Sem vložte zadání Vaší práce.

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Diplomová práce

E-learningový portál Mentica - modul správa testů

Bc. Olga Budnik

Vedoucí práce: Ing. Jiří Chludil

4. května 2015

Poděkování

Tímto bych chtěla poděkovat svému vedoucímu Ing. Jiřímu Chludilovi za cenné rady a pomoc v realizaci této práci a svým kolegům Bc. Jiřímu Matějkovi a Bc. Jaroslavu Tesařovi za spolupráci a vzájemnou podporu. Ráda bych také poděkovala svým rodičům za možnost studovat vysokou školu a za morální podporu během studia.

Prohlášení

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

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

V Praze dne 4. května 2015 .....

Odkaz na tuto práci

Budnik, Olga. *E-learningový portál Mentica - modul správa testů.* Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2015.

Abstrakt

Tato diplomová práce se zabývá návrhem a realizací modulu pro správu testů a otázek ve výukovém portálu Mentica. Práce je součástí týmového projektu, který je rozdělen do více modulů. Modul pro správu testů a otázek umožňuje uživatelům vytvářet vlastní testy, plnit je otázkami, editovat a poté publikovat. Otázky je možné vytvářet nezávisle na testech a seskupovat je do složek, což umožňuje znovupoužití otázek. Uživatel má na výběr mezi více druhy otázek a může tak sestavovat rozmanité testy. Výsledkem práce je prototyp aplikace, který je možné použít pro budoucí vývoj.

Klíčová slova Diplomová práce, vzdělávání, e-learning, Internet, test, otázka, studium, mentor, student, banka otázek

Abstract

This diploma thesis deals with the design and realization of module for test and questions management within Mentica learning portal. The thesis is a part of the team project, which is divided in more modules. The module for tests and questions management allows users to create their own tests, add questions to them, edit them and publish them. It is possible to create the questions independently of the tests and group questions in folders, which

allows the existing questions to be reused in more tests. User can choose from more question types, which helps him to create various tests. The result of the diploma thesis is a prototype of application that can be used in the future development.

Keywords Diploma thesis, studying, e-learning, test, question, mentor, student, bank of questions

Obsah

Seznam obrázků

Seznam tabulek

Úvod

V moderním světě rychle se vyvíjecích Internetových technologií a mobilních zařízení nabírá na popularitě vzdálené vzdělávání přes Internet neboli E-learning. Člověk již není omezen časem nebo místem – vzdělávat se může každý, stačí mít počítač a připojení k Internetu. Samovzdělávání začíná být přístupné v jakékoliv oblasti. Největší E-learningové portály mají širokou nabídku kurzů a lekcí na nejrůznější témata v různých úrovních složitosti. Každý uživatel si tak přijde na své a může si sestavit svůj vlastní vzdělávací program, který mu bude vyhovovat jak časově, tak i cenově a zároveň bude odpovídat úrovni jeho znalostí. E-learning se stále rozvijí, objevují se nové trendy a více a více lidí začíná využívat výhod vzdáleného vzdělávání. Avšak v České republice není tento trend ještě tak rozšířen a český trh vzdělávání nabízí zejména portály poskytující výuku pouze v určité oblasti, například cizích jazyků. Většina existujících portálů nabízí vlastní, již připravené kurzy, což přináší jistá omezení pro uživatele, kteří chtějí publikovat svůj vlastní výukový obsah.

S nápadem vytvořit rozsáhlý vzdělávací portál zaměřený na český trh přišel zadavatel diplomové práce Petr Mentberger. Cílem projektu Mentica je vytvořit moderní E-learningový portál, který nabídne vzdělávání ve více sférách a za pomoci výukových materiálů různých druhů. Uživatel zde může být jak studentem čerpajícím výukový obsah, tak i učitelem, který obsah vytváří. Výukové materiály jsou přístupné online v podobě kurzů a testů a zároveň je zde i možnost publikování a správy pozvánek na prezenční semináře nebo workshopy. Významnou roli hraje obohacení vzdělávacího procesu o elementy gamifikace. Uživatel může získávat speciální body a odměny za absolvování kurzů a testů nebo za publikování vlastních materiálů. Studenti si můžou porovnávat své výsledky s výsledky jiných studentů, což umožní motivovat uživatele k lepším výsledkům formou soutěže. O svém úspěchu a průběhu výuky se uživatel může podílet s přáteli prostřednictvím sociálních sítí, které umožní přilákat nové lidí do portálu.

Ve výsledku by se měl portál Mentica stát moderním výukovým portálem, který bude schopen uspokojit potřeby uživatelů v jakékoliv oblasti vzdělávání

a dokáže zaujmout co nejrozsáhlejší skupinu uživatelů.

Kapitola 1

Popis projektu a rozdělení práce

Projekt Mentica je velice rozsáhlý a má za úkol vyřešit mnoho netriviálních problémů. Proto bylo rozhodnuto rozdělit projekt do několika základních modulů řešících dílčí úlohy. Každý modul je navržen a popsán v jednotlivých diplomových pracích.

1.1 Slovník pojmů

V této kapitole jsou zavedené nejčastěji používané terminy v rámci projektu a jejích význam.

Mentor – synonymum pro učitele. Mentor je uživatel, který vede výuku, vytváří výukové materiály a testy a sleduje úspěchy svých studentů.

Student – uživatel, který odebírá výukový obsah a vyplňuje testy. Student prostřednictvím portálu získává znalosti.

Kurz – druh výuky, který slouží k poskytnutí znalostí studentům. Skládá se z kapitol a lekcí, které jsou tvořeny různými výukovými materiály.

Test – druh výuky, který slouží k ověření znalostí studentů. Test se skládá z otázek různých typů, na které musí uživatel odpovědět.

Banka otázek – úložiště otázek, které jsou tematicky seskupené do složek.

Gamifikace – je uplatnění herních prvků a principů za účelem nalákání nových uživatelů a zvýšení uživatelského zájmu.

1.2 Modul Uživatel

Tento modul se zabývá převážně správou uživatelů, jejich práv a účtů. Modul řeší základní úlohy, kterými jsou například registrace nebo propojení se sociálními sítěmi, a také složitější problémy, například přidělení práv uživatelům, vytváření uživatelských skupin a jejich správa. Každému uživateli se po registraci vytvoří jeho vlastní účet, který může následně spravovat a modifikovat. Tento účet je propojen i s uživatelskou aktivitou na portálu. Uživatel

má přehled o všech svých akcích, o absolvovaných kurzech nebo testech, jsou mu poskytnuté výukové výsledky v podobě statistik. Na základě svého profilu si uživatel může vytvořit životopis, do kterého si může přidat i výsledky ze certifikovaných kurzů a testů. Návrh a implementace modulu Uživatel jsou podrobně zpracovány v diplomové práci Bc. Jaroslava Tesaře [1].

1.3 Modul Kurz

Cílem modulu Kurz je vytváření a absolvování kurzů. Každý kurz se skládá z kapitol, které se dále skládají z lekcí. Lekce je postavena na různých výukových materiálech, například se může jednat o klasickou textovou přednášku, video lekci, prezentaci, tabulku nebo kombinaci více materiálů najednou. Vytváření kurzů má na starosti učitel neboli mentor, který u kurzů nastavuje atributy jako popis, podmínky ke splnění a později kurz zveřejňuje pro ostatní uživatele. Díky přiřazení kurzu do určité kategorie lze kurz snadno vyhledat a studenti se mohou do kurzů lehce zapsat. Návrh a implementace modulu Kurz jsou podrobně zpracovány v diplomové práci Bc. Jiřího Matějky [2].

1.4 Modul Gamifikace

Modul Gamifikace se zabývá realizací motivačních prvků portálu. Uživatel má možnost získávat body a odměny za své aktivity na portálu. Každá aktivita (například splnění testu) může být odměněna speciálním odznakem. Pokud se uživatel polepší ve svých výsledcích, může získat další odznak, což vede uživatele k většímu zájmu o studium a překonávání vlastních úspěchů. Uživatelé také získávají zkušenostní body, které mohou později využít k otevření přístupu k uzamčeným materiálům. Uživatelům jsou k dispozici statistiky, které jim zobrazují průběh výuky v čase. Uživatelé mají možnost navzájem hodnotit výukové materiály a nahlašovat chyby nebo nevhodný obsah v kurzech a testech. Produkty s větším hodnocením se zobrazí v horních pozicích ve vyhledávání, díky čemuž budou mít větší návštěvnost. Tento systém by měl motivovat mentory k vytváření a publikaci kvalitnějšího obsahu. Modul Gamifikace není v současné době předmětem žádné diplomové práce.

1.5 Modul Test

Posledním modulem je modul Test, který je věnován vytváření a správě otázek, ze kterých jsou sestavovány testy. Každý mentor může zkoušet znalosti svých studentů pomocí testů. Test se skládá z několika otázek různých typů a může být vytvořen manuálně přidáním nových otázek nebo automaticky vygenerován z existujících úloh na základě zadaných kritérií. Student může absolvovat test a v případě úspěšného pokusu získá určité body. Testy mohou sloužit k odemykání nových kapitol v kurzu, pokud jsou navazující kapitoly vázané na

splnění testu. Modul Test je podrobně zanalyzován, popsán a navržen dále v této diplomové práci.

Kapitola 2

Analýza

V kapitole analýza jsou popsány a zhodnoceny existující řešení na trhu online vzdělávání a to jak české, tak i zahraniční. Na základě této rešerše byly definovány uživatelské role v budoucí aplikace a sestaven seznam funkčních a nefunkčních požadavků, který je vstupním krokem do návrhu aplikace.

2.1 Analýza existujících řešení

Současný český trh nabízí velmi omezený počet kvalitních online řešení v oblasti vzdělávání. Větší část podobných projektů je úzce specifikovaná pouze na určitý předmět nebo oblast a zabývá se například jenom jazyky. Pokud by měl uživatel zájem o systém vzdělávacích testů ve více oblastech najednou, musel by využít služby zahraničního portálu.

Celkem byly vybrány pro analýzu konkurence 4 nejzajímavější projekty, které se nejvíce podobají plánované aplikaci Mentica. Každý projekt byl zhodnocen podle různých kritérií popsaných v tabulkách 2.1 - 2.5.

2.1.1 Zahraniční trh

V této kapitole jsou popsána existující řešení ze zahraničního trhu. První – Exam Professor (viz kapitola 2.1.1.1) – je samostatný portál, který nabízí pouze testy. Druhé řešení – ProProfs Quiz Maker (viz kapitola 2.1.1.2) – je součásti většího portálu, který nabízí mnoho dalších produktu v oblasti vzdělávání, například vzdělávací hry nebo tak zvané báze znalostí.

2.1.1.1 Exam Professor

Exam Professor (`www.examprofessor.com`) je webové rozhraní umožňující snadno a rychle vytvářet vlastní testy a kvízy a následně je publikovat a prodávat. Rozhraní je určeno především pro školy a organizace, cena za jeho použití se liší podle počtu studentů, učitelů a testů. Systém je převážně zaměřen na uči-

Hodnocení	Vysvětlení
1	Uživatel v rámci portálu může být pouze studentem, tj. pouze odebírá výukový obsah.
2	Uživatel může být jak učitelem, tak i studentem, avšak v roli učitele nemá plnou kontrolu nad obsahem.
3	Uživatel může být jak učitelem, tak i studentem. Pokud je uživatel v roli učitele, může vytvářet vlastní obsah a má plnou kontrolu nad jeho správou.

Tabulka 2.1: Hodnocení podle uživatelských rolí

Hodnocení	Vysvětlení
1	Aplikace nabízí výukové materiály pouze v jedné oblasti
2	Aplikace nabízí výukové materiály ve více oblastech

Tabulka 2.2: Hodnocení podle výukových oblastí

Hodnocení	Vysvětlení
1	Systém nabízí testovací otázky 1-2 typů.
2	Systém nabízí testovací otázky 3 a více typů.

Tabulka 2.3: Hodnocení podle typu otázek

tele, student zde pouze vyplňuje testy. V rámci analýzy byla otestovaná verze zdarma, která je omezena na jednu zkoušku a 10 studentů.

Pohled učitele

Učitel má na starosti vše týkající se vytváření a správy testů, uživatelů a nastavení cen a způsobů placení. K dispozici má několik modulů, které pomáhají strukturovat práci a poskytují lepší přehled o dostupných aktivi-

Hodnocení	Vysvětlení
1	Není poskytnuto žádné úložiště otázek, otázky existují pouze uvnitř testů a nejde s nimi manipulovat mimo test.
2	Otázky existují uvnitř testů, ale jde je mezi testy přesouvat a kopírovat.
3	Systém umožňuje vytvářet a spravovat otázky mimo testy, k čemuž slouží speciální úložiště.

Tabulka 2.4: Hodnocení podle existence úložiště otázek

Hodnocení	Vysvětlení
1	Proces výuky není obohacen o žádné motivační prvky.
2	Proces výuky je obohacen o některé motivační prvky (například videa, originální nápovědy), avšak chybějí ocenění a odměny za úspěšnou výuku.
3	Proces výuky je obohacen o motivační prvky v podobě ocenění a odměn.

Tabulka 2.5: Hodnocení podle gamifikačních a motivačních prvků

tách. V rámci analýzy budou rozebrány moduly Questions (Otázky) a Exams (Zkoušky).

Modul Exam

Modul Exam slouží pro vytvoření a správu jednotlivých zkoušek neboli testů. Zkoušky se dělí na privátní a veřejné. Privátní zkoušky jsou určené pouze pro registrované uživatele, pro něž jsou potom zaznamenávány statistiky a historie absolvování zkoušky. Veřejné zkoušky jsou dostupné komukoliv, ale musejí být předem sdílené učitelem na jeho vlastních stránkách.

Zkoušky mohou být zdarma nebo placené. Učitel si sám zvolí cenu za zkoušku a případně i za její opakování. Cena za každý další pokus je stejná.

U každé zkoušky má učitel možnost nastavit procentuální počet bodů potřebných k jejímu splnění a počet pokusů na zkoušku. Výsledky zkoušky mohou

být odeslány přes email jak učiteli, tak i studentovi. Případně je možné zadat i další emailové adresy, na které má být odeslána notifikace o absolvování zkoušky.

Zkoušky se dále dělí na časově omezené a neomezené. U časové omezených zkoušek má učitel možnost nastavit dobu, po které se zkouška automaticky vyhodnotí, a její výsledky se odešlou učiteli. Pokud časové omezení není nastaveno, student si sám určí, kdy je připraven vyhodnotit svoji odpovědi a celková zkouška se vyhodnotí po manuálním odesílání studentem.

U zkoušek je možné nastavit zobrazení nápovědy, která se bude zobrazovat studentovi v průběhu zkoušky a pomáhat mu při zodpovídání jednotlivých otázek.

Učitel má možnost zvolit si informace, které chce poskytovat studentovi po splnění zkoušky. Například student uvidí pouze svůj výsledek nebo se mu zobrazí kompletní přehled jeho odpovědí včetně označení, zda jsou zodpovězeny správně či ne. Učitel také vybírá a nastavuje pořadí otázek a odpovědí.

Každá zkouška je zamčena a není dostupná studentům, pokud mentor nedokončí všechny úpravy a nenastaví status zkoušky na „Aktivní“.

Modul Questions

Modul Questions slouží ke správě otázek, použitých v testech. Otázky jsou vždy přiřazeny k některé zkoušce, proto je potřeba před vytvářením otázek založit zkoušku. Na výběr jsou otázky typu pravda/nepravda nebo klasické otázky s výběrem z více možností. U každé otázky se musí povinně uvést zadání a přidat odpovědi. Učitel rovněž může zadat popis k otázce, který bude sloužit jako nápověda pro studenta, nebo přidat k otázce přílohu v podobě obrázku.

Učitel má možnost měnit pořadí otázek a také měnit jejich stav na „aktivní“ nebo „neaktivní“. Neaktivní otázky se neobjeví v testu z pohledu studenta. Pokud učitel spravuje více zkoušek, může vytvořené otázky zkopírovat z jedné zkoušky do jiné.

Pohled studenta

Aktivita studenta na portálu začíná potvrzením pozvánky od učitele. Studentovi se dále otevře zkouška, kterou může absolvovat. Před zahájením zkoušky se student vždy dozví, kolik procent otázek má zodpovědět správně, aby u zkoušky uspěl, a také zjistí, jaké případné omezení má tato zkouška. Samotné otázky se zobrazují po jedné na stránce a vždy je možnost mezi otázkami libovolně přecházet a vracet se na již zodpovězené. Po zodpovězení všech otázek a potvrzení ukončení zkoušky, se zkouška automaticky vyhodnotí a student získá svůj výsledek včetně informace, jestli zkouškou prošel a jaký je jeho výsledný čas. Pokud učitel povolí možnost náhledu do výsledků zkoušky, může se student podívat na statistiku svých otázek a zjistit, kde případně udělal chyby. V případě, že student nedosáhne požadované hranice bodů, může zkoušku znovu opakovat.

Celkové zhodnocení Podle kritérií definovaných v kapitole 2.1 bylo provedeno celkové zhodnocení webu Exam Professor a jeho výsledky jsou znázorněny v tabulce 2.6.

Uživatelské role	3
Výukové oblasti	2
Typy otázek	1
Úložiště otázek	2
Gamifikace a motivace	1

Tabulka 2.6: Celkové zhodnocení portálu Exam Professor

2.1.1.2 ProProfs Quiz Maker

Webový portál ProProfs nabízí spoustu nástrojů pro online vzdělávání, mezi nimiž je i Quiz Maker (`www.proprofs.com/quiz-school`). Díky široké sadě svých funkcí Quiz Maker umožňuje jednoduše vytvářet různé typy testů a kvízů podle potřeb uživatelů. Quiz Maker mohou používat firmy pro vzdělávání anebo i obyčejní uživatelé k ověření svých znalostí. Uživatelé zde mohou přebírat roli učitele neboli zadavatele testu a studenta, který test absolvuje.

Pohled učitele

Učitel má vlastní rozhraní sloužící k vytváření testů, ve kterém může spravovat jednotlivé otázky a testy a nahlížet do statistik.

Vytváření otázek

Učitel má na výběr mezi několika druhy otázek, které buď může sám vytvořit, nebo použit již existující otázky od jiných učitelů. Otázky lze vyhledávat podle kategorií nebo klíčových slov. Otázky a odpovědi mohou obsahovat multimediální soubory, které učitel nahraje z vlastních zdrojů nebo přidá pomocí odkazu.

Import otázek

Kromě postupného vytváření otázek jedné po druhé má učitel možnost importovat otázky. Otázky mohou být do testu naimportovány z jednoho ze svých dalších testů, z jiného veřejného testu anebo z Microsoft Excel souboru.

Pořadí otázek

Učitel může libovolně měnit pořadí otázek v testu. Může je setřídit podle abecedy anebo vytvořit vlastní pořadí pomocí technologie drag and drop.

Náhled testu

Po vytvoření testu je možné zobrazit náhled testu, který je totožný se zobrazením testu pro studenta a je možné vyzkoušet si absolvování testu.

Správa a zveřejnění testů

Po vytvoření a aktivaci testu se test zobrazí na seznamu testů učitele. Učitel má poté možnost test zveřejnit na sociálních sítích nebo ho přidat do vlastních stránek. Test lze vytisknout pro prezenční absolvování nebo učitel může zaslat studentům pozvánku na test prostřednictvím emailu.

Přehledy a statistiky

Ke každému testu má učitel k dispozici přehledy a statistiky. V přehledech se ukazují informace o jednotlivých studentech, kteří testem prošli, jejich výsledcích a času, který potřebovali k absolvování testu. V sekci statistik má učitel přehled o průběhu celého testu, jeho úspěšnosti, počtu absolvování apod. Všechna data se dají zobrazit i v podobě různých grafů, což dělá statistiky ještě přehlednější.

Nastavení

Po vytvoření testu jsou k dispozici různá nastavení. V této sekci učitel může definovat omezení pro test, určovat vzhled testu a otázek, zadávat jazyk, nastavovat privátnost či zadávat cenu testu.

Způsob, jakým se bude test vyhodnocovat, určuje také jeho tvůrce. Celkové bodové ohodnocení testu je prováděno vždy po zodpovězení všech otázek. Zadavatel může určit, zda se samotné otázky vyhodnotí hned po zodpovězení nebo se uživateli zobrazí přehled všech zodpovězených otázek na konci testu spolu s bodovým ohodnocením. Výsledky testu jsou zobrazené v podobě certifikátu, který obsahuje bodové a procentuální ohodnocení. Zadavatel testu může povolit uživatelskou kontrolu testu. V tomto případě má uživatel přehled o počtu správných a chybných odpovědí, stráveném času, počtu bodů potřebných k úspěšnému splnění testu a rekapitulaci otázek. Uživatelé mohou testy opakovat, pokud učitel tuto možnost povolil v nastavení testu.

Pohled studenta

Student si může volit testy podle různých kritérií. Nabízí se výběr podle kategorií, popularity, jazyka testu nebo lze testy seřadit podle času jejich publikování. Před tím, než uživatel spustí test, si může zvolit, jakým způsobem chce zobrazovat hodnocení anebo kolik otázek chce vyzkoušet. Vše za podmínky, pokud jsou tyto nastavení povolené tvůrcem testu. U některých testů se spolu s krátkým popisem také zobrazí náhled některých otázek. Po odeslání testu se uživatelům zobrazí vyhodnocení testu a dále mohou postupovat podle akcí, které jsou u testu povolené – například opakování testu.

Druhy otázek

Více možností odpovědí

Uživateli je nabídnuta možnost výběru z více odpovědí, kde jedna nebo více odpovědi jsou správné.

Pravda/Nepravda
Uživatel odpoví na otázku pouze ano nebo ne.

Doplňování
Uživatelé musí sami doplnit chybějící slovo. Není nabízen výběr z více možností. Odpověď studenta se musí přesně souhlasit s odpovědí, kterou uvedl učitel při vytváření testu.

Esej
Uživatel napíše krátkou esej na téma, které zadal učitel. Esej může být omezená na počet slov a vyhodnocuje se učitelem samostatně.

Přiřazování
Uživatel musí správně přiradit k sobě nabízená slova podle zadání.

Celkové zhodnocení
Tabulka 2.7 znázorňuje celkové zhodnocení portálu ProProf Quiz Maker podle kritérií stanovených v kapitole 2.1.

Uživatelské role	3
Výukové oblasti	2
Typy otázek	2
Úložiště otázek	2
Gamifikace a motivace	1

Tabulka 2.7: Celkové zhodnocení portálu ProProf Quiz Maker

2.1.2 Český trh

Z českého trhu, stejně jako ze zahraničního, byly vybrány dva portály. Oba portály se zabývají výukou jazyku a nabízí svým uživatelům kromě testů, také i vzdělávací kurzy.

2.1.2.1 Online jazyky

Online Jazyky je portál pro online výuku cizích jazyků, který je dostupný na adrese `www.onlinejazyky.cz`. Portál nabízí možnost výběru hned z několika jazyků a každý jazyk se vyučuje na více úrovních rozdělených podle složitosti. Online Jazyky převážně nabízejí online kurzy, které jsou zpoplatněny a jsou poskytovány jak pro jednotlivce, tak i pro školy nebo firmy. Uživatel odebírá kurzy a testy jako hotový produkt a nemá možnost vytvářet vlastní.

Uživatelé se na webu Online jazyky setkávají se dvěma druhy testů: vstupní test a test jako součást kurzu.

Vstupní test

Před zakoupením a zahájením kurzu je uživateli nabízen vstupní jazykový test, jehož cílem je pomoct uživateli ohodnotit své dosažené znalosti a podle výsledku si vybrat kurz na vhodné úrovni. Tento test se skládá ze šesti částí, kde každá část odpovídá určité jazykové úrovni. Uživatel projde testy od úrovně A1 do úrovně C2 a ve výsledku získá informace o své úrovni v určitém jazyce a také dostane doporučení na konkrétní jazykový kurz. Tento test je generován z náhodných otázek a je možné ho libovolně opakovat k ověření dosažených znalostí.

Vstupní test je možné přerušit a vyhodnotit kdykoliv v jeho průběhu. Po vyhodnocení uživatel dostane celkový výsledek spolu se statistikou pro jednotlivé části testu. Podle výsledku je doporučen vhodný jazykový kurz.

Test jako součást kurzu

Každý kurz se skládá z několika částí – lekcí a po každé lekci následuje test, který je určen pro procvičení látky probírané v kurzu. Otázky v těchto testech jsou rozmanité a interaktivní, čímž umožnují uživateli splňovat testy formou her. Na konci kurzu je uživateli předložen velký závěrečný test, který obsahuje otázky ze všech procvičených lekcí. Během celého kurzu doprovázejí uživatele nápovědy a typy, které mu pomáhají se zorientovat.

Test jako součást kurzu nemusí být splněn celý, test je možné kdykoliv přerušit. V tomto případě je testovací cvičení přidáno na seznam úkolu a uživatel má možnost absolvovat jej později. Celý test se vyhodnotí hned po vyplnění všech otázek a uživatel má k dispozici jak procentuální výsledky, tak i podrobný přehled správných a chybných odpovědí. Test je možné opakovat i hned po splnění.

Druhy otázek

V testech na Online jazycích se vyskytují otázky různých druhů od jednodušších pravda/nepravda po složitější otázky typu doplňování. Dále budou popsány všechny druhy otázek zde použitých.

Pravda/Nepravda

Nejjednodušší typ otázek. Na otázku tohoto typu může uživatel odpovědět pouze ano nebo ne.

Více možností odpovědí

Uživateli je předloženo několik možností odpovědí a musí si vybrat jednu z nich, kterou považuje za správnou.

Přiřazení

Uživateli je nabízen seznam obrázků a seznam slov. Uživatel pomocí technologii Drag and Drop potřebuje správně přiřadit slova k obrázkům.

Křížovka

Křížovka je jedním ze zábavních druhů testu. Uživatel má k dispozici seznam slov v češtině, které potřebuje přeložit a vyplnit do křížovky.

Seřazení

Uživatel dostane za úkol seřadit věty v textu nebo v dialogu. Má k dispozici několik vět a pomocí přetahování je umístí ve správném pořadí.

Doplňování

Uživatel má k dispozici odstavec textu, do kterého potřebuje doplnit chybějící slova. U jednodušších otázek je k dispozici seznam slov a uživatel potřebuje vybrat správné slovo a přetáhnout ho na vhodné místo v textu. Složitější otázky tuto možnost nenabízejí a uživatel musí potřebná slova doplnit sám.

Motivační prvky

Výuka na Online Jazycích je velice zajímavá a atraktivní díky využití motivačních prvků a obohacení o elementy zábavy. Během celého výukového procesu uživatele doprovázejí zajímavá videa od tvůrců portálu, které zároveň učí, napovídají a motivují uživatele. Výukový proces tak není nudný a dokáže zaujmout a uživatele udržet.

Celkové zhodnocení

V tabulce 2.8 jsou zobrazeny výsledky celkového zhodnocení výukového portálu Online Jazyky.

Uživatelské role	1
Výukové oblasti	1
Typy otázek	2
Úložiště otázek	1
Gamifikace a motivace	2

Tabulka 2.8: Celkové zhodnocení portálu Online Jazyky

2.1.2.2 LANGMaster

LANGMaster je dalším zástupcem online jazykových škol, který poskytuje svoje služby jak zdarma tak zpoplatněné. Tento portál se nachází na adrese `www.langmaster.cz`. Testy jsou k dispozici již hotové a uživatel nemá možnost zastupovat roli učitele a vytvářet vlastní testy.

Portál LANGMaster je možné rozdělit do dvou částí. Jedna se zabývá poskytováním služeb pro jednotlivce, druhá pro firmy nebo školy.

Jazykové kurzy pro jednotlivce

Pod pojmem kurz jsou na portálu reprezentovány sady testů a cvičení. Kurzy jsou nabízeny zadarmo a je možné si zvolit kurz dle různých úrovní složitosti od začátečníka až po vyšší pokročilého. Každý kurz obsahuje několik kapitol, které se dále dělí na lekce. Každá lekce se skládá z jednotlivých skupin cvičení nebo testů. Tyto testy potom slouží k procvičení určitého tématu.

Otázky v testu se vyhodnotí na přání uživatele. Uživatel má možnost otázku přeskočit a vrátit se k ní později anebo ihned po vyplnění ji vyhodnotit a přejít k další otázce. Při vyhodnocení se uživateli zobrazí procentuální výsledek a označí se správné a chybné odpovědi. Test je možné znovu opakovat. Po absolvování všech testů v lekci dostane uživatel celkové vyhodnocení, které obsahuje přehled o jednotlivých testech, celkovém hodnocení a času, které vyplnění testů trvalo.

Testovací platforma pro firmy a školy

LANGMaster nabízí online testovací platformu eTestMe.com (`www.langmaster.cz/lmcom/com/web/cs-cz/pages/companies/eTestMe.aspx`) k otestování znalosti studentů škol nebo zaměstnanců firmy. Pro použití platformy stačí mít připojení k Internetu a nainstalované prostředí Silverlight od společnosti Microsoft, na kterém platforma běží. Tato služba avšak není dostupná zdarma a pro její plné využití je potřeba zakoupit si licenci. Vyzkoušet platformu je možné pomocí demo verze zdarma.

Uživatel zde může přebírat několik rolí: testovaný účastník, firma/správce a tutor. V ostrém provozu jsou uživatelé rozděleni podle uživatelských rolí a každý využívá rozhraní aplikace pro svou roli. V demo verzi je možné mezi jednotlivými rolemi přepínat.

Testovaný účastník

Testovaný účastník je uživatel, který chce test vyplnit. Může to být student ve škole nebo zaměstnanec firmy. Test může absolvovat na základě pozvánky, kterou dostane od firmy/správce. Po úspěšném ukončení testu testovaný účastník dostane certifikát s výsledkem testu.

Firma/Správce

Správce má na starosti správu uživatelů a testů. Přidává do systému nové uživatele a zadává jim testy. Při zadávání testů vybírá druh testu a jeho složitost. Až správce prozkoumá výsledky testu konkrétního testovaného účastníka, může rozhodnout, co je potřeba provést dále a jaký jazykový kurz účastníkovi doporučit.

Tutor

Práce tutora je kontrolovat úkoly testovaného účastníka. Kontroluje pouze slovní úkoly a eseje. Výsledky pak zadává do systému a předává je tak pro další zpracování správcům.

Samotný test probíhá v několika krocích. Testovaný účastník je postupně vyzkoušen z gramatiky, čtení, poslechu, mluvení a psaní. Když uživatel vyplňuje test v prostředí testovací platformy, nemá možnost si průběžně zobrazovat výsledky. Musí postupně projít každou sekcí testu a až na konci dostane celkový přehled. Celkový výsledek obsahuje procentuální bodové hodnocení každé sekce a také čas, který uživatel strávil zodpovězením otázek. U otázek typu mluvení a psaní je potřeba počkat, než je opraví tutor a zadá hodnocení do systému. U tohoto druhu testu uživatel nemá přehled o tom, jaké otázky vyplnil chybně a případně jaká na ně byla správná odpověď.

Druhy otázek a odpovědí

Otázky jsou v testech LANGMasteru velice rozmanité a jeden druh otázky má více podob a vzhledů. Dále jsou popsány základní typy otázek, které se v testech vyskytují.

Doplňování

Uživatel doplní do prázdných políček správná slova, písmenka nebo čísla. Doplňování je možné na základě odposlechnuté nahrávky, textu nebo obrázku. U složitějších otázek, kde je potřeba rozsáhlejší odpověď, uživatel nevyplňuje políčko sám, ale vybere si z nabízených variant, které již obsahují potřebný text.

Pravda/Nepravda

Uživatel odpoví na otázku pouze ano nebo ne. Otázky mohou být jednoduché nebo jsou postaveny na poslechu nahrávky nebo přečtení textu. V rozšířené verzi otázky musí student na základě poslechu určit, zda je zadaná věta v souladu s obsahem nahrávky.

Více možností odpovědí

Uživatel vybere jednu správnou odpověď z několika nabízených. Odpovědi mohou být jak textové tak i například ve formě obrázku.

Seřazení

Uživatel potřebuje seřadit několik vět ve správném pořadí, aby dohromady vytvořily smysluplný odstavec.

Přiřazování

Uživatel muší správně přiřadit k sobě různé částí úkolu (například zvukovou nahrávku a její překlad). V rozšířené verzi otázky uživatel potřebuje seskupit více slov do definovaných skupin.

Křížovka

Uživatel odpovídá na otázky a odpovědi doplňuje do křížovky.

Otázky s psanou odpovědí

Uživatel potřebuje odpovědět na otázku jednou větou, kterou sám zadá do políčka pro odpověď. Odpověď musí být napsána podle zadané šablony. Odpověď studenta musí přesně odpovídat šabloně, jinak je uznána za chybnou, i když obsahově byla správná (například He's a He is se vyhodnotí odlišně).

Celkové zhodnocení

Výsledky zhodnocení portálu LANGMaster jsou uvedené v tabulce 2.9. Hodnocení bylo prováděno podle kritérií definovaných v kapitole 2.1.

Uživatelské role	2
Výukové oblasti	1
Typy otázek	2
Úložiště otázek	1
Gamifikace a motivace	1

Tabulka 2.9: Celkové zhodnocení portálu LANGMaster

2.1.3 Moodle

Moodle (`moodle.org`) je nejznámější výuková platforma, která se používá po celém světě a umožňuje vytvářet a studovat výukové materiály mnoha druhů. Moodle je OpenSource projekt, který je dostupný zdarma pod GNU General Public License pro komerční a nekomerční použití a koncový uživatelé mohou nejen využívat jeho funkcionalit, ale také přidávat vlastní nebo modifikovat existující. Moodle je dostupný v mnoha světových jazycích a nad jeho správou pracuje mnoho vývojářů, kteří původní systém obohacují o další funkcionality.

Aby uživatel mohl Moodle využívat, potřebuje ho stáhnout, nainstalovat a zprovoznit na vlastním serveru. Uživatelé mají k dispozici dokumentaci na oficiálních stránkách aplikace. Po zprovoznění aplikace na serveru je dostupná i pro ostatní uživatele (například studenty) prostřednictvím Internetu.

Moodle nabízí pro své uživatele více rolí. Kromě klasických rolí studenta a učitele, zde může uživatel přebírat roli tvůrce kurzů, manažera, administrátora, učitele bez práva editace nebo hosta. Tak například k výukovým materiálům má přístup více uživatelů a každý má nastavená svoje práva podle role.

Hlavním výukovým produktem je v Moodlu Kurz. Do kurzu lze přidávat různé studijní materiály a jejich škála je opravdu široká. Mezi tyto materiály patří i testy. Tato diplomová práce se zabývá testy a otázkami, proto bude analyzována převážně testovací část projektu Moodle.

Testy v rámci systému Moodle tvoří součást kurzu. Při vytváření testu jsou učiteli k dispozici různá nastavení tykající se například omezení testu, jeho vzhledu nebo známkování. Nastavení je mnoho a není jednoduché se v

nich orientovat. K většině bodů je avšak poskytnuta nápověda, která v případě potřeby poradí učiteli, jak má postupovat. Po vytvoření základních informací k testu má učitel možnost přejít k přidávání samotných testovacích otázek. Učitel může buď vytvořit nové otázky anebo přidat již vytvořené úlohy z banky úloh. Otázky se přidávají po jedné, systém se po každé zeptá učitele, kterou úlohu chce přidat a poté nabídne příslušný formulář pro její vytváření. Při vytváření nových úloh má učitel možnost výběru z následujících druhů otázek.

Dlouhá tvořená odpověď

Dlouhá tvořená odpověď je podobná eseji. Na otázku je možné zodpovědět více větami a otázka se vyhodnotí učitelem ručně.

Doplňovací úloha

Podstatou doplňovací úlohy je, že do prázdných políček v textu se doplní vhodná slova čí čísla. Moodle umožňuje několik možností doplňování. Doplnit se může buď slovo, nebo číslo anebo je tu možnost vybrat si z více variant, které jsou předem definovány. Tento typ úlohy je složitý na vytváření a vyžaduje znalost syntaxe speciálního editoru.

Krátká tvořená odpověď

Na otázku s krátkou tvořenou odpovědí je možné odpovědět jedním nebo několika slovy. Učitel může zadat více možností správné odpovědí, pokud není odpověď jednoznačná.

Numerická úloha

V případě numerické úlohy je odpověď tvořena číslem. Při vytváření této úlohy je možné zadat jednotky a toleranci.

Pravda/Nepravda

Úloha typu Pravda/Nepravda nabízí dvě odpovědi (ano nebo ne), ze kterých je jedna správná.

Přiřazování

Přiřazovací úloha je tvořena seznamem otázek a seznamem odpovědí, které je potřeba k sobě přiřadit. Odpověď nemusí být vždy přiřazená a otázka nemusí mít přiřazovací par.

Přiřazování z krátkých odpovědí

Tento druh úlohy je podoben přiřazování, avšak seznam otázek a odpovědí se generuje náhodně z již existujících otázek s krátkou tvořenou odpovědí. Při vytvoření této otázky je potřeba vybrat kategorii otázek a systém musí zkontrolovat, jestli daná kategorie obsahuje dostatečný počet otázek s krátkou tvořenou odpovědí.

Výběr z možných odpovědí

Tento druh úlohy je klasická testová otázka, ve které má uživatel k dispozici seznam odpovědí a potřebuje z nich vybrat jednu nebo více možností.

Jednoduchá vypočítávaná úloha a vypočítávaná úloha

Vypočítávaná úloha patří mezi numerické úlohy, ale nejsou zde konkrétní výsledky. Vypočítávané úlohy se automaticky generují systémem podle vzorce zadaného učitelem, což vede k tomu, že každý student získá jiné zadání a tím pádem má správnou jinou odpověď. Z pohledu učitele je vytváření úlohy ztíženo o nutnost znalosti práce s vzorci a jejích zadávání přes editor.

Vypočítávaná úloha s více možnostmi

Vypočítávaná úloha s více možnostmi je založená na stejném principu jako vypočítávaná úloha, ale je obohacena o více možností odpovědí. Odpovědí se opět vypočítají systémem podle určitého vzorce.

U všech otázek zadává učitel název, počet bodů, určitá omezení a nápovědy pro studenta. Stejně jako při zadání testu je více možností nastavení, které nejsou vždy úplně jasné.

Banka úloh

Učitel má k dispozici banku úloh, do které může přidávat testovací úlohy pro následné použití. Otázky jsou setříděné do kategorií a lze je mezi kategoriemi přesouvat. Otázky je možné editovat nebo duplikovat. U každé otázky je vedena evidence, kdo naposledy editoval otázku. Učitel má také možnost zobrazit otázku tak, jak ji vidí student. Otázky, které se nachází v bance úloh, jsou dostupné pro další použití a učitel je může přidávat do svých testů.

Celkové zhodnocení

Po analýze systému Moodle bylo provedeno jeho zhodnocení a sestaven seznam výhod a nevýhod. V úvahu se braly jak obecné parametry, tykající se celého systému, tak i parametry, podle nichž se hodnotil testovací modul. Tabulka 2.10 znázorňuje seznam výhod a nevýhod.

Celkově lze říct, že Moodle je rozsáhlý výukový systém, který má promyšleno mnoho funkcionalit tykajících se vytváření a správy výukových materiálů. Díky podpoře více jazyků a dostupnosti systému zadarmo, je Moodle hodně rozšířen po celém světě a používá se jak ve školách, tak i ve firmách. Moodle má však jednu velkou nevýhodu a tou je jeho uživatelské rozhraní a složitost. Moodle poskytuje příliš mnoho funkcionalit, které obyčejný uživatel nepotřebuje a tak může být při použití systému zmatený. Je těžké navrhnout uživatelské rozhrání pro takový rozsah funkcionalit. I přes to, že se rozhraní mění a zlepšuje se v novějších verzích systému, stále není dokonalé a obsahuje hodně chyb.

Z těchto důvodu bylo rozhodnuto nevyužít systém Moodle a vyvinout vlastní aplikaci, která bude jednodušší a zároveň více přívětivá a intuitivní pro uživatele.

2.1.4 Shrnutí

Existující řešení popsané v kapitolách 2.1.1.1 - 2.1.2.2 nejsou jediné dostupné na trhu. Existuje řada dalších hůře čí lépe provedených aplikací a systémů.

V případě E-learningu nelze jednoznačně zvolit nejlepší řešení, protože každá aplikace může být zaměřená na něco jiného. Například jedna aplikace poskytuje pouze testy, druhá pouze výukové kurzy a třetí může být kombinaci obou. Některé aplikace jsou více zaměřené na kvalitu obsahu, jiné například na gamifikaci. Na základě provedené rešerše vznikly určité funkční a nefunkční požadavky, které jsou popsány v kapitole 2.3. Po provedení analýzy českého trhu bylo zjištěno, že trh nenabízí mnoho kvalitních výukových portálů, které by poskytovaly různé výukové materiály ve více kategoriích. Z analýzy vyplývá, že aplikací nabízejících možnost tvoření vlastního výukového obsahu je nedostatek. Projekt Mentica počítá s touto možností, což by mohla být v budoucnu jeho výhoda. Pokud jde o vytváření a správu otázek, jako dobrá volba se jeví zavedení otázkového úložiště, které by umožnilo lepší evidenci otázek a jejích znovu použitelnost ve více testech. K požadavkům definovaným v zadání bylo rozhodnuto přidat založení banky otázek, ve které je možné vytvářet a spravovat otázky nezávisle na testech. Otázky v bance otázek budou seskupeny do složek a správu složek bude mít na starosti sám mentor.

2.1.4.1 Výběr typů otázek

Součástí výsledku analýzy existujících řešení je přehled nejčastěji používaných druhů otázek. Přehled otázek je seřazen podle počtů výskytů v konkurenčních řešeních v tabulce 2.11.

Z tabulky 2.11 vyplývá, že nejčastěji se používají klasické otázky typu pravda/nepravda a otázky s možností výběru z více odpovědí. Tyto otázky jsou z pohledu studenta nejjednodušší, stačí vybrat jednu z možností, kterou nabízí systém. Zároveň se tento typ otázek velice často používá také u tištěných (písemných) testů a je pro uživatele běžný. Z těchto důvodu bylo rozhodnuto použít otázky těchto typů v modulu Test portálu Mentica.

Druhými nejčastějšími otázkami jsou otázky typu přiřazování a doplňování. Otázka typu přiřazování vyžaduje menší aktivitu studenta, čili student potřebuje pouze správně přiřadit nabízené odpovědi a nemusí vymýšlet vlastní. Otázky typu přiřazování bude modul Test nabízet v různých variantách. Přiřazovací pár může být tvořen jak slovy, tak i obrázky anebo kombinací obou možností.

Otázka typu doplňování vyžaduje od studenta hlubší znalosti, student již nemůže vybírat z nabízených možností a potřebuje vymyslet vlastní odpověď, kterou považuje za správnou. Použitím typu otázky doplňování lze předcházet náhodnému výběru správné odpovědi v případě neznalosti studenta. Otázka typu doplňování bude rovněž použitá v modulu Test, do první verze systému se s ní však nepočítá kvůli složitější implementaci.

Na místo otázek typu doplňování bylo rozhodnuto do první verze přidat otázky s „jednoslovní“ odpovědí, ve kterých potřebuje uživatel ručně zadat odpověď, aniž by měl předdefinované možné odpovědi. Tento druh otázek zahrnuje otázky s numerickou a psanou odpovědí. Odpověď může být číslo, zlo-

mek, datum nebo slovo. Otázky se nevyhodnocují mentorem, ale systémem. Mentor by měl při zadávání otázky zvážit všechny možné správné odpovědi. Například u otázky z dějepisu lze napsat jméno Karel IV jako Karel 4 nebo Karel čtvrtý a všechny tyto možnosti musí být uznány jako správné. Při vytváření otázek s jednoslovnou odpovědí bude mít mentor prostor k zadání dalších možných odpovědí nebo intervalu hodnot v případě numerické odpovědi.

2.2 Uživatelské role

V modulu Test projektu Mentica existuje více uživatelských rolí neboli aktérů. Tyto role slouží k odlišení uživatelů a ke každé roli jsou přiřazeny určité funkcionality. Modul Test dědí většinu uživatelských rolí od celého projektu. Následující kapitoly popisují základní uživatelské role a jejich funkcionality.

Nepřihlášený uživatel

Nepřihlášený uživatel je uživatel, který není v systému registrován nebo není do systému přihlášen pomocí svých přihlašovacích údajů. Základní funkcionality, které se nepřihlášenému uživateli nabízí, jsou registrace nebo přihlášení do systému v případě, že uživatel již má v systému svůj účet. Problém registrace a uživatelských účtu je řešen v modulu Uživatel (viz diplomová práce Bc. Jaroslava Tesaře [1]).

Nepřihlášený uživatel má zcela omezenou funkcionalitu - může pouze prohlížet veřejně dostupné materiály a náhledy testů. Uživatel může vyhledávat testy a získávat k nim úvodní informace, například název, popis, počet bodů apod. Pokud je test nebo kurz veřejný a je dostupný zadarmo, může si uživatel tento produkt vyzkoušet. Pro absolvování privátních a placených testů je avšak potřeba mít vlastní uživatelský účet a být přihlášen do systému.

Přihlášený uživatel

Přihlášeny uživatel má v systému vlastní účet, který získal při registraci, a je do systému přihlášen pomocí svých přihlašovacích údajů. Tento aktér má v systému podstatně více práv a možností než nepřihlášený uživatel. Přihlášený uživatel nabývá dalších dvou podrolí: Mentor a Student. Stejný uživatel může být jak studentem, tak i mentorem zároveň a v závislosti na jeho současné roli se mu přidělí určité funkcionality.

Student

Student je uživatelem, který nepublikuje žádný vlastní obsah, ale čerpá obsah vytvořený mentorem. Student může nahlížet do testů, absolvovat je, hodnotit a komentovat. Za úspěšné splnění testu student může získat ocenění v podobě odznaků nebo zkušenostních bodů. Student má také přehled o všech absolvovaných testech a může nahlížet do vlastních statistik.

Mentor

Pojem mentor v rámci projektu Mentica znamená učitel. Mentor je uživatel, který má možnost vytváření vlastních výukových materiálů a následně

jejich publikaci pro studenty. V případě modulu Test má mentor k dispozici rozhraní pro vytváření testů, jejich publikaci a analýzu výsledků. Každému mentorovi je přidělena banka otázek, kterou může sám spravovat a přidávat do ní vlastní otázky, které může následně použít v testech.

Systém

Samotný systém je také jednou z uživatelských rolí. Jeho účelem je správně reagovat na akce prováděné ostatními aktéry a poskytovat zpětnou vazbu. Systém například může sloužit k opravě testů nebo validaci formuláře.

2.3 Funkční a nefunkční požadavky

Po analýze zadání a provedení rešerše existujících řešení byl sestaven seznam funkčních a nefunkčních požadavků definujících funkcionality, které poskytne výsledný portál a nároky kladené na portál. Po splnění těchto požadavků by měl portál poskytovat uživateli vše potřebné a práce s celým systémem by měla být uživatelsky příjemná.

2.3.1 Funkční požadavky

Funkční požadavky definují co má systém umožňovat a vztahují se k jeho funkcionalitám. Požadavky jsou vytvořené na základě analýzy zadání práce a rešerši existujících řešení. Seznam požadavků je rozdělen do dvou skupin, kde první skupina zahrnuje požadavky z pohledu mentora a druhá skupina pokrývá požadavky z pohledu studenta.

2.3.1.1 Pohled mentora – F1

F1.1. Založení, správa a mazání složek v bance otázek

Mentor má k dispozici banku otázek, ve které si může zakládat nové složky, do nichž bude následně přidávat svoje otázky. Struktura práce se složkami je podobná práci se složkami v počítači. Při vytvoření složky stačí zadat pouze její jméno. Každá složka může mít neomezený počet podsložek a může obsahovat jak podsložky, tak i otázky. Mentor není nucen přiřazovat složky do kategorií. Pojmenování a struktura složek jsou pouze záležitostí mentora, který je může uspořádat podle vlastních potřeb. Složky je možné editovat a mazat. Při smazání složky se smaže veškerý její obsah včetně podsložek a otázek.

F1.2. Vytvoření, editace a mazání otázky

Mentor může vytvářet otázky různých druhů v rámci testů nebo složek. Při zadávání otázky mentor potřebuje vyplnit její název, který bude sloužit pro identifikaci otázky, zadání, případnou nápovědu a počet bodů. Ke každé otázce dále musí zadat odpovědi a definovat, které jsou správné. K otázce je možné přidat multimediální soubory, které mohou patřit jak k zadání otázky, tak i k obsahu odpovědí. Mentor může vytvořenou otázku následně editovat

nebo ji může smazat. Pokud je otázka přidaná do nějakého testu, tak se při její editaci nebo smazání přepočítá celkový počet bodů za test.

F1.3. Verzování otázky

Pokaždé když mentor změní otázku, systém nabídne možnost aplikovat změny rovnou na otázku, kterou edituje, nebo vytvoří novou otázku. Pokud se vytvoří nová verze, je považována za nejaktuálnější a zobrazuje se v přehledu otázek jako první. Mentor má také přehled o předchozích verzích otázky, může je editovat, mazat nebo nastavit dřívější verzi jako aktuální.

F1.4. Vytvoření, správa a mazání testu

Mentor má možnost vytvářet testy k ověření znalostí studenta. Test může být jako samostatný produkt, nebo může být součástí kurzu. U testu se zadávají různé parametry potřebné pro absolvování testu, například pořadí otázek, časové omezení nebo počet bodů nutných k úspěšnému absolvování. Testy lze editovat a mazat. Dokud není test publikován, je možné měnit všechny informace, které se testu tykají, včetně otázek a bodování. Jakmile je test zveřejněn, jeho editace se omezí. Editace testu znamená změnu úvodních informací k testu, například popisu nebo omezení kladených na test. Je také možné editovat otázky v testu a měnit bodové hodnocení. Při mazání testu se smažou i otázky, které do testu patří.

F1.5. Zobrazení náhledu testu z pohledu studenta

Mentor si může zobrazit test tak, jak ho uvidí student. Tato možnost pomůže mentorovi sestavit kvalitnější a srozumitelnější test, který pak bude lépe ohodnocen studenty.

F1.6. Verzování testu

Když je test publikován a mentor chce provést změny, které se tykají vyplnění testu, je potřeba vytvořit novou verzi tohoto testu kvůli předcházení konfliktů na straně studenta. Nová verze je kopií původního testu, avšak jeví se jako nový test, který ještě nebyl publikován, a proto je možné měnit jeho obsah.

F1.7. Publikace testu

Po provedení všech úprav a přidání otázek do testu, má mentor možnost test zveřejnit neboli publikovat. Test není dostupný pro studenta, dokud není mentorem publikován. Po zveřejnění testu, je jeho obsah zablokován pro editaci.

F1.8. Automatické generování testů

Mentor má možnost vygenerovat test z náhodných otázek pomocí nastavení různých kritérií pro generování, například počet otázek a počet bodů za test. Otázky je možné použít z banky otázek a také je možné vygenerovat více odlišných variant stejného testů.

F1.9. Vložení otázek do testu
Do testu je možné přidávat jak nové otázky, tak i již existující, které má mentor ve své bance otázek.

F1.10. Přesouvání otázek mezi složkami
Mentor může přesunout otázky z jedné složky do druhé. Při této akci se otázky smažou z první složky a přibudou do druhé složky.

F1.11. Kopírování otázek mezi složkami
Mentor může zkopírovat otázky z jedné složky do druhé. Při této akci se původní otázky zachovají v první složce a jejich kopie se přidá do druhé složky.

2.3.1.2 Pohled studenta – F2

F2.1. Vyhledávání testu
Student může vyhledávat testy podle názvů, mentorů nebo kategorií. Z výsledných testů si potom vybere ten, který ho nejvíce zajímá a vyhovuje mu obsahem.

F2.2. Zobrazení úvodních informací k testu
Před zahájením testu má student možnost prohlédnout si základní informace o testu. Student se dozví, jestli má test určité omezení, kolik obsahuje otázek a kolika bodů je potřeba dosáhnout pro splnění testu. Student si může zobrazit i statistiky jiných uživatelů a dozví například jakou má test úspěšnost a hodnocení.

F2.3. Vyplnění testu
Student postupně prochází otázkami v testu a zodpovídá je. K otázkám se může později vracet a měnit své odpovědi. Pokud je test časově omezen, vyhodnotí se hned po vypršení časového limitu a student již nemůže zodpovědět zbylé otázky.

F2.4. Zobrazení výsledků testu
Po skončení a vyhodnocení testu je studentovi nabídnut náhled jeho výsledků a informace o jeho úspěšnosti. Výsledky obsahuje informace o počtu bodů a počtu správně zodpovězených otázek. Pokud mentor povolí náhled celého testu, může se student podívat, ve kterých otázkách chyboval.

F2.5. Opakování testu
Pokud mentor umožní opakování testu, student může test opakovat. Počet pokusů závisí na nastavení testu, které definuje mentor.

2.3.2 Nefunkční požadavky

Nefunkční požadavky definují, jak systém bude splňovat požadované funkcionality a popisuje nároky a omezení kladené na samotný systém. Tyto požadavky vycházejí ze zadání a zahrnují technické možnosti a omezení.

N1. Dostupnost aplikace ve všech prohlížečích včetně mobilních verzi

Aplikace má bezchybně fungovat a poskytovat stejné rozhraní alespoň pro poslední verze všech současných internetových prohlížečů (Chrome 24+, Safari 5.1+, Firefox 18+, Opera 12.1+, IE 9+). Aplikace musí být přístupná i prostřednictvím mobilních zařízení (Android 2.1+, iOS 3.2+) a musí poskytovat alespoň základní funkcionality, které neomezí práci s celým systémem.

N2. Responzivní design

Webové rozhraní aplikace bude přizpůsobeno jak desktopovým aplikacím, tak i mobilním zařízením.

N3. Omezení vlivu výpadku Internetového spojení nebo pád OS.

Pokud na straně uživatele nastanou technické problémy nebo problémy s připojením k Internetu, uživatel by neměl přijít o rozdělanou práci. Mentor například nepřijde o již vytvořené otázky.

N4. Minimalizace zátěže serveru

Větší část logiky bude běžet na aplikačním serveru, avšak část funkcionalit, která může být vyřešena na straně klientského zařízení, na něj bude přenesena.

N5. Použití PHP frameworku Symfony 2 pro serverovou část aplikace

Aplikace bude napsána v jazyku PHP s využitím frameworku Symfony 2 a ORM rozšíření Doctrine 2. Výběr frameworku vychází z diplomové práce Bc. Jaroslava Tesaře [1].

N6. Použití SASS CSS rozšíření

Front end aplikace bude vyřešen za pomoci SASS CSS rozšíření, které zjednoduší a zpřehlední práci s kaskádovými styly.

N7. Databáze MySQL

Data budou uložena v databázovém systému MySQL. Výběr databázového systému je popsán v diplomové práci Bc. Jiřího Matějky [2].

Výhody	Nevýhody
Vícejazyčnost – moodle je nabízen ve více jazycích, mezi nimiž je i čeština	Nutnost instalace – moodle je potřeba instalovat, takže je nutné mít k dispozici aplikační a databázový server a mít zkušenosti s provozem webových aplikací
Dostupnost – moodle je nabízen zadarmo jako OpenSource	Složité uživatelské rozhraní – uživatelské rozhraní moodlu není zcela intuitivní, uživatel často neví jak se dostat k určitým krokům
Dokumentace – moodle obsahuje rozsáhlou a podrobnou dokumentaci včetně nápověd k prováděným akcím uvnitř kurzu	Mnoho funkcionalit „navíc“ – při vytváření testů nebo otázek je poskytnuto mnoho nastavení, které uživatel může provádět, avšak jejich velká část může být nevyužitá
Nabídka výukových materiálů – moodle poskytuje širokou nabídku výukových materiálů	Nutnost prostudování dokumentace – při vytváření otázek se učitel často musí obracet na dokumentaci, kde je popsána speciální syntaxe, kterou klasický editor nenabízí
Nabídka druhů otázek – moodle nabízí mnoho druhů testovacích úloh od těch nejjednodušších k nejtěžším	
Banka úloh – moodle nabízí banku testovacích úloh pro ukládání otázek, což umožňuje lepší správu otázek a jejích znovupoužitelnost	

Tabulka 2.10: Zhodnocení moodlu

Typ otázky	Počet řešení, ve kterých se vyskytuje
Výběr z více možností	5
Pravda / Nepravda	5
Doplňování	4
Přiřazování	4
Seřazení	2
Křížovka	2
Esej	2
Psaná odpověď	1
Numerická odpověď	1
Vypočítávána odpověď	1

Tabulka 2.11: Nejčastěji používané druhy otázek seřazené podle počtu výskytu

Kapitola 3

Návrh

Tato kapitola se zabývá návrhem výsledné aplikace. Součástí návrhu je sestavení případů užití na základě funkčních požadavků, vytvoření doménového a databázového modelu a také příprava wireframů pro návrh uživatelského rozhraní.

3.1 Případy užití

Případy užití popisují chování aplikace na základě uživatelských akcí. Případy užití zachycují jednotlivé funkční požadavky a jsou rozdělené podle uživatelských rolí. Každý případ užití může zahrnovat další případy užití nebo je může vhodně doplňovat. Tyto vztahy jsou na diagramech označeny jako „include“ a „extend“. [3].

Vzhledem k rozsáhlosti projektu, byly případy užití rozděleny do více skupin. Případy užití jsou znázorněné v následujících kapitolách.

Banka otázek – správa složek

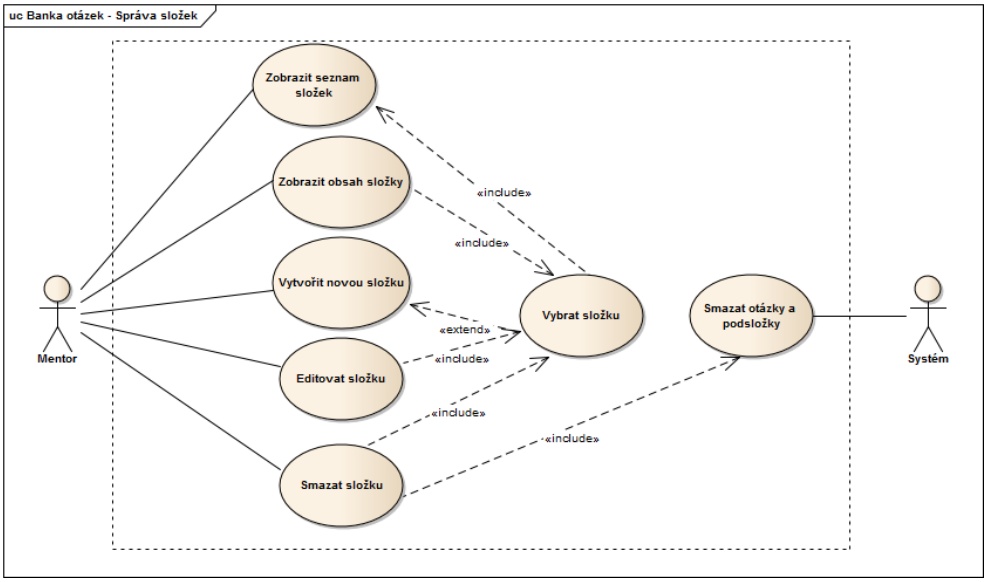
Tato skupina případů užití vychází z funkčního požadavku F1.1. a popisuje práci mentora se složkami v bance otázek. Diagram případů užití je zobrazen na obrázku 3.1.

Banka otázek – správa otázek

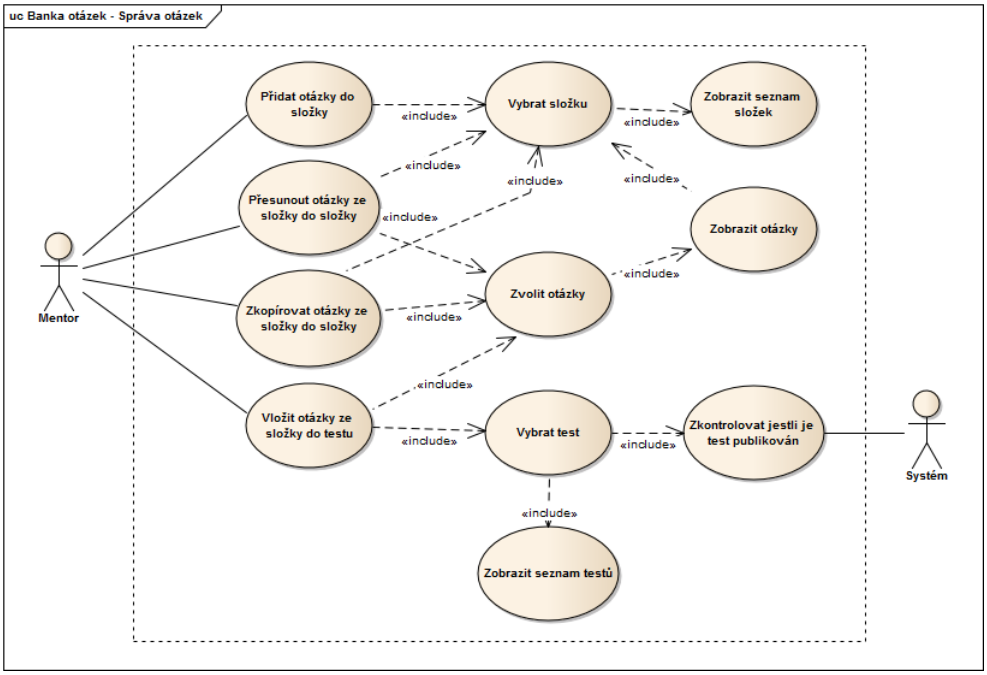
Skupina případů užití znázorňuje práci mentora s otázkami uvnitř složek. Tento případ užití vychází z funkčních požadavků F1.2., F1.10. a F1.11. a je znázorněn na obrázku 3.2.

Zkopírovat otázky ze složky do složky

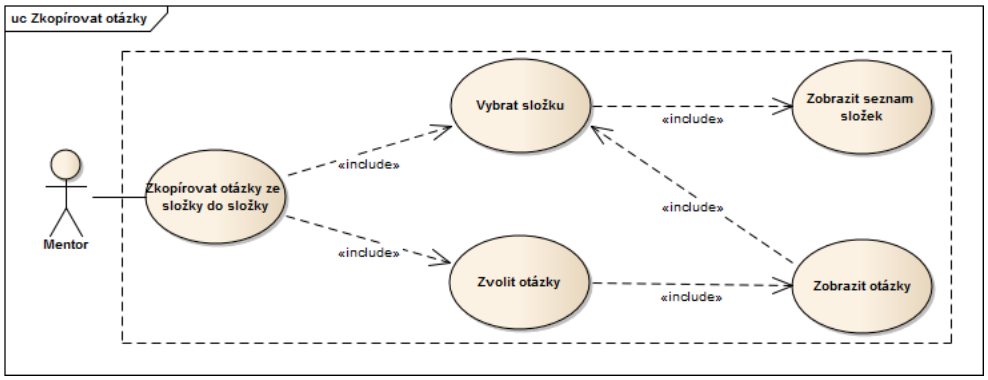
Tento případ užití je součásti skupiny případů užití „Banka otázek – správa otázek“. Případ užití „Zkopírovat otázky ze složky do složky“ slouží k popisu akcí mentora a systému v případě, že mentor chce zkopírovat existující otázky z jedné složky do jiné. Případ užití je znázorněn na obrázku 3.3 a je k němu poskytnout kompletní scénář.



Obrázek 3.1: Případy užití „Správa složek“



Obrázek 3.2: Případy užití „Správa otázek“



Obrázek 3.3: Případ užití „Zkopírovat otázky“

UC Zkopírovat otázky ze složky do složky

Hlavní scénář:

1. Případ užití začíná, když chce uživatel zkopírovat otázky z jedné složky do druhé.
2. INCLUDE **UC Zvolit otázky**.
3. Uživatel zvolí akci kopírování ze seznamu akcí.
4. Systém poskytne seznam složek.
5. Uživatel zvolí cílovou složku – INCLUDE **UC Vybrat složku**.
6. Uživatel potvrdí kopírování kliknutím na tlačítko.
7. Systém provede kopírování ze zdrojové složky do cílové.

Výjimky:

7a. Žádná otázka nebyla zvolena

1. Systém vrací chybovou hlášku a požádá o zvolení otázek.
2. Uživatel se vrátí k bodu 2. Hlavního scénáře.

7b. Cílová složka nebyla vybrána

1. Systém vrací chybovou hlášku a požádá uživatele o zvolení složky.
2. Uživatel se vrátí k bodu 4. Hlavního scénáře.

Další případy užití

UC Zvolit otázky

Hlavní scénář:

1. Případ užití začíná, když chce uživatel zvolit jednu a více otázek k manipulaci.
2. INCLUDE **UC Zobrazit otázky**.
3. Uživatel vybere otázky, se kterými chce následně manipulovat.
4. Systém označí vybrané otázky a nabídne akce k manipulaci.

UC Zobrazit otázky

Hlavní scénář:

1. Případ užití začíná, když chce uživatel zobrazit otázky z určité složky.
2. INCLUDE **UC Vybrat složku**.
3. Uživatel klikne na ikonku zobrazení otázek.
4. Systém poskytne uživateli seznam otázek ze složky.

Výjimky:

4a. Složka neobsahuje žádné otázky

1. Systém poskytne hlášku o tom, že ve vybrané složce nejsou žádné otázky.
2. Uživatel se vrátí k bodu 2. Hlavního scénáře.

UC Vybrat složku

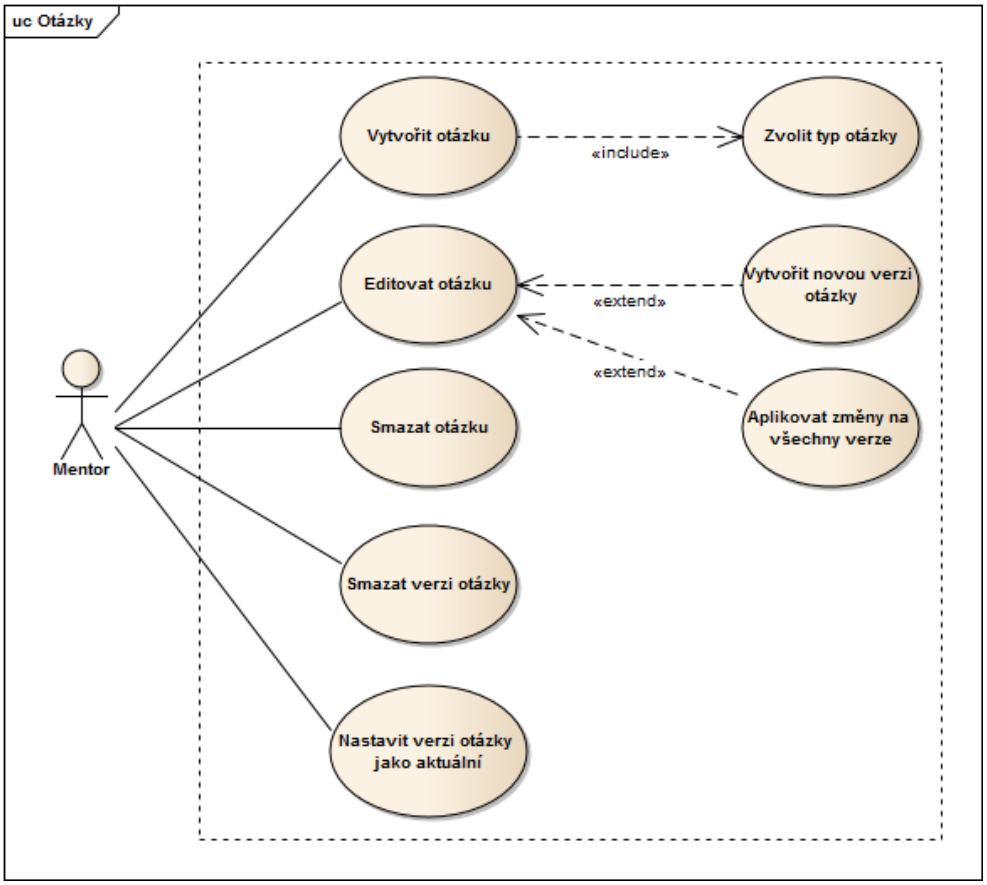
Hlavní scénář:

1. Případ užití začíná, když chce uživatel zobrazit obsah určité složky.
2. INCLUDE **UC Zobrazit seznam složek**.
3. Uživatel klikne na vybranou složku.
4. Systém poskytne uživateli výpis obsahu složky, který tvoří podsložky a otázky.

UC Zobrazit seznam složek

Hlavní scénář:

1. Případ užití začíná, když chce uživatel zobrazit výpis všech svých složek.



Obrázek 3.4: Případy užití ze skupiny „Otázky“

2. Uživatel požádá o výpis svých složek.
3. Systém poskytne uživateli seznam jeho složek.

Otázky

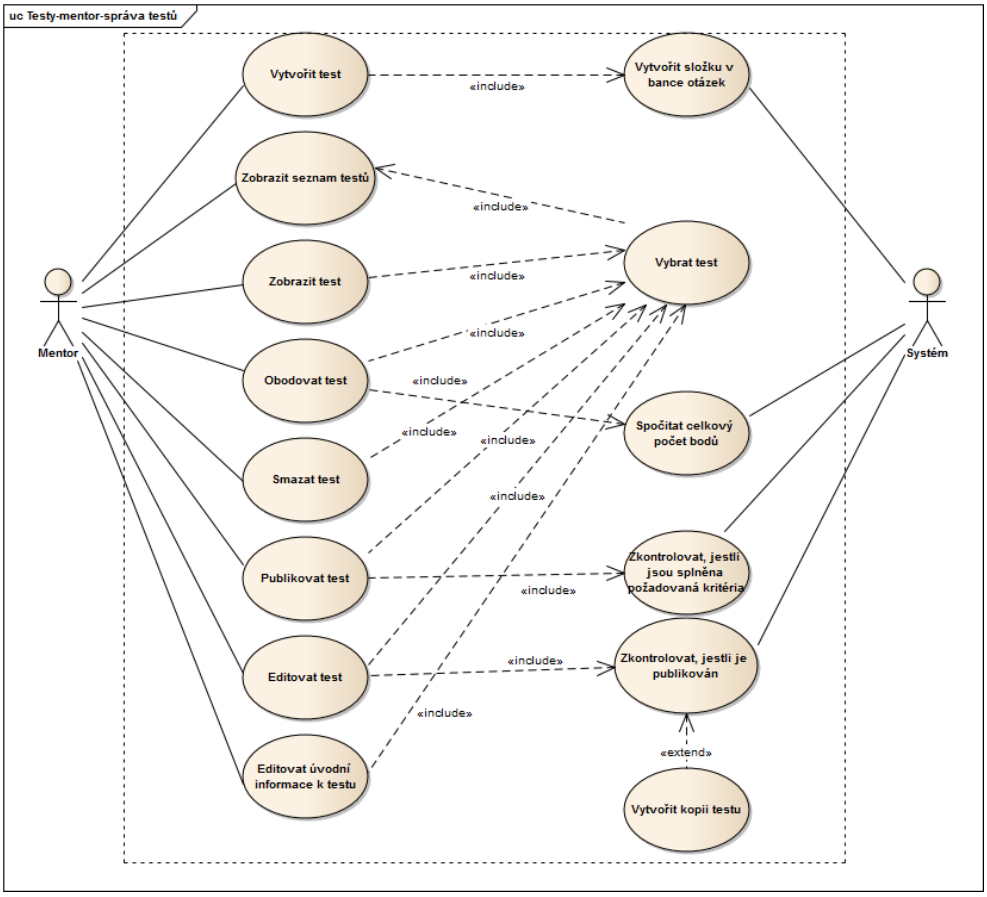
Tato skupina případů užití znázorňuje práci mentora s otázkami a vychází z požadavků F1.2., F1.3. Jejich diagram je vidět na obrázku 3.4.

Testy – pohled mentora – správa testů

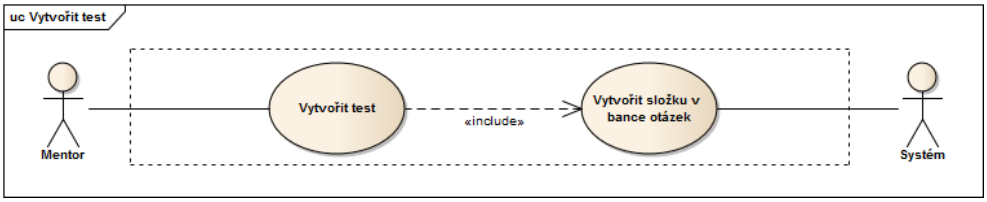
Na obrázku 3.5 je znázorněn diagram případů užití odpovídajících správě testů z pohledu mentora. Tyto případy užití vychází z funkčních požadavků F1.4, F1.6. - F1.8.

Vytvořit test

Případ užití „Vytvořit test“ patří do skupiny případů užití „Mentor-správa testů“. Diagram případu užití znázorňuje akce mentora a reakci systému při vytvoření nového testu. Diagram je zobrazen na obrázku 3.6. Dále je také popsán scénář pro tento případ užití.



Obrázek 3.5: Případy užití ze skupiny „Správa testů“



Obrázek 3.6: Případ užití „Vytvořit test“

UC Vytvořit test

Hlavní scénář:

1. Případ užití začíná, když uživatel chce vytvořit nový test.
2. Systém nabídne uživateli formulář pro zadání nového testu.
3. Uživatel vyplní formulář a odešle ho ke zpracování.
4. Systém zpracuje formulář a uloží test.
5. INCLUDE **UC Vytvořit složku v bance otázek**.

Výjimky:

4a. Některé položky formuláře jsou nevyplněné nebo jsou vyplněné chybně

1. Systém vrací chybovou hlášku a požádá uživatele o opravu.
2. Uživatel se vrátí k bodu 3. Hlavního scénáře.

Alternativní scénář:

Zrušení přidání testu
Uživatel může provedené akce odvolat tlačítkem „Storno“.

Další případy užití

UC Vytvořit složku v bance otázek

Hlavní scénář:

1. Případ užití začíná, když uživatel vytvoří nový test.
2. Systém vygeneruje složku v bance otázek, která bude odpovídat vytvořenému testu a obsahovat otázky z tohoto testu.

Testy – pohled mentora – správa otázek

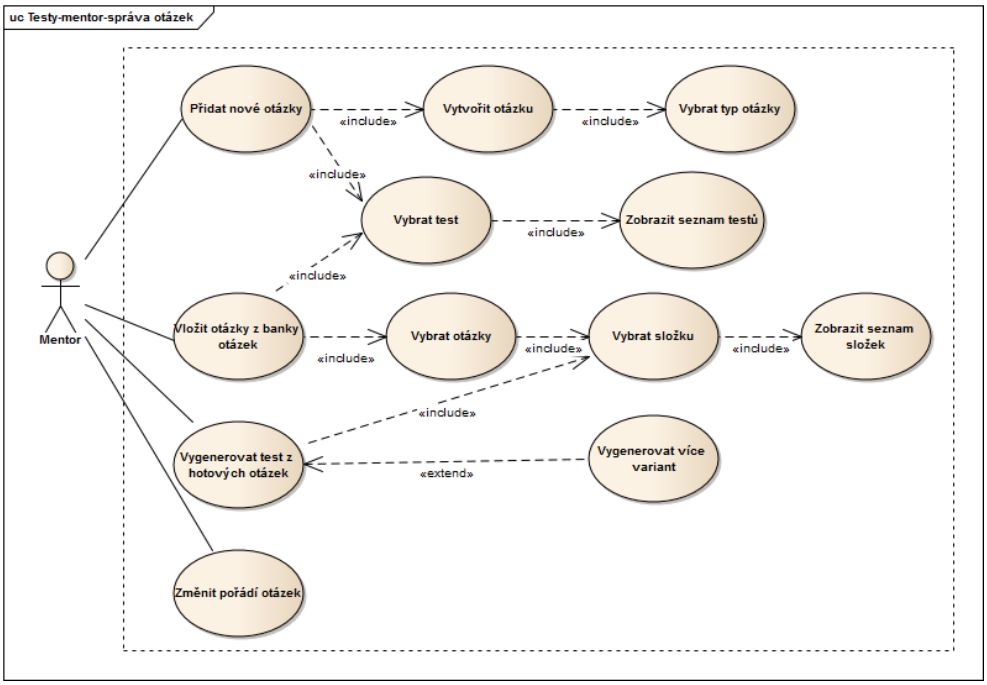
Diagram případu užití ze skupiny „Mentor-správa otázek“ je znázorněn na obrázku 3.7. Tyto případy užití vychází z požadavků F1.2.,F1.9. a popisují chování mentora a systému při přidání a správě testovacích otázek.

Testy – pohled studenta

Diagram případů užití popisující práci s testy z pohledu studenta je znázorněn na obrázku 3.8. Skupina těchto případů užití vychází z funkčního požadavku F2.

Opakovat test

Případ užití „Opakovat test“ spadá do skupiny případů užití „Testy – pohled studenta“ a definuje, jak se chová student a systém v případě, pokud



Obrázek 3.7: Případy užití ze skupiny „Test – správa otázek“

chce student znovu opakovat test. Tento případ užití je znázorněn na obrázku 3.9. Dále je popsán scénář případu užití „Opakovat test“.

UC Opakovat test

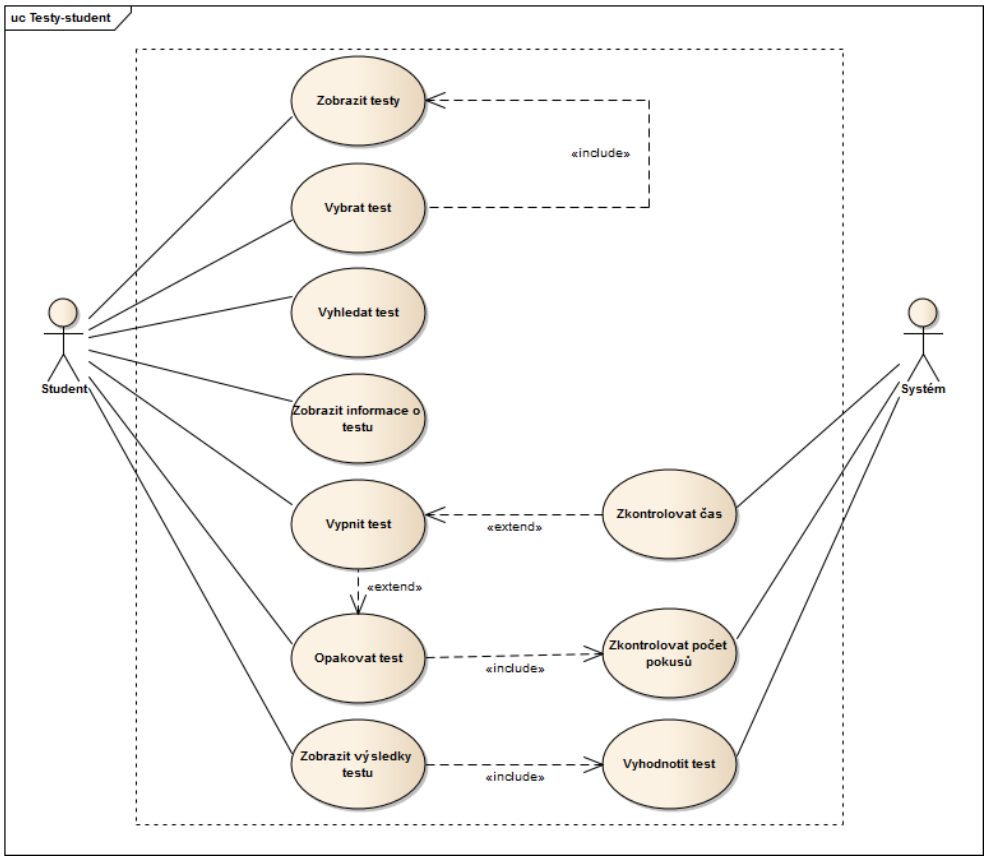
Hlavní scénář:

1. Případ užití začíná, když chce uživatel opakovat test.
2. INCLUDE **UC Zkontrolovat počet pokusů**.
3. Systém povolí opakování.
4. Uživatel vyplní test.

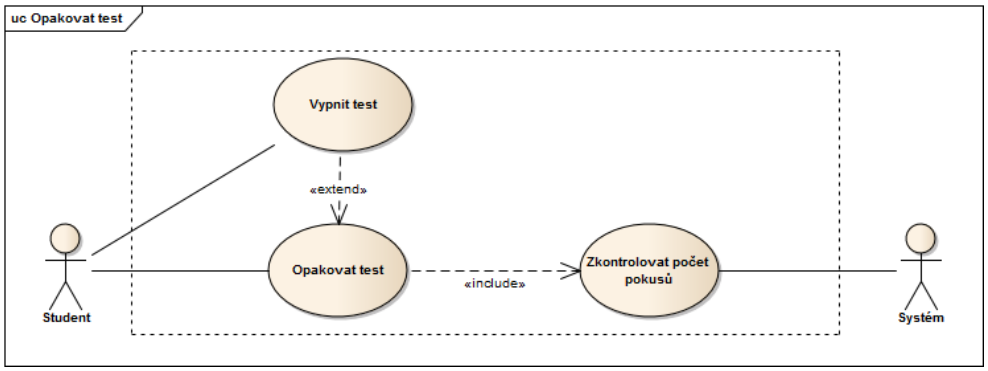
Výjimky:

3a. Systém nepovolí opakování, protože uživatel vyčerpal všechny pokusy

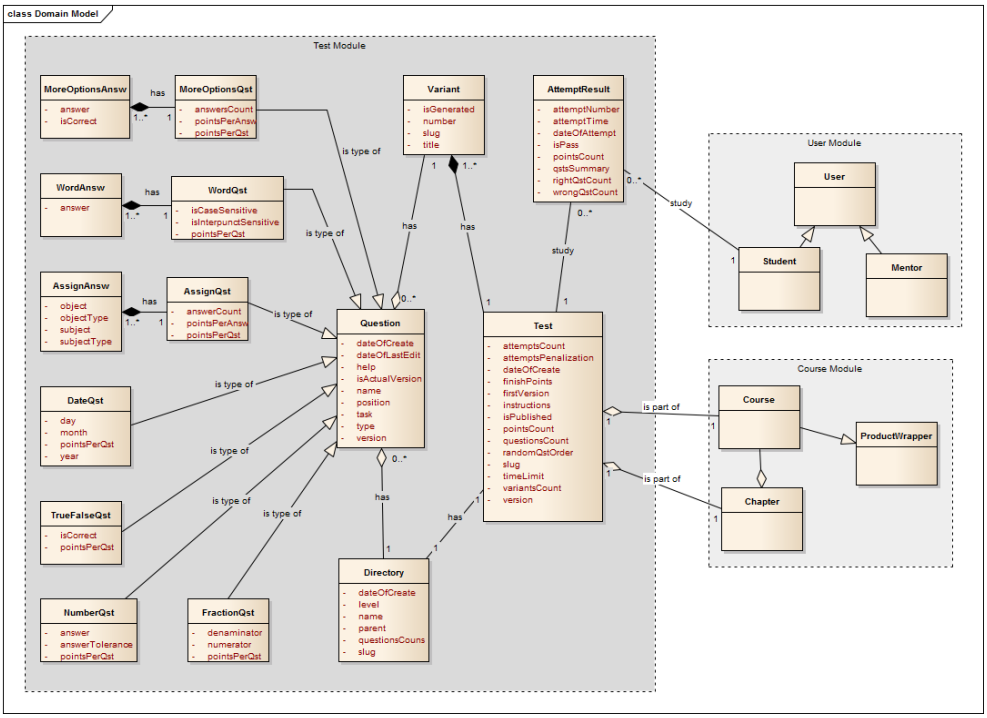
1. Systém vrátí chybovou hlášku.
2. Hlavní scénář tímto končí.



Obrázek 3.8: Případy užití ze skupiny „Testy – pohled studenta“



Obrázek 3.9: Případ užití „Opakovat test“



Obrázek 3.10: Doménový model

Další případy užití

UC Zkontrolovat počet pokusů

Hlavní scénář:

1. Systém zkontroluje, jestli je test omezen na počet pokusů a případně jestli uživateli zbývají další pokusy.

3.2 Doménový model

Doménový model neboli diagram tříd je UML diagram, který slouží k popisu tříd a vazeb mezi nimi. Na obrázku 3.10 je znázorněn doménový model pro modul Test. Diagram je rozdělen do tří bloků, kde každý blok zastupuje určitý modul aplikace. Modul Uživatel a modul Kurz jsou pouze naznačeny pro popis jejich vztahů s modulem Test. Doménové modely těchto modulů jsou popsány v diplomových pracích Bc. Jaroslava Tesaře [1] (modul Uživatel) a Bc. Jiřího Matějky [2] (modul Kurz).

Tato kapitola se zabývá návrhem modulu Test. Hlavním stavebním kamenem modulu je otázka. Otázky je možné vytvářet samostatně uvnitř složek nebo je přidávat do testů. Test může být vygenerován z otázek a může tak obsahovat více variant. V návrhovém modelu proto otázky nepatří přímo do

testu, ale do jeho varianty. Vztah mezi třídou `Question` a třídami `Variant` a `Directory` je definován jako agregace, protože otázka je vždy připojena k některému produktu/položce, ale zároveň může patřit i jinam než do testu nebo složky.

Otázka má dále více typů, které jsou definovány pomocí dědičnosti. Všechny otázky mají několik společných atributů jako například název, text zadání nebo nápověda. Každý typ pak má nějaké atributy „navíc“, které jsou platné pouze pro něj. Například u numerické otázky je definována hodnota a tolerance, u otázky s jednoslovní odpovědí se uvádí, jestli se mají rozlišovat velká a malá písmena. Otázky z více možnostmi odpovědí (třída `MoreOptionsQst`), přiřazovací otázky (třída `AssignQst`) nebo otázky s jednoslovní odpovědí (třída `WordQst`) mohou obsahovat více odpovědí. Třídy sloužící pro definici odpovědí jsou ve vztahu mnoha ku jedné s třídami příslušných otázek. Vzhledem k tomu, že odpověď nemůže existovat sama o sobě a vždy musí být propojená s otázkou, je vazba mezi otázkou a odpovědí definována jako kompozice.

Propojení s modulem User a modulem Course je uskutečněno pomocí rozhraní třídy Test. Test vždy patří buď ke kurzu, nebo ke kapitole. Test může být zároveň samostatný produkt nabízený podobně jako kurzy, z hlediska návrhu je ale vždy test součástí kurzu. V případě samostatného testu je změněn typ kurzu na `testOnly`, ve kterém uživatelé nemohou vytvářet studijní materiály. Pro uživatele jsou tyto kurzy zobrazeny jako samostatné testy. Pokud by třída `Test` v návrhu tříd dědila od třídy `Produkt`, přebírala by i všechny atributy produktu. Díky tomu by testy nemohly být propojeny s kapitolami. Jak je patrné z doménového modelu třída `Test` je propojena s třídami `Chapter` a `Course` pomocí agregace, což znamená, že třída `Test` vždy patří k nějaké další třídě.

Třída `Test` a třída `Student` z modulu User se nachází ve vztahu mnoho ku mnoha. Student může studovat více testů a stejný test může studovat více studentů. V případě vztahu `Test-User` bylo potřeba navrhnout pomocnou třídu, která slouží nejenom k zajištění vztahu mezi dvěma třídami, ale také definuje atributy navíc, které tento vztah popisuje. Třída se jmenuje `AttemptResult` a obsahuje takové atributy jako evidenci čísla pokusu na test, počet bodů dosažených studentem, jestli student test splnil apod.

3.3 Databázový model

Databázový model slouží pro návrh struktury tabulek a jejích vazeb v databázovém úložišti. V případě modulu Test je databázový model (viz obrázek 3.11) velmi podobný doménovému. Přibývá zde podrobnější definice vazeb, definují se primární a cizí klíče. Tabulky v databázovém modelu přesně odpovídají třídám z diagramu tříd. Navíc byly přidány pouze tabulky `directory_closure` a `question_type`. Tabulka `question_type` obsahuje data tykající se typu otázek. Tabulka `directory_closure` slouží k definici stromové struktury pro

práci se složkami. Vytvoření této tabulky a cíl jejího přidání je podrobněji popsán v kapitole 4.3.3.

3.4 Wireframy

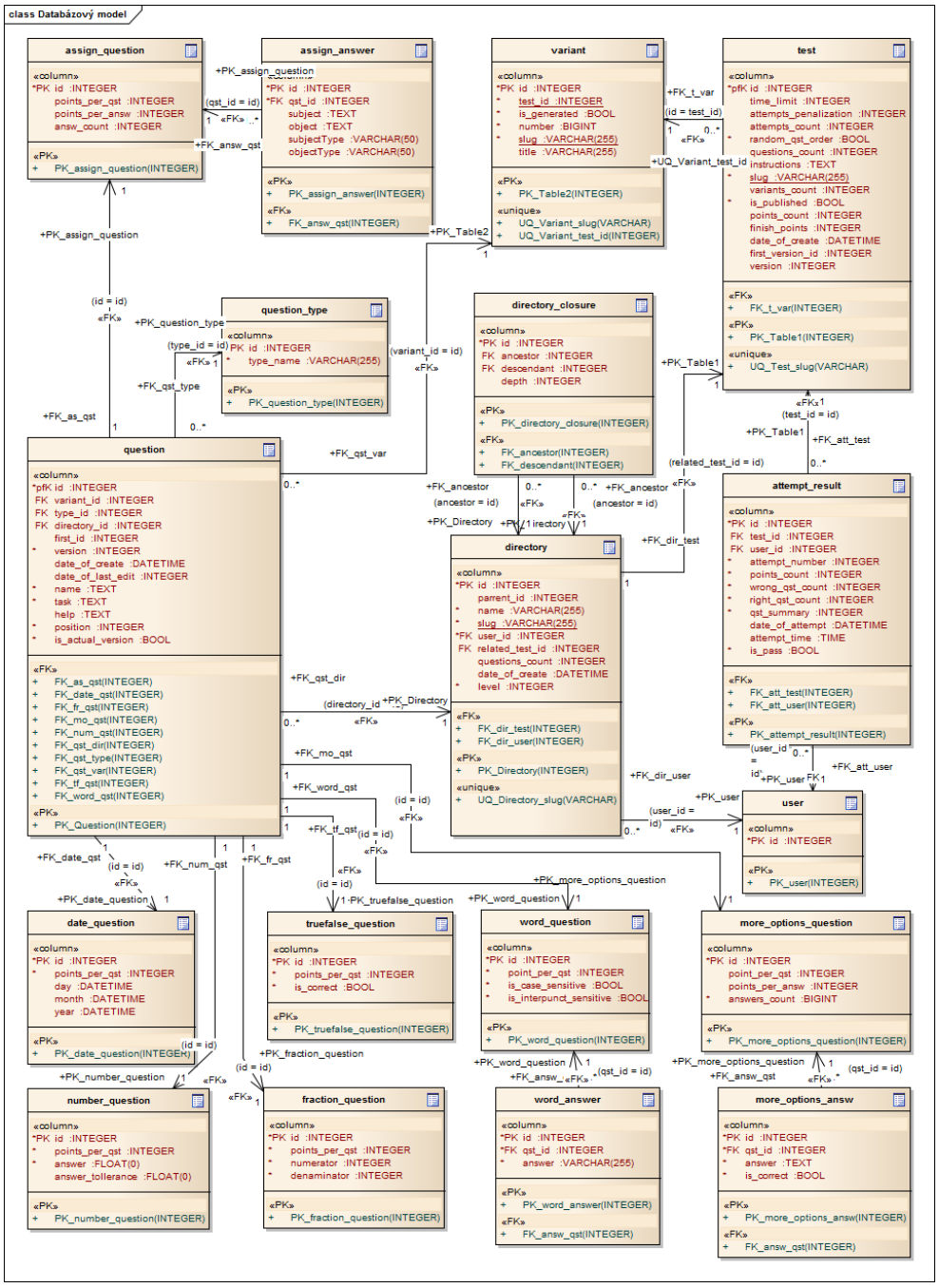
Wireframe je model aplikace, který znázorňuje informace a navigační prvky, které se zobrazí v jednotlivých krocích či na jednotlivých stránkách aplikace. Wireframe je v podstatě nástroj k popsání uživatelského rozhrání. Pomocí wireframů lze zdůraznit hlavní prvky aplikace, navrhnout rozložení prvků na stránce a popsat dynamické chování aplikace v závislosti na uživatelských akcích. Wireframy se také často používají k testování aplikací a umožňují tak zjistit, jestli je navržená logika srozumitelná pro koncového uživatele. [4].

V následujících kapitolách jsou popsány vybrané wireframy, které znázorňují nejdůležitější stránky webu a jejich chování.

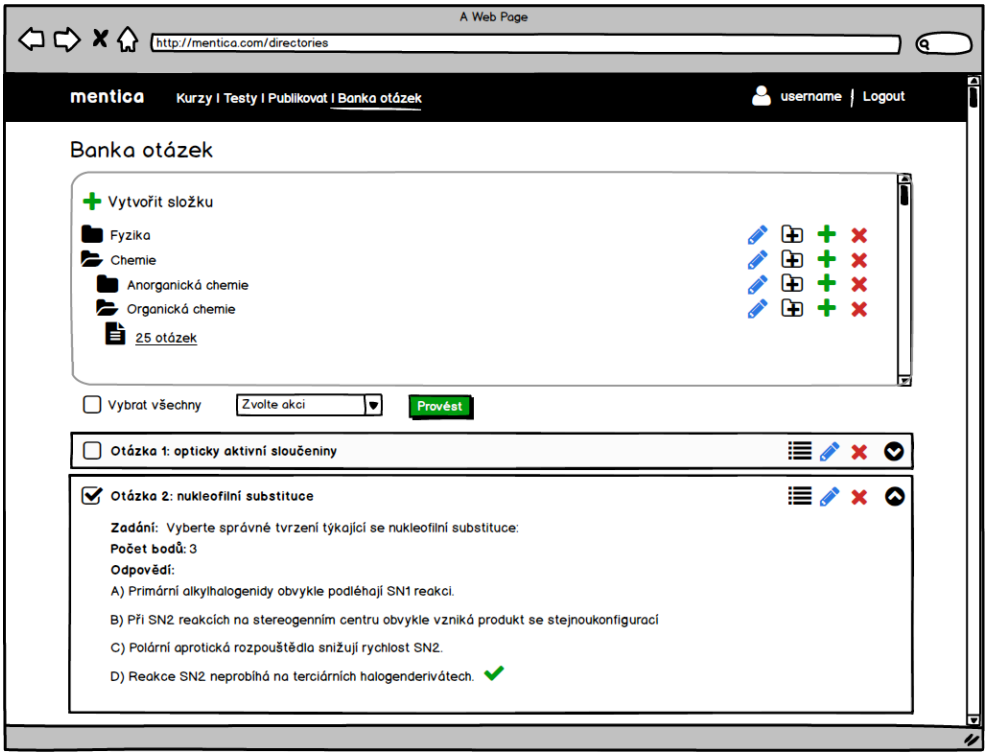
3.4.1 Banka otázek

Na obrázku 3.12 je znázorněn wireframe, který reprezentuje úvodní stránku banky otázek. Uživatel má k dispozici seznam všech svých složek, kterými může procházet a zobrazovat jejích obsah. Dále je možné vytvářet nové složky a podsložky nebo editovat již existující. Proces přidání, editace nebo mazaní složky se děje dynamicky, například uživatel není přesměrován na další stránku, formulář pro provedení příslušné akce se zobrazí ve vyskakujícím okně (viz obrázek 3.13). Po provedení akce se uživateli zobrazí obnovený seznam složek a upozornění, která akce byla vykonána a jestli byla úspěšná. Odstranění složky musí být vždy potvrzeno uživatelem pro případ zabránění chybnému smazaní (viz obrázek 3.14). Do každé složky lze přidávat nové otázky. Pokud chce uživatel provést tuto akci, je přesměrován na jinou stránku, kde je mu zobrazen formulář pro zadání otázek.

Uživatel má také přehled o dříve vytvořených otázkách, které jsou umístěny do složek. Kliknutím na příslušný odkaz složky se uživateli načtou potřebné otázky. Otázky se dynamicky mění podle vybrané složky a zobrazují se na stejné stránce jako složky. Uživatel má tak přehled o všech potřebných informacích na jedné stránce. Uživatel si může rovnou zvolit jednu nebo více otázek a provést s nimi potřebnou akci, například přesunout je do jiné složky či smazat. Při každé akci vyžaduje systém potvrzení uživatele, že opravdu chce tuto akci vykonat. Uživatel může překlikávat mezi základním a rozšířeným pohledem na otázku. Základní pohled obsahuje pouze číslo a název otázky a seznam akcí, které lze s otázkou provést. Rozšířený pohled obsahuje navíc celý náhled otázky včetně označení správných odpovědí. Stejně jak v případě složek, může uživatel otázky editovat a mazat a tyto akce se nabízejí hned u výpisu každé otázky. Kliknutím na příslušnou ikonu si uživatel může zobrazit všechny verze otázky. V tomto případě je uživatel přesměrován na jinou stránku, která mu nabídne přehled verzí otázky a možnosti k jejich správě.



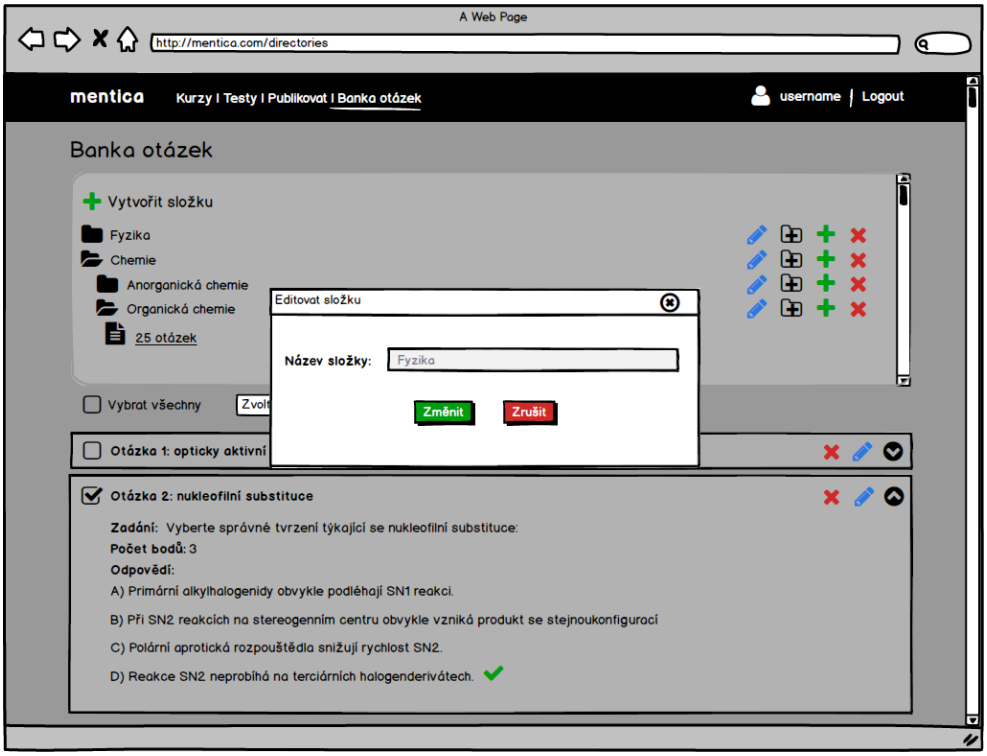
Obrázek 3.11: Databázový model



Obrázek 3.12: Wireframe „Banka otázek“

3.4.2 Vytváření otázek

Na obrázku 3.15 je zobrazen wireframe odpovídající vytváření nových otázek. Nové otázky se přidávají buď do banky otázek, nebo do testů. V obou případech je pro vytvoření nových otázek poskytnutá stránka, kde uživatel může přidat více otázek najednou a také může měnit jejích pořadí. Uživateli je nabídnuto postranní menu (viz obrázek 3.16), ze kterého si může vybrat druh nově přidávané otázky. Menu pro výběr otázek je možně minimalizovat a rozšířit podle potřeby. Po výběru druhu otázky je uživateli nabídnut formulář pro její vytvoření. Pokud otázka obsahuje více odpovědí (viz obrázek 3.17) je poskytnutá možnost přidávání dalších odpovědí nebo odstranění existujících. Výchozí počet odpovědí je nastaven na 4. Přidané otázky lze ze stránky odstranit ještě před odesláním celého formuláře a uložení do databáze. Formulář pro zadávání otázek je možné minifikovat a uvolnit tak další prostor pro přidání nových otázek. Uživatel může měnit pořadí otázek pomocí přetahování, což umožňuje technologie Drag and Drop.



Obrázek 3.13: Wireframe „Editace složky“

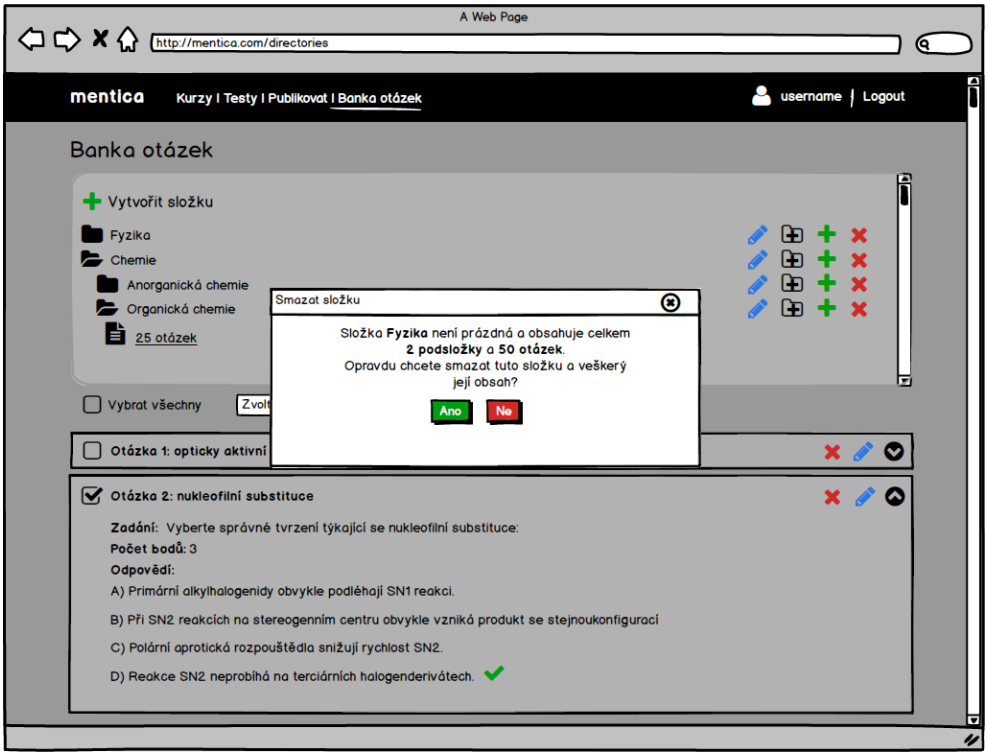
3.5 Použité technologie

Po sestavení funkčních a nefunkčních požadavků a analýze požadavků zadavatele byly vybrány technologie, pomocí kterých bude portál realizován.

3.5.1 PHP

PHP (PHP: Hypertext Preprocessor) je skriptovací jazyk, který se používá převážně pro programování serverových částí webových stránek. V současné době je to jeden z předních programovacích jazyků sloužících k vytváření webů. PHP je velice populární mezi vývojáři díky své jednoduchosti, dostupnosti a široké sadě funkcionalit a nástrojů, které jsou do jazyka implementovány (viz statistiky [5]). PHP je také podporován většinou poskytovatelů webhostingů, což umožňuje snadné nasazení a spuštění webů v ostrém provozu. Velkou roli hraje spolupráce jazyka s velkým počtem databázových systémů, například MySQL, Oracle, PostgreSQL. Od verze 5 jazyk plně podporuje objektové programování[6].

Na základě PHP je postaveno mnoho různých frameworků, které ulehčují práci vývojářům a starají se například o bezpečnost nebo validaci.



Obrázek 3.14: Wireframe „Mazání složky a jejího obsahu“

3.5.2 HTML

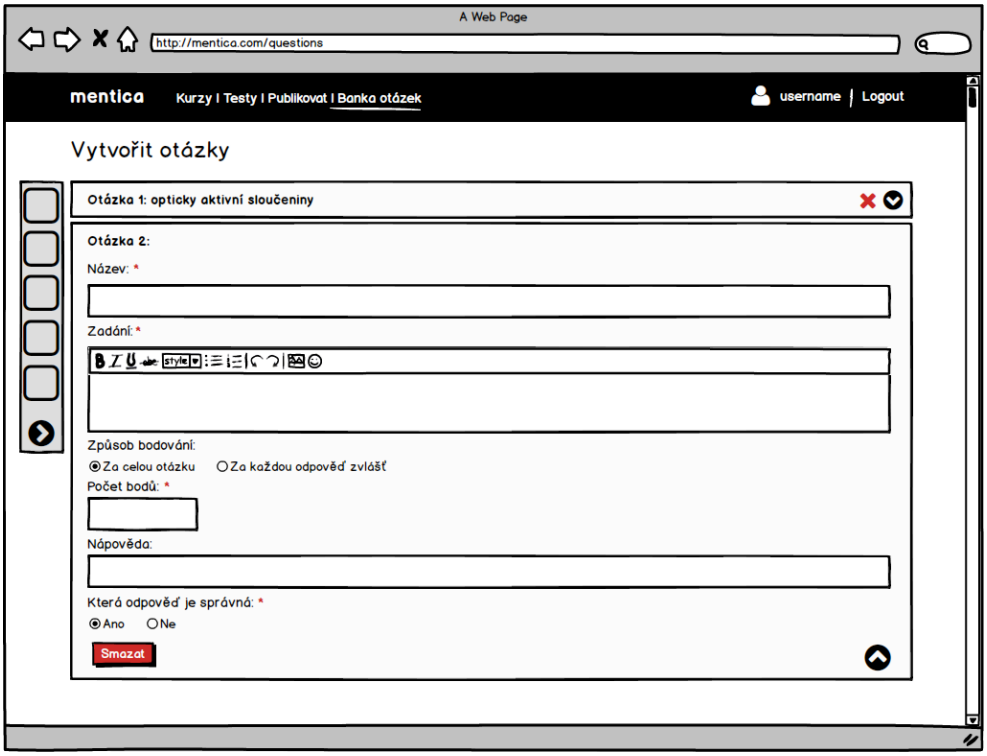
HTML (HyperText Markup Language) je standardní značkovací jazyk, který je používán ve většině webových stránek. HTML kód je tvořen HTML značkami a webový prohlížeč si interpretuje tento kód do podoby, která je srozumitelná pro člověka.

Od roku 2014 podporují nejpoužívanější prohlížeče verzi HTML 5 [7], jejíž hlavní výhodou je obohacení dokumentu o sémantický obsah a podpora multimediálních souborů.

3.5.3 CSS

CSS (Cascading Style Sheets) je formální jazyk, který se používá k popisu vzhledu HTML dokumentu. CSS umožňuje definovat barvy, písmo, rozložení bloků na stránce a mnoho dalších prvků týkajících se designu.

Hlavním účelem CSS je oddělení logiky struktury webové stránky od popisu jejího vzhledu. CSS styly jsou uložené ve zvláštním souboru, který je připojen k HTML dokumentu. Pokud je potřeba změnit vzhled dokumentu, stačí změnit pravidla, napsané v CSS souboru.



Obrázek 3.15: Wireframe „Vytváření otázek“

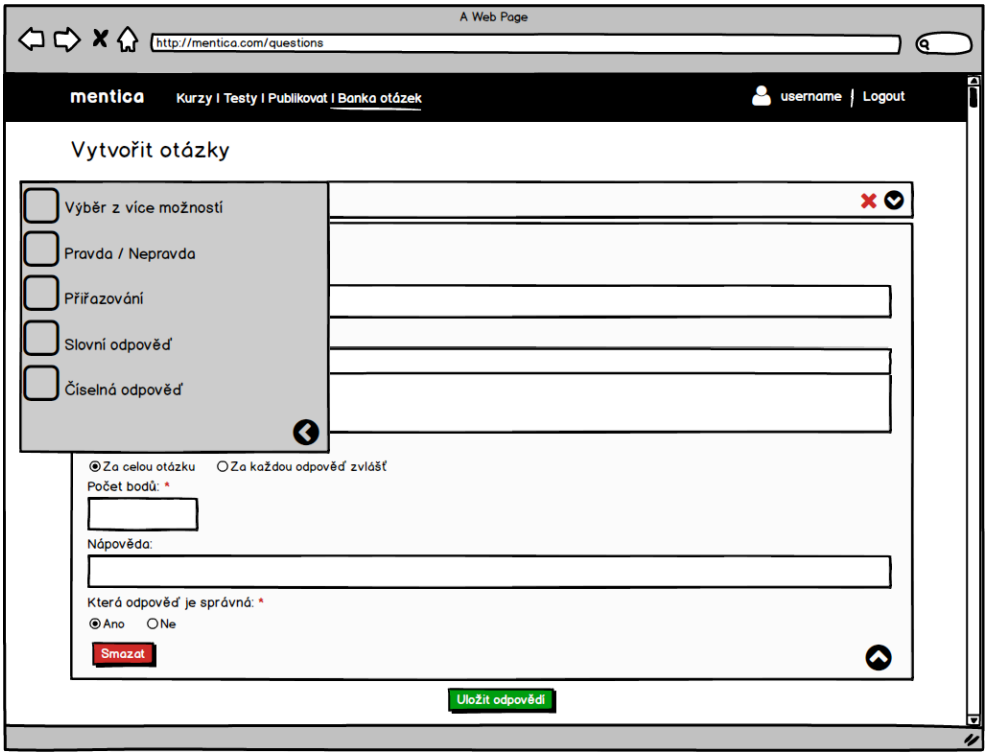
Díky CSS je také možné definovat více stylů najednou, což umožňuje přizpůsobit zobrazení dokumentu pro různé výstupy (například zobrazení na stolním počítači nebo při tisku).

Aktuální verze CSS 3 zavádí mnoho nových funkcionalit [8] a posunuje vývoj vzhledu webových aplikací na další úroveň. V současné době je kladen velký důraz na mobilní zařízení, především chytré telefony a tablety. Proto je nutné mít webové stránky přizpůsobené k prohlížení i na těchto zařízeních. K zajištění optimalizace vzhledu CSS 3 poskytuje funkci media queries, která umožňuje přizpůsobit vzhled stránek pro měnící se šířku obrazovky. CSS 3 také přebírá i některé funkce JavaScriptu a umožňuje například vytvoření jednoduché animace.

3.5.4 JavaScript

JavaScript je skriptovací programovací jazyk, který se převážně používá při implementaci klientské části webových aplikací a umožňuje obohatit chování aplikace o interaktivní prvky. Skript, který je napsán v JavaScriptu se nekompiluje a stačí ho připojit k HTML stránce.

Pomocí JavaScriptu je možné dynamicky měnit strukturu HTML stránky, editovat kaskádové styly a vzhled HTML prvků v závislosti na akcích prová-



Obrázek 3.16: Wireframe „Vytváření otázek – postranní menu“

děných uživatelem nebo spouštět různé akce závislé na událostech. Výhodou JavaScriptu je také možnost odlehčit práci serveru a přenést část výpočtu na klienta, což je umožněno díky technologii Ajax (viz kapitola 3.5.5).

V jazyce JavaScript je napsáno mnoho rozšíření, které jsou dostupné ke stažení a programátor tak nemusí vymýšlet vlastní implementace některých funkcionalit.

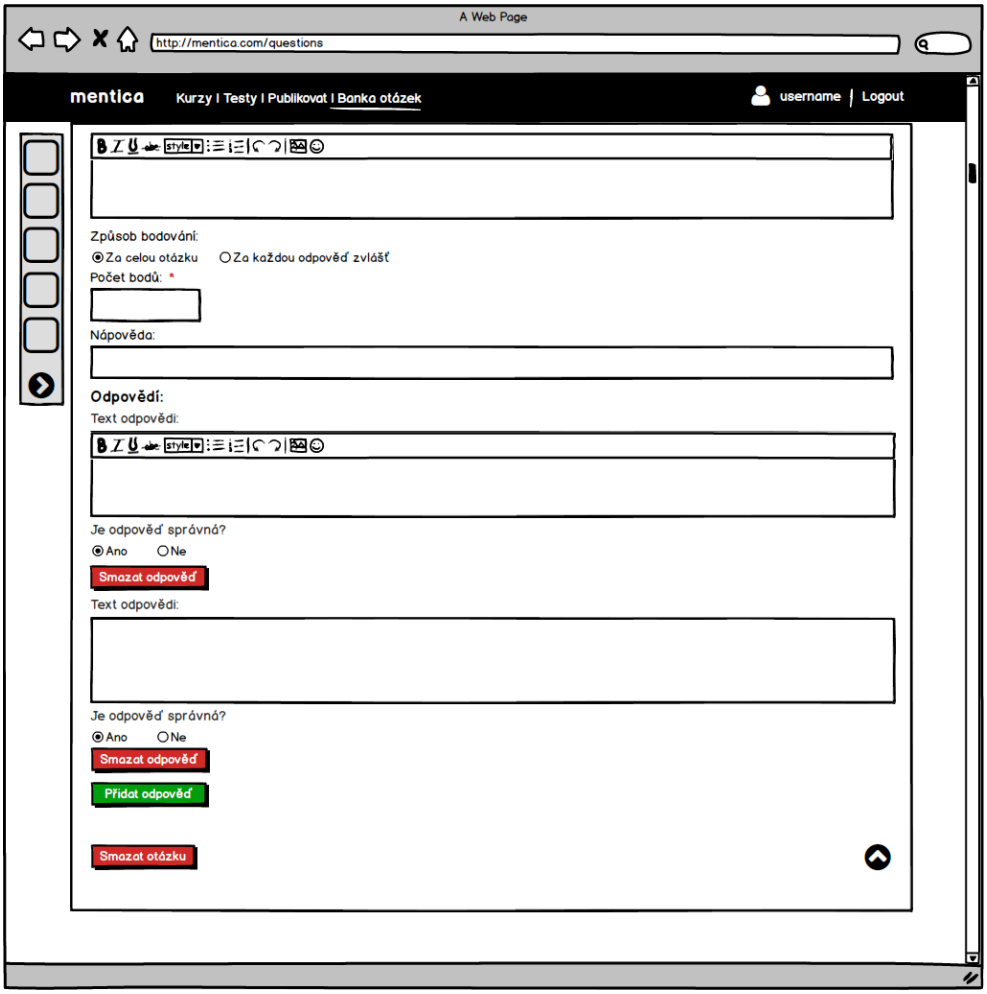
3.5.5 Ajax

Ajax (Asynchronous JavaScript and XML) je mezivrstva mezi klientem a serverem, která umožňuje posílání asynchronních požadavků na server. Při zpracování asynchronního požadavku se celá stránka nemusí načítat znovu, změní se pouze její část v závislosti na odpovědi serveru. Ajax tak pomáhá zrychlit práci s webovou aplikací a snížit zátěž serveru.

3.5.6 MySQL

MySQL je relační databázový systém, který využívá dotazovacího jazyku SQL[1]. Díky své jednoduchosti a dostupnosti je MySQL velmi rozšířeno mezi vývojáři

[1]SQL - Structured Query Language



Obrázek 3.17: Wireframe „Vytváření otázky typu výběr z více možností“

(viz statistiky [9]) a je zahrnuto spolu s PHP a Apache[2] do vývojových prostředí WAMP[3], XAMPP[4] a podobně. Výběr MySQL je popsán v diplomové práci Bc. Jiřího Matějky [2].

3.6 Použité frameworky a nástroje

Na základě vybraných technologií byly zvoleny frameworky a knihovny, které jsou na těchto technologiích založeny a slouží k usnadnění práce programátora.

[2]Apache - webový server
[3]WAMP – vývojové prostředí `http://www.wampserver.com/en/`
[4]XAMPP – vývojové prostředí `https://www.apachefriends.org/index.html`

3.6.1 Symfony 2

Symfony 2 je framework postavený na PHP5 a principech MVC[5]. Díky použití frameworku může programátor vyvíjet aplikace rychleji a snadněji. Symfony 2 je jedním z nejpoužívanějších frameworků (viz statistiky [10]), který se stále vyvíjí a případné chyby se opravují. Framework má rozsáhlou komunitu uživatelů a vývojářů, které do frameworku přispívají pomocí vývoje různých rozšíření a doplňků. Symfony 2 poskytuje širokou nabídku již naimplementovaných funkcionalit a programátor se tak může zabývat složitějšími řešeními a neztrácet čas nad něčím, co je již vyvinuto. Symfony 2 nabízí rozsáhlou dokumentaci a příručky, které pomáhají správně Framework nainstalovat a používat. Podrobněji o vlastnostech a výhodách frameworku Symfony 2 je popsáno v diplomové práci Bc. Jaroslava Tesaře [1].

3.6.2 Doctrine 2

Doctrine 2 je ORM[6] Framework, který využitím objektového přístupu umožňuje zajistit abstrakci databázové vrstvy. Dotazování na databázi se uskutečňuje pomocí vlastního jazyku DQL[7], který je podobný jazyku SQL. Doctrine2 je nezávislý na databázovém systému. Hlavním stavebním prvkem Doctrine2 je Entita, která reprezentuje řádek tabulky. Entity obsahují atributy, které reprezentují sloupce v tabulkách a metody pro jejich získání nebo vytvoření. Entity mají mezi sebou stejné vztahy jako příslušné tabulky v databázi. [11].

Doctrine 2 je zaintegrovaná do Symfony 2 frameworku, což bylo rozhodujícím kritérium pro její využití.

3.6.3 jQuery

JQuery je JavaScriptová knihovna, která umožňuje jednoduše přistupovat k DOM [8] elementům a jejich obsahu a umožňuje snadnou manipulaci s nimi. JQuery také poskytuje API[9] pro podporu AJAX požadavků.

JQuery je velmi rozšířeno mezi vývojáři webů a má rozsáhlou komunitu uživatelů. Díky tomu je dostupná široká sada rozšíření, které je možné použít při tvorbě vlastních stránek a dosáhnout tak potřebných funkcionalit.

3.6.4 SASS CSS

Sass (Syntactically Awesome Stylesheets) – je speciální framework založený na CSS, který slouží k zvýšení úrovně abstrakce CSS kódu a vytváření přehlednějších kaskádových stylů. Sass umožňuje například definování proměnných

[5]MVC - Model-view-controller
[6]ORM - Object-relational mapping
[7]DQL - Doctrine Query Language
[8]DOM - Document Object Model
[9]API - Application Programming Interface

nebo zavádí dědičnost [12]. Psaní CSS kódu se tak stává jednodušší a srozumitelnější.

Kapitola 4

Implementace

V této kapitole jsou popsány detaily implementace vybraných částí aplikace a použitých knihoven. Podrobně je rozebrán výběr a implementace strategie pro ukládání dat ve stromové struktuře nebo realizace polymorfismu v relační databázi. Na závěr je ukázána práce s vnořenými formulářovými kolekcemi.

4.1 MVC

MVC je návrhový vzor, jehož cílem je rozdělit datový model, uživatelské rozhraní a aplikační logiku do třech vrstev, které se nazývají Model, View a Controller.

Model zajišťuje práci s daty, jejich ukládání a manipulaci s nimi. Model nedefinuje výsledné zobrazení dat.

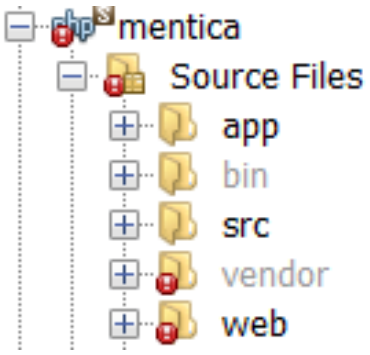
View neboli pohled slouží pro zobrazení dat. Na vstupu view jsou již připravená data, která se na výstupu zobrazí ve srozumitelné podobě uživatelům.

Controller zajišťuje komunikaci mezi vrstvami Model a View. Controller prostřednictvím View získá požadavek uživatele, zpracuje ho a pošle dotaz na Model. Model vrátí určitá data, která Controller směřuje zpět na vrstvu View pro zobrazení uživateli.

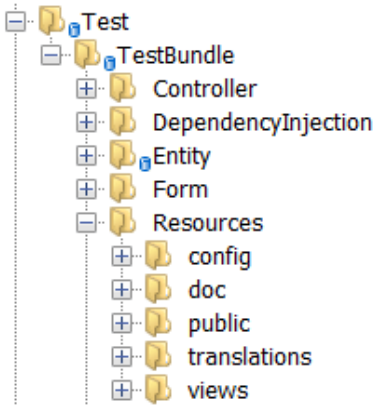
MVC architektura je plně podporována frameworkem Symfony 2. Model zde zastupují Doctrine entity reprezentující tabulky. K vytvoření pohledů používá Symfony 2 vlastní šablonovací systém Twig, který poskytuje rozsáhlou nabídku funkcionalit a umožňuje rychle vytvářet i složitější šablony. Struktura aplikace v Symfony 2 je podrobněji popsána v následující kapitole.

4.2 Struktura aplikace

Na obrázku 4.1 je zobrazena celková struktura aplikace. Hlavními složkami jsou zde `app`, `src` a `web`.



Obrázek 4.1: Struktura aplikace



Obrázek 4.2: Struktura modulu Test

Složka `app` obsahuje konfigurační soubory, které slouží k nastavení konfigurace samotné aplikace, například databázového připojení, definice rozšíření apod.

Složka `web` je jedinou veřejnou složkou projektu a obsahuje CSS soubory, JavaScript soubory, fonty a obrázky, které jsou společné pro všechny moduly.

Složka `src` je rozdělená do více složek a každá podsložka reprezentuje určitý modul. V každé složce jsou uloženy controllery, modely i view pro příslušný modul.

4.2.1 Struktura modulu Test

Na obrázku 4.2 je znázorněná adresářová struktura modulu Test. Složka `Controller` obsahuje MVC controllery. Každá část modulu Test je reprezentována pomocí vlastního controlleru. Například `DirectoryController` slouží pro správu adresářů, `QuestionController` je určen ke správě otázek a akcí s tím spojených. Složka `Entity` reprezentuje MVC model a obsahuje třídy entit, které odpovídají tabulkám v databázi. Složka `Form` obsahuje znovupo-

užitelné šablony pro vytvoření formulářů různých typů. Složka `Resources` je spojená s vzhledovou stránkou aplikace a obsahuje tři důležité složky – `config, public, views`. Složka `config` obsahuje soubory, ve kterých se definují například pravidla přesměrování. Ve složce `public` se nachází CSS a JavaScript soubory určující vzhled jednotlivých stránek. Složka `views` reprezentuje MVC šablony. Jsou v ní uložené jednotlivé šablony stránek rozdělený podle „tématu“ stejně tak jako controllery.

4.3 Stromová hierarchie

Jak již bylo řečeno, při práci s otázkami v bance otázek může uživatel seskupovat otázky do složek. Složky mohou obsahovat libovolně mnoho podsložek a to do více úrovní hloubky. Tato struktura je v databázi reprezentována jako strom a k její implementaci bylo použito Doctrine 2 rozšíření Tree-Nested Set behaviour [10]. Toto rozšíření podporuje tři hlavní strategie pro ukládání stromu a práci s jeho elementy: Materialized Path, Nested Set a Closure Table.

4.3.1 Strategie Materialized Path

Principem strategie Materialized Path je, že každý uzel ve stromu obsahuje v sobě celou cestu od kořene stromu až k tomuto uzlu [13]. Tato cesta může být reprezentována jako ID uzlu a je na vývojáři jakým způsobem cestu vygeneruje.

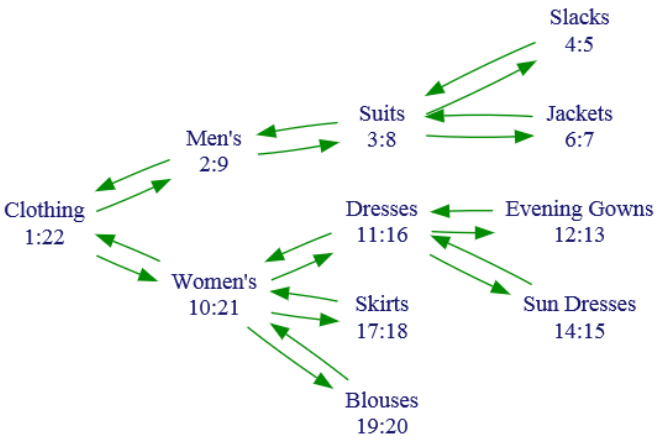
Výhodou strategie je jednoduché a rychlé procházení stromu. Uložení celé absolutní cesty ke každému uzlu umožňuje dotazovat na přímé a nepřímé potomky pouze v jednom jednoduchém dotazu. Avšak tento přístup přináší i svoje nevýhody. Přidání a editace záznamů je složitější a klade větší nároky na výpočet. Přidání záznamu znamená vypočítání nového ID, které se skládá z cesty k rodiči a ID aktuálního záznamu. Při změně struktury stromu je potřeba přepočítat jak cestu editovaného záznamu, tak i cesty jeho potomků.

Strategie Materialized Path je vhodná použít u stromových dat, jejichž struktura je předem definována a nebude se často měnit.

4.3.2 Strategie Nested Set

Strategie Nested Set je podobná strategii Materialized Path, ale je o něco složitější jak na pochopení, tak i na implementaci. V případě Nested Set se neukládá absolutní cesta k uzlům. Místo toho jsou ke každému uzlu zaznamenávány levé a pravé indexy. Strom se při vytváření prochází dvakrát a každý uzel je tak navštíven jednou při počáteční cestě a po druhé při cestě zpátky. Při každém procházení se uloží index počátečního zastavení (levý index) a

[10] Tree-Nested Set behaviour - dostupné z `http://github.com/Atlantic18/DoctrineExtensions/blob/master/doc/tree.md`



Obrázek 4.3: Procházení stromem při strategii Nested Set (zdroj en.wikipedia.org)

index zastavení při zpáteční cestě (pravý index) [13]. Algoritmus Nested Set je znázorněn na obrázku 4.3.

Strategie Nested Set se jeví jako velice efektivní a rychlá při vyhledávání ve stromu a při práci s podstromy. Například pro zjištění počtu potomků určitého uzlu stačí vědět jeho levý a pravý index, procházet všechny podstromy již není nutné.

Nevýhodou je stejně jako u Materialized Path náročnější ukládání a editace dat kvůli nutnosti přepočítání indexu uzlů. Proto Nested Set se jeví vhodnější pro stromové hierarchie, kde ukládání, mazaní a editace dat není prováděna často.

4.3.3 Strategie Closure Table

Strategie Closure Table se liší od předchozích dvou. Při použití Closure Table se namísto jedné tabulky vytvoří dvě, kde první tabulka obsahuje pouze samotná data a druhá tabulka slouží k ukládání vztahů mezi daty [14]. Na obrázku 4.4 je znázorněna struktura těchto tabulek.

Jak je vidět z obrázku 4.4 v tabulce vztahu jsou uložena data o vztazích všech přímých a nepřímých potomků. Každý záznam je potomkem sám sebe v nulové hloubce a může mít více potomků v různých hloubkách. Například položka C je „synem“ B a zároveň „vnukem“ A. V tabulce vztahů se zaznamenává vztah položky C jak k rodiči, tak i k prarodiči a příslušná úroveň hloubky.

Celkový počet záznamů se v tabulce vztahů rovná počtu záznamu v základní tabulce vynásobeným průměrnou hloubkou stromu. V nejhorším případě může obsahovat n^2 záznamů, což je v praxi nepravděpodobné. Ukládání

Základní tabulka			Tabulka vztahů			
id	parent_id	item	Id	parent_id	child_id	depth
1	0	A	1	1	1	0
2	1	B	2	2	2	0
3	2	C	3	3	3	0
			4	1	2	1
			5	2	3	1
			6	1	3	2

Obrázek 4.4: Struktura tabulek při strategii Closure Table

většího množství dat se tak jeví jako nevýhoda této strategie.

Vkládání a editace dat při použití Closure Table je jednodušší a rychlejší než u strategií Nested Set a Materialized Path. Není zde potřeba přepočítávat indexy uzlů, pouze se vytvoří další záznam v tabulce vztahů. Většina databázových systémů podporuje triggery, díky nimž je možné automatizovat proces vytváření, mazání a editaci dat v tabulce vztahů.

Uložení záznamu o hloubce pomáhá rychle a jednoduše dotazovat na rodiče a potomky různých úrovní. Proces získání potomků a například jejích seřazení je díky zavedení tabulky vztahů také jednodušší než u předchozích strategií.

4.3.4 Výběr strategie

Tabulka 4.1 obsahuje souhrn popsaných strategií.

Strategie	Počet tabulek	Dotazování na potomky	Dotazování na podstrom	Modifikace stromu
Materialized Path	1	Jednoduché	Jednoduché	Náročné
Nested Set	1	Náročné	Jednoduché	Náročné
Closure Table	2	Jednoduché	Jednoduché	Jednoduché

Tabulka 4.1: Souhrn strategií pro ukládaní stromových hierarchií (převzato a upraveno z www.slideshare.net)

Při práci se složkami bude uživatel často měnit jejich strukturu, přidávat nové složky a mazat existující. Zároveň potřebuje mít přehled o celé struktuře složek, vědět jak jsou vnořené a kolik mají potomků. Po provedení analýzy a sestavení celkového vyhodnocení byla zvolena jako nejvhodnější strategie Closure Table.

4.3.5 Realizace

K realizaci Closure Table je potřeba vytvořit dvě entity – jedna odpovídá základní tabulce, druhá slouží k reprezentaci tabulky vztahů. Následující kód popisuje definici entit pomocí anotací.

```
1  use Gedmo\Mapping\Annotation as Gedmo;
2  use Doctrine\ORM\Mapping as ORM;
3  /**
4   * @Gedmo\Tree(type="closure")
5   * @Gedmo\TreeClosure(class="Test\TestBundle\Entity\DirectoryClosure")
6   * @ORM\Entity(repositoryClass="Test\TestBundle\Entity\DirectoryRepository")
7   * @ORM\Table(name="mtc_directory")
8   */
9  class Directory {
10     public function addClosure(DirectoryClosure $closure) {
11         $this->closures[] = $closure;
12     }
13 }
```

```
1  use Gedmo\Tree\Entity\MappedSuperclass\AbstractClosure;
2  use Doctrine\ORM\Mapping as ORM;
3  /**
4   * @ORM\Entity
5   * @ORM\Table(name="mtc_directory_closure")
6   */
7  class DirectoryClosure extends AbstractClosure{
8  }
```

Entita `DirectoryClosure` je potomkem entity `AbstractClosure`, která je již implementována v Doctrine 2 Tree rozšíření. V této entitě jsou předdefinovány atributy reprezentující předka, potomka a hloubku vnoření a metody pro jejich získání a nastavení.

K získání vytvořené stromové struktury složek se v Controlleru používá metoda `childrenHierarchy()`, která v závislosti na vstupních parametrech vrátí hierarchii složek uspořádanou do pole (viz následující kód).

```
1  $loggedInUserFolder = $repository->getRootDirectory($this->getUser());
2  $directoryTree = $repository->childrenHierarchy(
3          $loggedInUserFolder,
4          false,
5          array(
6          'decorate' => false,
7          'childSort' => array('field'=>'name','dir'=>'asc')
8          ));
```

První parametr (řádek 3) určuje uzel, jehož potomci mají být vráceny. V tomto případě budou vraceny všechny potomky uzlu odpovídajícího kořenové složce aktuálního uživatele. Druhý parametr, který je nastaven na `false` (řádek 4), říká, že budou vráceny všechny potomky elementů, jak přímé, tak i nepřímé.

Jako třetí parametr (řádek 5-8) se uvádí pole hodnot definujících vzhled hierarchie. Pokud je parametr `decorate` nastaven na `true`, výsledná hierarchie se zobrazí jako nečíslovaný HTML seznam. Volitelně je možné definovat také HTML tagy, do kterých bude seznam obalen. Parametr `childSort` (řádek 7) definuje, jak bude hierarchie seřazená. Je potřeba definovat, podle kterého parametru se provede řazení a definovat způsob řazení.

Výsledek metody `childrenHierarchy()` se posílá do šablony, kde se vykreslí struktura složek. Složky se vykreslují rekurzivně od rodiče k potomkům pomocí použití speciálních maker. Dále následuje ukázka kódu pro rekurzivní vykreslení složek.

```
1  {% for directory in directoryTree %}
2          <div class="directoryRow">
3              {{ _self.display_cl_tree(directory) }}
4          </div>
5  {% endfor %}
6  . . .
7  {% macro display_cl_tree(directory) %}
8          <ul>
9          <li>
10             <a href="#" class="packDirectory">
11                 <span class="folderIcon"></span>{{ directory.name }}
12             </a>
13             . . .
14              <div class="subDirectories">
15                  {% if directory.___children|default() %}
16                      {% for child in directory.___children %}
17                          {{ _self.display_cl_tree(child) }}
18                      {% endfor %}
19                  {% endif %}
20              . . .
21              </div>
22         </li>
23     </ul>
24 {% endmacro %}
```

4.4 Polymorfizmus v datovém úložišti

Při práci s otázkami má mentor možnost výběru z více druhů, kde každý druh otázky se liší podle možnosti odpovědí. Všechny otázky však mají společné nastavení jako například název, text zadání, datum vytvoření apod. To znamená, že otázka a její typ se nacházejí ve vztahu dědičnosti. Při implementaci této funkcionality nastává problém s realizací dědičnosti na úrovni databázového úložiště.

V objektovém programovacím jazyku se podobné problémy řeší pomocí polymorfizmu. V tomto případě se vytvoří třída rodič a několik tříd potomků.

Rodičovská třída obsahuje společné atributy a metody pro všechny svoje „potomky“, Třídy potomků dědí od rodičovské třídy a přidávají vlastní unikátní atributy nebo přepisují existující metody. Polymorfizmus umožňuje přistupovat k různým typům obecně a kompilátor sám zjistí, o který typ se jedná a zavolá potřebnou funkci.

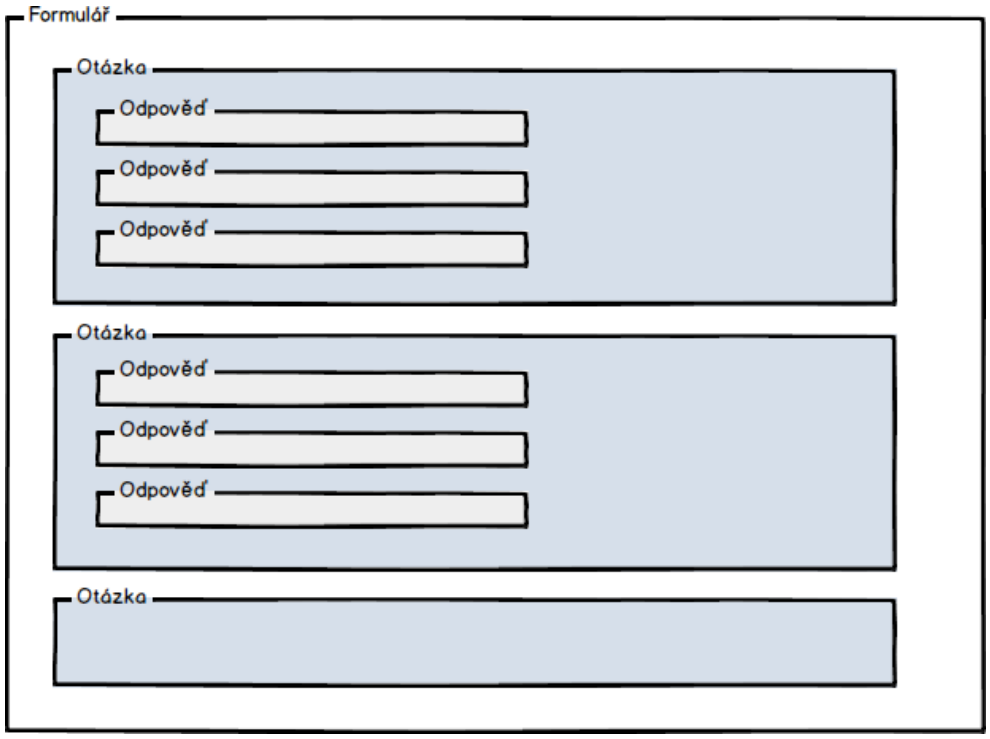
Zajistit polymorfizmus v relační databázi je složitější úkol. Hlavním problémem je zajištění unikátního ID skrz několik tabulek. Použití stejného ID pro rodičovskou tabulku a tabulky potomků zajistí, že každý sloupec má identifikátor, který je jedinečný v rámci všech potomků, a zjednoduší tak SQL příkazy pro přístup k datům. Zajištění funkcionality je řešeno pomocí ORM, které má na starosti kontrolu ID.

Nejprve je potřeba vytvořit entitu reprezentující rodičovskou tabulku. Následující kód popisuje vytvoření entity Question a její definici pomocí anotací.

```
1  /**
2   * Question
3   *
4   * @ORM\Table(name="mtc_question")
5   * @ORM\Entity(repositoryClass="Test\TestBundle\Entity\QuestionRepository")
6   * @ORM\InheritanceType("JOINED")
7   * @ORM\DiscriminatorColumn(name="discr" , type="string")
8   * @ORM\DiscriminatorMap({"moreoptionsqst"="MoreOptionsQst",
9                             "truefalseqst"="TrueFalseQst",
10                            "assignqst"="AssignQst"})
11  */
12
13 class Question {
14     /**
15      * @var integer
16      *
17      * @ORM\Column(name="id", type="integer")
18      * @ORM\Id
19      * @ORM\GeneratedValue(strategy="AUTO")
20      */
21     private $id;
22         . . .
23 }
```

Anotace (řádek 6 - 8) definují polymorfní vztah a musí být uvedené v entitě rodiče. Anotace `@ORM\DiscriminatorColumn` a `@ORM\DiscriminatorMap` slouží k specifikaci entit potomků. `@ORM\DiscriminatorColumn` definuje pouze klíčové slovo pro potomka, `@ORM\DiscriminatorMap` pak mapuje toto klíčové slovo na entitu potomka a specifikuje tak typ každého řádku v tabulce Question.

U entit potomků se nemusí specifikovat žádné speciální vlastnosti pomocí anotací. Entita potomka ale musí dědit od rodičovské entity a nesmí mít definované ID, které přebírá od rodiče.



Obrázek 4.5: Formulář s více vnořenými formuláři

Pokud je polymorfní vztah navržen a naimplementován správně, volání metody `getQuestion($id)` s různými hodnotami ID bude vracet otázky různých typů a poskytovat pro každou příslušné metody a atributy.

4.5 Dynamická kolekce otázek

Jak již bylo zmíněno při návrhu wireframů, na jedné stránce se vytváří více otázek pomocí jednoho formuláře. Pro každou otázku existuje vlastní formulář. Na stránce může být více otázek, což znamená, že je potřeba umožnit přidávání více formulářů do jednoho a počet vnořených formulářů se může dynamicky měnit. Navíc pokud otázka obsahuje více odpovědí, je potřeba znovu vložit formulář do již vnořeného formuláře. Tím pádem vzniká složitá formulářová struktura, která se může rozšiřovat do hloubky (viz obrázek 4.5).

Tuto funkcionalitu je možné v Symfony 2 realizovat pomocí takzvaných formulářových kolekcí. Kolekce je v podstatě typ formulářového pole a do formuláře se přidá stejně jako jakékoliv jiné textové pole, avšak je zde navíc několik parametrů platných pouze pro kolekci. Následující kód obstarává přidání kolekci odpovědí do formuláře pro zadávání otázky typu „výběr z více možností“.

```
1  $builder->add('name', 'textarea',
2                  array('label' => 'Název',
3                        'label_attr' => array('class' => 'cRequired')))
4          ->add('answers', 'collection', array(
5                      'type' => new MoreOptionsAnswType(),
6                      'label' => "Odpovedí",
7                      'allow_add' => true,
8                      'allow_delete' => true,
9                      'by_reference' => true,
10                     'prototype_name' => '__moreoptansw_prot__',
11         ));
```

Parametr `type` (řádek 3) definuje typ kolekce, respektive typ každého jejího prvku. Tento typ může být jak předdefinovaný, například email, tak i vlastní. V případě přidání kolekce odpovědí do otázky jde o další formulář, proto jako parametr `type` se nastaví instance vlastního formulářového typu, jehož vytvoření je popsáno níže. Parametry `allow_add` (řádek 5) a `allow_delete` (řádek 6) slouží k zajištění dynamického přidání nebo mazaní prvků kolekce a musí být nastaveny na hodnotu `true`. Po odesílání formuláře se provede porovnání nově získané kolekce s existující kolekcí. Pokud přibudou nebo ubudou některé elementy, provede se automatické přidání/mazání těchto elementů do/z kolekce. Atribut `prototype_name` (řádek 8) je určen k vykreslení formuláře v šabloně a díky němu lze obstarat dynamické přidání formulářových prvků pomocí JavaScriptu na straně klienta.

Jak již bylo řečeno pro přidání kolekce do formuláře je potřeba definovat vlastní formulářový typ, který bude odpovídat této kolekci. Následující kód ukazuje vytvoření formulářového typu kolekce pro seznam odpovědí u otázky typu „výběr z více možností“.

```
1  class MoreOptionsAnswType extends AbstractType {
2      public function buildForm(FormBuilderInterface $builder, array $options) {
3          $builder
4                  ->add('answer','ckeditor',array('label' => 'Odpoveď:'))
5                  ->add('isCorrect', 'choice', array(
6                      'choices' => array('1' => 'Ano', '0' => 'Ne'),
7                      'label' => 'Je odpoveď správná?',
8                      'expanded' => true,
9                      'multiple' => false,
10         ));
11     }
```

Vlastní třída `MoreOptionsAnswType` dědí od abstraktní třídy `AbstractType` (řádek 1), která je společná pro všechny formulářové typy. Funkce `buildForm()` (řádky 2-10) slouží k sestavení formuláře na základě uvedených formulářových polí. Pro vytvoření odpovědi je potřeba zadat její text, který je typu „ckeditor“. Dále je potřeba uvést, jestli je odpověď správná, což je ve formuláři reprezentováno pomocí „radio buttonu“ a samotné pole je typu „choice“. Typ „ckeditor“ odpovídá formulářovému poli pro zadávání textu ve Wiki syntaxi.

U každé kolekce vlastního typu musí být také uvedena databázová entita, která tento typ zastupuje, a formulářové prvky v kolekci potom odpovídají atributům entity. Následující funkce znázorňuje propojení formuláře kolekce s příslušnou entitou.

```
1  public function setDefaultOptions(OptionsResolverInterface $resolver) {
2          $resolver->setDefaults(array(
3              'data_class' => 'Test\TestBundle\Entity\MoreOptionsAnsw',
4          ));
5      }
```

Postup vytváření kolekcí popsaný výše se aplikuje na všechny typy otázek, případně i na jejich odpovědi. Vytvořené kolekce se přidají do výsledného hlavního formuláře jako obyčejné formulářové pole.

Přidání více otázek do stejného formuláře je dále potřeba vyřešit na straně klienta. Základem je definice následujícího kódu v HTML šabloně a použití vlastností prototype definované při přidání kolekcí do formuláře.

```
1  <div class="collection" id="form-prototypes"
2   data-prototype-moreopt=
3   "{{_self.moreopt_prototype(form.moreOptionsQsts.vars.prototype)|e }}"
4   data-prototype-moreoptansw=
5   "{{_self.moreoptansw_prototype
6   (form.moreOptionsQsts.vars.prototype.children['answers'].vars.prototype) | e }}"
7   . . .
8  >
9          {% for qst in form.moreOptionsQsts %}
10             {{ _self.moreopt_prototype(qst) }}
11         {% endfor %}
12         {% for answ in form.moreOptionsQsts.vars.prototype.children['answers'] %}
13             {{ _self.moreoptansw_prototype(answ) }}
14         {% endfor %}
15         . . .
16     </div>
```

Blok kódu (řádky 1-13) slouží k zajištění přidávání nových otázek či odpovědí a určuje, jak bude vypadat formulář pro přidané elementy. Samotný blok se na stránce nezobrazí. Důležitou věcí je zde definice prototypů. Atribut `prototype` v sobě obsahuje kompletní HTML kód formuláře pro každou otázku a slouží v podstatě jako šablona pro její vytváření. Každý typ otázky má definovaný vlastní prototyp, podle kterého je možné otázku rozpoznat. Dále následuje ukázka formulářového pole uvnitř atributu `prototype`.

```
1  <textarea id="form_moreOptionsQsts____moreopt_prot____name"
2  name="form[moreOptionsQsts][__moreopt_prot__][name]" required="required">
3  </textarea>
```

Při přidávání nové otázky se pomocí JavaScriptu získá obsah příslušného atributu `prototype`, který se obalí do nového `div` bloku a přidají se k němu další HTML elementy, například tlačítko smazat. Pomocí JavaScriptu se také

zjistí počet již existujících otázek na stránce a název prototypu ve formulářových polích nové otázky se změní na její pořadové číslo získané z celkového počtu otázek. Ukázka implementace tohoto kroku je vidět v následujícím kódu.

```
var prototypes = $('#form-prototypes');
var qstPrototype = prototypes.attr('data-prototype-moreopt');
qstPrototype = qstPrototype.replace(/__moreopt_prot__/g, moreoptCount);
var newQuestion = $('<div class="form-group qst sortableEl ui-state-default"
                    id="moqst-' + moreoptCount + '"></div>').html(qstPrototype);
```

Příklad nového formulářového pole, které bylo vygenerováno po provedení JavaScriptového přidání otázky je uveden níže.

```
<textarea id="form_moreOptionsQsts_0_name"
                name="form[moreOptionsQsts][0][name]" required="required">
</textarea>
```

Nově vygenerovaný blok se následně vloží do stránky v podobě formuláře pro přidávání otázky a uživatel poté může zadávat a spravovat otázku pomocí tohoto formuláře. Po odesílání kompletního formuláře se všemi vygenerovanými otázkami se iterativně zpracují vnořené formuláře-kolekce a výsledek se uloží do databáze.

Kapitola 5

Testování

Testování aplikace je důležitou součásti vývoje, které pomáhá odhalit chyby a nedodělky ještě před spuštěním aplikace. Existuje několik druhů testování, které se používají při testování webových stránek, a jejich stručný popis je uveden dále [15].

Funkční testování

Tento druh testování je určen k testování funkcionalit, které stránka nabízí. Provádí se zde kontrola korektnosti odkazů, validace formulářů, kontroluje se, jestli existují „mrtvé“ stránky nebo jestli správně funguje přesměrování. Kromě toho se také provádí testování komunikace s databází nebo práce s cookies.

Testování použitelnosti

Testování použitelnosti má odhalit, jestli je aplikace uživatelsky přívětivá, jestli je práce s ní intuitivní a případně jaké problémy může mít potenciální uživatel při práci s aplikací. Testování použitelnosti se provádí s několika vybranými uživateli, kteří mají za úkol procházet aplikací a snažit se splnit požadované úkoly.

Testování rozhraní

Účelem tohoto testování je zjistit, jestli rozhraní aplikace dokáže korektně komunikovat se serverovým a databázovým rozhraním. Aplikace musí správně reagovat na přerušení spojení a poskytovat uživateli srozumitelnou hlášku o případných komplikacích.

Testování kompatibility

Testování kompatibility se provádí v různých prohlížečích, případně na různých zařízeních a jeho účelem je zjistit, zda se aplikace všude zobrazuje korektně a nejsou omezené její funkcionality.

Výkonnostní testování

Při výkonnostním testování se provádí různé zátěžové testy, které mají zjistit, jestli je aplikace schopná obsloužit velké množství uživatelů a případně

jestli je odolná vůči zátěžovým útokům. Provádí se zde testy rychlosti, které odhalují, jak rychle se stránky načítají a jak dlouho trvá vyřízení požadavku serveru.

Testování bezpečnosti

Testování bezpečnosti slouží k odhalení potenciálních rizik při používání aplikace. Data, kterými aplikace disponuje, musí být dobře chráněna proti různým útokům typu Cross-Site Scripting nebo Injection.

Tato kapitola je věnována především funkčnímu testování a testování použitelnosti. Vzhledem k tomu, že požadavkem zadavatele bylo neprovádět veřejné testování, jsou pro testování připraveny potřebné podklady, ale nejsou poskytnuty výsledky.

5.1 Funkční testování

Testování funkčnosti aplikace se provádí pomocí testovacích scénářů a testovacích případů. Testovací scénáře definují, která funkcionalita má být otestována, testovací případy poté určují, jak bude testování prováděno. Testovací scénáře vycházejí z jednotlivých případů užití a skládají se z více testovacích případů [16].

Testovací případ popisuje konkrétní akce, které je možné s aplikací provádět. Tyto akce mohou být validní či nikoliv a pomocí testovacích případů lze sledovat, jestli systém reaguje na akce uživatele podle očekávání. Testovací případ je reprezentován pomocí tabulky a obsahuje následující informace:

ID - pořadové číslo kroku v testovacím případu

Krok - popis kroku v testovacím případu

Očekávaný výsledek - popis výsledku, který je očekáván po provedení kroku

Výsledek - popis výsledku, který byl skutečně dosažen po provedení kroku

Rozhodnutí - rozhodnutí, jestli je test splněn nebo ne

Dále jsou popsány scénáře, podle kterých by mělo být provedeno testování. U všech scénářů se předpokládá, že uživatel je přihlášen do systému. Tato kapitola obsahuje seznam testovacích scénářů. Testovací případy k vybraným scénářům jsou uvedeny v příloze B.

5.1.1 Testovací scénáře

1. **Zobrazit seznam složek**

   Uživatel si může zobrazit složky, které se nachází v bance otázek. Scénář slouží k ověření, zda je uživatel přesměrován na odpovídající stránku a zda se banka otázek na této stránce zobrazí korektně.

2. Vytvořit složku v bance otázek

Každý uživatel by měl mít možnost vytvořit složku pro ukládání otázek ve vlastní bance otázek. Scénář musí ověřit, jestli lze vytvořit složku pouze s validním názvem a jestli přidání složky v tomto případě proběhne úspěšně. Testovací případy ke scénáři jsou k nalezení v příloze B.1.

3. Vytvořit podsložku v bance otázek

Uživatel musí mít možnost vytvořit podsložku uvnitř dříve vytvořené složky. Scénář musí ověřit, jestli je možné vytvořit podsložku pouze s validním názvem a jestli přidání podsložky v tomto případě proběhne úspěšně a podsložka se zařádí do správné složky.

4. Editovat složku v bance otázek

Aplikace musí umožnit editaci dříve vytvořené složky. Scénář slouží k ověření, zda editace složky proběhne úspěšně a uloží se pouze složka s validním názvem. Pokud se uživatel pokusí o vytvoření nevalidní složky, musí mu být zobrazena chybová hláška a provedené změny se neuloží.

5. Smazat složku z banky otázek

Aplikace musí umožňovat smazání složky a veškerého jejího obsahu. Scénář slouží k ověření, zda se potřebná složka smaže pouze po potvrzení uživatele a zda mazání složky a jejího obsahu proběhne úspěšně.

6. Přidat otázky do složky

Aplikace musí umožnit přidat otázky do vybrané složky. Scénář slouží k ověření, jestli je uživateli poskytnutá stránka pro zadávání otázek, na které se mu nabídne formulář pro vytvoření nových otázek. Pokud chce uživatel uložit otázky, musí se uložit pouze validní otázky.

7. Vytvořit otázku typu „výběr z více možností“

Uživatel může vytvořit otázku typu „výběr z více možností“. Scénář je určen k ověření chování formuláře pro vytvoření otázky. Pouze validní otázka s validním počtem odpovědí by měla být vytvořena. Testovací případy ke scénáři jsou uvedené v příloze B.2.

8. Vytvořit otázku typu pravda/nepravda

Aplikace musí umožnit uživateli vytvořit otázku typu „pravda/nepravda“. Scénář slouží k otestování formuláře pro zadávání otázky.

9. Vytvořit numerickou otázku

Uživatel může vytvořit numerickou otázku, u které se jako odpověď uvede samotná hodnota a interval tolerance. Scénář slouží k ověření, jestli jde zadat pouze validní otázka.

10. Vytvořit otázku typu „zlomek“

Aplikace musí umožňovat vytvoření otázky typu „zlomek“, u které se jako odpověď udává čitatel a jmenovatel správného zlomku. Testovací scénář pomůže ověřit, zda je možné vytvořit otázku pouze s validně vyplněnými daty.

11. Vytvořit otázku typu „přiřazení“

Uživatel může vytvořit otázku typu „přiřazení“. Scénář je určen k ověření chování formuláře pro vytvoření otázky. Pouze validní otázka, která má požadovaný počet validních odpovědí, může být vytvořena. Scénář má také ověřit přidávání různých typu přiřazovacích páru a jejich validaci dle zvoleného typu.

12. Zobrazit otázky ve složce

Scénář slouží k ověření, zda je možné zobrazit kompletní seznam otázek z vybrané složky a zda je k těmto otázkám poskytnut seznam akcí, které je možné s otázkami provádět.

13. Zkopírovat otázky z vybrané složky do jiné složky

Scénář by měl ověřit fungování kopírování otázek mezi složkami. Otázky by měly zůstat ve zdrojové složce a jejich kopie by měly přibýt ve správné cílové složce. Pokud se kopírování nezdaří, měla by uživateli být zobrazena příslušná hláška.

14. Přesunout otázky z vybrané složky do jiné složky

Scénář by měl ověřit, jestli přesouvání otázek z jedné složky do jiné funguje správně. Otázky by se měly odebrat ze zdrojové složky a poté se objevit ve správné cílové složce. Pokud se přesouvání nezdaří, měla by být uživateli poskytnuta příslušná hláška.

15. Přidat test ke kurzu/kapitole

Scénář slouží k ověření, jestli je možné přidat závěrečný test ke kapitole či kurzu. Test se musí vytvořit pouze pokud je validní, v opačném případě má být uživatel informován o chybách při zadávání testu.

16. Přidat otázky do testu

Aplikace musí umožňovat přidat otázky do vybraného testu. Scénář slouží k ověření, jestli je uživateli zobrazena stránka pro zadávání otázek, na které je formulář pro vytvoření nových otázek. Pokud uživatel chce uložit otázky, musí být uloženy pouze validní otázky.

17. Editovat test

Scénář je určen k ověření, zda jde editovat test a pokud ano, zda se uloží pouze validní změny provedené při změně testu. Testovací případy ke scénáři jsou uvedené v příloze B.3.

18. Publikovat test

Uživatel má možnost publikovat vytvořený test. Scénář musí ověřit, zda jde publikovat pouze validní test, který obsahuje potřebný počet testovacích otázek. V opačném případě musí být uživatel informován o zákazu publikování testu a test není publikován.

5.2 Testování použitelnosti

Existuje více druhů testování použitelnosti. Jedním z nich je testování s uživatelem, které má zjistit, zda je uživatel schopen pracovat s aplikací intuitivně a bez problémů. Pro účely testování se vytváří seznamy úkolů, které se musí uživatel pokusit splnit. Během testování se zaznamenává celý průchod uživatele systémem a jeho reakce v určitých krocích. Po provedení testování se vyhodnotí úspěšnost různých úkolů. Na základě úspěšnosti se rozhodne, které případné chyby v aplikaci mají být opraveny podle jejich závažnosti.

Screener

Screener je dotazník, který se používá k výběru participantů ze skupiny uživatelů. Screener se skládá z veřejné a neveřejné částí. Veřejná část je dotazník, který vyplňují uživatelé. Neveřejná část popisuje očekávání od vyplnění veřejného screeneru, tj. požadované odpovědi, na jejichž základě jsou vybráni participanti. Vzhledem k tomu, že současný prototyp aplikace obsahuje pouze část mentora, tj. část pro zadávání testů, nikoli pro jejich absolvování, byla příprava testování zaměřena pouze na potenciální mentory. Veřejné a neveřejné screenery jsou k nalezení v příloze C.

Pre a post test dotazníky

Pre test dotazník je rozšířením screeneru a participant ho musí vyplnit před zahájením samotného testování. Pre test dotazník slouží k získání doplňujících informací tykajících se zkušeností participanta a jeho očekávání. Post test dotazník vyplní participant po skončení samotného testování. Tento dotazník slouží ke zjištění názoru participanta na testovanou aplikaci. Pre a post test dotazníky jsou uvedené v přílohách C.3 a C.4.

Seznam úkolů

Samotný proces testování spočívá ve splnění určitých úkolů uživateli. Úkoly se vytvářejí na základě funkcionalit, které jsou potřeba otestovat. Znění úkolu musí být obecné, úkoly se nerozepisují po krocích a uživatel potřebuje sám přijít na způsob splnění úkolu. Díky tomu je možné sledovat chování uživatele a odhalit případné problémy v uživatelském rozhrání. Seznam úkolů pro testování modulu Test je k nalezení v příloze D. Při testování modulu Test se počítá s tím, že uživatel je již přihlášen do aplikace.

Vyhodnocení testu

Po provedení testu se vyhodnotí jeho výsledky a výsledky pre a post test dotazníku. Na základě těchto výsledků lze posoudit, v jakém stavu se nachází

současná verze aplikace a které chyby je potřeba opravit pro její vylepšení. Některé postřehy uživatelů z post test dotazníku je možné použít k analýze budoucího vývoje aplikace.

Závěr

Hlavním cílem diplomové práce bylo navrhnout a realizovat modul Test portálu Mentica, který by měl sloužit k přidávání a správě testů a práci s testovacími otázkami. V rámci práce byl proveden rozbor zadání a rešerše existujících řešení ve světě e-learningu. Na základě těchto analýz vznikly funkční a nefunkční požadavky, které byly pokryty při sestavění návrhu aplikace. Dalším krokem po vytvoření návrhu byla implementace aplikace. Kvůli časovému omezení a přehodnocení určitých počátečních požadavků, nebyly některé požadované funkcionality realizovány. Výsledná aplikace je brána jako prototyp, který poslouží základem pro budoucí vývoj. Pro výsledný prototyp aplikace byly vytvořené podklady pro testování, díky kterým bude možné aplikaci otestovat. V následující kapitole jsou popsány splněné a nesplněné cíle při implementaci aplikace.

Splněné a nesplněné cíle

Vytvoření a správa testů

Výsledná aplikace umožňuje mentorovi vytvářet nové testy a přidávat je ke kapitole nebo ke kurzu. U každého testu se uvádí základní informace tykající se kritérií splnění testu. K testům je možné přidávat různé otázky a měnit jejich pořadí. Hotové testy lze publikovat nebo editovat, pokud již nebyly publikovány. Editování publikovaného testu je zakázáno a uživatel potřebuje vytvořit kopii tohoto testu, ze které se následně stane nový test.

Tento cíl lze považovat za splněný.

Vytvoření a správa více druhů otázek

Výsledná aplikace umožňuje přidání 5 typů otázek. Jsou to otázky typu „výběr z více možností“, „pravda/nepravda“, „přiřazování“, „zlomek“ a numerická otázka. Implementace otázek typu „slovní odpověď“ a „datum“ nebyla realizovaná. U otázky typu „přiřazování“ zbývá několik drobných nedostatků,

které je potřeba co nejdříve odstranit. Vytvořené otázky je možné editovat a při každé úpravě se vytvoří nová verze otázky.

Cíl je možné považovat za částečně splněný.

Vytvoření a správa složek v bance otázek (funkcionalita navíc)

Výsledná aplikace umožňuje mentorovi vytvářet a spravovat vlastní složky v rámci banky otázek. Do každé složky lze přidávat další podsložky nebo otázky. Mentorovi je poskytnut kompletní přehled složek a jejich obsahů. Tento požadavek nebyl zahrnut v zadání a přibyl během provedení analýzy.

Cíl je považován za splněný.

Správa otázek uvnitř složek (funkcionalita navíc)

Výsledná aplikace umožňuje kopírovat a přesouvat otázky mezi dvěma vybranými složkami. Tento požadavek přibyl po zavedení banky otázek a rozšiřuje seznam počátečních požadavků.

Cíl je možné považovat za splněný.

Generování testů

Generování testů z různých otázek podle vybraných kritérií nebylo naimplementováno. Avšak byl proveden návrh této funkcionality a připravená databáze pro jeho realizaci. S generováním testů se počítá do další verze aplikace.

Cíl je považován za nesplněný.

Absolvování testů z pohledu studenta

Absolvování testů z pohledu studenta bylo kompletně navrženo, ale jeho implementace nebyla provedena.

Cíl je považován za nesplněný.

Budoucí vývoj

Aplikace naimplementovaná v rámci diplomové práci je základním prototypem pro budoucí aplikaci. Nehledě na to, že některé funkcionality nebyly v prototypu realizovány, provedená analýza a návrh jsou kompletní a jsou dobrým základem pro pokračování ve vývoji a implementování finální verze aplikace. V budoucnu se počítá s rozšířením výukových materiálů, v případě testů například o přidání nových typů otázek. Otázky by bylo možné také sdílet mezi více mentory a importovat z externích souborů. V úvahu také připadá možnost generování písemných testů na základě jejich elektronické podoby. Důležitou kapitolou budoucího vývoje je realizace modulu statistik a gamifikace, které obohatí celý proces výuky a poslouží jako propagační prostředek pro aplikaci.

Literatura

[1] Tesař, J.: *E-learningový portál Mentica - modul uživatelský modul*. Diplomová práce, České vysoké učení technickév Praze, Fakulta informačních technologií.

[2] Matějka, J.: *E-learningový portál Mentica - modul správa kurzů*. Diplomová práce, České vysoké učení technickév Praze, Fakulta informačních technologií.

[3] Komárek, M.: Požadavky a jejich modelování pomocí případů užití. 2011, přednaška v rámci předmětu Úvod do softwarového inženýrství.

[4] Žikovský, P.: Návrh UI, prototypy. 2011, přednaška v rámci předmětu Návrh uživatelského rozhrání.

[5] Programming Language Popularity. Dostupné z: `http://langpop.com/`

[6] Plotz, M.: Principles Of Object Oriented Programming in PHP. Dostupné z: `http://www.htmlgoodies.com/beyond/php/article.php/3909681`

[7] W3Schools: HTML5 Browser Support. Dostupné z: `http://www.w3schools.com/html/html5_browsers.asp`

[8] W., A.: CSS3 Introduction – New Features, What it Can Do, and Resources. Dostupné z: `http://www.1stwebdesigner.com/css3-introduction/`

[9] DB-Engines. Dostupné z: `http://db-engines.com/en/`

[10] Skvorc, B.: Best PHP Framework for 2015 – SitePoint Survey Results. 2015. Dostupné z: `http://www.sitepoint.com/best-php-framework-2015-sitepoint-survey-results/`

[11] 2, V. D.: *About Doctrine Project*. 2014. Dostupné z: `http://www.doctrine-project.org/about.html`

[12] Sass, V.: *Sass (Syntactically Awesome StyleSheets)*. 2015. Dostupné z: http://sass-lang.com/documentation/file.SASS_REFERENCE.html

[13] Vasiliev, A.: 3 Patterns for Storing Tree Structures in the RDBMS. 2013. Dostupné z: http://railsware.com/blog/2013/07/15/storing-tree-structures-in-the-rdbms/

[14] PJE: The simplest(?) way to do tree-based queries in SQL. 2010. Dostupné z: http://dirtsimple.org/2010/11/simplest-way-to-do-tree-based-queries.html

[15] Web Testing: Complete guide on testing web applications. 2015. Dostupné z: http://www.softwaretestinghelp.com/web-application-testing/

[16] Test Cases Vs Test Scenarios – Which is Better? (My Experience). 2015. Dostupné z: http://www.softwaretestinghelp.com/test-cases-vs-test-scenarios/

Příloha **A**

Seznam použitých zkratek

PHP PHP: Hypertext Preprocessor

HTML HyperText Markup Language

CSS Cascading Style Sheets

Ajax Asynchronous JavaScript and XML

Sass Syntactically Awesome Style Sheets

API Application Programming Interface

ORM Object-relational mapping

MVC Model-view-controller

SQL Structured Query Language

DQL Doctrine Query Language

Příloha B

Testovací scénáře

B.1 Testovací scénář 2: Vytvořit složku v bance otázek

Testovací případ 1: Vytvoření validní složky

Předpoklad: uživatel se nachází na stránce banky otázek.
Testovací případ je znázorněn v tabulce B.1.

ID	Krok	Očekávaný výsledek	Výsledek	Rozhodnutí
1	Kliknout na ikonku „Vytvořit novou složku“.	Uživateli je zobrazeno okno s formulářem pro zadání názvu složky.		
2	Zadat validní název složky a kliknout na tlačítko „Uložit“.	Zavře se okno s formulářem, uživateli se zobrazí původní stránka s výpisem složek a hláška o úspěšném přidání složky, v seznamu složek se objeví nová složka.		

Tabulka B.1: Testovací případ 1: Vytvoření validní složky

Testovací případ 2: Vytvoření složky s nevalidním názvem

Předpoklad: uživatel se nachází na stránce banky otázek.
Testovací případ je znázorněn v tabulce B.2.

ID	Krok	Očekávaný výsledek	Výsledek	Rozhodnutí
1	Kliknout na ikonku „Vytvořit novou složku“.	Uživateli je zobrazeno okno s formulářem pro zadání názvu složky.		
2	Zadat nevalidní název složky nebo nechat název složky prázdný a kliknout na tlačítko „Uložit“.	Uživateli se zobrazí chybová hláška a uživatel je požádán o opětovné zadání validního názvu v uvedeném formátu.		

Tabulka B.2: Testovací případ 2: Vytvoření složky s nevalidním názvem

B.2 Testovací scénář 7 : vytvoření otázky typu „výběr z více možností“

Testovací případ 1: Vytvoření validní otázky typu „výběr z více možností“

Předpoklad: uživatel se nachází na stránce pro vytvoření otázek.
Testovací případ je znázorněn v tabulce B.3.

ID	Krok	Očekávaný výsledek	Výsledek	Rozhodnutí
1	Kliknout na ikonku „Vytvořit otázku typu výběr z více možností“.	Na stránce se objeví formulář pro zadání otázky.		
2	Zadat validní údaje do formulářových políček včetně vyplnění možných variant odpovědí.	Otázka je zadána a uživatel může přidat další otázku nebo uložit již přidané.		

Tabulka B.3: Testovací případ 1: Vytvoření validní otázky typu „výběr z více možností“

Testovací případ 2: Přidání a odebírání odpovědí do/z otázky typu „výběr z více možností“

Předpoklad: uživatel má k dispozici formulář pro vytvoření otázky typu „výběr z více možností“.

Testovací případ je znázorněn v tabulce B.4.

ID	Krok	Očekávaný výsledek	Výsledek	Rozhodnutí
1	Kliknout na tlačítko „Přidat další odpověď“.	Do formuláře na vytvoření otázky se přidají další políčka pro vytvoření nové odpovědi.		
2	Zadat validní údaje do nových formulářových políček.	Odpověď je přidána a uživatel může přidat další odpověď nebo smazat existující.		
3	Vybrat odpověď a kliknout na tlačítko „Smazat odpověď“.	Uživateli se zobrazí okénko s požadavkem o potvrzení smazání odpovědi.		
4	Potvrdit smazání odpovědi.	Potvrzovací okénko se zavře, zobrazí se stránka pro zadávání otázek, smazaná otázka na stránce již není.		

Tabulka B.4: Testovací případ 2: Přidání a odebírání odpovědí do/z otázky typu „výběr z více možností“

Testovací případ 3: Vytvoření nevalidní otázky typu „výběr z více možností“

Předpoklad: uživatel se nachází na stránce pro vytvoření otázek.

Testovací případ je znázorněn v tabulce B.5.

B.3 Testovací scénář 17 : editace testu

Testovací případ 1: Editovat nepublikovaný test

Předpoklad: uživatel se nachází na stránce vybraného testu.

Testovací případ je znázorněn v tabulce B.6.

ID	Krok	Očekávaný výsledek	Výsledek	Rozhodnutí
1	Kliknout na ikonku „Vytvořit otázku typu výběr z více možností“.	Na stránce se objeví formulář pro zadání otázky.		
2	Zadat nevalidní údaje do formulářových políček nebo nevyplnit požadována údaje (včetně možných variant odpovědí).	Formulářová pole, která neodpovídají požadovanému formátu se označí červeně a uživatel je požádán o opravu těchto polí. Dokud se neprovede úprava údajů podle požadavků, není možné přejít k přidání další otázky nebo uložit již vytvořené otázky.		
3	Zadat pouze jednu odpověď nebo nevyplnit žádnou.	Uživateli je zobrazeno upozornění, že minimální požadovaný počet odpovědí je 2 a uživatel je požádán a doplnění potřebného počtu odpovědí. Dokud se neprovede přidání odpovědí podle požadavků, není možné přejít k přidání další otázky nebo uložit již vytvořené otázky.		

Tabulka B.5: Testovací případ 3: Vytvoření nevalidní otázky typu „výběr z více možností“

ID	Krok	Očekávaný výsledek	Výsledek	Rozhodnutí
1	Kliknout na tlačítko „Editovat test“.	Ve vyskakujícím okně se objeví formulář pro editaci testu s předvyplněnými poli.		
2	Zadat validní údaje do formulářových políček a kliknout na tlačítko uložit.	Test je upraven, uživateli se znovu zobrazí stránka s výpisem testu, který obsahuje změněné údaje.		

Tabulka B.6: Testovací případ 1: Editovat nepublikovaný test

Testovací případ 2: Editovat publikovaný test

Předpoklad: uživatel se nachází na stránce vybraného testu, který je publikován.

Testovací případ je znázorněn v tabulce B.7.

ID	Krok	Očekávaný výsledek	Výsledek	Rozhodnutí
1	Kliknout na tlačítko „Editovat test“.	Uživateli se zobrazí hláška, že test již byl zveřejněn a není možné ho editovat. Uživateli jsou poskytnuty na výběr dvě možnosti: vytvořit nový test, který bude kopií existujícího, nebo zrušit editaci.		
2	Zvolit možnost vytvoření kopie testu.	Uživatel je přesměrován na stránku nově vytvořené kopie testu, která nebyla publikována a uživatel může tento test editovat podle testovacího případu 1.		

Tabulka B.7: Testovací případ 2: Editovat publikovaný test

Testovací případ 3: Editovat nepublikovaný test (nevalidní údaje)

Předpoklad: uživatel se nachází na stránce vybraného testu.

Testovací případ je znázorněn v tabulce B.8.

ID	Krok	Očekávaný výsledek	Výsledek	Rozhodnutí
1	Kliknout na tlačítko „Editovat test“.	Ve vyskakujícím okně se objeví formulář pro editaci testu s předvyplněnými poli.		
2	Zadat nevalidní údaje do formulářových políček nebo smazat údaje z povinných polí a kliknout na tlačítko uložit.	Uživateli je poskytnutá chybová hláška a uživatel je požádán o znovu vyplnění údajů podle uvedeného formátu. Změny v testu se neprovedou, dokud nebude formulář validní.		

Tabulka B.8: Testovací případ 3: Editovat nepublikovaný test (nevalidní údaje)

Příloha C

Screenery a pre/post test dotazníky

C.1 Veřejný screener

1. Vaše pohlaví
   a) Muž
   b) Žena
2. Váš věk
   a) 18-25
   b) 26-35
   c) 36-45
   d) 45-55
   e) Více než 55
3. Dosažené vzdělání
   a) Základní
   b) Střední
   c) Vyšší odborné
   d) Vysoké
4. Obor vzdělávání
   a) Technický
   b) Humanitní
   c) Přírodní vědy

d) Právo

e) Ekonomika

f) Jiné

5. Jak byste ohodnotil/a svoji počítačovou znalost?

   a) Začátečník (používám počítač na běžné uživatelské úrovni, dokážu se zorientovat v prostředí, ale v případě problémů vyhledávám technickou pomoc)

   b) Pokročilý (počítač používám běžně, dokážu samostatně vyřešit některé problémy a zvládám instalaci softwaru)

   c) Velmi pokročilý (orientuju se ve světě počítačů, mám zkušenosti s programováním)

   d) Znáte a používáte online vzdělávací portály?

   e) Znám a mám zkušenost s jejich použitím

   f) Znám, ale nikdy jsem je nevyzkoušel/a

   g) Neznám

6. Máte zkušenost s vytvářením vlastních testů?

   a) Ano

   b) Ne

7. Pokud jste zodpověděli Ano na předchozí otázku, jaké druhy testů jste vytvářel/a?

   a) Písemné

   b) Elektronické

   c) Obojí

C.2 Neveřejný screener

Neveřejný screener je znázorněn v tabulce C.1

C.3 Pre test dotazník

1. Co si představujete pod pojmem online vzdělávání a co byste čekal/a od portálu poskytujícího online vzdělání?

2. Máte již zkušenost s online vzdělávacími aplikacemi? Pokud ano, které aplikace to jsou?

C.4 Post test dotazník

1. Jaký máte dojem z aplikace?
2. S čím jste během testování měl/a největší problém?
3. Používal/a byste podobnou aplikaci v budoucnu?

Otázka	Očekávané odpovědi
1. Vaše pohlaví	3x muž 3x žena
2. Váš věk	2x 26-35 2x 36-45 2x 45-55
3. Dosažené vzdělání	4x Vysoké 2x Vyšší odborné
4. Obor vzdělávání	1x Technické 1x Humanitní 1x Přírodní vědy 1x Právo 1x Ekonomika 1x Jiné
5. Jak byste ohodnotil/a svoji počítačovou znalost?	2x Začátečník 3x Pokročilý 1x Velmi pokročilý
6. Znáte a používáte online vzdělávací portály?	2x Znám a mám zkušenost s jejich použitím 2x Znám, ale nikdy jsem je nevyzkoušel/a 2x Neznám
7. Máte zkušenost s vytvořením vlastních testů?	4x Ano 2x Ne
8. Pokud jste zodpověděli Ano na předchozí otázku, jaké druhy testů jste vytvářel/a?	2x Písemné 1x Elektronické 1x Obojí

Tabulka C.1: Neveřejný screener

Příloha D

Seznam úkolů pro testování použitelnosti

1. Vytvořit složku v bance otázek.
2. Vytvořit podsložku ve složce.
3. Editovat složku.
4. Přidat otázky do složky.
   a) Přidat otázku typu „výběr z více možností“.
   b) Přidat otázku typu „pravda/nepravda“.
   c) Přidat otázku typu „přiřazení“.
   d) Přidat numerickou otázku.
   e) Přidat otázku typu „zlomek“.
5. Zobrazit otázky ve složce.
6. Zkopírovat otázky ze složky do jiné složky.
7. Přesunout otázky ze složky do jiné složky.
8. Editovat otázku.
9. Smazat otázku.
10. Smazat složku.
11. Přidat test k vybrané kapitole vybraného kurzu.
12. Editovat test.
13. Publikovat test.

Příloha E

Instalační příručka

Pro zprovoznění aplikace na lokálním počítači je potřeba mít vlastní lokální server. Za tímto účelem lze použit hotová vývojová prostředí, která obstarají instalaci a základní konfiguraci server a databáze, například XAMPP.

1. Zkopírovat složku „mentica“ včetně jejího obsahu do kořenového adresáře lokálního serveru.
2. V konzoli spustit příkaz pro aktualizaci balíčků použitých v projektu: `php composer.phar update` (více informace zde `getcomposer.org/doc/`)
3. Vytvořit databázi „mentica“ v databázovém úložišti a naimportovat do vytvořené databázi tabulky a s iniciačními daty ze souboru `mentica.sql`.
4. Nastavit údaje pro přistup k databázi v konfiguračním souboru `app/config/parameters.yml`.
5. Pokud instalace a konfigurace proběhla úspěšně, je výsledná aplikace dostupná na adrese `http://localhost/mentica/web/app_dev.php`

Příloha F

Uživatelská příručka

Přihlášení

1. Přihlásit se pomocí přihlašovacích údajů (viz login.txt)

Banka otázek

1. Kliknout na odkaz „Banka otázek“ v hlavním menu.

2. Na zobrazené stránce kliknout na ikonku „plus“ – vytvořit novou složku. V novém dialogovém okně zadat název složky a kliknout na tlačítko „uložit“.

3. Z akcí dostupných pro nově vytvořenou složku zvolit „Editovat“. V novém dialogovém okně zadat upravený název složky a kliknout na tlačítko „uložit“.

4. Na stránce s výpisem složek vybrat složku a kliknout na ikonku „plus“ – přidat podsložku do složky. V novém dialogovém okně zadat název podsložky a kliknout na tlačítko „uložit“.

5. Na stránce s výpisem složky kliknout na název složky. Zobrazí se obsah složky včetně podsložek a odkazu na seznam otázek.

6. Kliknout na odkaz „[X] otázek“.

7. Ze seznamu otázek, který se objevil na stránce, vybrat potřebné otázky a zvolit akci „Zkopírovat do jiné složky“. Ze seznamu složek vybrat potřebnou složku a kliknout na tlačítko „Provést“.

8. Ze seznamu otázek, který se objevil na stránce, vybrat potřebné otázky a zvolit akci „Přesunout do jiné složky“. Ze seznamu složek vybrat potřebnou složku a kliknout na tlačítko „Provést“.

9. Ze seznamu otázek, který se objevil na stránce, vybrat jednu otázku a zvolit u ní akci editovat. V novém dialogovém okně upravit potřebné údaje a kliknout na tlačítko „uložit“.

10. Na stránce s výpisem složek vybrat složku a kliknout na ikonku „křížek“ – smazat složku. V novém dialogovém okně potvrdit nebo zrušit smazaní složky.

11. Na stránce s výpisem složek vybrat složku a kliknout na ikonku „plus“ – přidat otázky do složky. Na nové stránce vytvořit otázky (viz vytvoření otázek). Otázky se po vytvoření objeví v příslušné složce.

Vytvoření otázek

1. Na stránce pro vytvoření otázek zvolit „výběr z více možností“ z postranní nabídky otázek. V zobrazeném formuláři vyplnit potřebná pole. V případě potřeby přidat další odpověď nebo odebrat existující.

2. Na stránce pro vytvoření otázek zvolit „přiřazení“ z postranní nabídky otázek. V zobrazeném formuláři vyplnit potřebná pole. Zadat typ přiřazovacího páru a podle toho zadat odpovědí (buď zadat text, nebo nahrát obrázek). V případě potřeby přidat další odpověď nebo odebrat existující.

3. Na stránce pro vytvoření otázek zvolit „pravda/nepravda“ z postranní nabídky otázek. V zobrazeném formuláři vyplnit potřebná pole.

4. Na stránce pro vytvoření otázek zvolit „numerická otázka“ z postranní nabídky otázek. V zobrazeném formuláři vyplnit potřebná pole.

5. Na stránce pro vytvoření otázek zvolit „zlomek“ z postranní nabídky otázek. V zobrazeném formuláři vyplnit potřebná pole.

6. Na stránce pro vytvoření otázek minifikovat otázku kliknutím na ikonku šípky. Zobrazí se zkrácená verze otázky. Pro rozšíření otázky pro její editaci znovu kliknout na ikonku „šípky“.

7. Na stránce pro vytvoření otázek zvolit otázku a kliknout na ikonku „křížek“ – odebrat otázku. Otázka bude odebrána ze seznamu otázek.

8. Na stránce pro vytvoření otázek kliknout na tlačítko uložit.

Vytváření testů

1. V hlavním menu kliknout na odkaz „Kurzy“ a vybrat kurz, ke kterému bude přidán test, případně vybrat kapitolu.

2. Kliknout na tlačítko „Přidat test“. Na nové stránce zadat do formuláře potřebné údaje a kliknout na tlačítko „pokračovat“. Na stránce pro vytvoření otázek (viz vytvoření otázek) přidat otázky a kliknout na „Pokračovat“. Na další stránce zadat počet bodů potřebných pro splnění testu na základě celkového počtu bodů vypočítaného ze zadaných otázek.

3. Na stránce testu kliknout na tlačítko „Editovat“. V novém dialogovém okně upravit údaje ve formuláři a kliknout na tlačítko „Uložit“.

4. Na stránce testu kliknout na tlačítko „Přidat další otázky“. Na nové stránce vytvořit otázky (viz vytvoření otázek) a kliknout na tlačítko „Uložit“.

5. Na stránce testu kliknout na tlačítko „Publikovat“. V novém dialogovém okně potvrdit publikaci testu.

Příloha G

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `inst` ..... soubory potřebné k instalaci
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF