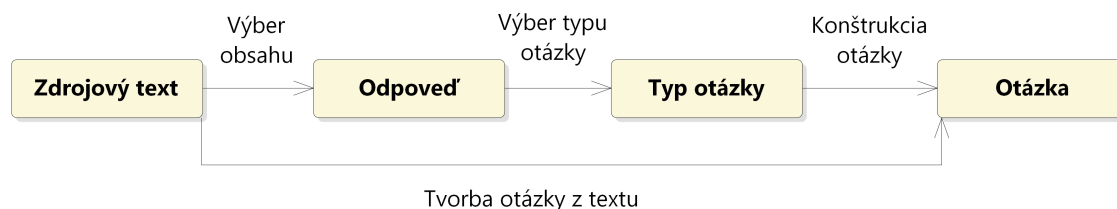
- zapamätanie (*zopakovať, rozpoznať, reprodukovať, vymenovať*),
- porozumenie (*vysvetliť, rozlíšiť, zaradiť, zhrnúť*),
- aplikácia (*použiť, navrhnuť*),
- analýza (*kriticky myslieť, skúmať*),
- syntéza (*spájať, odvodíť, tvoriť*),
- hodnotenie (*posúdiť*).



Obr. 2.1: Úrovne Bloomovej taxonómie.

2.5 Proces tvorby otázky

Generovanie otázky je proces, ktorého úlohou je spracovať poskytnutú vetu a ako výstup vrátiť otázku, ktorá je vytvorená na základe zdrojovej vety a pýta sa na informáciu, ktorá sa v nej nachádza. Tento proces je možné rozdeliť na 3 základné časti (Rus et al., 2009).



Obr. 2.2: Tvorba otázky z textu (Rus et al., 2009).

2.5.1 Výber obsahu otázky

Prvým krokom pri tvorbe otázky je výber toho, na čo sa bude otázka pýtať. Na túto úlohu je možné použiť rôzne postupy, no v princípe sa všetky usilujú o to, aby sa otázky týkali informácií, ktoré sú v poskytnutom kontexte dôležité (Rus et al., 2012). Aby sa na otázku dalo jednoznačne odpovedať, požadovaná odpoveď by mala vyplývať z informácií, ktoré boli poskytnuté.

2.5.2 Výber typu otázky

Po identifikovaní informácie, na ktorú sa otázka bude pýtať, je potrebné zvoliť vhodný typ otázky, ktorý bude použitý. Inými slovami ide o to, že keď už vieme, na čo sa ideme spýtať, musíme rozhodnúť ako sa na to najvhodnejšie spýtať (Rus et al., 2012). Opäť ide o zložitejší proces, keďže množina typov otázok je veľmi veľká. Je potrebné rozhodnúť, či otázka bude vytvorená zmenou slovosledu alebo vynechaním priestoru, či pôjde o otvorenú alebo uzavretú otázku, prípadne či budú poskytnuté možnosti.

2.5.3 Konštrukcia otázky

Na záver je otázku potrebné na základe predchádzajúcich dvoch krokov vytvoriť. V závislosti od zvolených možností je vykonaná konštrukcia, ktorej cieľom je vytvoriť korektnú otázku.

2.6 Problémy a výzvy pri tvorbe otázok

Automatické generovanie otázok z textu však so sebou prináša relatívne veľké množstvo problémov a výziev. Heilman ich vo svojej práci rozdelil do troch kategórií (Heilman, 2011):

- lexikálne (kapitola 2.6.1),
- syntaktické (kapitola 2.6.2),
- diskurz (kapitola 2.6.3).

2.6.1 Lexikálne výzvy

Prvou lexikálnou výzvou pri tvorbe otázok je správne mapovanie otázok na odpovede. Inými slovami, opytovacie zámeno musí korešpondovať s odpoveďou k otázke. Vďaka nástrojom na spracovanie prirodzeného jazyka sme schopní pomerne presne rozpoznávať entity a vďaka tomu zvoliť vhodné opytovacie zámeno, avšak tieto nástroje nie sú dokonalé a môžu slovo označovať nesprávne, z čoho následne pramení zvolenie nesprávneho opytovacieho zámena a tým pádom nesprávna konštrukcia otázky (Heilman, 2011).

Príklad:

Santiago Bernabéu je štadión Realu Madrid.

V prípade, ak systém nerozpozná, že Santiago Bernabéu nereprezentuje v tejto vete osobu, ale názov štadiónu, môže vzniknúť nasledujúca otázka.

Kto je štadión Realu Madrid?

Ďalší problém môže nastať v prípade, ak by sme sa snažili z vety vytvoriť všetky možné otázky. Rôzne otázky môžu mať rovnaký význam a pýtať sa na zhodnú informáciu. Klasickým príkladom je variácia rovnakej vety v činnom a trpnom rode, keď majú dve rozdielne vety totožný význam. Vytvoriť všetky možné otázky je preto vzhľadom k veľkému množstvu ich variácií veľmi náročné (Heilman, 2011).

Príklad:

Joanne Rowlingová napísala Harryho Pottera.

Pri tejto vete je možné vytvoriť veľké množstvo otázok s totožným významom. Automatické vygenerovanie všetkých možností nie je na základe poskytnutej vety možné, napriek tomu, že všetky sú správne.

Kto napísal Harryho Pottera?

Kto je autorom Harryho Pottera?

Kým bol napísaný Harry Potter?

Posledným lexikálnym problémom sú ustálené slovné spojenia, ktoré sú veľmi ťažko rozpoznateľné systémom. Pri ustálených slovných spojeniach nie je možné určiť ich význam na základe významov jednotlivých členov a z toho dôvodu často spôsobujú nekorektne vytvorenú otázku, keďže systém pracoval so slovami ako oddelenými členmi a tým ignoroval význam ustáleného slovného spojenia (Heilman, 2011).

Príklad:

Vojaci zrovnali mesto so zemou.

Zrovnať so zemou je ustálené slovné spojenie, ktorého význam nemusí vyjadrovať fyzické zrovnanie nejakého objektu so zemou. Bez pochopenia skutočného významu tohto ustáleného slovného spojenia by mohla otázka vyzeráť nasledovne.

S čím zrovnali vojaci mesto?

2.6.2 Syntaktické výzvy

Pri práci so syntaxou vety sa spoliehame výlučne na nástroje, ktoré zabezpečujú spracovanie prirodzeného jazyka. Tieto nástroje poskytujú možnosť identifikovať slovné druhy vo vete, avšak ani v tejto úlohe nedosahujú perfektné výsledky. Ak príde k nesprávnemu označeniu slovného druhu, môže z toho vyplývať nesprávna reprezentácia vzťahov medzi jednotlivými slovami a z toho dôvodu aj nesprávne vytvorená veta (Heilman, 2011).

Príklad (príklad je uvedený v anglickom jazyku kvôli jasnejšej demonštrácii problému):

The city's chief manufactures are processed food, lumber, and textiles.

Slovo manufactures sa môže vyskytovať v niektorých vetách aj vo forme slovesa. V prípade, že v tejto vete bude označené ako sloveso, môže prísť k vytvoreniu nekorektnej otázky.

Who manufactures are processed food, lumber, and textiles? (Heilman, 2011)

Na druhej strane, ani správne označkovanie slov nemusí byť postačujúce k vytvoreniu správnej otázky. Problémy so sebou prinášajú dlhé a zložité vety, ktoré obsahujú veľa informácií. Pri takýchto vetách je často využívaný postup zjednodušovania viet, keď sú na základe vzťahov medzi slovami identifikované jednoduché vety a otázky sú vytvárané z nich (Heilman, 2011).

Príklad:

Hlavným mestom Slovenska je Bratislava, najväčšie mesto na našom území, ktorého dominantou je Bratislavský hrad dokončený v 18. storočí.

Ide o súvetie zložené z dvoch viet, ktoré navyše obsahujú pomerne veľké množstvo informácií. Pri vetách podobného typu je vhodné pred samotnou tvorbou otázok danú vetu zjednodušiť a rozdeliť na jednoduché vety, ktoré budú spoločne obsahovať všetky informácie z pôvodnej vety.

Hlavným mestom Slovenska je Bratislava.

Bratislava je najväčšie mesto Slovenska.

Dominantou Bratislavy je Bratislavský hrad.

Bratislavský hrad bol dokončený v 18. storočí.

2.6.3 Výzvy diskurzu

Diskurz je definovaný ako textový útvar, ktorý presahuje jednu vetu (Bočák, 2009). S týmto sú často spojené rôzne súvislosti a prepojenia medzi jednotlivými vetami, prípadne s inými nadobutnutými vedomosťami. V prirodzenom jazyku si to často neuvedomujeme, avšak po vytrhnutí vety z kontextu môže táto veta zmeniť, prípadne stratiť svoj význam.

Iným prípadom je situácia, keď veta obsahuje informáciu, ktorá nevyplýva priamo zo vzťahov medzi slovami, no napriek tomu sa vo vete nachádza. Systém nemá schopnosť domyslieť si súvislosti, ktoré si dokáže domyslieť človek na základe nadobutných znalostí a napriek tomu, že pre nás je určitá informácia v texte úplne zrejmá, systém ju nedokáže rozpoznať.

Extrémny prípad nastane vtedy, keď sa prejaví fakt, že systém nemá základné poznatky o svete a potrebný rozhľad. Toto nastane v prípade, keď zdrojová veta bola vytvorená za predpokladu, že jej čitateľ je ľudská bytosť a má určitý všeobecný rozhľad. Vo vete preto nie je myšlienka vyjadrená do absolútnych detailov, keďže predpokladá istú dávku znalostí a toto spôsobuje neschopnosť systému vytvoriť z vety otázku.

Príklad:

V rámci celej Wikipedia stránky o Abrahamovi Lincolnovi nie je ani v jednej vete priamo spomenuté, kto ho zavraždil. V texte sa síce nachádzajú nepriame informácie, z ktorých to jasne vyplýva, ale ak systém nie je schopný z týchto informácií zistiť meno vraha Abrahama Lincolna, nemôže sa na túto informáciu spýtať. Naopak pre človeka, ktorý má schopnosť domyslieť si súvislosti, informácia o tom, že niekto na Lincolna namieril zbraň a vystrelil jasne znamená, že práve tento človek je jeho vrahom (Heilman, 2011).

2.7 Zhrnutie prístupov pri tvorbe otázok

V priebehu analýzy dostupných riešení sme narazili na rozličné prístupy a metódy. V práci (Hussein et al., 2014) sa autori zameriavajú na generovanie otázok na základe naučených pravidiel. Systém najskôr prechádza tréningovou fázou, počas ktorej si ukladá jednotlivé vzory viet a k nim prislúchajúce typy otázok, ktoré je možné z nich vytvoriť. Po ukončení tréningovej fázy sú naučené pravidlá aplikované na základe identifikovaných vzorov vo vetách na vstupe.

Podobné riešenie bolo opísané v práci (Agarwal et al., 2011), v ktorej sa autori venujú tvorbe otázok zo súvetí. Pre rôzne typy súvetí definovali pravidlá, ktorými je možné z daného typu súvetia vytvoriť otázku. Pri testovaní najskôr identifikovali typ súvetia na základe spojky a následne aplikovali pripravené pravidlo.

V práci (Ali et al., 2010) autori nevytvárajú otázky priamo z poskytnutej vety, nakoľko od predchádzajúcich dvoch riešení. Pred samotnou tvorbou otázky najskôr extrahujú jednotlivé informácie z viet reprezentované elementárnymi vetami. Otázky sú tvorené až z týchto jednoduchých viet.

Odlišný prístup zvolili autori v práci (Agarwal – Mannem, 2011). Základným rozdielom je, že pri tvorbe otázok nepracovali so vzťahmi medzi slovami ani stavbou vety. Vytvárali testové otázky, v ktorých na mieste kľúčových slov bolo voľné miesto, do ktorého bolo potrebné doplniť správnu odpoveď. Pri výbere vhodných kandidátov na vynechané miesta vytvorili vlastné pravidlá, ktoré aplikovali. Ohodnotením jednotlivých kandidátov na vytvorenie otázky sa zaoberajú autori v práci (Beinborn et al., 2015).

Pri takomto type testových môžu byť otázky zodpovedané bez pochopenia. Tomuto problému sa venujú autori v práci (Lin et al., 2007), ktorí pri výbere možností nepoužili slovo, ktoré sa pôvodne nachádzalo v texte, ale použili jeho synonymum.

Kapitola 3

Spracovanie prirodzeného jazyka

Spracovanie prirodzeného jazyka je oblasť, ktorá skúma ako môžu počítače porozumieť a pracovať s prirodzeným jazykom, za účelom vykonávať užitočnú činnosť. Výskum je do veľkej miery orientovaný na spôsoby, akými ľudia chápu a používajú prirodzený jazyk. Na základe toho sa snažia vyvíjať vhodné spôsoby, ktorými sa snažia doceliť, aby sa počítače čo najviac priblížili pri spracovaní prirodzeného jazyka k ľuďom (Chowdhury, 2003). To je však veľmi náročné, vzhľadom k nasledujúcim skutočnostiam:

- Ľudia používajú veľké množstvá jazykov.
- V rôznych jazykoch sa nachádzajú rôzne špecifické kombinácie slov, ako napríklad homonymá, synonymá alebo antonymá.
- Pre každý jazyk môžu existovať rôzne pravidlá pri stavbe vety.

Tomuto problému je možné sa vyhnúť zameraním sa na špecifický jazyk, no aj pri takejto špecializácii sa objavujú nové problémy. Jazyk sa neustále vyvíja, objavujú sa nové slová, zastarané slová sa vytrácajú, ľudia používajú slangové a nespisovné slová.

Napriek týmto problémom sa s postupom času podarilo vyvinúť relatívne kvalitné nástroje, ktoré vykonávajú rôzne úlohy spojené so spracovaním prirodzeného jazyka (kapitola 3.1).

3.1 Úlohy spracovania prirodzeného jazyka

Vzhľadom k tomu, aké veľké množstvo výziev so sebou prináša oblasť spracovania prirodzeného jazyka, vzniklo aj veľké množstvo úloh, ktorým sa venuje. Na základe analýzy sme ako kľúčové identifikovali nasledujúce:

- značkovanie slovných druhov (*angl. Part-Of-Speech Tagging*) (kapitola 3.1.1),
- rozpoznanie názvoslovných entít (*angl. Named Entity Recognition*) (kapitola 3.1.2),
- lematizácia (*angl. Lemmatization*) (kapitola 3.1.3),
- identifikácia koreferencií (*angl. Coreference Resolution*),
- určovanie sémantických rolí (*angl. Semantic Role Labeling*),
- extrahovanie informácií (*angl. Information Extraction*).

3.1.1 Značkovanie slovných druhov

Označenie slovných druhov je proces, pri ktorom je ku každému slovu priradená značka, ktorá značí jeho syntaktickú úlohu vo vete, inými slovami jeho slovný druh (Collobert et al., 2011). Pri tvorbe otázok ide o jednu z najzákladnejších úloh spracovania prirodzeného jazyka, keďže slovný druh je takmer najzákladnejšou syntaktickou informáciou o slove.

Označenie slovného druhu slova je možné využiť napríklad pri ohodnotení dôležitosti slova v rámci vety, keďže môžeme predpokladať, že podstatné meno je vhodnejší kandidát na odpoveď ako spojka.

Problémy nastávajú pri slovách, ktoré sa môžu v rôznych vetách vyskytovať ako rôzne slovné druhy. V takom prípade môže nastať chybné označenie slovného druhu, z ktorého môžu následne vyplývať ďalšie nepresnosti.

3.1.2 Rozpoznanie názvoslovných entít

Názvoslovné entity vyjadrujú informáciu, že slovo spadá do určitej špecifickej kategórie (Collobert et al., 2011). Týchto kategórií je veľké množstvo, avšak my sme medzi základné názvoslovné entity zaradili:

- osoba (angl. person),
- lokalita (angl. location),
- organizácia (angl. organization),
- dátum (angl. date),
- číslo (angl. number),
- poradie (angl. ordinal),
- trvanie (angl. duration),
- peniaze (angl. money),
- percentá (angl. percent).

Názvoslovné entity majú široké uplatnenie pri tvorbe otázok, keďže prinášajú veľmi špecifickú informáciu o slovách a navyše je možné pomocou nich identifikovať slová, ktoré nesú dôležitú informáciu.

3.1.3 Lematizácia

Lematizácia je proces hľadania normalizovanej formy slova (Plisson et al., 2004). Vďaka tejto normalizovanej forme je možné pri generovaní možných odpovedí dodržať pravidlo, aby všetky možnosti boli v rovnakom tvare. Pri tvorbe otázok zmenou slovosledu je možné vďaka normalizovanému tvaru slovesa (neurčitok) zmeniť pozíciu slovesa a aj jeho tvar do požadovanej formy. Pre rôzne slovné druhy sú použité rôzne pravidlá pre normalizovaný tvar, napríklad pre slovesá ide o neurčitok a pre podstatné mená ide o nominatív jednotného čísla.

Kapitola 4

Návrh systému na tvorbu testov

Po vykonaní analýzy sme sa rozhodli pripraviť návrh systému, ktorý bude zameraný na tvorbu otázok a testov z učebného textu. Pri návrhu sme sa zamerali na návrh architektúry systému (kapitola 4.1), spôsobu spracovania vstupného textu (kapitola 4.2), spôsobu tvorby otázok a testov (kapitoly 4.3 a 4.5) a návrhu klientskej aplikácie (kapitola 4.6).

Systém je navrhnutý tak, aby bolo možné tvoriť otázky z celých odsekov a aby bolo možné tieto testy vyplňať a vyhodnocovať. Zamerali sme sa na jednotlivé prípady použitia, ktoré zabezpečia požadovanú funkcionálnu systém. Spôsob spracovania textu a uchovania informácií o texte je navrhnutý tak, aby sme pri tvorbe otázok mali k dispozícii všetky potrebné informácie. Systém bude pracovať s anglickým jazykom.

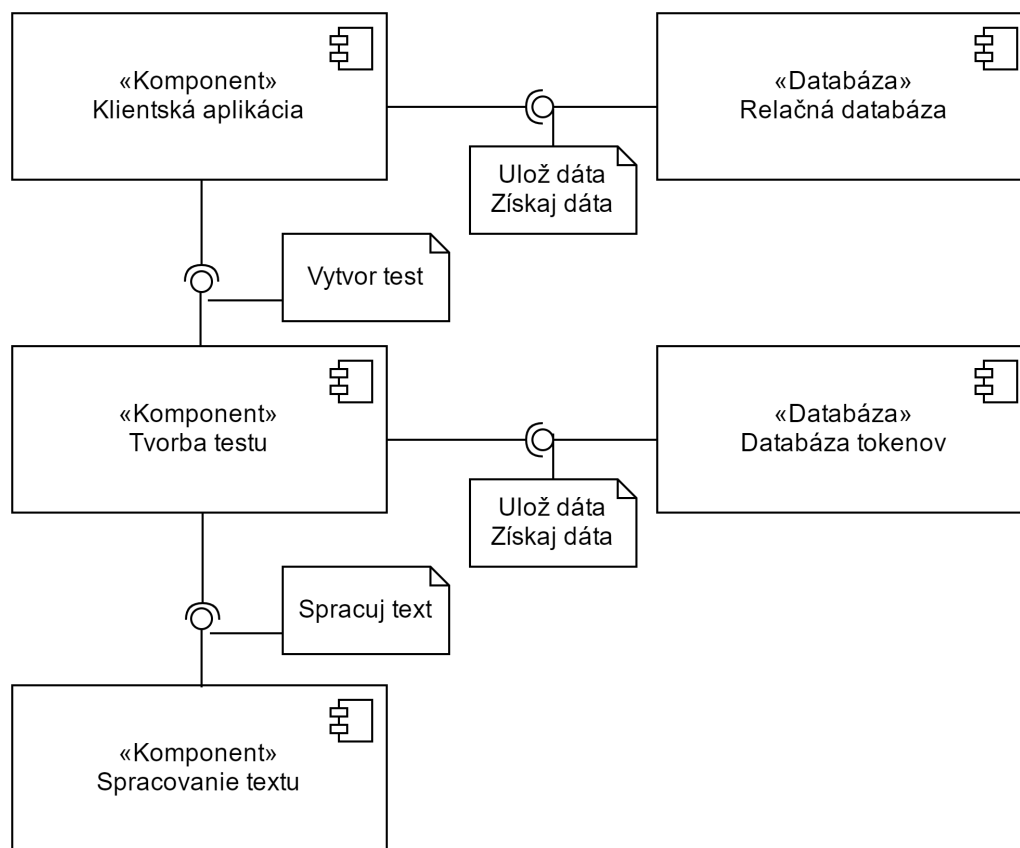
Využívanie takto navrhnutého systému by zefektívnilo prácu učiteľov, ktorí by nemuseli manuálne identifikovať kľúčové informácie v texte, k správnym odpovediam vymýšľať ďalšie možnosti a ani vytvárať viacero rôznych testov pre väčšiu skupinu študentov. Všetky tieto činnosti budú v nami navrhnutom systéme vykonávané automatizovane.

4.1 Architektúra systému

Tvorba testu zo vstupného textu je komplexný proces, ktorý je možné rozdeliť medzi tri základné úlohy:

- spracovanie vstupného textu a uchovanie potrebných informácií (kapitola 4.2),
- vytvorenie otázok a z nich zloženého testu (kapitoly 4.3, 4.4 a 4.5),
- zobrazenie informácií a interakcia používateľa so systémom (kapitola 4.6).

Každá z uvedených úloh je špecifická a preto sme pre každú z nich navrhli samostatný komponent. Vďaka tomuto prístupu sa budeme môcť zamerať na vývoj jednotlivých komponentov samostatne a znížime závislosti medzi ich implementáciou. Jednotlivé komponenty budú medzi sebou komunikovať prostredníctvom navrhnutých rozhraní a systém bude možné v prípade potreby rozširovať o ďalšie komponenty.



Obr. 4.1: Diagram komponentov navrhnutého systému.

Komponent *spracovanie textu* bude slúžiť na získanie potrebných informácií o texte, ktoré bude možné využiť pri tvorbe otázok. Rozhranie tohto komponentu bude poskytovať prístup k týmto informáciám a vďaka nemu budú môcť tieto informácie získať aj iné komponenty, ktoré sú zamerané na iné úlohy. Výstupom budú informácie o zadanom texte štruktúrované podľa jednotlivých úrovní, na ktoré sa tieto informácie viažu (kapitoly 4.2.1, 4.2.2, 4.2.3 a 4.2.4). Komponent nebude závislý od iných komponentov aplikácie a bude schopný fungovať samostatne. V prípade, že v budúcnosti bude potrebné pridať k textu novú informáciu, zmenu bude možné vykonať v rámci tohto komponentu a informácia bude pridaná do jeho výstupu.

Komponent *tvorba testu* bude slúžiť na vytvorenie otázok a z nich zloženého testu. Rozhranie komponentu bude poskytovať možnosť vygenerovania testu na základe zadaného vstupného textu. Pred samotnou tvorbou otázok je potrebné zadaný text predspracovať a získať potrebné informácie, na čo slúži komponent *spracovanie textu*. Vďaka jeho rozhraniu bude možné získať potrebné informácie o texte a následne vytvoriť test (kapitoly 4.3, 4.4 a 4.5). Výstupom komponentu budú informácie o teste a vytvorených otázkach.

So systémom na tvorbu testov bude potrebné komunikovať a umožniť používateľom využívať jeho možnosti. Pre tieto úlohy je navrhnutý komponent *klientská aplikácia*, ktorá je určená na interakciu používateľa so systémom. Pri požiadavke na vytvorenie testu bude *klientská aplikácia* komunikovať s komponentom *tvorba testu*, od ktorého si vyžiada vytvorenie testu z textu, ktorý zadal do rozhrania používateľ. Po vytvorení testu bude mať používateľ možnosť tento test spravovať a poskytnúť ho ostatným používateľom na vyplnenie.

4.2 Spracovanie vstupného textu

Spracovanie vstupného textu je úplne prvou úlohou, ktorú je potrebné v rámci procesu tvorby testu vykonať. Spracovanie prirodzeného jazyka (kapitola 3) sa venuje veľkému množstvu úloh a my sme z nich vybrali tie, ktoré plánujeme využiť pri tvorbe otázok. Informácie o texte budú uchované v štruktúre, ktorá vychádza zo štruktúry samotného textu. Niektoré informácie sú špecifické pre jednotlivé tokeny v texte, zatiaľ čo iné informácie je vhodné uchovať na úrovni viet, prípadne celého článku. Okrem informácií na týchto troch úrovniach plánujeme využiť OpenIE trojice, čiže zjednodušené formy pôvodných viet, ktoré sú tvorené z troch častí (podmet, vzťah a predmet). Tieto trojice nám budú pri komplexnejších vetách poskytovať konkrétnejšie informácie nachádzajúce sa v týchto vetách.

4.2.1 Token

Najzákladnejšou jednotkou textu, ku ktorej budú uchované informácie, je token. Ide o jednotlivé elementy, z ktorých sa skladá veta. Môže ísť o slová alebo o interpunkčné znamienka, napríklad bodka alebo čiarka.

Ku každému tokenu si plánujeme uchovať nasledujúce informácie:

- pôvodný text,
- léma (kapitola 3.1.3),
- pozícia vo vete,
- pozícia v texte,
- pozícia prvého znaku v texte,
- pozícia posledného znaku v texte,
- slovný druh (kapitola 3.1.1),
- názvoslovná entita (kapitola 3.1.2),
- pozícia vety v texte,
- predchádzajúci znak,
- nasledujúci znak.

4.2.2 OpenIE trojica

OpenIE trojica je zjednodušená forma vety, ktorá obsahuje informáciu nachádzajúcu sa v pôvodnej vete. Každá trojica je zložená z podmetu, predmetu a ich vzťahu. Pre každú z týchto častí budú uchované informácie, z ktorých bude možné trojicu zrekonštruovať, prípadne pristupovať k detailnejším informáciám o tokenoch nachádzajúcich sa v jednotlivých častiach trojice.

Ku každej trojici si teda plánujeme uchovať nasledujúce informácie:

- zoznam tokenov tvoriacich podmet spolu s informáciami o daných tokenoch,
- pôvodný text podmetu,
- pozícia prvého tokenu podmetu v texte,
- pozícia posledného tokenu podmetu v texte,
- pozícia prvého znaku podmetu v texte,
- pozícia posledného znaku podmetu v texte,
- zoznam tokenov tvoriacich predmet spolu s informáciami o daných tokenoch,
- pôvodný text predmetu,
- pozícia prvého tokenu predmetu v texte,
- pozícia posledného tokenu predmetu v texte,
- pozícia prvého znaku predmetu v texte,
- pozícia posledného znaku predmetu v texte,
- zoznam tokenov tvoriacich vzťah spolu s informáciami o daných tokenoch,
- pôvodný text vzťahu,
- pozícia prvého tokenu vzťahu v texte,
- pozícia posledného tokenu vzťahu v texte,
- pozícia prvého znaku vzťahu v texte,
- pozícia posledného znaku vzťahu v texte.

4.2.3 Veta

O vete si nebudeme uchovávať žiadne špecifické informácie, pretože v našom systéme bude veta reprezentovaná zoznamom tokenov, ktoré obsahujú všetky potrebné informácie. Okrem tokenov, ktoré sa nachádzajú v danej vete, bude obsahovať každá veta aj zoznam OpenIE trojíc, ktoré z nej vznikli.

Ku každej vete teda budeme uchovávať:

- zoznam tokenov spolu s informáciami o daných tokenoch,
- zoznam OpenIE trojíc a informácií o daných trojiciach.

4.2.4 Článok

Najrozsiahljším elementom, ktorý bude reprezentovať samotný text, ktorý bol spracovávaný, je článok. V rámci článku sa budú na rôznych úrovniach nachádzať všetky informácie, ktoré si uchováame počas spracovávania textu. Článok sa bude skladať zo zoznamu viet, v rámci ktorých budú obsiahnuté všetky informácie o daných vetách a ich prvkoch. Okrem zoznamu

viet bude článok obsahovať aj zoznam nájdených tokenov pre každú názvoslovnú entitu. Tieto zoznamy bude taktiež možné využiť pri výbere možných odpovedí pri tvorbe otázok.

Článok bude obsahovať:

- zoznam viet a informácií o daných vetách,
- zoznam tokenov, ktorých názvoslovná entita je *osoba*,
- zoznam tokenov, ktorých názvoslovná entita je *miesto*,
- zoznam tokenov, ktorých názvoslovná entita je *organizácia*,
- zoznam tokenov, ktorých názvoslovná entita je *dátum*,
- zoznam tokenov, ktorých názvoslovná entita je *číslo*,
- zoznam tokenov, ktorých názvoslovná entita je *poradie*,
- zoznam tokenov, ktorých názvoslovná entita je *trvanie*,
- zoznam tokenov, ktorých názvoslovná entita je *peniaze*,
- zoznam tokenov, ktorých názvoslovná entita je *percentá*.

4.2.5 Štruktúra spracovaného textu

Po získaní všetkých potrebných informácií o texte ich bude potrebné uložiť do určitej štruktúry a v tejto forme ich poskytovať ďalším komponentom. Pri definovaní štruktúry výsledných dát sme využili reprezentáciu v strome, ktorú je možné efektívne používať pri ich prenose. Táto štruktúra vychádza zo spracovaného článku (kapitola 4.2.4) a vďaka tomu, že jednotlivé úrovne elementov sú zoradené hierarchicky, ak na pozíciu koreňového prvku vložíme tento článok (kapitola 4.2.4), automaticky všetky získané informácie vytvoria stromovú štruktúru.

4.3 Tvorba otázok

Po spracovaní vstupného textu a uložení potrebných informácií do vhodnej štruktúry (kapitola 4.2.5) bude možné prejsť k tvorbe samotných otázok. Keďže existuje veľké množstvo rôznych typov otázok (kapitola 2.1), rozhodli sme sa zamerať na jeden konkrétny typ. Všetky navrhnuté typy otázok (kapitola 4.4) budú mať rovnakú štruktúru a pôjde o otázky, pri ktorých je úlohou študenta doplniť správnu odpoveď do textu. Okrem správnej odpovede, ktorá sa nachádzala v pôvodnom texte, budú medzi možné odpovede vygenerované alebo vybrané tri ďalšie možnosti, ktoré budú podobné ako správna odpoveď. Pri tvorbe otázok budú využité informácie o texte, ktoré boli získané pri jeho spracovaní.

V systéme bude navrhnutých niekoľko typov otázok, ktoré budú mať definované vlastné pravidlá, ktoré budú musieť byť splnené pre úspešné vytvorenie otázky. Systém bude postupne skúšať aplikovať pravidlá jednotlivých typov otázok na spracovaný text a v prípade, že budú tieto pravidlá splnené, vytvorí sa otázka príslušného typu.

4.4 Navrhnuté typy otázok

V rámci nášho systému sme navrhli niekoľko typov otázok. Každý typ otázky bude mať na vstupe kompletný článok so všetkými o ňom získanými informáciami (kapitola 4.2.4). Pri každom type bude definované, akým spôsobom má spracovať tieto informácie a pokúšať sa na ne aplikovať svoje pravidlá. V prípade, že na určitú časť článku bude možné aplikovať pravidlá daného typu otázky, na tomto mieste bude vytvorená otázka tohto typu s príslušnými možnosťami. Po vytvorení otázky bude systém pokračovať v spracovávaní článku. Po spracovaní celého článku a vytvorení všetkých možných otázok daného typu budú tieto otázky vrátené na výstup. Takýmto spôsobom bude systém postupne aplikovať všetky definované typy otázok na spracovávaný článok.

4.4.1 Otázka so zameraním na rok

Pri otázke so zameraním na rok bude systém spracovávať jednotlivé tokeny nachádzajúce sa v článku. Pre každý token bude testovať podmienku, či je názvoslovná entita tokenu označená ako dátum. V prípade, že táto podmienka bude splnená, systém otestuje, či je text tokenu číslo väčšie ako -3000 a zároveň menšie ako 3000. V prípade, že aj táto podmienka bude splnená, systém prejde k tvorbe otázky.

Systém vytvorí novú otázku, ktorej atribúty nastaví na základe príslušných atribútov tokenu. Pri generovaní zoznamu ďalších možností sa do tohto zoznamu pridajú čísla, ktorých hodnota je maximálne o 10 nižšia alebo vyššia ako hodnota správnej odpovede. Z týchto čísel sa následne náhodne vyberú tri, ktoré budú pridané k otázke.

4.4.2 Otázka so zameraním na mesiac

Pri otázke so zameraním na mesiac sa bude postupovať veľmi podobne ako pri otázke so zameraním na rok (kapitola 4.4.1). Rozdielom bude časť, pri ktorej sa nebude testovať, či je text tokenu číslo, ale či text tokenu označuje niektorý z mesiacov. V prípade, že text tokenu označuje niektorý mesiac, systém prejde k tvorbe otázky.

Pri generovaní zoznamu ďalších možností sa do tohto zoznamu pridajú všetky označenia mesiacov, okrem toho, ktorý je označený ako správna odpoveď. Následne sa z tohto zoznamu náhodne vyberú tri mesiace, ktoré budú pridané k otázke.

4.4.3 Otázka so zameraním na dátum v úplnom tvare

Pri tomto type otázky bude systém opäť spracovávať tokeny v článku. Pri testovaní podmienky však nebude token testovaný s pomocou jeho názvoslovnej entity, ale regulárnym výrazom. Systém bude mať definovanú množinu regulárnych výrazov, ktoré budú akceptovať reťazce, ktoré reprezentujú dátum v úplnom tvare. V prípade, že text tokenu bude akceptovaný niektorým z týchto regulárnych výrazov, systém prejde k tvorbe otázky.

Pri generovaní zoznamu ďalších možností sa do tohto zoznamu pridá 20 ďalších dátumov vytvorených náhodnými kombináciami dní, mesiacov a rokov, ktorých hodnoty budú maximálne o 4 menšie alebo o 4 väčšie ako sú hodnoty pri správnej odpovedi. Z týchto dátumov sa následne vyberú tri dátumy, ktoré budú pridané k otázke.

4.4.4 Otázka so zameraním na názvoslovné entity

Tento typ otázky združuje viacero typov otázok. Ide o zovšeobecnenie otázok zameraných na:

- osobu,
- organizáciu,
- miesto,
- číslo,
- trvanie,
- peniaze,
- poradie,
- percentá.

Pri všetkých uvedených typoch sme navrhli rovnaký princíp, pri ktorom bude systém postupne spracovávať OpenIE trojice z článku (kapitola 4.2.2). Pri každej trojici môže byť otázka vytvorená na základe podmetu alebo predmetu tejto trojice. Podmienka pre tieto otázky bude splnená, ak všetky tokeny v podmete alebo predmete majú rovnakú názvoslovnú entitu, ktorá patrí do zoznamu uvedeného vyššie v tejto časti.

V prípade splnenia uvedenej podmienky systém prejde k vytvoreniu otázky, ktorej atribúty budú nastavené na základe atribútov jednotlivých tokenov podmetu alebo predmetu v závislosti od toho, ktorý z nich splnil uvedenú podmienku. Pri generovaní zoznamu ďalších možností systém využije zoznamy tokenov z článku (kapitola 4.2.4). V prípade, že nájdená názvoslovná entita bola osoba, miesto alebo organizácia, systém do zoznamu možností pridá aj všetky známe tokeny danej entity, získané zo všetkých spracovaných článkov. Z tohto zoznamu možností sa následne vyberú tri tokeny, ktoré budú pridané k otázke.

4.5 Tvorba testu a štruktúra informácií v teste

Pri tvorbe testu zo zadaného textu bude potrebné vykonať viacero úloh. Test bude zložený z otázok, ktoré bude systém schopný pre zadaný text vytvoriť, avšak ešte pred samotnou tvorbou otázok bude potrebné tento text spracovať (kapitola 4.2). Spracovaný článok bude dočasne uložený a na základe tohto článku budú vytvárané otázky pre test.

Po spracovaní článku bude vytvorený prázdny test, čiže test bez vytvorených otázok (kapitola 4.5.2). Tento test však už bude obsahovať všetky ostatné informácie, napríklad pôvodný text alebo zoznam všetkých tokenov v texte, ktorých názvoslovná entita je *osoba*. To znamená, že ak by nebolo možné vytvoriť žiadnu otázku, výstupom tvorby testu bude test v korektnom formáte a s požadovanými informáciami, avšak bez vytvorených otázok.

Keď bude spracovaný článok a pripravený test, systém prejde k vytváraniu otázok. Postupne bude posilať spracovaný článok jednotlivým typom otázok a ich výstupom bude zoznam otázok daného typu, ktoré sa podarilo vytvoriť zo zadaného článku. Pre všetky vytvorené otázky bude následne otestované, či sa na určenom mieste v texte ešte nenachádza niektorá zo skôr vytvorených otázok. V prípade, že bude otázka na voľnom mieste v rámci textu,

táto otázka bude pridaná do zoznamu otázok v teste. Test sa stane kompletným po vytvorení otázok zo všetkých definovaných typov.

4.5.1 Otázka

Pre každú vytvorenú otázku bude potrebné uchovať informácie, ktoré sú pre ňu špecifické a budú potrebné pre ďalšiu prácu s danou otázkou. Keďže pôjde o otázky, pri ktorých je cieľom doplnenie správnej možnosti na vynechané miesto v pôvodnom texte, nevyhnutnou informáciou bude pozícia, na ktorej sa táto správna odpoveď nachádzala. V nami definovanej štruktúre otázky bude táto pozícia definovaná pozíciou prvého a posledného znaku v texte. Nevyhnutnou informáciou bude správna odpoveď na otázku. Okrem správnej odpovede bude každá otázka obsahovať aj ďalšie, nesprávne možnosti, ktoré systém pre túto otázku vygeneroval. Poslednou informáciou pri otázke bude názvoslovná entita (kapitola 3.1.2), ktorá bola identifikovaná pre správnu možnosť. Názvoslovná entita bude súčasťou otázky kvôli jej možnému využitiu pri návrhu možností rovnakého typu v prípade, že sa používateľ rozhodne vygenerované možnosti upravovať. Otázka teda bude obsahovať tieto informácie:

- pozícia prvého znaku správnej odpovede v texte,
- pozícia posledného znaku správnej odpovede v texte,
- správna odpoveď,
- zoznam nesprávnych možností,
- názvoslovná entita správnej odpovede.

4.5.2 Test

Základom testu budú otázky, ktoré sa pre tento test podarilo systému vytvoriť. Okrem otázok však bude test obsahovať aj ďalšie informácie, z dôvodu použiteľnosti a budúcej práce s jeho štruktúrou. Test preto obsahuje aj pôvodnú verziu textu, pre ktorý bol daný test vytvorený. Keďže test bude možné upravovať aj po jeho vytvorení, rozhodli sme sa uľahčiť tieto úpravy už vo fáze vytvárania testu a k samotnému testu pripojiť zoznamy slov, pre ktoré boli počas ich spracovania identifikované špecifické názvoslovné entity. Kvôli relevantnosti jednotlivých slov pre aktuálny test sú tieto zoznamy rozdelené na dva typy:

- zoznam tokenov z článku,
- zoznam všetkých známych tokenov.

Každý vytvorený test teda obsahuje nasledujúce informácie:

- pôvodný text článku,
- zoznam vytvorených otázok a informácií o daných otázkach,
- zoznam tokenov z článku, ktorých názvoslovná entita je *osoba*,
- zoznam tokenov z článku, ktorých názvoslovná entita je *lokalita*,
- zoznam tokenov z článku, ktorých názvoslovná entita je *organizácia*,
- zoznam tokenov z článku, ktorých názvoslovná entita je *dátum*,

- zoznam tokenov z článku, ktorých názvoslovná entita je *číslo*,
- zoznam tokenov z článku, ktorých názvoslovná entita je *poradie*,
- zoznam tokenov z článku, ktorých názvoslovná entita je *trvanie*,
- zoznam tokenov z článku, ktorých názvoslovná entita je *peniaze*,
- zoznam tokenov z článku, ktorých názvoslovná entita je *percentá*,
- zoznam všetkých známych tokenov, ktorých názvoslovná entita je *osoba*,
- zoznam všetkých známych tokenov, ktorých názvoslovná entita je *lokalita*,
- zoznam všetkých známych tokenov, ktorých názvoslovná entita je *organizácia*.

4.6 Návrh klientskej aplikácie

Jediným komponentom systému, ktorý bude určený na interakciu s používateľom bude klientská aplikácia. Táto aplikácia bude primárne určená pre učiteľov a študentov, ktorí sa do nej budú prihlasovať. Každý používateľ bude mať možnosť vytvoriť si vlastný účet a využívať možnosti aplikácie. Používatelia nebudú rozdelení na učiteľov a študentov na úrovni našej aplikácie a všetci používatelia budú mať rovnaké práva. To znamená, že možnosť vytvorenia testu nebude prístupná len učiteľom, ale aj študentom a naopak, možnosť odoslať odpoveďový hárok bude taktiež dostupná nielen študentom, ale aj učiteľom.

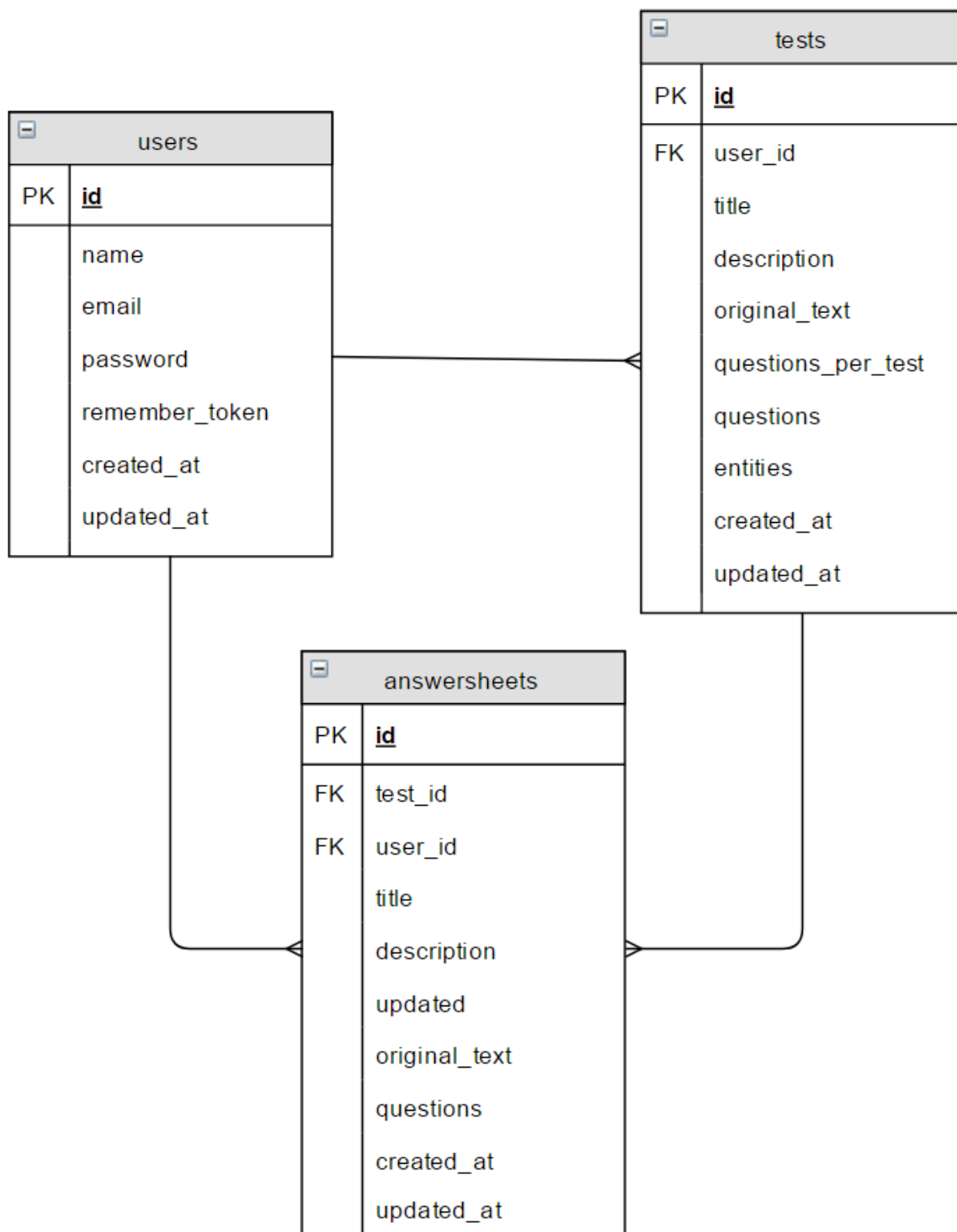
Po prihlásení do aplikácie získa používateľ prístup k informáciám, na ktorých zobrazenie má právo. Okrem zobrazenia dostupných informácií bude mať používateľ po prihlásení možnosť využívať jednotlivé funkcionality systému. Každý používateľ bude mať možnosť:

- vytvoriť nový test (kapitola 4.6.2),
- zobrazíť a upravovať svoje testy (kapitoly 4.6.3, 4.6.4, 4.6.5, 4.6.6, 4.6.7 a 4.6.8),
- vyplniť test po zadaní jeho kľúča a odoslať vyplnený odpoveďový hárok (kapitoly 4.6.9 a 4.6.10),
- zobrazíť svoje odoslané odpoveďové hárky spolu s ich vyhodnotením (kapitoly 4.6.11 a 4.6.12),
- zobrazíť odpoveďové hárky k svojim testom spolu s ich vyhodnotením (kapitoly 4.6.11 a 4.6.13).

4.6.1 Uchovanie dát v databáze

Keďže aplikácia bude pracovať s rôznymi dátami, ktoré bude potrebné vhodne ukladať, navrhli sme dátový model (obrázok 4.2), v ktorom sú dáta rozdelené do troch tabuliek:

- používatelia (*users*),
- testy (*tests*),
- odpoveďové hárky (*answersheets*).



Obr. 4.2: Dátový model pre klientskú aplikáciu.

Dátový model je navrhnutý pre relačnú databázu, avšak v niektorých častiach, ktoré sa vyznačujú väčšou variabilitou, sme sa rozhodli v určitom bode nevytvoriť novú tabuľku, ale časť dát uložiť ako celý objekt. Napríklad pre jednotlivé otázky v rámci testov nie je navrhnutá vlastná tabuľka, ale každá otázka je uložená ako objekt vo vhodnom textovom formáte.

Detailný opis dátového modelu sa nachádza v prílohe A.3.

4.6.2 Vytvorenie testu

Na začiatku životného cyklu testu je jeho vytvorenie. Pri vytváraní testu bude prebiehať komunikácia s komponentom, ktorého úlohou je tvorba testu zo zadaného textu (kapitola 4.5). Používateľ bude mať možnosť zadať jednotlivé požiadavky na vytvorenie testu. Zadané požiadavky k testu bude prebiehať cez formulár, ktorý bude ponúkať možnosť zadania názvu, popisu testu a vstupného textu, ku ktorému bude vytvorený test. Názov a popis testu nebudú mať vplyv na vytvorený test a sú určené len na identifikáciu testu medzi vytvorenými testami v klientskej aplikácii. Všetky tri položky formulára budú vyžadovať ich vyplnenie a v prípade nekorektného vyplnenia formuláru bude používateľ vyzvaný k jeho opätovnému vyplneniu.

Po schválení formulára vytvorí klientská aplikácia požiadavku na vytvorenie testu, ktorej vstupným parametrom bude požadovaný text zadaný do formulára. Následne bude vytvorený návrh testu s otázkami v definovanej štruktúre (kapitola 4.5.2).

Po úspešnom vytvorení návrhu testu bude potrebné ho spracovať a uložiť do databázy (kapitola 4.6.1). Do jednotlivých stĺpcov budú uložené nasledujúce hodnoty:

- Stĺpec *id* bude obsahovať unikátnu hodnotu v rámci tabuľky *tests*.
- Stĺpec *created_at* bude obsahovať dátum a čas v momente, kedy bol test vytvorený.
- Stĺpec *updated_at* bude obsahovať dátum a čas v momente, kedy bol test vytvorený.
- Keďže vytvorenie testu bude umožnené len prihláseným používateľom, počas vytvárania testu bude možné získať *id* používateľa, ktorý je práve prihlásený. Táto hodnota bude následne uložená do stĺpca *user_id*.
- Stĺpec *title* bude obsahovať názov testu zadaný do formulára.
- Stĺpec *description* bude obsahovať popis testu zadaný do formulára.
- Do stĺpca *original_text* nebude uložená hodnota priamo z formulára, ale hodnota vrátená v rámci návrhu testu. Toto je vykonané kvôli prípadu, kedy bude počas vytvárania návrhu testu upravený pôvodný text, napríklad odstránením nežiadúcich alebo neznámych znakov. Informácie o otázkach a ich pozíciách v texte budú v takom prípade viazané na tento upravený text a nie text zadaný do formulára.
- Do stĺpca *questions* bude prekopírovaný zoznam vytvorených otázok z návrhu testu v ich pôvodnej štruktúre (kapitola 4.5.1).
- Do stĺpca *entities* nebude priamo uložený žiadny z pôvodných zoznamov tokenov vytvorených počas návrhu testu, ale pôjde o zoznam zložený zo všetkých týchto zoznamov.
- Hodnota, ktorá bude uložená do stĺpca *questions_per_test* bude závisieť od počtu vytvorených otázok. V prípade, že počet vytvorených otázok bude vyšší ako 10, do stĺpca bude vložená hodnota 10 a v opačnom prípade bude do tohto stĺpca vložená hodnota rovná počtu vytvorených otázok.

Po vyplnení všetkých stĺpcov bude nový test uložený do databázy.

4.6.3 Zobrazenie prehľadu testov

Keďže každý používateľ bude môcť vytvoriť viacero testov, bude vhodné mu ponúknuť možnosť zobrazenia prehľadu všetkých jeho testov. V prehľade budú zobrazené všetky testy, ktorých autorom je práve prihlásený používateľ. Pri každom teste bude zobrazený jeho názov, popis a možnosť zobrazíť viac informácií o danom teste.

4.6.4 Zobrazenie detailov testu

Ku každému vytvorenému testu bude možné zobrazíť jeho detaily. Zobrazenie týchto detailov bude umožnené len autorovi testu a pri tomto zobrazení budú dostupné všetky známe informácie o konkrétnom teste. Používateľovi budú zobrazené tieto informácie:

- názov testu,
- popis testu,
- pôvodný text,
- dátum a čas vytvorenia testu,
- dátum a čas poslednej úpravy testu,
- počet prijatých odpovedových hárkov,
- počet otázok testu,
- počet otázok na jeden odpovedový hárok,
- text testu s vynechaným miestom na doplnenie pre jednotlivé otázky,
- prehľad otázok a ich možností so zvýraznením správnej odpovede.

Okrem zobrazenia informácií o teste bude možné pri zobrazení detailov testu vykonať niektorú z možností úprav daného testu. Bližší popis jednotlivých úprav sa nachádza v kapitolách 4.6.5, 4.6.7 a 4.6.8. Pôjde o nasledujúce možnosti:

- zmeniť názov testu,
- zmeniť popis testu,
- zmeniť počet otázok vybraných pre odpovedový hárok,
- zobrazíť odpovedové hárky pre test,
- vymazať test,
- vytvoriť novú otázku,
- upraviť možnosti k existujúcej otázke,
- vymazať existujúcu otázku.

4.6.5 Úpravy informácií o teste

Prvou z možných zmien v rámci základných informácií o teste je zmena jeho názvu. Túto zmenu bude možné vykonať prostredníctvom formulára, do ktorého používateľ vloží nový názov. Po odoslaní formulára sa zmení názov testu v databáze a zobrazia sa detaily testu s novým názvom. V prípade, že nový názov nebude zadaný v korektnom tvare, zmena sa neprejaví a bude zobrazený test v pôvodnom tvare.

Podobným spôsobom, akým bude možné zmeniť názov testu, bude možné zmeniť aj popis testu. Opäť pôjde o zmenu prostredníctvom formulára, do ktorého bude vložený nový popis testu. Po odoslaní formulára sa taktiež vykoná kontrola správnosti tvaru tohto popisu a v prípade, že popis nebude v korektnom tvare, zmena sa nijako neprejaví. V opačnom prípade bude zmena uložená do databázy a zobrazí sa test s upraveným popisom.

Počet otázok na jeden odpoveďový hárok môže obsahovať len číselnú hodnotu, ktorá nemôže byť menšia ako 0 a väčšia ako počet otázok v teste. Preto pre vykonanie tejto zmeny bude zobrazený formulár, ktorý umožní vybrať požadované číslo z tohto rozsahu. Po odoslaní formulára sa táto hodnota uloží do databázy a zobrazí sa test s upravenou hodnotou. Po zmene tejto hodnoty budú všetky nové odpoveďové hárky pre test obsahovať daný počet otázok.

Používateľ bude mať taktiež možnosť vytvorený test vymazať. Po vymazaní testu bude zobrazený prehľad vytvorených testov.

4.6.6 Zobrazenie odpoveďových hárkov k testu

Pri zobrazení detailov testu bude mať používateľ taktiež možnosť zobrazenia prehľadu všetkých odpoveďových hárkov, ktoré boli vytvorené k zobrazenému testu. Ku každému odpoveďovému hárku v tomto prehľade bude zobrazené meno autora hárku, emailová adresa autora hárku, dátum a čas vytvorenia hárku, dátum a čas odoslania vyplneného hárku, počet správnych odpovedí a celkový počet otázok. Okrem týchto informácií bude možné zobraziť bližšie detaily o danom odpoveďovom hárku.

4.6.7 Vytvorenie otázky

V prípade, že používateľ identifikuje v texte časť, na ktorú by chcel vytvoriť otázku, bude to môcť vykonať prostredníctvom formulára pre pridanie novej otázky. V rámci tohto formulára bude najskôr zobrazený pôvodný text testu, v ktorom používateľ vyberie cieľovú časť, na ktorú sa bude nová otázka pýtať.

Pri testoch bude platiť obmedzenie, že každý znak pôvodného textu sa bude môcť nachádzať iba v jednej otázke. Po výbere cieľovej časti pre novú otázku sa preto vykoná kontrola, či zvolená časť nie je súčasťou jednej z vytvorených otázok. V prípade, že zvolená časť bude zasahovať do inej otázky, používateľ bude upozornený a vyzvaný k výberu inej časti textu.

Okrem výberu cieľovej časti textu bude mať používateľ možnosť zvoliť, či správna odpoveď patrí do niektorej kategórie slov, definovaných názvoslovnými entitami. Používateľ teda bude mať napríklad možnosť zvoliť, či vybraná časť textu predstavuje osobu alebo organizáciu.

Po výbere požadovanej časti textu a kategórie otázky bude zobrazený formulár pre určenie zvyšných ponúkaných možností. Ak pri výbere kategórie otázky používateľ nezvolí niektorú z ponúkaných kategórií, zvyšné možnosti budú určené priamym vstupom od používateľa. V prípade, že používateľ zvolí niektorú z kategórií, okrem priameho zadania odpovede bude mať možnosť zvoliť niektoré zo známych slov danej kategórie, rozdelených medzi slová nájdené v aktuálnom texte a ostatné.

Po určení ostatných možností bude môcť používateľ novú otázku odoslať. Po odoslaní bude otázka pridaná medzi vytvorené otázky a zobrazí sa test s pridanou otázkou.

4.6.8 Úpravy existujúcej otázky

Okrem vytvorenia novej otázky bude mať používateľ možnosť upravovať existujúce otázky. Pri jednotlivých otázkach nebude možné zmeniť kategóriu otázky, pozíciu v texte ani tvar správnej odpovede. Jedinou možnou zmenou pri jednotlivých otázkach bude zmena ponúknutých možností. Túto úpravu bude možné vykonať prostredníctvom formulára, v ktorom budú zobrazené aktuálne tvary ponúkaných možností, ktoré bude môcť používateľ priamo upraviť a odoslať upravenú otázku. V prípade, že k otázke bola pri jej vytváraní priradená niektorá z kategórií, okrem priamej úpravy možností bude môcť používateľ vybrať jedno zo známych slov danej kategórie, rozdelených medzi slová nájdené v aktuálnom texte a ostatné, rovnako ako pri vytváraní novej otázky. Po odoslaní upravenej otázky bude pôvodná otázka v databáze zmenená a zobrazí sa test s upravenou otázkou.

Druhou operáciou, ktorá sa viaže na existujúce otázky a bude ponúknutá používateľovi, je vymazanie existujúcej otázky. Po vymazaní otázky sa zobrazí test, ktorý už danú otázku nebude obsahovať.

4.6.9 Vytvorenie odpoveďového hárku

Každý používateľ bude mať právo vytvoriť odpoveďový hárk ku každému testu na základe *id* daného testu. Po zadaní *id* do formulára bude vykonané overenie, či test so zadaným *id* existuje. V prípade, že nebude existovať žiadny test so zadaným *id*, používateľ bude upozornený a vyzvaný k zadaniu *id* existujúceho testu.

Ak sa podarí vyhľadať existujúci test, systém vykoná overenie, či pre daný test existuje odpoveďový hárk vytvorený práve prihláseným používateľom. V prípade, že takýto odpoveďový hárk bude nájdený, používateľovi nebude umožnené vytvorenie nového odpoveďového hárku a bude mu zobrazený hárk, ktorý vytvoril pre vyhľadávaný test v minulosti.

Po úspešnom overení podmienok, že test so zadaným *id* existuje a prihlásený používateľ v minulosti nevytvoril pre tento test žiadny odpoveďový hárk, je vytvorený nový hárk s nasledujúcimi hodnotami:

- Stĺpec *id* bude obsahovať unikátnu hodnotu v rámci tabuľky *answersheets*.
- Stĺpec *test_id* bude obsahovať *id* testu, pre ktorý bol vytvorený.
- Keďže vytvorenie odpoveďového hárku bude umožnené len prihláseným používateľom, počas vytvárania testu bude možné získať *id* používateľa, ktorý je práve prihlásený. Táto hodnota bude následne uložená do stĺpca *user_id*.

- Stĺpec *title* bude obsahovať názov testu, pre ktorý bol vytvorený.
- Stĺpec *description* bude obsahovať popis testu, pre ktorý bol vytvorený.
- Stĺpec *original_text* bude obsahovať originálny text z testu, pre ktorý bol vytvorený.
- Stĺpec *updated* bude obsahovať hodnotu false, ktorá značí, že hárok zatiaľ nebol vyplnený.
- Počet otázok v hárku bude závisieť od hodnoty atribútu *questions_per_test* testu, pre ktorý bol vytvorený. Zo zoznamu otázok v teste následne bude zvolený daný počet otázok, ktoré budú pridané do stĺpca *questions* odpovedového hárku. Tieto otázky však nebudú priamo prekopírované, ale mierne pozmenia aj svoju štruktúru. Do zoznamu ponúkaných možností bude k vygenerovaným nesprávnym možnostiam pridaná aj správna odpoveď a tento zoznam bude následne premiešaný. K otázke bude navyše pridaný atribút *chosenAnswer*, ktorý bude určený na uloženie zvolenej možnosti pri vyplňaní testu. Počas vytvárania hárku však bude hodnota tohto atribútu nastavená na *null*.
- Stĺpec *created_at* bude obsahovať dátum a čas v momente, kedy bol odpovedový hárok vytvorený.
- Stĺpec *updated_at* bude obsahovať dátum a čas v momente, kedy bol odpovedový hárok vytvorený.

Po vyplnení všetkých stĺpcov bude nový odpovedový hárok uložený do databázy a zobrazí sa formulár na vyplnenie tohto odpovedového hárku.

4.6.10 Vyplnenie a odoslanie odpovedového hárku

Po vytvorení odpovedového hárku bude mať používateľ možnosť tento odpovedový hárok vyplniť a následne odoslať. Odpovedový hárok môže vyplniť a odoslať len používateľ, ktorý je jeho autorom. Počas vyplňania hárku bude používateľovi zobrazený text s vynechaným miestom na doplnenie odpovedí pre jednotlivé otázky, ktoré boli vybrané pre daný hárok. Pre každú otázku budú zobrazené možnosti, ktoré sa nachádzajú v jej zozname možností. Používateľovou úlohou bude pre každú otázku zvoliť odpoveď, ktorá sa nachádzala na požadovanom mieste v pôvodnom texte. Po vyplnení jednotlivých otázok bude môcť používateľ odpovedový hárok odoslať na vyhodnotenie.

Po odoslaní odpovedového hárku bude najskôr vykonaná kontrola, či aktuálny odpovedový hárok ešte nebol od jeho vytvorenia nikdy upravený. V prípade, že hárok bude stále v rovnakom stave ako keď bol vytvorený, je možné ho upraviť. V opačnom prípade bude zobrazené vyhodnotenie odpovedového hárku. Ak bude možné hárok upraviť, možnosti označené počas vyplňania hárku budú uložené k jednotlivým otázkam do atribútu *chosenAnswer*.

Po uložení zvolených možností k otázkam sa zobrazia detaily a vyhodnotenie odpovedového hárku.

4.6.11 Zobrazenie detailov a vyhodnotenie odpoved'ového hárku

Každý odpoved'ový hárk, ktorý bol vyplnený a odoslaný bude možné zobraziť používateľom. Právo na zobrazenie odpoved'ového hárku bude mať okrem autora hárku aj autor testu, pre ktorý bol hárk vytvorený.

Pri zobrazení odpoved'ového hárku budú zobrazené tieto informácie:

- meno autora hárku,
- emailová adresa autora hárku,
- dátum a čas vytvorenia hárku,
- dátum a čas odoslania vyplneného hárku,
- počet správnych odpovedí,
- celkový počet otázok,
- text testu s vynechaným miestom na doplnenie pre jednotlivé otázky,
- prehľad otázok s vyznačenou zvolenou možnosťou a správnu odpoveďou.

4.6.12 Zobrazenie prehľadu odoslaných odpoved'ových hárkov

Vzhľadom k tomu, že používateľ bude môcť odoslať odpoved'ový hárk k viacerým testom, bude mu ponúknutá možnosť zobrazenia prehľadu odpoved'ových hárkov, ktoré odoslal. V tomto prehľade bude pre každý hárk zobrazený dátum a čas vytvorenia hárku, dátum a čas odoslania vyplneného hárku, počet správnych odpovedí a celkový počet otázok. Okrem týchto informácií bude možné zobraziť bližšie detaily o danom odpoved'ovom hárku.

4.6.13 Zobrazenie prehľadu prijatých odpoved'ových hárkov

Používateľ bude môcť taktiež zobraziť prehľad odpoved'ových hárkov odoslaných k testom, ktoré vytvoril. Pre jednotlivé hárky budú zobrazené rovnaké informácie ako pri prehľade odoslaných hárkov, doplnené o meno a emailovú adresu autora hárku.

4.7 Zhrnutie

V nami navrhnutom systéme budú môcť používatelia vykonávať rôzne úlohy prostredníctvom klientskej aplikácie. Navrhnutá klientská aplikácia bude ponúkať možnosť vytvorenia testu a následnej správy vytvorených testov.

Po odoslaní požiadavky na vytvorenie testu bude vytvorený test z otázok, ktoré sme navrhli. Jednotlivé otázky budú využívať informácie, ktoré budú získané spracovaním zadaného textu. Na základe týchto informácií o texte budú typy otázok aplikovať definované pravidlá a v prípade splnenia podmienok bude na danom mieste vytvorená otázka. Z vytvorených otázok bude zložený test, ktorý bude uložený a zobrazený používateľom.

Kapitola 5

System na tvorbu testov

V tejto časti sa venujeme opisu implementácie nami navrhnutého systému na tvorbu testov (kapitola 4). Na základe navrhutej architektúry systému (kapitola 4.1) sme sa rozhodli celý systém vyvíjať ako dve samostatné aplikácie:

- systém na tvorbu testov (kapitola 5),
- klientská aplikácia (kapitola 6).

System na tvorbu testov zabezpečuje spracovanie vstupného textu a následné vytvorenie testu v navrhutej štruktúre, pričom táto funkcionálna je sprístupnená prostredníctvom definovaného rozhrania. Na interakciu s používateľom je určená klientská aplikácia, ktorá je opísaná v ďalšej časti. Pri opise implementácie systému na tvorbu testov sa venujeme použitým technológiám (kapitola 5.1), spôsobu spracovania vstupného textu (kapitola 5.2) a následného uchovania získaných informácií (kapitola 5.3). Následne opisujeme akým spôsobom sú vytvárané testy (kapitola 5.4) a akým spôsobom je táto funkcionálna sprístupnená klientskej aplikácii (kapitola 5.5).

5.1 Použité technológie

Pred samotnou implementáciou aplikácie sme sa museli rozhodnúť, ktoré technológie spĺňajú naše požiadavky a sú najvhodnejšie pre využitie v našej aplikácii.

Ako nástroj na spracovanie prirodzeného jazyka sme zvolili *Stanford CoreNLP* (Manning et al., 2014). K tomuto výberu prispelo najmä množstvo informácií, ktoré dokáže poskytnúť o zadanom texte. Medzi týmito informáciami sa nachádzajú všetky informácie, ktoré sme pri návrhu identifikovali ako kľúčové pre nami definované typy otázok. Nástroj dokáže spracovať vstup v šiestich rôznych jazykoch, medzi ktorými nechýba ani angličtina, teda jazyk ktorému sa v našej práci venujeme. Medzi výhody patrí aj podpora pre veľké množstvo programovacích jazykov.

Pre samotnú aplikáciu sme zvolili programovací jazyk *Java*. Tento jazyk sme zvolili kvôli jeho robustnosti a možnostiam, ktoré so sebou prináša. Okrem týchto prirodzených výhod jazyka k nášmu výberu prispelo aj to, že pre prácu s nástrojom *Stanford CoreNLP* je najviac odporúčaný práve programovací jazyk *Java*. Keďže sme funkcionálnu aplikáciu potrebovali sprístupniť pre klientskú aplikáciu, zvolili sme formu webovej aplikácie a implementovanie API (rozhranie pre programovanie aplikácií, angl. Application Programming Interface). V

programovacím jazyku *Java* je pre tieto účely vhodné využiť technológiu *Java Servlet*, ktorá zabezpečuje spracovanie HTTP požiadaviek. Pre aplikáciu sme museli zvoliť vhodný aplikačný server, na ktorom bude spustená a tak bude poskytovať definované API. Pre tieto účely sme zvolili *Apache Tomcat*, ktorý spĺňa všetky naše požiadavky.

Na ukladanie nájdených tokenov pre jednotlivé názvoslovné entity sme zvolili databázový systém *SQLite*, ktorý spĺňa všetky naše požiadavky a nie je zbytočne zložitý a komplikovaný.

Viac informácií je možné nájsť v prílohe A.1.

5.2 Spracovanie vstupného textu

Na spracovanie vstupného textu sme použili nástroj *Stanford CoreNLP*. Tento nástroj poskytuje viacero možností použitia, pričom tými najzákladnejšími sú:

- priama integrácia tried do aplikácie,
- spustenie vo forme servera.

Obidve možnosti poskytujú rovnaký výstup, avšak pri priamej integrácii do aplikácie sa prejavilo, že pri každej žiadosti na spracovanie textu musí nástroj opätovne načítať pomerne veľké množstvo externých knižníc, čo spôsobilo dlhý čas tohto spracovania. Z tohto dôvodu sme sa rozhodli spustiť nástroj vo forme servera, ktorý tieto knižnice načíta len pri prvej požiadavke a pri každej ďalšej požiadavke pracuje s načítanými knižnicami, čo zabezpečuje vyššiu rýchlosť spracovania.

So spusteným serverom *Stanford CoreNLP* je možné komunikovať prostredníctvom HTTP požiadaviek. Pre zjednodušenie komunikácie so serverom sa medzi dodávanými knižnicami nachádza trieda *StanfordCoreNLPClient*, ktorá predstavuje rozhranie medzi prostredím jazyka *Java* a nástrojom *Stanford CoreNLP*. Po vhodnom nakonfigurovaní je vytvorený klient, ktorý zabezpečuje komunikáciu so serverom a všetky získané informácie o texte mapuje na atribúty triedy *Annotation*, z ktorej je možné tieto informácie získať a spracovať do navrhnutej štruktúry (kapitola 4.2.5).

5.3 Uchovanie informácií o texte

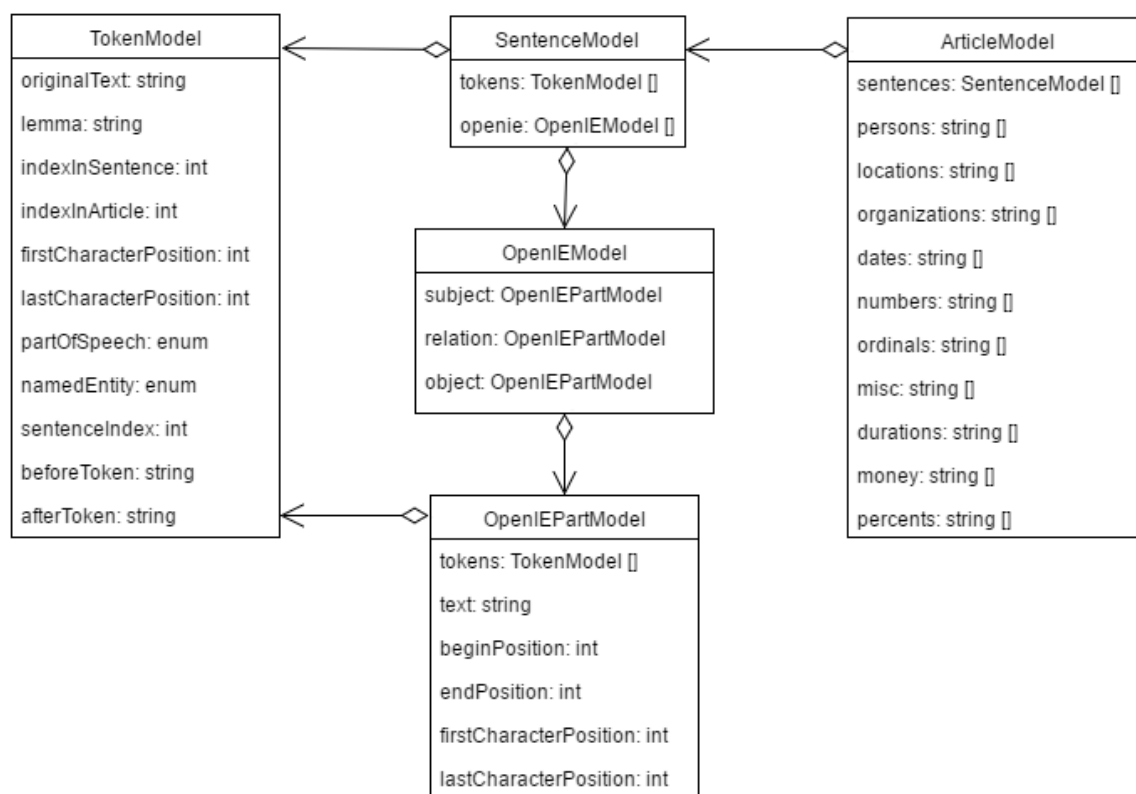
Po spracovaní textu a vytvorení príslušného objektu triedy *Annotation* je tento objekt spracovaný v triede *ModelBinder*, ktorá zabezpečuje namapovanie informácií z anotácie na jednotlivé triedy modelu, ktorých štruktúra sa zhoduje so štruktúrou definovanou v návrhu (kapitoly 4.2.1, 4.2.2, 4.2.3 a 4.2.4). Ide o triedy:

- *TokenModel* (model pre token),
- *OpenIEPartModel* (model pre subjekt, objekt alebo vzťah z OpenIE trojice),
- *OpenIEModel* (model pre OpenIE trojicu),
- *SentenceModel* (model pre vetu),
- *ArticleModel* (model pre článok).

Informácie z objektu triedy *Annotation* sú získavané vďaka jednotlivým anotáciám, z ktorých každá predstavuje získanie inej informácie. Napríklad názvoslovnú entitu z tokenu získavame prostredníctvom anotácie *CoreAnnotations.NamedEntityTagAnnotation*. Podobným spôsobom je možné získať všetky potrebné informácie, vďaka ďalším dostupným anotáciám.

Zoznamy tokenov jednotlivých názvoslovných entít sú inicializované pri vytvorení modelu článku ako jeho atribúty. Do týchto zoznamov sú pridávané tokeny v prípade, že pri mapovaní údajov z anotácie na model tokenu bolo zistené, že daný token má priradenú niektorú z podporovaných názvoslovných entít.

Atribúty jednotlivých tried modelu sú implementované tak, aby dodržovali štruktúru opísanú v návrhu. To znamená, že pre vstupný text je po jeho spracovaní vytvorený jeden model článku (*ArticleModel*), z ktorého sa dá prístup k jednotlivým modelom viet, z ktorých sa dá prístup k modelom tokenov a podobne. Takýmto spôsobom sme vytvorili navrhnutú štruktúru (kapitola 4.2.5), ktorú je možné získať priamo ako objekt triedy *ArticleModel* alebo vo formáte *JSON*.

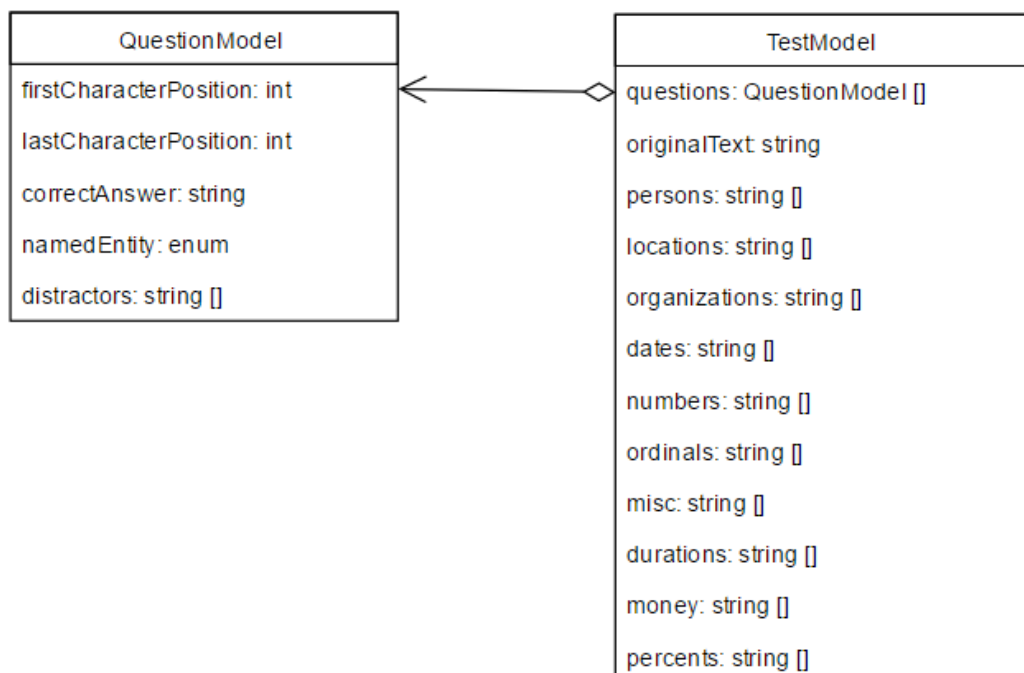


Obr. 5.1: Diagram tried modelu pre uchovanie informácií o texte.

5.4 Tvorba testu

Po kompletnom vytvorení modelu článku sme mohli prejsť k tvorbe testu. Pri tvorbe testu sme opäť vytvorili modelové triedy podľa štruktúry z návrhu. Týmto triedami sú:

- **QuestionModel** (model pre otázku),
- **TestModel** (model pre test).



Obr. 5.2: Diagram tried modelu pre návrh testu.

Na začiatku tvorby testu je vytvorený model testu s prázdny zoznamom otázok. Pred vytváraním otázok je potrebné vstupný text spracovať a preto sme z neho najskôr vytvorili model článku. Po vytvorení modelu článku sme na základe jeho zoznamov tokenov naplnili tieto zoznamy aj v rámci modelu testu. Model testu obsahuje okrem zoznamov tokenov nájdených v texte aj zoznamy všetkých známych tokenov pre niektoré názvoslovné entity. Na ukladanie a opätovné načítanie týchto tokenov sme použili databázový systém *SQLite*, ktorý napriek svojej jednoduchosti spĺňal všetky naše požiadavky. Okrem zoznamov tokenov sme pred vytvorením otázok uložili do testu aj pôvodný text.

Po naplnení testu statickými hodnotami prechádzame k tvorbe otázok. Každá otázka je implementovaná ako samostatná trieda a každá z týchto tried obsahuje metódu, ktorej vstupným parametrom je model článku (*ArticleModel*) a výstupným parametrom je zoznam vytvorených otázok (*QuestionModel*) daného typu. Pri tvorbe testu sú postupne volané jednotlivé typy otázok, ktoré ako výstup poskytujú zoznam otázok. Z každého zoznamu otázok sú do testu pridané tie otázky, ktorých pozícia v texte neprichádza do konfliktu s niektorou otázkou nachádzajúcou sa v teste.

Všetky implementácie jednotlivých typov otázok majú podobnú štruktúru. Vstupným parametrom pre typ otázky je model článku, na ktorého časti sa pokúša typ otázky uplatniť svoje pravidlá. V prípade, že sa podarí aplikovať pravidlá na určitú časť článku, vytvorí sa nový model otázky, ktorého hodnoty atribútov sú nastavené v závislosti od časti článku, na ktorú sa otázka vzťahuje. Pri generovaní nesprávnych možností je najskôr vytvorený zoznam prijateľných možností, ktorý nemá obmedzenú veľkosť a každý typ otázky môže mať vlastný algoritmus, ktorým tieto možnosti generuje. Následne sú z tohto zoznamu náhodne vybrané tri možnosti, ktoré sú uložené do modelu otázky. To znamená, že pre rovnaký vstup môže byť pri opakovanom vytváraní testu vytvorený vždy iný test. Po vytvorení otázok všetkých typov je test kompletný a je možné k nemu pristupovať ako k objektu triedy *TestModel* alebo ho získať vo formáte *JSON*.

5.5 API

Aby sme sprístupnili funkcionálnosť vytvoreného systému pre klientskú aplikáciu, zvolili sme využitie technológie *Java Servlet*. Ide o technológiu, ktorá umožňuje spracovanie HTTP požiadaviek a poskytnutie odpovedí na tieto požiadavky. Základným spôsobom využitia opisovanej aplikácie je vytváranie testov, pre ktoré sme vytvorili *Servlet* na adrese */gettest* (kapitola 5.5.1). Okrem možnosti vytvorenia testu sme sprístupnili aj možnosť získať model článku, teda informácií o texte, s akými pracujeme pri vytváraní testu. Táto možnosť je sprístupnená na adrese */gettags* (kapitola 5.5.2).

5.5.1 Požiadavka na vytvorenie testu

Webová služba, ktorá spracováva požiadavky na vytvorenie testu je služba dostupná na adrese */gettest*. Ide o metódu typu GET a vstupným parametrom požiadavky na vytvorenie testu je text, na základe ktorého má byť test vytvorený. Názov tohto parametru je *inputText*. Odpoveďou na túto požiadavku je vytvorený test, ktorý je odoslaný vo formáte *JSON*. Požiadavku je možné vykonať bez akejkoľvek autorizácie a k funkcionalite aplikácie má prístup ktokoľvek.

5.5.2 Požiadavka na spracovanie článku

Webová služba, ktorá poskytuje vytvorenie modelu článku je veľmi podobná požiadavke na vytvorenie testu. Ide o metódu typu GET na adrese */gettags* a vstupným parametrom je text, ktorý má byť spracovaný. Odpoveďou na požiadavku je model článku, ktorý je odoslaný vo formáte *JSON*. Túto funkcionálnosť sme sprístupnili za účelom podpory vývojárov, ktorý by mali záujem vytvoriť ďalšie nástroje na tvorbu otázok, prípadne testov. Požiadavku je taktiež možné vykonať bez autorizácie.

5.6 Zhrnutie

Systém na tvorbu testov poskytuje verejne prístupné služby na spracovanie textu (kapitola 5.5.2) a vytvorenie testu (kapitola 5.5.1). Pri spracovaní textu sa do veľkej miery spoliehame na prácu s nástrojom *Stanford CoreNLP* a informácie získané vďaka tomuto nástroju mapujeme na jednotlivé modelové triedy, ktorých štruktúra vytvára model článku s rôznymi informáciami. Okrem spracovania vstupného textu a poskytnutia získaných informácií systém ponúka aj možnosť vytvorenia testu. Pri tvorbe testu pracujeme s vytvoreným modelom článku, pre ktorý sa pokúšame vytvoriť otázky pre jednotlivé typy otázok.

Kapitola 6

Klientská aplikácia

Okrem vytvorenia systému na tvorbu testov sme sa venovali aj vytvoreniu klientskej aplikácie, ktorej používateľom poskytujeme rôzne možnosti spojené s vytváraním, správou a vyplňaním testov. Táto aplikácia využíva funkcionality systému na tvorbu testov (kapitola 5).

6.1 Použité technológie

Klientskú aplikáciu sme sa rozhodli implementovať ako webovú aplikáciu, založenú na PHP frameworku *Laravel*. Tento framework nám zjednodušuje prácu s dátami a spracováva jednotlivé žiadosti na vytvorenie *HTML* stránok pre prehliadače.

Na ukladanie dát sme zvolili databázový systém *PostgreSQL*, ktorý spĺňa všetky naše požiadavky.

Pri generovaní jednotlivých *HTML* stránok využívame technológie *JavaScript* a *jQuery*. Pri vytváraní dizajnu stránky sme použili CSS framework *Bootstrap*, vďaka ktorému má stránka responzívny dizajn.

Viac informácií je možné nájsť v prílohe A.1.

6.2 Spracovanie dát a ich zobrazenie v prehliadači

Jednotlivé stránky sú vytvárané na základe prijatých HTTP požiadaviek, ktoré sú spracované prostredníctvom príslušných metód. Aplikácia najskôr identifikuje o aký typ požiadavky ide a na základe jej typu a adresy je zavolaná príslušná metóda, ktorá pripraví požadované dáta. Na prácu s dátami využívame špecifické možnosti frameworku *Laravel*. Pre úspešné zobrazenie akejkoľvek poskytovanej podstránky je potrebné byť v aplikácii prihlásený. Prehľad požiadaviek je dostupný v prílohe A.5.

Po prijatí HTTP požiadavky a spracovaní potrebných dát je potrebné tieto dáta zobraziť používateľovi. Pre každú podstránku sme pripravili vzor v podobe *HTML* stránky, ktorá však neobsahuje žiadne dáta. Dáta, ktoré majú byť zobrazené na tejto *HTML* stránke sú posunuté do prehliadača ako premenná, ktorá je po načítaní stránky v prehliadači dostupná pre skripty napísané v jazyku *JavaScript*. Po načítaní vzorovej *HTML* stránky je spustená *JavaScript* funkcia, ktorá z týchto dát vytvorí príslušné elementy a zapracuje ich do vzorovej stránky.

V prípadoch, kedy je potrebné z prehliadača odoslať dáta na server, sú tieto dáta odosielané prostredníctvom formulárov požadovaného typu na požadované adresy. V niektorých prípadoch je formulár priamo súčasťou HTML stránky a v niektorých prípadoch je tento formulár vytváraný dynamicky prostredníctvom funkcie v jazyku *JavaScript*.

6.3 Vytvorenie testu

Pri vytváraní nového testu je odoslaná požiadavka na webovú službu systému na tvorbu testov (kapitola 5.5.1). Po prijatí odpovede od systému je potrebné túto odpoveď spracovať a nový test uložiť do databázy.

Vytvorenie testu začína prijatím požiadavky typu *POST* na adresu */tests* a zavolaní príslušnej metódy. Táto požiadavka je vytvorená a odoslaná prostredníctvom formulára, ktorého vstupmi sú názov testu, popis testu a vstupný text. Tieto parametre sú preto dostupné aj v rámci metódy, ktorá spracováva vytvorenie testu. Nadviazanie komunikácie so systémom na tvorbu testov je zabezpečené odoslaním HTTP požiadavky na príslušnú adresu webovej služby (kapitola 5.5.1). Vstupným parametrom tejto požiadavky je text, ktorý zadal používateľ do formulára.

Po prijatí odpovede od servera je táto odpoveď spracovaná a vytvorený test obsahuje otázky, ktoré obsahoval test vytvorený systémom na tvorbu testov. Okrem otázok sú z odpovede využité aj hodnoty originálneho textu a zoznamy tokenov. Vytvorený test je uložený do databázy a zobrazený používateľovi.

6.4 Úpravy vytvoreného testu

Všetky podporované úpravy vytvoreného testu sú vykonávané prostredníctvom *JavaScriptu* v prehliadači na stránke, kde sú zobrazené detaily testu. Medzi premennými je dostupný objekt, ktorý reprezentuje zobrazený test. Po vykonaní požadovanej úpravy je tento test upravený a následne je dynamicky vytvorený formulár, do ktorého atribútov je vložený upravený test a formulár sa automaticky odošle ako HTTP požiadavka typu PUT. Táto požiadavka je následne spracovaná príslušnou metódou, ktorá na základe odoslaného testu upraví príslušný test v databáze a zobrazí detaily upraveného testu.

Pri vytváraní novej otázky alebo úprave existujúcej otázky má používateľ možnosť napísať nesprávne odpovede priamo do príslušných vstupov. Pri otázkach, ktoré majú definovanú niektorú z podporovaných názvoslovných entít existuje okrem toho možnosť zvoliť niektorú zo známych možností, na základe názvoslovnej entity otázky. Pri vytváraní stránky sú okrem samotného testu prístupné aj zoznamy tokenov nájdených v texte, rovnako ako zoznamy všetkých známych tokenov. Pri načítaní formuláru na úpravu možností sú tieto tokeny načítané do príslušnej časti formuláru.

6.5 Odoslanie odpovedového hárku

Každý používateľ má možnosť odoslať jeden odpovedový hárk na každý vytvorený test, po zadaní *id* požadovaného testu. Po odoslaní formuláru je vykonaná kontrola, či existuje test so

zadaným *id*. V prípade, že takýto test neexistuje, používateľ je presmerovaný späť na formulár pre zadanie *id* testu. V opačnom prípade je vykonaná kontrola, či existuje odpoveďový hárok od prihláseného používateľa k požadovanému testu. Ak existuje takýto odpoveďový hárok, používateľovi je zobrazené vyhodnotenie tohto hárku. Ak bol test nájdený a prihlásený používateľ ešte nevytvoril žiadny hárok k tomuto testu, vytvorí sa nový odpoveďový hárok.

Jednotlivé atribúty odpoveďového hárku sú vyplnené na základe príslušných atribútov testu. Do zoznamu otázok sa pridajú náhodné otázky zo zoznamu otázok v teste. Počet vybraných otázok závisí od hodnoty atribútu *questions_per_test* aktuálneho testu. Medzi atribúty vybraných otázok bude navyše pridaný atribút *chosenAnswer*, ktorého hodnota sa nastaví na *null*. Vytvorený hárok sa uloží do databázy a používateľovi sa zobrazí formulár na vyplnenie tohto hárku.

Pred zobrazením formuláru na vyplnenie hárku sa však ešte vykoná kontrola, či hodnota atribútu *updated* je nastavená na *false*. Ak táto podmienka nie je splnená, zobrazí sa používateľovi vyhodnotenie daného odpoveďového hárku. Táto kontrola je implementovaná kvôli obmedzeniu viacnásobného vyplňovania hárkov. Ak hárok ešte nebol vyplnený, zobrazí sa vzorová HTML stránka pre vyplňanie hárkov, ku ktorej sa sprístupní objekt reprezentujúci odpoveďový hárok, ktorý sa má zobraziť. Po načítaní vzorovej stránky je spustená *JavaScript* funkcia, ktorá vytvorí formulár s otázkami. Po označení odpovedí je možné tento formulár odoslať ako HTTP požiadavku typu PUT.

Pred uložením označených odpovedí je sa opäť vykoná kontrola, či je hodnota atribútu *updated* nastavená na *false*. Táto kontrola znemožní viacnásobné odosielanie hárkov a ich uloženie. Ak hárok ešte nebol od jeho otvorenia upravený, znamená to, že ide o prvé odoslanie odpovedí pre tento hárok. Pri prijatí hárku sa však neukladá žiadne vyhodnotenie testu, ale ide len o uloženie označených možností do atribútov *chosenAnswer* pre jednotlivé otázky. Po uložení týchto odpovedí je zmenená aj hodnota atribútu *updated* na *true*, vďaka čomu je zabezpečené, že aktuálne spracovanie hárku bude zároveň jediné. Po uložení upraveného hárku je zobrazené jeho vyhodnotenie.

6.6 Vyhodnotenie a zobrazenie odpoveďového hárku

Vyhodnotenie otázok sa zobrazí len pre tie hárky, pri ktorých je atribút *updated* nastavený na *true*. Pre hárky, ktoré síce boli vytvorené, ale ešte neboli odoslané, sa síce stránka zobrazí, ale správne odpovede nie sú vyznačené. V prípade, že hárok bol odoslaný, pri jednotlivých otázkach je v atribúte *chosenAnswer* uložená označená odpoveď a v atribúte *correctAnswer* správna odpoveď.

Po prijatí požiadavky na zobrazenie detailov hárku je vytvorená vzorová HTML stránka a sprístupnená premenná reprezentujúca požadovaný hárok. Po načítaní stránky sú prostredníctvom *JavaScriptu* spracované základné informácie o hárku, ktoré sú zobrazené v postrannom paneli. Pri spracovaní otázok a odpovedí je vykonávané porovnanie hodnôt atribútov *chosenAnswer* a *correctAnswer*. Ak sú tieto hodnoty rovnaké, znamená to, že používateľ označil správnu odpoveď a táto odpoveď je pri vypisovaní možností vyznačená zelenou farbou. V opačnom prípade je možnosť označená ako *chosenAnswer* označená červenou farbou a možnosť označená ako *correctAnswer* zelenou farbou. V prípade, že atribút *chosenAnswer* má hodnotu *null*, znamená to, že na túto otázku nebola odoslaná odpoveď a v takomto prípade nie je vyznačená žiadna odpoveď.

6.7 Zhrnutie

Klientská aplikácia je implementovaná ako webová aplikácia, primárne určená pre učiteľov a študentov. Aplikácia ponúka rôzne možnosti spojené s tvorbou testov a ich následnou správou. Vytváranie testov prebieha prostredníctvom nášho systému (kapitola 5). Po vytvorení testu je možné upravovať informácie o teste, rovnako ako obsah otázok.

Okrem vytvárania testov ponúka aplikácia možnosť vyplnenia testu a následného automatického vyhodnotenia odoslaného odpovedového hárku. Otázky pre odpovedové hárky sú náhodne vyberané z otázok príslušného testu, vďaka čomu je zabezpečené automatické vytváranie rôznych testov pre rôznych študentov.

Kapitola 7

Evaluácia

Pri evaluácii systému sme sa zamerali na dve základné oblasti:

- množstvo vytvorených otázok,
- ohodnotenie vytvorených otázok.

Pri prvej časti (kapitola 7.2) sme sa zamerali na počet viet zo vstupnej množiny, na ktoré bola vytvorená aspoň jedna otázka. Pri druhej časti (kapitola 7.3) bolo vykonané ohodnotenie vytvorených otázok používateľmi, ktorých úlohou bolo zaradiť otázky do jednotlivých kategórií.

7.1 Vstupná množina viet

Pri evaluácii systému sme použili množinu viet¹ použitú pri evaluácii systémov na tvorbu otázok v rámci konferencie QGSTEC v roku 2010 (Rus et al., 2012). Vety boli vybrané z troch zdrojov:

- OpenLearn,
- Wikipedia,
- Yahoo Answers.

Pri výbere jednotlivých viet bol kladený dôraz na to, aby na tieto vety bolo možné vytvoriť otázky. K jednotlivým vetám boli následne ľudskými silami vytvorené otázky rôznych typov. K množine viet bolo vytvorených celkom 180 otázok, avšak dosiahnutie tohto čísla je veľmi náročné práve preto, že nejde o otázky vytvorené žiadnym systémom, ale ľuďmi. Na každú vetu z množiny bola vytvorená aspoň jedna otázka.

Pri našom experimente sme tieto vety žiadnym spôsobom neupravovali a použili sme všetky vety z množiny. Tieto vety sme vložili do nášho systému, ktorý sa na ne pokúsil vytvoriť otázky. Výstupy pre jednotlivé vety sme uložili ako samostatné testy v rámci klientskej aplikácie.

¹<https://github.com/bjwyse/QGSTEC2010/blob/master/TaskB/DevData/QuestionsFromSentences.Development.xml>

7.2 Pokrytie viet

Prvým experimentom, na ktorý sme sa zamerali, bolo pokrytie vstupnej množiny viet naším systémom. Pri tomto experimente sme vychádzali z predpokladu, že na každú vetu bolo možné vytvoriť aspoň jednu otázku. Po spracovaní otázok vytvorených naším systémom sme vypočítali percentuálne množstvo viet, na ktoré náš systém vytvoril aspoň jednu otázku.

Pokrytie viet sme vypočítali ako pomer počtu viet, na ktoré bola vytvorená otázka a celkového počtu viet. Celkový počet viet v množine bol 81 a nášmu systému sa podarilo vytvoriť aspoň jednu otázku na 48 viet z tejto množiny. Percentuálne pokrytie množiny, po zaokrúhlení na 5 desatinných miest, bolo 59,25926 %.

7.3 Ohodnotenie vytvorených otázok

Na ohodnotenie kvality otázok sa podieľalo päť používateľov systému, ktorým boli zobrazené otázky vytvorené systémom. Zo vstupnej množiny viet (kapitola 7.1) systém vytvoril spolu 95 otázok. Úlohou používateľov bolo zaradiť tieto otázky do jednej z nasledujúcich kategórií:

- Otázka aj ponúknuté možnosti sú korektné.
- Otázka je korektná, ale 1 z ponúknutých možností nie je korektná.
- Otázka je korektná, ale 2 z ponúknutých možností nie sú korektné.
- Otázka je korektná, ale 3 z ponúknutých možností nie sú korektné.
- Otázka nie je korektná.

Okrem ohodnotenia otázok sme sa zamerali aj na kategorizovanie otázok podľa ich typu. Používatelia mali za úlohu zaradiť otázky podľa ich typu do týchto kategórií:

- Otázka zameraná na osobu.
- Otázka zameraná na dátum.
- Otázka zameraná na organizáciu.
- Otázka zameraná na miesto.
- Otázka nepatrí do žiadnej z uvedených kategórií.

Po získaní dát od používateľov sme tieto dáta spracovali do tabuľky 7.1.

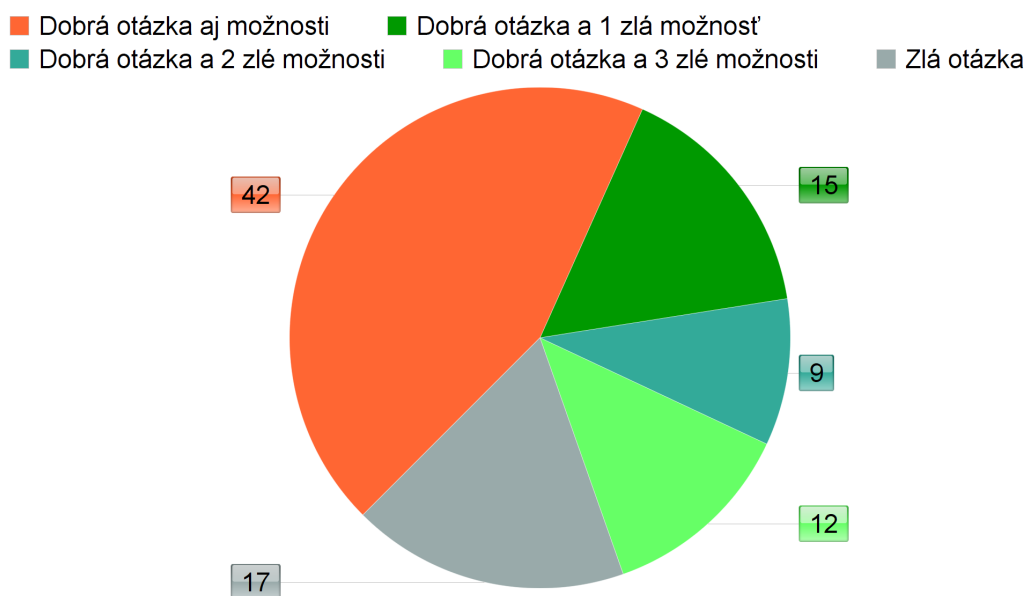
Tabuľka 7.1: Ohodnotenie otázok používateľmi.

	Osoba	Dátum	Organizácia	Miesto	Iné	Spolu
Dobrá otázka aj možnosti	1	33	1	7	0	42
Dobrá otázka a 1 zlá možnosť	8	0	1	6	0	15
Dobrá otázka a 2 zlé možnosti	4	0	1	2	2	9
Dobrá otázka a 3 zlé možnosti	1	0	2	3	6	12
Zlá otázka	0	0	0	0	14	17
Spolu	14	33	5	21	22	95

Získané údaje sme následne spracovali do grafov. Na obrázku 7.1 je možné vidieť rozdelenie otázok do kategórií na základe ich typov. Na obrázku 7.2 je znázornený celkový prehľad ohodnotenia kvality otázok a na obrázkoch 7.3, 7.4, 7.5, 7.6 a 7.7 je znázornené ohodnotenie kvality otázok rozdelených podľa ich typov. Ukážky otázok z jednotlivých kategórií sa nachádzajú v prílohe D.



Obr. 7.1: Rozdelenie otázok do kategórií podľa typu otázky.



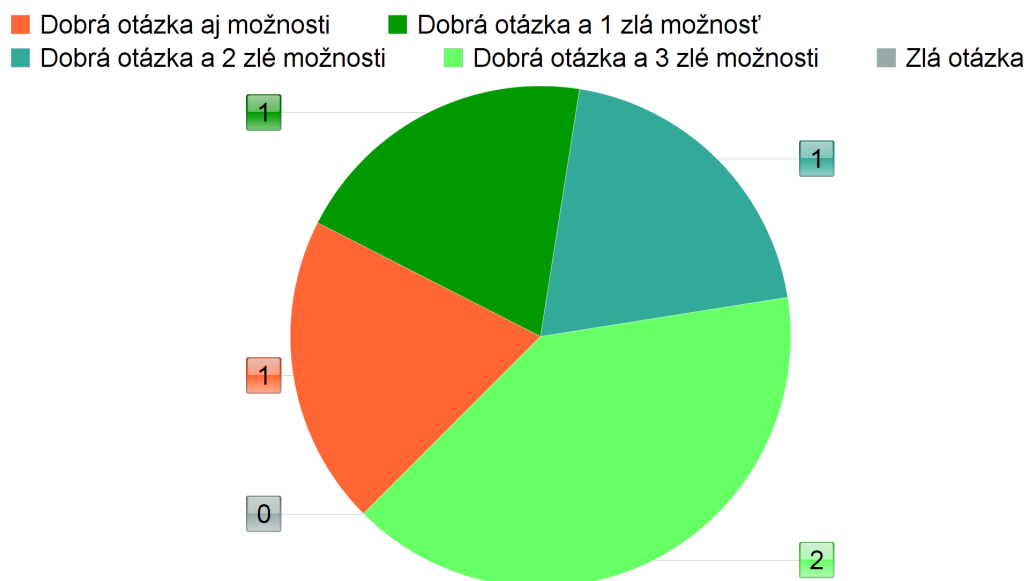
Obr. 7.2: Ohodnotenie kvality vytvorených otázok.



Obr. 7.3: Ohodnotenie kvality vytvorených otázok zameraných na osobu.



Obr. 7.4: Ohodnotenie kvality vytvorených otázok zameraných na dátum.



Obr. 7.5: Ohodnotenie kvality vytvorených otázok zameraných na organizáciu.



Obr. 7.6: Ohodnotenie kvality vytvorených otázok zameraných na miesto.



Obr. 7.7: Ohodnotenie kvality zvyšných otázok.

7.4 Zhrnutie

Evaluáciou systému sme získali viaceré výstupy. V prvej časti evaluácie sme zistili, že sa nám podarilo vytvoriť aspoň jednu otázku na 59,25926 % viet zo vstupnej množiny, pričom vety vo vstupnej množine boli zvolené tak, aby sa na každú z nich dala vytvoriť otázka. K jednotlivým vetám zo vstupnej množiny bolo ľudskými silami navrhnutých celkom 180 otázok. Nášmu systému sa podarilo celkom vytvoriť 95 otázok, avšak porovnanie týchto hodnôt nie je ideálne, pretože pri automatickej tvorbe otázok je veľmi náročné dosiahnuť úroveň a počet otázok, ktoré manuálne vytvorili ľudia.

V druhej časti experimentu bolo úlohou používateľov zaradiť otázky do jednej z piatich kategórií na základe typu otázky a následne ohodnotiť jej kvalitu. Veľká časť otázok označených ako korektných, patrila do kategórie otázok zameraných na dátum. Toto vyplýva z faktu, že dátumy je možné jednoznačnejšie identifikovať a naformátovať do jednotného formátu. Preto je možné generovať možnosti k otázkam tohto typu presnejšie a kvalitnejšie. Pri otázkach zameraných na osobu, organizáciu alebo miesto boli otázky rozdelené podľa kvality do všetkých piatich kategórií. Pri otázkach, ktoré neboli zaradené do žiadnej z predchádzajúcich štyroch kategórií boli výsledky menej kvalitné, keď veľká časť týchto otázok bola označená ako nekorektná.

Kapitola 8

Zhodnotenie

V práci sme sa venovali systému na tvorbu otázok z učebného textu. Po analýze oblasti spracovania prirodzeného jazyka sme vytvorili návrh systému, ktorý je zložený z dvoch samostatných aplikácií. Prvá časť systému je zameraná na spracovanie textu a následné vytvorenie testu. V tejto časti sme navrhli niekoľko typov otázok, ktoré sú aplikované na text v prípade splnenia pravidiel definovaných pre jednotlivé typy. Tieto pravidlá využívajú informácie o texte, ktorý je spracovaný pred samotným vytváraním otázok. Pri spracovaní textu sme sa primárne zamerali na informácie o jednotlivých tokenoch, napríklad názvoslovné entity alebo slovné druhy. Všetky typy otázok majú rovnakú formu, pri ktorej je cieľová časť z pôvodného textu vynechaná a úlohou študenta je na toto miesto doplniť správnu odpoveď. Pre všetky otázky sa systém pokúša vygenerovať aj ďalšie možnosti, ktoré budú študentovi ponúknuté spolu so správnou odpoveďou. Spôsob generovania možností je definovaný samostatne pre každý typ otázky. Niektoré otázky využívajú pri generovaní možností databázu tokenov kategorizovaných podľa ich názvoslovných entít a niektoré otázky generujú tieto možnosti na základe definovaného algoritmu pre konkrétny typ otázky. Pri spracovaní textu sme využili nástroj Stanford CoreNLP a aplikáciu sme implementovali v jazyku Java. Podporovaným jazykom je angličtina. Funkcionalitu aplikácie na tvorbu testov sme sprístupnili prostredníctvom API.

Druhou časťou systému je klientská aplikácia, implementovaná vo forme webovej aplikácie. Pri implementácii sme využili PHP framework Laravel, v kombinácii s technológiami HTML, CSS, JavaScript a jQuery. Aplikácia je určená primárne pre učiteľov a študentov. Pri návrhu aplikácie sme identifikovali a navrhli jednotlivé prípady použitia, ktoré sme následne implementovali. Aplikácia umožňuje vytváranie testov vďaka využitiu nášho poskytovaného API. Vďaka tomuto procesu a vytvorenému návrhu testu budú mať učitelia menšie starosti s identifikáciou kľúčových informácií v texte pri tvorbe testových otázok. Navrhnutý test môže učiteľ následne upravovať a tým ho dostať do cieľovej podoby. Vytvorené testy sú ukladané do databázového systému PostgreSQL. Okrem vytvárania a správy testov aplikácia ponúka používateľom možnosť vyplniť niektorý z vytvorených testov po zadaní id daného testu. Vďaka tejto funkcionalite môžu študenti vyplňať testy priamo v aplikácii a ich odpovedové hárky sa automaticky vyhodnotia. Okrem automatického vyhodnotenia odpovedových hárkov je v aplikácii zabezpečený náhodný výber otázok pri vytváraní nových hárkov, vďaka čomu je pre každého študenta vytvorený jeho vlastný test.

Systém sme testovali prostredníctvom dvoch testov. Pri prvom teste sme sa zamerali na to, koľko viet zo vstupnej množiny dokáže náš systém pokryť aspoň jednou otázkou. Pri tomto

teste sme zistili, že náš systém vytvoril otázku na 59,25926 % viet. Pri druhom teste sme pri otázkach skúmali ich kvalitu a typ. Na tomto teste sa zúčastnilo päť používateľov, ktorí nám poskytli požadované údaje k otázkam. Z celkového počtu 95 otázok bolo 42 označených ako korektných a 17 otázok ako nekorektných. Zvyšné otázky boli označené ako korektné s chybou medzi ponúknutými možnosťami. Spomedzi vytvorených otázok bolo najviac otázok zaradených do kategórie otázok so zameraním na dátum.

Vzhľadom k tomu, že náš systém nie je zameraný priamo na spracovanie prirodzeného jazyka, ale až na tvorbu otázok na základe informácií o texte, pri možných vylepšeniach systému sme sa zamerali na možnosti vylepšenia kvality vytvorených otázok. Pri ďalšom vývoji systému je možné vytvárať nové typy otázok, ktoré budú mať definované svoje pravidlá. Okrem vytvárania nových otázok je priestor na zlepšenie aj pri generovaní odpovedí pre implementované typy otázok. Najmä pri otázkach zameraných na miesto by mohla byť ku každému nájdenému miestu pridaná informácia o tom, o aký typ miesta ide. Pri generovaní možností by boli následne pre jednotlivé typy miest vyberané len miesta zhodného typu. Týmto zlepšením by sa odstránili chyby, pri ktorých sa pri otázke zameranej na mesto nachádzala medzi možnosťami rieka a podobne.

Najviac chýb, ktoré sa pri tvorbe otázok vyskytujú, vyplýva z nesprávneho spracovania textu nástrojom Stanford CoreNLP. V našom systéme nie sme schopní overovať správnosť dostupných informácií o texte a preto v prípade nesprávneho spracovania textu môžu byť vytvorené aj nesprávne otázky.

Systém bol odprezentovaný na výskumnej konferencii IIT.SRC v kategórii inovatívnych aplikácií. Publikovaný článok sa nachádza v prílohe F.

Literatúra

- AGARWAL, M. – MANNEM, P. Automatic gap-fill question generation from text books. In *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 56–64. Association for Computational Linguistics, 2011.
- AGARWAL, M. – SHAH, R. – MANNEM, P. Automatic question generation using discourse cues. In *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 1–9. Association for Computational Linguistics, 2011.
- ALI, H. – CHALI, Y. – HASAN, S. A. Automation of question generation from sentences. In *Proceedings of QG2010: The Third Workshop on Question Generation*, pages 58–67, 2010.
- ALI, H. D. A. D. *Automatic question generation: a syntactical approach to the sentence-to-question generation case*. PhD thesis, Lethbridge, Alta.: University of Lethbridge, Dept. of Mathematics and Computer Science, c2012, 2012.
- BEINBORN, L. – ZESCH, T. – GUREVYCH, I. Candidate Evaluation Strategies for Improved Difficulty Prediction of Language Tests. In *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 1–11, 2015.
- BOČÁK, M. Diskurz: neurčitá cesta kulturných, mediálnych a komunikačných štúdií do centra svojho záujmu. *Kultura–médiá–komunikace*. 2009, 1, pages 117–146.
- CHOWDHURY, G. G. Natural language processing. *Annual review of information science and technology*. 2003, 37, 1, pages 51–89.
- COLLOBERT, R. et al. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*. 2011, 12, Aug, pages 2493–2537.
- FOREHAND, M. Bloom's taxonomy. *Emerging perspectives on learning, teaching, and technology*. 2010, pages 41–47.
- GRAESSER, A. – RUS, V. – CAI, Z. Question classification schemes. In *Proc. of the Workshop on Question Generation*, 2008.
- HEILMAN, M. *Automatic factual question generation from text*. PhD thesis, Carnegie Mellon University, 2011.
- HUSSEIN, H. – ELMOGY, M. – GUIRGUIS, S. Automatic English Question Generation System Based on Template Driven Scheme. *International Journal of Computer Science Issues (IJCSI)*. 2014, 11, 6, pages 45.

- LIN, Y.-C. – SUNG, L.-C. – CHEN, M. C. An automatic multiple-choice question generation scheme for English adjective understanding. In *Workshop on Modeling, Management and Generation of Problems/Questions in eLearning, the 15th International Conference on Computers in Education (ICCE 2007)*, pages 137–142, 2007.
- MANNING, C. D. et al. The stanford corenlp natural language processing toolkit. In *ACL (System Demonstrations)*, pages 55–60, 2014.
- PLISSON, J. et al. A rule based approach to word lemmatization. In *Proceedings C of the 7th International Multi-Conference Information Society IS 2004*, volume 1, pages 83–86. Citeseer, 2004.
- RUS, V. – ARTHUR – GRAESSER, C. The Question Generation Shared Task and Evaluation Challenge. In *The University of Memphis. National Science Foundation*, 2009.
- RUS, V. et al. A detailed account of the first question generation shared task evaluation challenge. *Dialogue and Discourse*. 2012, 3, 2, pages 177–204.

Dodatok A

Technická príručka

A.1 Použité technológie

Pri implementácii sme použili rôzne technológie. V tejto časti sa nachádzajú detaily k jednotlivým technológiám. Okrem spomenutých technológií sme použili knižnice pre Javu a Laravel. Všetky potrebné knižnice sa nachádzajú na elektronickom médiu ako súčasť klient-skej aplikácie alebo aplikačného serveru Apache Tomcat.

A.1.1 Java Development Kit 8

Verzia: jdk1.8.0_111

Odkaz: <http://www.oracle.com/technetwork/java/javase/8u111-relnotes-3124969.html>

Informácie: V tomto prostredí bola vyvíjaná Java aplikácia a spúšaný Stanford CoreNLP Server.

A.1.2 PostgreSQL

Verzia: 9.4

Odkaz: <https://www.postgresql.org/download/>

Informácie: Datázový systém bol použitý pri klientskej aplikácii.

A.1.3 Stanford CoreNLP

Verzia: 3.7.0

Odkaz: <https://stanfordnlp.github.io/CoreNLP/history.html>

Návod na spustenie vo forme servera: <https://stanfordnlp.github.io/CoreNLP/corenlp-server.html>

A.1.4 Laravel

Verzia: 5.4

Odkaz: <https://laravel.com/docs/5.4/installation>

Informácie: Klientská aplikácia je postavená na frameworku Laravel vo verzií 5.4. Tento framework je možné nainštalovať rôznymi spôsobmi. Naša aplikácia spolu s Laravelom sa nachádza na elektronickom médiu.

A.1.5 SQLite

Verzia: 3

Odkaz: <https://www.sqlite.org/download.html>

Informácie: Databázový systém SQLite 3 bol použitý pri systéme na tvorbu testov. Súbor, do ktorého sú ukladané dáta sa nachádza na elektronickom médiu v rámci aplikačného serveru Apache Tomcat.

A.1.6 Bootstrap

Verzia: 3.3.7

Odkaz: <http://blog.getbootstrap.com/2016/07/25/bootstrap-3-3-7-released/>

Informácie: Bootstrap v3.3.7 bol použitý pri klientskej aplikácii. Jednotlivé súbory sa nachádzajú v klientskej aplikácii na elektronickom médiu.

A.1.7 jQuery

Verzia: 3.1.1

Odkaz: <https://blog.jquery.com/2016/09/22/jquery-3-1-1-released/>

Informácie: jQuery 3.1.1 bol použitý pri klientskej aplikácii a nachádza sa v klientskej aplikácii na elektronickom médiu.

A.1.8 Apache Tomcat

Verzia: 8.0.39

Odkaz: <https://tomcat.apache.org/download-80.cgi>

Informácie: Aplikačný server Apache Tomcat 8.0.39 bol použitý na spúšťanie Java aplikácie na tvorbu testov. Aplikačný server sa nachádza na elektronickom médiu. Vo verzií nachádzajúcej sa na elektronickom médiu sa nachádzajú všetky potrebné knižnice na korektné fungovanie aplikácie.

A.1.9 PHP

Verzia: 7.1.2

Odkaz: http://php.net/releases/7_1_2.php

Informácie: PHP vo verzií 7.1.2 je potrebné na správne fungovanie klientskej aplikácie.

A.1.10 Composer

Odkaz: <https://getcomposer.org/>

Informácie: Composer je potrebný pre spustenie klientskej aplikácie nachádzajúcej sa na elektronickom médiu.

A.2 Inštalácia a spustenie aplikácie

Systém bol vyvíjaný a testovaný na operačnom systéme *Windows 10*. Počítač mal *8GB RAM* a procesor *Intel Core i7-6500U*.

Pri vývoji sme pracovali v prostrediach *IntelliJ Idea 2016.3.1* a *JetBrains PhpStorm 2017.1.2*.

Pre úspešné spustenie je potrebné nainštalovať všetky potrebné technológie vo verziách popísaných v predchádzajúcej časti. Ide o technológie:

- Java Development Kit 8 (aj s vytvorením premennej prostredia),
- PostgreSQL 9.4 (spustený na adrese *localhost* a na porte *5432*) s vytvorenou databázou *testgen_client* a používateľom *postgres* s heslom *heslo123*,
- Stanford CoreNLP 3.7.0 (spustený vo forme servera podľa príslušného návodu na porte *9000*),
- PHP 7.1.2,
- Composer.

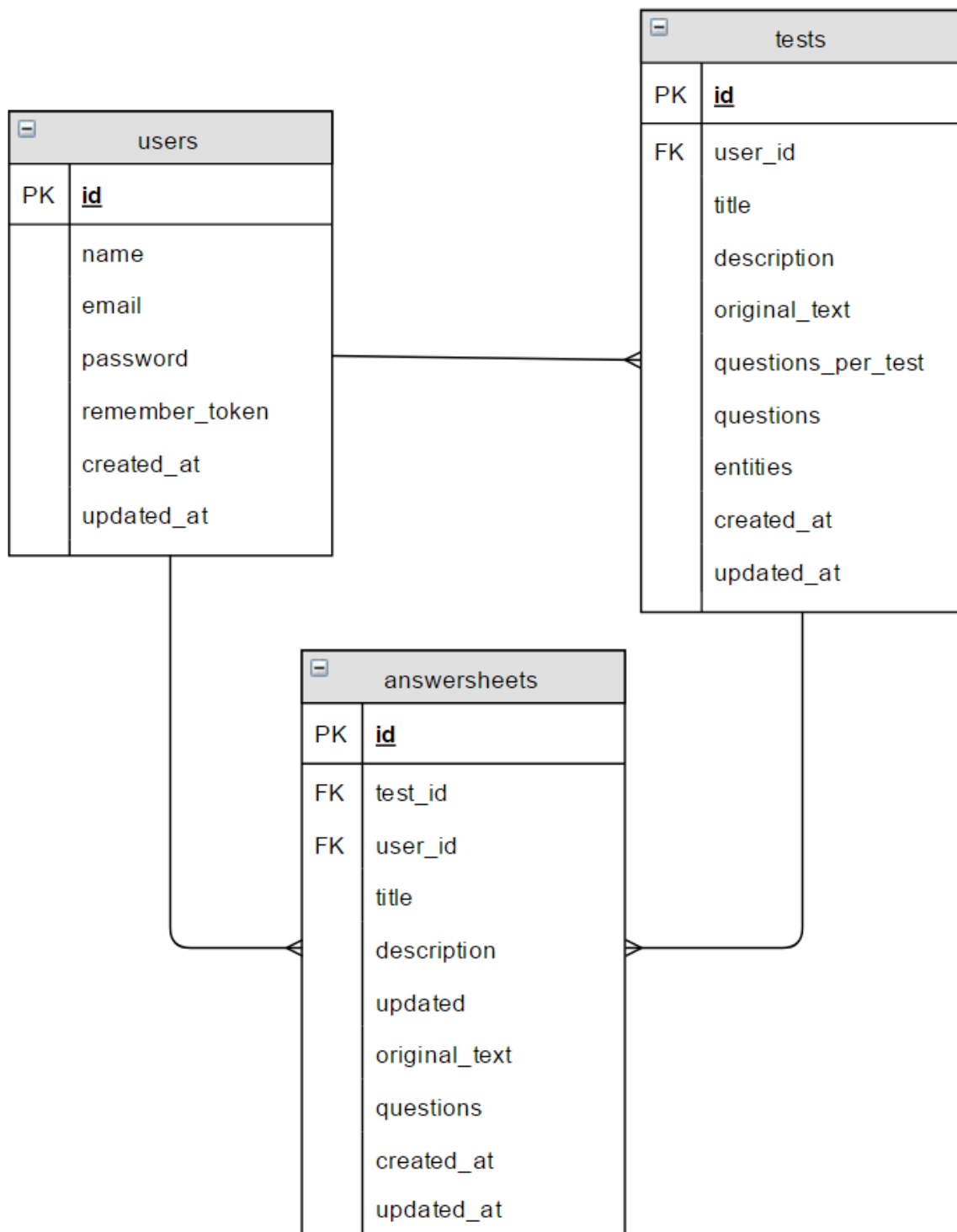
Po nainštalovaní a spustení všetkých potrebných nástrojov je možné spustiť aplikačný server *Apache Tomcat*. Aplikačný server je potrebné spustiť z elektronického média, nakoľko dodávaná verzia obsahuje všetky potrebné knižnice pre správne fungovanie aplikácie. Aplikačný server spustíte prostredníctvom súboru */system_na_tvorbu_testov/apache-tomcat-8.0.39/bin/startup.bat*. V rámci aplikačného servera sa spustí naša aplikácia, ktorá sa nachádza v priečinku */system_na_tvorbu_testov/apache-tomcat-8.0.39/webapps/ROOT*. Aplikačný server musí byť spustený na adrese *localhost* a na porte *8080*.

Po spustení aplikačného servera spustíte klientskú aplikáciu. Klientská aplikácia má nastavené údaje pre prístup k databáze v súbore */klientska_aplikacia/testgen_client/.env*. Tabuľky v databáze vytvoríte spustením Powershell príkazu *php artisan migrate* v priečinku */klientska_aplikacia/testgen_client*. Klientskú aplikáciu spustíte prostredníctvom Powershell príkazu *php artisan serve* spusteného v priečinku */klientska_aplikacia/testgen_client*. Po spustení klientskej aplikácie je možné k nej prísť na adrese *http://localhost:8000/*.

Pre úspešné spustenie systému je potrebné mať všetky nástroje spustené v správnej verzii a na správnom porte.

A.3 Dátový model klientskej aplikácie

Na obrázku A.1 sa nachádza dátový model klientskej aplikácie. V tejto časti sa venujeme popisu jednotlivých atribútov tabuliek.



Obr. A.1: Dátový model pre klientskú aplikáciu.

Atribúty tabuľky používateľov (angl. users):

- id - unikátny identifikátor používateľa,
- name - meno používateľa,
- email - emailová adresa používateľa (využíva sa pri prihlasovaní do aplikácie),
- password - heslo používateľa (využíva sa pri prihlasovaní do aplikácie),
- remember_token - token slúžiaci pri zapamätaní prihlásenia používateľa,
- created_at - dátum a čas vytvorenia záznamu,
- updated_at - dátum a čas poslednej úpravy záznamu.

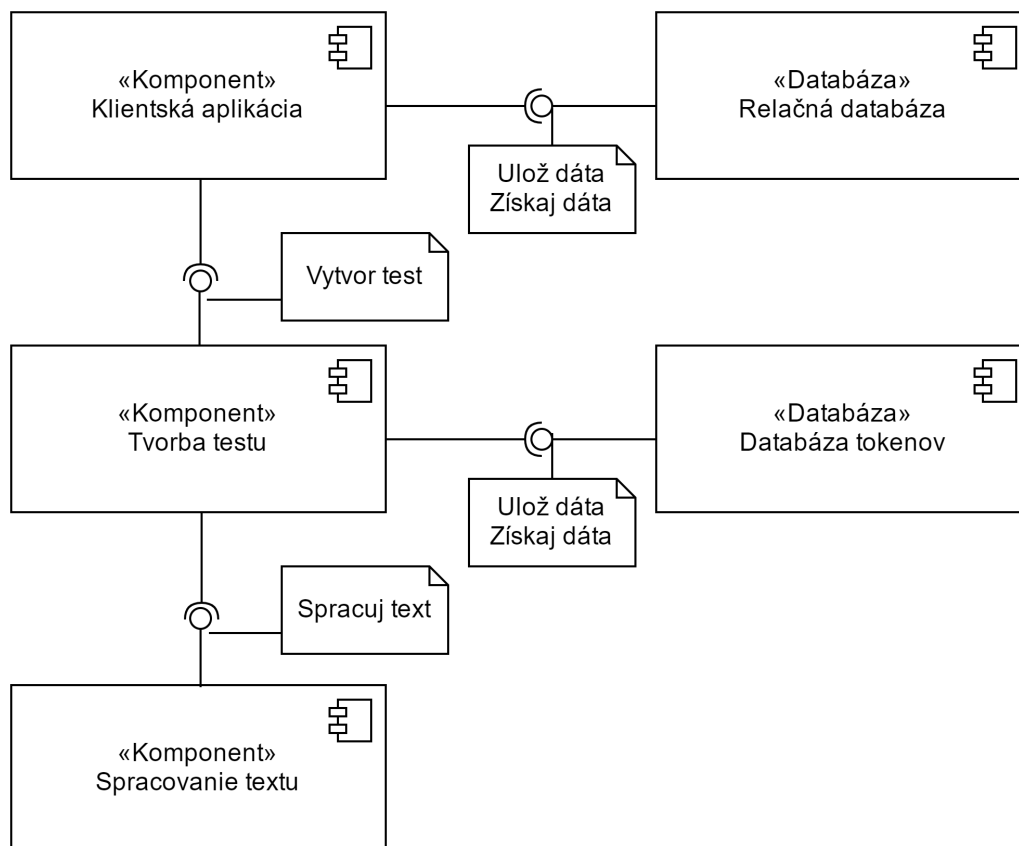
Atribúty tabuľky testov (angl. tests):

- id - unikátny identifikátor testu
- user_id - id autora testu v tabuľke používateľov,
- title - názov testu,
- description - popis testu,
- original_text - pôvodný text testu,
- questions_per_test - počet otázok určených pre jeden odpoveďový hárok,
- questions - zoznam otázok testu,
- entities - zoznam tokenov pre jednotlivé názvoslovné entity,
- created_at - dátum a čas vytvorenia záznamu,
- updated_at - dátum a čas poslednej úpravy záznamu.

Atribúty tabuľky odpoveďových hárkov (angl. answer sheets):

- id - unikátny identifikátor odpoveďového hárku
- test_id - id testu v tabuľke testov,
- user_id - id autora odpoveďového hárku v tabuľke používateľov,
- title - názov odpoveďového hárku (názov testu v čase vytvorenia hárku),
- description - popis odpoveďového hárku (popis testu v čase vytvorenia hárku),
- updated - informácia, či bol odpoveďový hárok upravený,
- original_text - pôvodná podoba textu pre odpoveďový hárok (text testu v čase vytvorenia hárku),
- questions - zoznam otázok vybraných pre odpoveďový hárok,
- created_at - dátum a čas vytvorenia záznamu,
- updated_at - dátum a čas poslednej úpravy záznamu.

A.4 Diagram komponentov



Obr. A.2: Diagram komponentov.

A.5 Prehľad HTTP požiadaviek pre klientskú aplikáciu

Klientská aplikácia dokáže spracovať určitú množinu požiadaviek. Jednotlivé požiadavky sú na základe ich adresy a typu spracované priradenými metódami. Prehľad požiadaviek, ich typov a priradených metód sa nachádza v tabuľke A.1.

Tabuľka A.1: Požiadavky a k nim priradené metódy.

Adresa	Typ	Priradená trieda	Priradená metóda
/	GET	HomeController	index
/tests	GET	TestController	index
/tests/create	GET	TestController	create
/tests	POST	TestController	store
/tests/{test}	GET	TestController	show
/tests/{test}/edit	GET	TestController	edit
/tests/{test}	PUT	TestController	update
/tests/{test}	DELETE	TestController	destroy
/answersheets	GET	AnswersheetController	index
/answersheets/create	GET	AnswersheetController	create
/answersheets	POST	AnswersheetController	store
/answersheets/{answersheet}	GET	AnswersheetController	show
/answersheets/{answersheet}/edit	GET	AnswersheetController	edit
/answersheets/{answersheet}	PUT	AnswersheetController	update
/answersheets/{answersheet}	DELETE	AnswersheetController	destroy

Dodatok B

Obsah elektronického média

Na elektronickom médiu sa nachádza aplikačný server Apache Tomcat s potrebnými knižnicami a skompilovanou verziou Java aplikácie na tvorbu testov. Okrem skompilovanej verzie Java aplikácie sa na médiu nachádzajú aj zdrojové kódy tejto aplikácie. Na médiu sa nachádza aj klientská aplikácia postavená na frameworku Laravel. Okrem aplikácií sa na médiu nachádza aj táto práca vo formáte PDF.

Obsah elektronického média:

- */dokument/pdf/* - dokument s anotáciou vo formáte PDF
- */klientska_aplikacia/testgen_client/* - kompletná klientská aplikácia spolu s frameworkom Laravel
- */system_na_tvorbu_testov/zdrojovy_kod/* - zdrojový kód Java aplikácie na tvorbu testov
- */system_na_tvorbu_testov/apache-tomcat-8.0.39/* - aplikačný server s potrebnými knižnicami a skompilovanou verziou Java aplikácie na tvorbu testov
- */system_na_tvorbu_testov/apache-tomcat-8.0.39/webapps/ROOT/* - skompilovaná verzia Java aplikácie na tvorbu testov
- */readme.txt* - popis obsahu média

Dodatok C

Používateľská príručka

V tejto časti sa nachádza používateľská príručka ku klientskej aplikácii. Neprihlásený používateľ má len možnosti registrácie a prihlásenia. Všetky ostatné možnosti sú prístupné až po prihlásení do aplikácie. Navigácia sa nachádza na hornom okraji obrazovky a pri krokoch návodu slovom navigácia máme na mysli túto navigáciu na hornom okraji obrazovky.

C.1 Registrácia a prihlásenie

Registračný formulár (obrázok C.1) sa zobrazí po kliknutí na tlačidlo *Register* v navigácii. Používateľ následne môže formulár vyplniť požadovanými údajmi a odoslať kliknutím na tlačidlo *Register* vo formulári. Pri formulári je vyžadované vyplnenie všetkých vstupných polí, emailová adresa musí byť korektná a heslo musí byť zložené z minimálne šiestich znakov. Po úspešnom zaregistrovaní používateľa je tento používateľ automaticky prihlásený do aplikácie.

Formulár pre prihlásenie (obrázok C.2) sa zobrazí po kliknutí na tlačidlo *Login* v navigácii. Po zadaní emailovej adresy a hesla môže byť formulár odoslaným kliknutím na tlačidlo *Login* vo formulári.

C.2 Vytvorenie testu

Formulár pre vytvorenie testu je možné zobrazíť kliknutím na tlačidlo *My Tests* v navigácii a následnom kliknutí na tlačidlo *Create new test* v hornej časti obrazovky s prehľadom vytvorených testov (obrázok C.3).

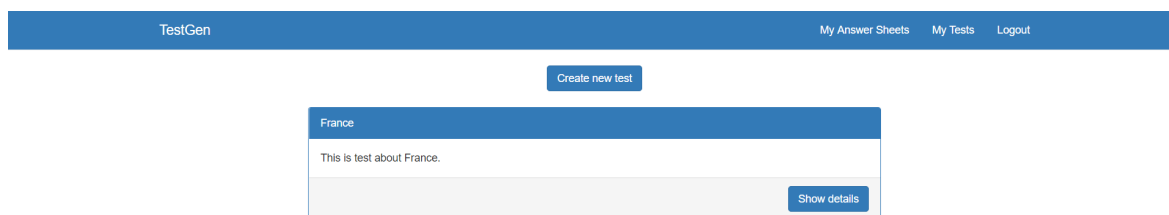
Do zobrazeného formulára (obrázok C.4) je možné zadať názov testu (Title), popis testu (Description) a text testu (Text). Test sa vytvorí po kliknutí na tlačidlo *Create* vo formulári. Po vytvorení testu sa zobrazí obrazovka s detailami testu (obrázok C.5).

The screenshot shows the 'TestGen' application header with 'TestGen' on the left and 'Login Register' on the right. Below the header is a 'Register' form. The form has a title bar 'Register' and contains the following fields: 'Name' with placeholder 'Your Name', 'E-Mail Address' with placeholder 'your@email.com', 'Password' with placeholder '*****', and 'Confirm Password' with placeholder '*****'. A blue 'Register' button is at the bottom right of the form.

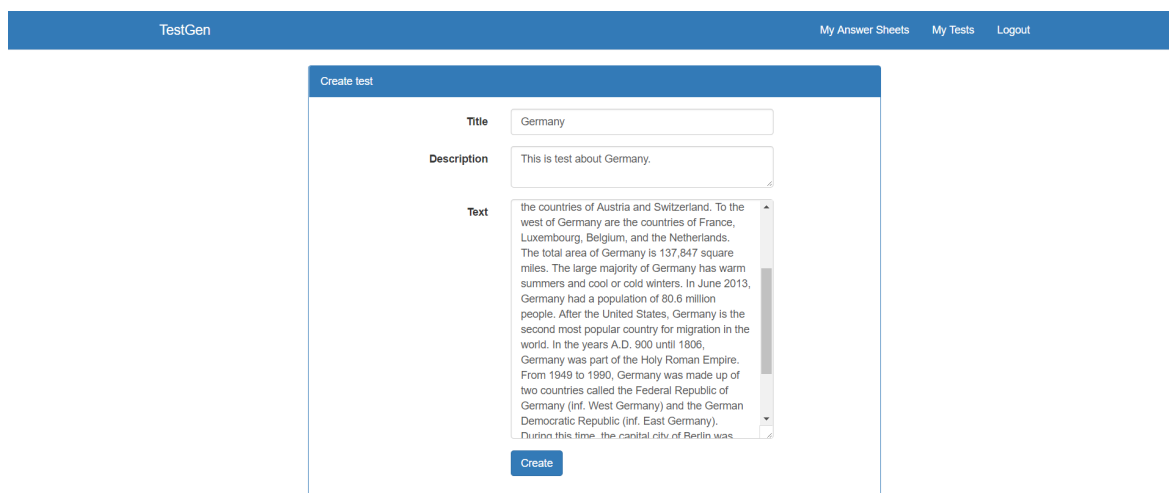
Obr. C.1: Obrázovka pre registráciu.

The screenshot shows the 'TestGen' application header with 'TestGen' on the left and 'Login Register' on the right. Below the header is a 'Login' form. The form has a title bar 'Login' and contains the following fields: 'E-Mail Address' with placeholder 'your@email.com' and 'Password' with placeholder '*****'. Below the password field is a checkbox labeled 'Remember Me'. A blue 'Login' button is at the bottom right of the form.

Obr. C.2: Obrázovka pre prihlásenie.



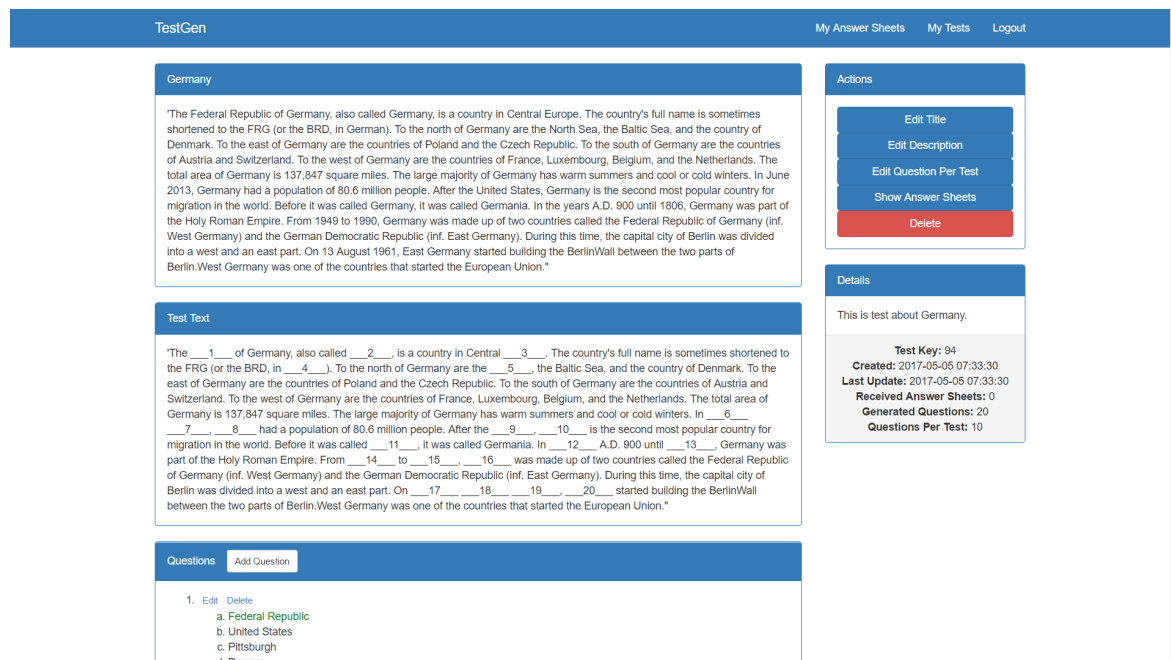
Obr. C.3: Obrazovka s prehľadom vytvorených testov.



Obr. C.4: Obrazovka pre vytvorenie testu

C.3 Správa testov

Všetky možnosti správy testu je možné vykonať z obrazovky s detailami testu (obrázok C.5). Pre túto časť predpokladáme, že je práve zobrazená táto obrazovka.



Obr. C.5: Obrazovka s detailami testu.

C.3.1 Vytvorenie otázky

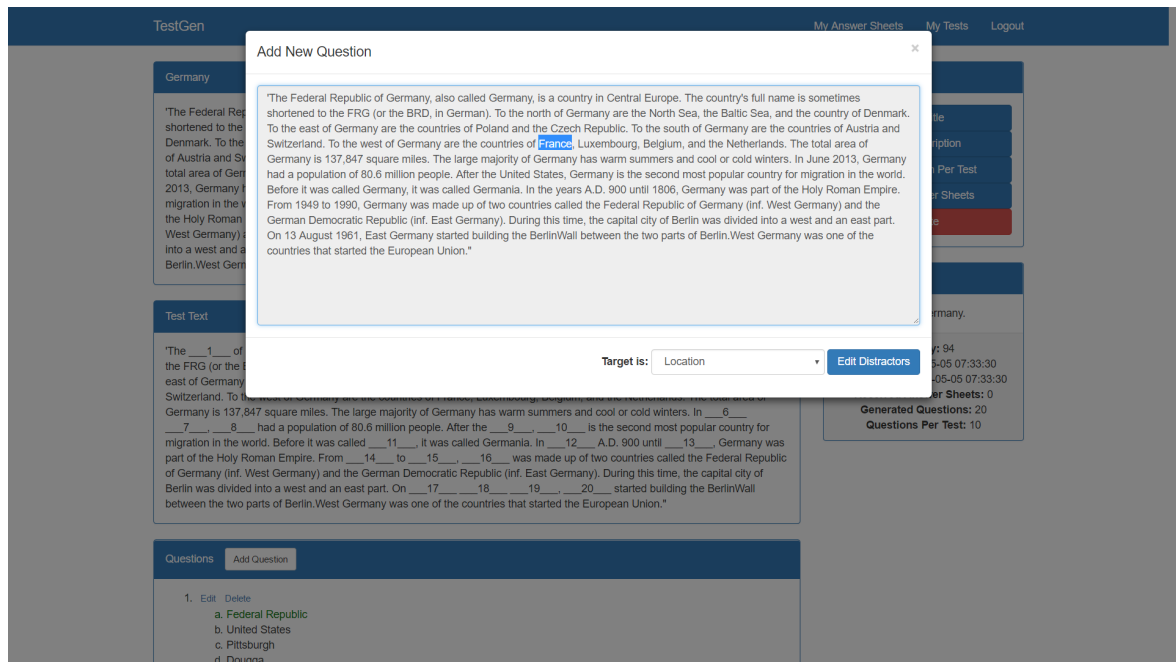
Otázku je možné vytvoriť kliknutím na tlačidlo *Add Question* v hlavičke panelu s názvom *Questions*. Po kliknutí na tlačidlo sa zobrazí okno s textom testu (obrázok C.6). V tomto okne používateľ môže vybrať časť textu, na ktorú má byť vytvorená otázka a zvoliť, či táto časť patrí do niektorej z podporovaných kategórií.

Po kliknutí na tlačidlo *Edit Distractors* sa zobrazí formulár pre zvolenie ďalších ponúknutých možností pri otázke (obrázok C.7). Po kliknutí na tlačidlo *Save Question* sa táto otázka vytvorí a opäť sa zobrazí obrazovka s detailami testu (obrázok C.5).

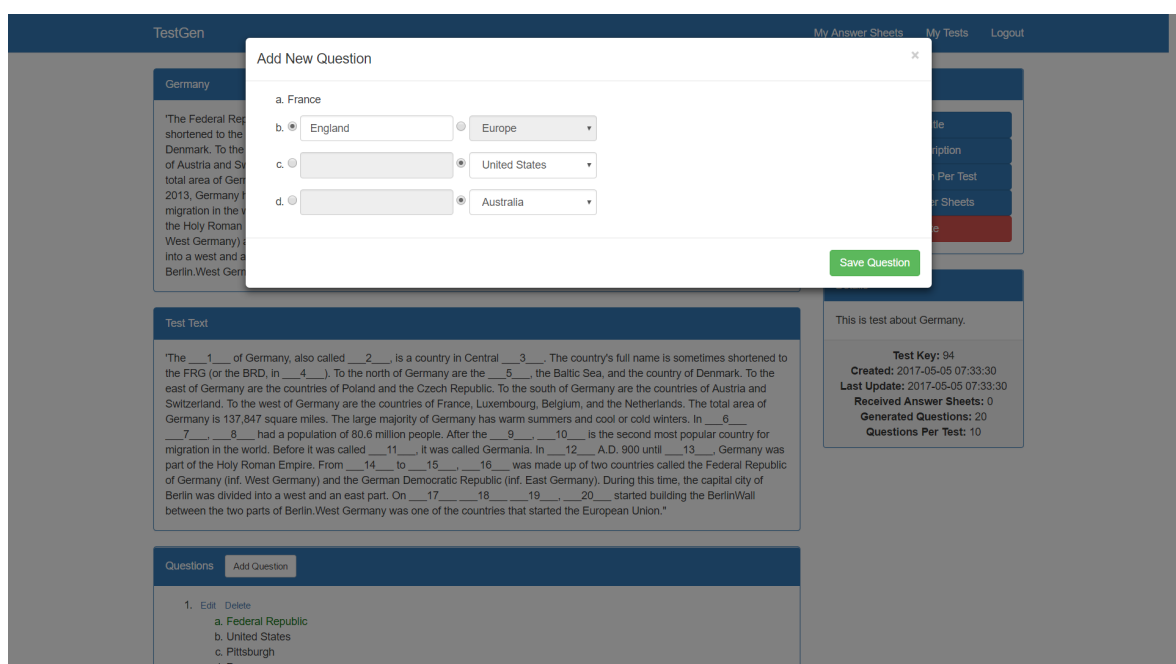
C.3.2 Úprava otázky

Po kliknutí na tlačidlo *Edit* pri niektorej z otázok v časti *Questions* sa zobrazí formulár pre zvolenie ďalších ponúknutých možností pri otázke (obrázok C.8). Po výbere požadovaných možností a kliknutí na tlačidlo *Save Question* sa otázka upraví a zobrazí sa obrazovka s detailami testu (obrázok C.5).

Vytvorenú otázku je možné zmazať kliknutím na tlačidlo *Delete* pri danej otázke v časti *Questions*.



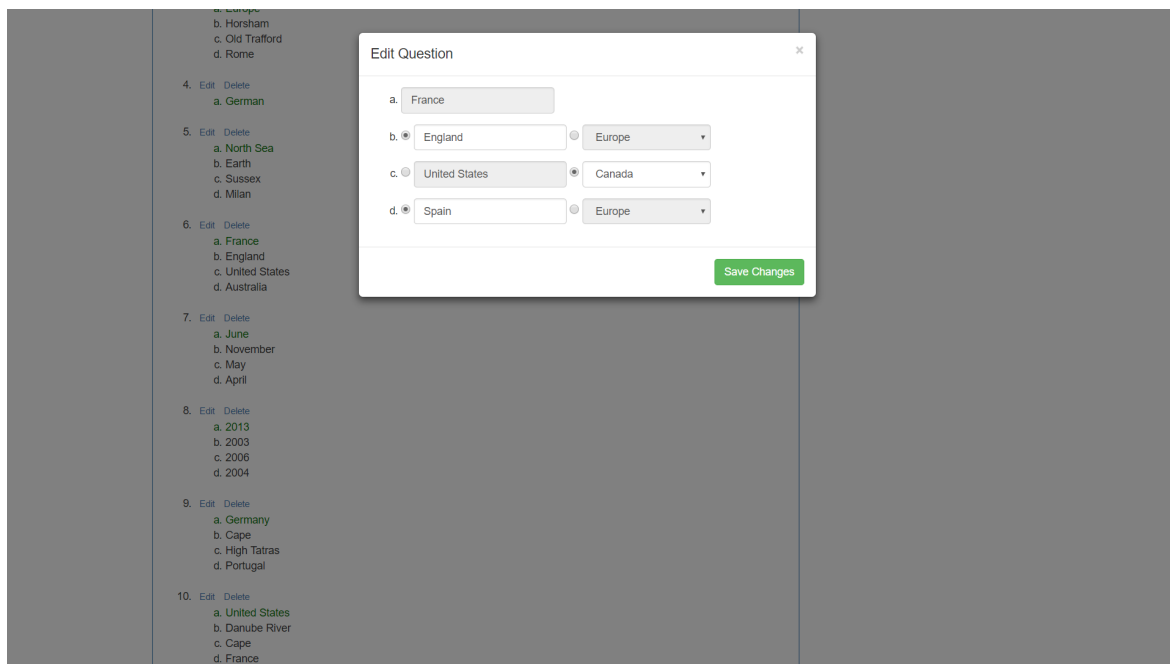
Obr. C.6: Obrazovka pre vytvorenie novej otázky.



Obr. C.7: Obrazovka pre výber ďalších možností k novej otázke.

C.3.3 Úprava informácií o teste

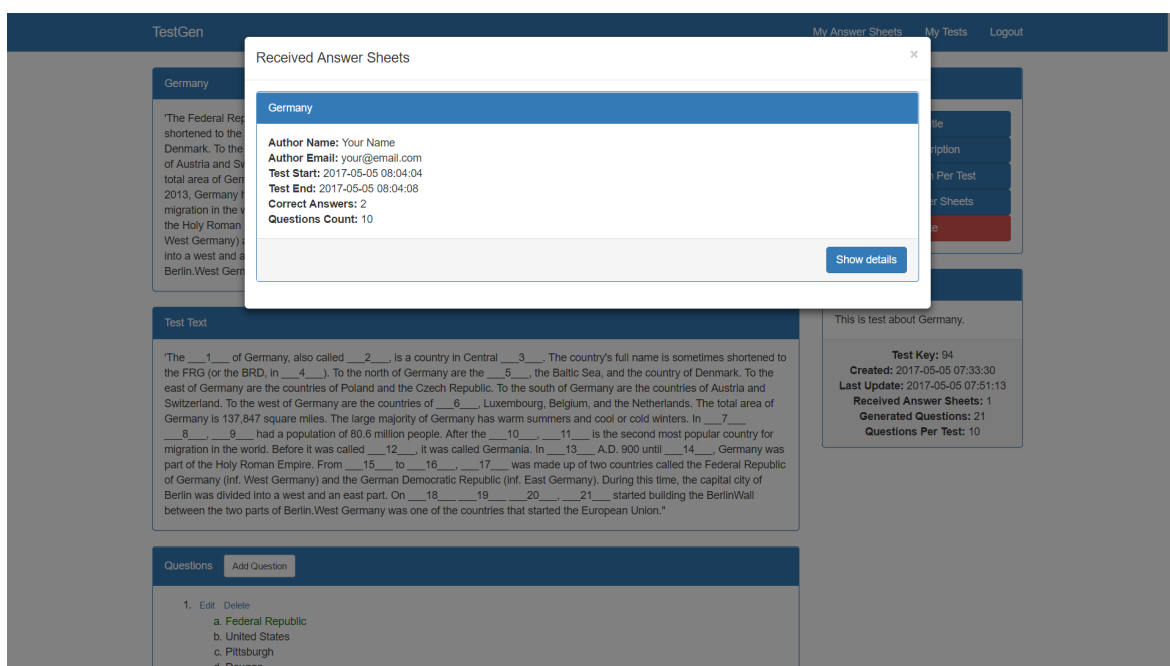
Upraviť základné informácie o teste je možné prostredníctvom tlačidiel v časti *Actions*. Po kliknutí na niektoré z tlačidiel sa zobrazí požadovaný formulár. Po kliknutí na tlačidlo *Delete* sa test vymaže. Upraviť názov testu je možné tlačidlom *Edit Title*, popis testu tlačidlom *Edit Description* a počet otázok pre jeden odpoveďový hárok tlačidlom *Edit Question Per Test*.



Obr. C.8: Obrázovka pre výber ďalších možností k otázke.

C.3.4 Zobrazenie odpoved'ových hárkov k testu

Zobraziť odpoved'ové hárky prijaté k zobrazenému testu je možné kliknutím na tlačidlo *Show Answer Sheets* v časti *Actions*. Po kliknutí na tlačidlo sa zobrazí prehľad týchto odpoved'ových hárkov (obrázok C.9).



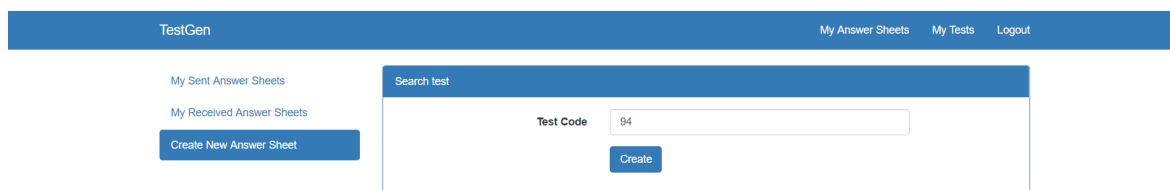
Obr. C.9: Obrázovka pre zobrazenie odpoved'ových hárkov prijatých k testu.

C.4 Správa odpovedových hárkov

V tejto časti popisujeme ako vytvoriť, vyplniť a odoslať odpovedový hárok k testu.

C.4.1 Vytvorenie odpovedového hárku

Odpovedový hárok je možné vytvoriť po kliknutí na tlačidlo *My Answer Sheets* v navigácii a následnom kliknutí na tlačidlo *Create New Answer Sheet* v ľavej časti obrazovky. Do zobrazeného formulára (obrázok C.10) je následne možné zadať *id* požadovaného testu a vytvoriť odpovedový hárok kliknutím na tlačidlo *Create* vo formulári.



The screenshot shows the TestGen web application interface. At the top is a blue navigation bar with the text 'TestGen' on the left and 'My Answer Sheets', 'My Tests', and 'Logout' on the right. Below the navigation bar, on the left side, there is a vertical menu with three items: 'My Sent Answer Sheets', 'My Received Answer Sheets', and 'Create New Answer Sheet' (which is highlighted with a blue button). To the right of this menu is a large white rectangular area with a blue header bar that says 'Search test'. Inside this area, there is a label 'Test Code' followed by a text input field containing the number '94'. Below the input field is a blue button labeled 'Create'.

Obr. C.10: Obrazovka pre vytvorenie odpovedového hárku.

C.4.2 Vyplnenie a odoslanie odpovedového hárku

Po vytvorení odpovedového hárku sa zobrazí obrazovka, na ktorej sa v ľavej časti nachádza text testu s vynechaným miestom pre jednotlivé otázky (obrázok C.11). V pravej časti sa nachádzajú otázky, ktoré boli priradené k danému odpovedovému hárku. Po výbere možností je možné tento hárok odoslať tlačidlom *Submit* v spodnej časti panelu s otázkami. Po odoslaní hárku sa zobrazí obrazovka s vyhodnotením hárku (obrázok C.12).

TestGen My Answer Sheets My Tests Logout

Germany

"The ____1____ of Germany, also called ____2____, is a country in Central Europe. The country's full name is sometimes shortened to the FRG (or the BRD, in German). To the north of Germany are the ____3____, the Baltic Sea, and the country of Denmark. To the east of Germany are the countries of Poland and the Czech Republic. To the south of Germany are the countries of Austria and Switzerland. To the west of Germany are the countries of ____4____, Luxembourg, Belgium, and the Netherlands. The total area of Germany is 137,847 square miles. The large majority of Germany has warm summers and cool or cold winters. In June 2013, Germany had a population of 80.6 million people. After the ____5____, Germany is the second most popular country for migration in the world. Before it was called ____6____, it was called Germania. In ____7____ A.D. 900 until ____8____, Germany was part of the Holy Roman Empire. From ____9____ to 1990, Germany was made up of two countries called the Federal Republic of Germany (inf. West Germany) and the German Democratic Republic (inf. East Germany). During this time, the capital city of Berlin was divided into a west and an east part. On 13 August ____10____, East Germany started building the BerlinWall between the two parts of Berlin. West Germany was one of the countries that started the European Union."

Questions

Question #1

- ☐ United States
- ☐ Pittsburgh
- ☒ Federal Republic
- ☐ Dougga

Question #2

- ☒ Germany
- ☐ Brantford
- ☐ England
- ☐ Munich

Question #3

- ☒ Earth
- ☐ Milan
- ☐ North Sea
- ☐ Sussex

Question #4

- ☐ France
- ☒ Australia
- ☐ England
- ☐ United States

Question #5

- ☒ France
- ☐ Cape
- ☐ United States
- ☐ Danube River

Question #6

Obr. C.11: Obrazovka pre vyplnenie odpoved'ového hárku.

TestGen My Answer Sheets My Tests Logout

Germany

"The ____1____ of Germany, also called ____2____, is a country in Central Europe. The country's full name is sometimes shortened to the FRG (or the BRD, in German). To the north of Germany are the ____3____, the Baltic Sea, and the country of Denmark. To the east of Germany are the countries of Poland and the Czech Republic. To the south of Germany are the countries of Austria and Switzerland. To the west of Germany are the countries of ____4____, Luxembourg, Belgium, and the Netherlands. The total area of Germany is 137,847 square miles. The large majority of Germany has warm summers and cool or cold winters. In June 2013, Germany had a population of 80.6 million people. After the ____5____, Germany is the second most popular country for migration in the world. Before it was called ____6____, it was called Germania. In ____7____ A.D. 900 until ____8____, Germany was part of the Holy Roman Empire. From ____9____ to 1990, Germany was made up of two countries called the Federal Republic of Germany (inf. West Germany) and the German Democratic Republic (inf. East Germany). During this time, the capital city of Berlin was divided into a west and an east part. On 13 August ____10____, East Germany started building the BerlinWall between the two parts of Berlin. West Germany was one of the countries that started the European Union."

Questions & Answers

1.

- a. United States
- b. Pittsburgh
- c. Federal Republic
- d. Dougga

2.

- a. Germany
- b. Brantford
- c. England
- d. Munich

3.

- a. Earth
- b. Milan
- c. North Sea
- d. Sussex

4.

- a. France
- b. Australia

Details

This is test about Germany.

Author Name: Your Name
Author Email: your@email.com
Test Start: 2017-05-05 08:11:09
Test End: 2017-05-05 08:17:59
Correct Answers: 3
Questions Count: 10

Obr. C.12: Obrazovka s vyhodnotením odpoved'ového hárku.

Dodatok D

Príklady otázok vytvorených počas testovania

Počas evaluácie systému sme vytvorili celkom 95 otázok. Používatelia tieto otázky zaraďovali do kategórií a hodnotili ich kvalitu. V tejto časti sa nachádzajú príklady otázok zo všetkých kombinácií kategórie a kvality, do ktorých bola zaradená aspoň jedna otázka. Pri príkladoch sú uvedené aj vygenerované možnosti. Pri každej otázke je správna odpoveď označená ako možnosť a).

D.1 Otázky zamerané na osobu

Ohodnotenie: Otázka aj ponúknuté možnosti sú korektné.

Otázka: _____'s acting career began in Horsham, Sussex.

- a) Caine
- b) Wolfgang Amadeus Mozart
- c) Jacques Charles
- d) Tycho Brahe

Ohodnotenie: Otázka je korektná, ale 1 z ponúknutých možností nie je korektná.

Otázka: When _____ and his Normans came to Britain in the eleventh century, a vast number of words, derived both from Norman French and from written Latin, entered English.

- a) William Conqueror
- b) Rasmus Lerdorf
- c) Rijndael
- d) Stephen Schwarzman

Ohodnotenie: Otázka je korektná, ale 2 z ponúknutých možností nie sú korektné.

Otázka: This method was much used by astronomers towards the end of the sixteenth century, particularly by the great Danish astronomer Tycho Brahe, who was visited by a young friend of Napier, _____, in 1590.

- a) John Craig
- b) Steven
- c) Martin
- d) Michael Wade

Ohodnotenie: Otázka je korektná, ale 3 z ponúknutých možností nie sú korektné.

Otázka: ----- had one sibling, his younger brother, Hilary Arthur Reuel, who was born on 17 February 1894.

- a) Tolkien
- b) Destiny
- c) Europe Union
- d) Barcelona

D.2 Otázky zamerané na dátum

Ohodnotenie: Otázka aj ponúknuté možnosti sú korektné.

Otázka: Following his horse-riding interests, he enlisted in ----- in the Yeomanry, and was sent to the front in France in early 1916.

- a) 1915
- b) 1907
- c) 1923
- d) 1911

D.3 Otázky zamerané na organizáciu

Ohodnotenie: Otázka aj ponúknuté možnosti sú korektné.

Otázka: In March 2009, ----- CEO Stephen Schwarzman said that up to 45% of global wealth had been destroyed in little less than a year and a half.

- a) Blackstone Group
- b) Apple
- c) Unicorn-Systems
- d) Royal Oak

Ohodnotenie: Otázka je korektná, ale 1 z ponúknutých možností nie je korektná.

Otázka: The ----- was launched in a converted factory in 1903 with \$28,000 in cash from twelve investors, most notably John and Horace Dodge (who would later found their own car company).

- a) Ford Motor Company
- b) Barcelona
- c) Unicorn-Systems
- d) SAP

Ohodnotenie: Otázka je korektná, ale 2 z ponúknutých možností nie sú korektné.

Otázka: The ----- is one of two institutions created at the Bretton Woods Conference in 1944.

- a) World Bank
- b) Google
- c) AES
- d) Raggett

Ohodnotenie: Otázka je korektná, ale 3 z ponúknutých možností nie sú korektné.

Otázka: Montenegro is a potential candidate for membership in the ----- and uses the Euro.

- a) European Union
- b) Destiny
- c) Raggett
- d) IBM

D.4 Otázky zamerané na miesto

Ohodnotenie: Otázka aj ponúknuté možnosti sú korektné.

Otázka: Following his horse-riding interests, he enlisted in 1915 in the Yeomanry, and was sent to the front in ----- in early 1916.

- a) France
- b) Slovakia
- c) England
- d) Japan

Ohodnotenie: Otázka je korektná, ale 1 z ponúknutých možností nie je korektná.

Otázka: DNA research shows that the inhabitants of Ireland came from ----- in prehistoric times.

- a) Spain
- b) France
- c) Caribbean
- d) Britain

Ohodnotenie: Otázka je korektná, ale 2 z ponúknutých možností nie sú korektné.

Otázka: With a total area of about 106.4 million square kilometres (41.1 million square miles), it covers approximately one-fifth of the -----'s surface and about one-quarter of its water surface area.

- a) Earth
- b) UK
- c) London
- d) Caribbean

Ohodnotenie: Otázka je korektná, ale 3 z ponúknutých možností nie sú korektné.

Otázka: Just after Columbus sailed across the ----- the Pope, asserting the 'right' to control non-Christian lands, drew a line to separate the spheres of influence of Spain and Portugal.

- a) Atlantic Ocean
- b) Horsham
- c) US
- d) River Bodrog

Ohodnotenie: Otázka nie je korektná.

Otázka: Their ancestors migrated through the middle and near east into Europe, beginning some 30-40 thousand years ago.

- a) east
- b) Edmonton
- c) London
- d) North Sea

D.5 Ostatné otázky

Ohodnotenie: Otázka je korektná, ale 2 z ponúknutých možností nie sú korektné.

Otázka: The ancient city of Thugga is often known by its modern name,

- a) Dougga
- b) Great Britain
- c) South Korea
- d) Holland

Ohodnotenie: Otázka je korektná, ale 3 z ponúknutých možností nie sú korektné.

Otázka: And later in the same century the Emperor Marcus Aurelius dispatched a trading mission by land which communicated directly with Huan-ti, the emperor of China.

- a) Roman
- b) Tatar
- c) Montenegro
- d) United States

Ohodnotenie: Otázka nie je korektná.

Otázka: With a total area of about 106.4 million square kilometres (41.1 million square miles), it covers approximately one-fifth of the Earth's surface and about of its water surface area.

- a) one-quarter
- b) about

Dodatok E

Plán na letný semester

V rámci pokračovania v priebehu letného semestra sme sa plánovali zamerať na návrh a implementáciu systému. Úlohy potrebné na úspešné dokončenie systému sme sa rozhodli rozdeliť nasledovne:

- 1. týždeň - návrh architektúry systému a testovanie možností komunikácie medzi jednotlivými komponentami,
- 2. týždeň - vytvorenie základnej kostry systému,
- 3. týždeň - identifikovanie jednotlivých typov otázok, ktoré budú generované systémom,
- 4. týždeň - implementácia niektorých typov otázok a testovanie systému,
- 5. týždeň - implementácia zvyšných typov otázok,
- 6. týždeň - testovanie vytvorenej časti systému a identifikácia možných chýb,
- 7. týždeň - oprava odhalených chýb v systéme,
- 8. týždeň - návrh používateľského rozhrania a vytvorenie prototypu, testovanie komunikácie medzi používateľským rozhraním a aplikáciou na generovanie otázok,
- 9. týždeň - implementácia používateľského rozhrania,
- 10. týždeň - implementácia a testovanie používateľského rozhrania,
- 11. týždeň - testovanie systému a spracovanie výsledkov,
- 12. týždeň - vyhradené pre nepredvídateľné komplikácie.

Splnenie plánu sme vyhodnotili nasledovne:

- 1. týždeň - Úspešne sme navrhli architektúru systému a otestovali spôsob komunikácie medzi jednotlivými komponentami.
- 2. týždeň - Identifikovali sme základné typy otázok, ich pravidiel a spôsob generovania možností.
- 3. týždeň - Implementovali sme spracovanie vstupného textu a uchovanie získaných informácií. Túto časť sme v pláne nemali zahrnúť, avšak návrh typov otázok sme zvládli v týždňovom predstihu.
- 4. týždeň - Implementovali sme jednotlivé typy otázok, tak ako sme navrhli v našom pláne.
- 5. týždeň - Testovali sme implementované typy otázok a opravovali chyby, ktoré sa vyskytli.
- 6. týždeň - V predchádzajúcom týždni sme identifikovali niektoré ďalšie typy otázok, ktoré sme v tomto týždni implementovali a aj testovali.
- 7. týždeň - Začali sme implementovať klientskú aplikáciu. Návrh aplikácie sme zvládli v predstihu a v tomto týždni sme úspešne zvládli implementovať komunikáciu s aplikáciou na tvorbu testov.
- 8. týždeň - Pokračovali sme v implementácii klientskej aplikácie. Oproti pôvodnému plánu sme boli v predstihu, avšak počas implementácie sme narazili na niekoľko problémov, ktoré spôsobili, že klientskú aplikáciu sme implementovali pomalšie ako sme plánovali.
- 9. týždeň - Naďalej sme implementovali klientskú aplikáciu. Počas implementácie sme objavovali nové problémy v aplikácii na tvorbu testov, ktoré sme priebežne opravovali.
- 10. týždeň - Implementácia klientskej aplikácie stále pokračovala. Podľa pôvodného plánu sme sa v tomto týždni mali venovať už testovaniu klientskej aplikácie, avšak kvôli objaveným chybám v aplikácii na tvorbu testov sme sa viac venovali testovaniu tejto aplikácie.
- 11. týždeň - S týždňovým oneskorením sme testovali klientskú aplikáciu. Počas testovania sme opravovali objavené chyby, no popri testovaní klientskej aplikácie sme sa zároveň venovali spracovaniu výsledkov a evaluácii systému.
- 12. týždeň - Opravovali sme nájdené chyby v oboch aplikáciách. Boli sme v miernom časovom sklze, avšak tento týždeň bol vyhradený práve na takýto typ úloh. Počas posledných dní sme aplikáciu upravili do požadovanej a stabilnej podoby.

Dodatok F

Článok z IIT.SRC

Prácu sme prezentovali na konferencií IIT.SRC v kategórií inovatívnych aplikácií. Niektoré z pripomienok účastníkov sme stihli zapracovať do tejto práce.

Question Generation from Educational Text

Tomáš GÁBRŠ*

*Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia*

`gabrstomas@gmail.com`

Abstract. One of the key factors of learning is evaluation of student's knowledge. Knowledge tests are the most common form of testing. With increasing number of studying materials in electronic form increases also interest in automatically generated tests from these materials. We focus on development application for teachers, which can help during test preparation to automatically generate questions from text. The main objective is to make the teacher's work more efficient and to increase the quality of the tests.

1 Introduction

The field of information technology is one of the most rapidly expanding disciplines. Nowadays, it is also common part of the educational process and it is important to use it as effective as possible. Regular part of educational process in schools and universities is paper exam. Exams are often created manually, so teacher has to identify target parts of text, construct the question and finally generate the whole exam in suitable format for printing.

Our goal is to simplify this process and generate questions for exam automatically based on the input text. This should be done with our proposed application. Questions can be edited by user and the final exam can be saved or exported. We obtain important information from the articles by various tools like in [1, 2, 3].

2 Proposed system

The system is divided into three components. First is external tool Stanford CoreNLP, which is used for parsing plain text and retrieving information about the structure of the sentences [4]. These information are necessary, because questions are generated on the basis of these information. With use this tool to enrich information about tokens from sentences, as well as information about whole sentences. Besides that, we are able to retrieve information about relations between the tokens. We also work with OpenIE (Open information extraction) elements, which refer

to the extraction of triples: subject, relationship and object.

There are many possibilities how to use CoreNLP. For us is the best option to use it as a standalone application, CoreNLP Server, which provides an API for our application. The most important advantage of this usage is the response time, because it keeps the annotators loaded after the initial request and there is no need for loading it separately for every request. There is also a library for Java client, which is very comfortable to use.

Second component of the application, server-side application, is responsible for business logic and question generation based on information received from CoreNLP. It provides an API for the client application and main parameter of the request is input text. There are not any restrictions for the length of the text – it can be simple sentence as well as whole paragraph. There are also optional parameters and they can be used for specifying requirements by user. It is possible to specify the need of generating distractors for the questions or the maximum number of generated questions. If input text is the article with title, it can be also sent within parameters.

After receiving request, CoreNLP API is called via provided CoreNLP client. It is configured according to the needs of the request and then API call is made. Response time of CoreNLP is based on the length of input text. When response is received, we bind provided information to a model classes. There are multiple classes, which represent various levels of information. These classes contain required attributes, which are filled in pursuance of CoreNLP response and their structure can be represented as tree with one root element, which is model class for article. This structure can be easily transformed to JSON string and passed to the other components of application.

When all information about the input text are available and set as parameters of the model classes, we focus on implementing algorithms for question generation as effective and reliable as possible. First

* Bachelor degree study programme in field: Informatics
Supervisor: Ing. Miroslav Blšták, Institute of Informatics, Information Systems and Software Engineering, Faculty of Informatics and Information Technologies STU in Bratislava

step is to identify the target parts of input text. We implemented multiple types of questions and every question has its own rules. These rules can have various complexity. All rules are based on the information provided by CoreNLP and saved within the model classes. Algorithm iterates through the tokens and every token is tested for every question rule. If rule is marked as fulfilled, all required information about question are written into the output. During iterations, there are another information written into the output, for example every token representing person included in input text is written into the list of names. These additional information can be used for generating distractors, because words with similar meaning are stored in the same list.

Our system generates fill-the-blank questions. Target part of question is blanked and student has to fill correct answer. Answer can consist from one or few words. Distractors are selected on the basis of information about the tokens. One of the key information for distractors selection is named entity tag. For example, when correct answer is person, distractors are also selected from the list of persons. After generation of all possible questions from the input text, the output is provided to the client.

Client application is available for the teachers and it provides various functionality. Teacher has to register to get access to all functionality of system. After registration and login, teacher can use system for exam generation. When he sends the request with the input text, server-side application is called to generate new exam. When the exam is completely generated, it is returned to the client application and displayed in the interface.

Exam is composed of the fill-the-blank questions. The test screen shows the original text with gaps for putting the correct answers. When the teacher selects one of these blanked parts, system shows the possible answers (correct answer and distractors if they were generated). Teacher can edit them, delete them or add new options. There is also possibility to create questions manually. Teacher can select part of text and generate new question related to marked part. After adding this question, it is equivalent to questions generated by application, so next options can be also added.

At the end, text contains multiple questions with correct answers and optional distractors and teacher can save the exam. It is inserted into database and it can be load in the future.

When exam is completed and ready for exporting, teacher can configure various output parameters. He does not have to manually create multiple variations of exam for every student, but he can simply set number of students and for every student is generated

unique exam. There is also option to specify if distractors will be exported too.

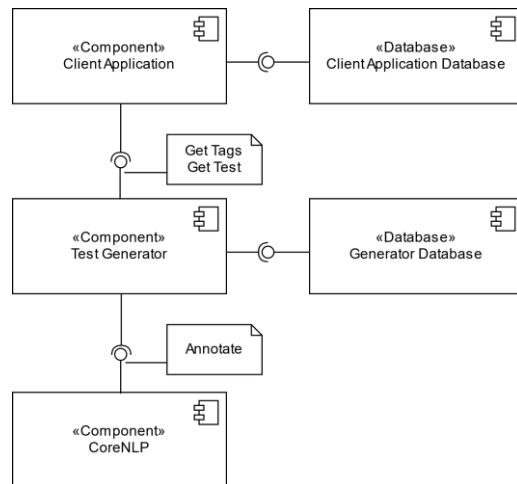


Figure 1. Component diagram of proposed question generation system.

3 Conclusions

Our application focuses to factual question generation from text. It was designed as a tool for teachers. It allows to create, save and reuse questions for tests. The tests can be also automatically evaluated.

Acknowledgement: This work was partially supported by the Scientific Grant Agency of Slovak Republic, grant No. VG 1/0752/14.

References

- [1] ALI, Husam Deeb Abdullah Deeb. Automatic question generation: a syntactical approach to the sentence-to-question generation case. 2012. PhD Thesis. Lethbridge, Alta.: University of Lethbridge, Dept. of Mathematics and Computer Science, c2012.
- [2] AGARWAL, Manish; MANNEM, Prashanth. Automatic gap-fill question generation from text books. In: *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 2011. p. 56-64.
- [3] BEINBORN, Lisa; ZESCH, Torsten; GUREVYCH, Iryna. Candidate evaluation strategies for improved difficulty prediction of language tests. In: *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*. 2015. p. 1-11.
- [4] MANNING, Christopher D., et al. The stanford corenlp natural language processing toolkit. In: *ACL (System Demonstrations)*. 2014. p. 55-60.