Sem vložte zadání Vaší práce.

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra teoretické informatiky

Diplomová práce

E-learningový portál Mentica uživatelský modul

Bc. Jaroslav Tesař

Vedoucí práce: Ing. Jiří Chludil

3. května 2015

Poděkování

Děkuji vedoucímu své diplomové práce Ing. Jiřímu Chludilovi za odborné vedení, cenné rady a připomínky. Za podporu při vypracovávání této práce děkuji rodině a přátelům. Velké poděkování patří kolegům z týmu, studentům Bc. Olze Budnik a Bc. Jiřímu Matějkovi.

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 3. května 2015 .....

České vysoké učení technické v Praze
Fakulta informačních technologií

Odkaz na tuto práci

Tesař, Jaroslav. *E-learningový portál Mentica
uživatelský modul*. Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2015.

Abstrakt

Diplomová práce se zabývá analýzou, návrhem a implementací uživatelského modulu online vzdělávací aplikace Mentica. Práce je součástí týmového projektu. Na základě analýzy existujících řešení jsou navrženy hlavní procesy výsledné aplikace. Modul je implementován v jazyce PHP s využitím frameworku Symfony 2. Výsledkem práce je modul, který umožňuje uživateli tvorbu a správu virtuálního účtu, potřebného k odebírání vzdělávacího obsahu.

Klíčová slova e-learning, online vzdělávání, uživatel, kurzy, testy

Abstract

The thesis deals with analysis, design and implementation of the users module, which is a part of online educational aplication named Mentica. This work is a part of a team project. Based on analysis of existing applications, main processes af the application are designed. Implementation is realized in PHP using Symfony 2 framework. The result of the thesis is a module that allows the user to create and manage virtual account required to retrieve educational content.

Keywords e-learning, online education, user, courses, tests

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Vzdělávání a získávání informací, které obohatí člověka o ceněné znalosti a zkušenosti je jednou z hlavních domén společnosti. I jednoduchou konverzací se člověk vzdělává a utváří si balíček hodnot, které definují kvalitu člověka jako tvůrce budoucnosti pro své děti a další generace. Forma vzdělávání se s časem mění a postupné přijímání informačních technologií do každodenního života poskytuje i pohodlnější, přístupnější, možná kvalitnější a mnohdy i levnější cestu ke znalostem. Trendem posledních let je takzvaný e-learning, který definovali Pollard a Hillage (2001) jako "poskytování a správu příležitostí ke vzdělávání a jejich podporu pomocí počítačové, síťové a internetové technologie za účelem napomáhání pracovnímu výkonu a rozvoji jedinců". [1]

Vyústěním narůstající poptávky po vzdělání a trendu využívání informačních technologií je vytvoření internetového portálu `Mentica`, který poskytne uživatelům online vzdělání jako alternativu a doplněk k prezenční výuce. Hlavním cílem není nahrazení, jako spíše rozšíření a doplnění učení a vzdělávání tváří v tvář.

S nápadem vytvořit vzdělávací aplikaci v prostředí internetu, která by poskytovala nějakou formu interakce s uživatelem, který by mohl předávat svoje zkušenosti a znalosti dál, odebírat obsah a testovat svoje znalosti formou vyplňování testů, přišel externí zadavatel této diplomové práce, BcA. Petr Mentberger. Prvotním cílem bylo vytvořit virtuální prostředí, jehož užíváním a získávání výsledků by si uživatel vybudoval jak znalostní základ využitelný v praxi, tak i získal hmatatelný výstup v podobě certifikátu nebo seznamu absolvovaných online kurzů nebo workshopů od kapacit v oboru.

Cíle aplikace

Primárním cílem této práce je vytvořit modul aplikace `Mentica`, který umožní a zjednoduší uživatelský přístup ke tvorbě a odebírání obsahu. Modul správy a tvorby obsahu v podobě kurzů nebo testů je součástí prací Bc. Jirky Matějky a Bc. Olgy Budnik. K vytvoření funkčního modelu bude využita analýza existujících řešení, funkčních a nefunkčních požadavků zadavatele, ze kterých vyplynou silné a slabé stránky. Ty budou využity pro navržení a následně implementaci a testování výsledného řešení.

Mezi hlavní požadavky zadavatele patří vytvoření systému registrace a správy uživatelů s důrazem na bezpečnost, také využití existujících služeb pro registraci a přihlašování, zejména pak služby `Facebook`. Z analýzy a implementace první verze aplikace za použití `PHP frameworku Nette` vyplynuly další požadavky, jako vytvoření systému správy skupin uživatelů, přes které se bude poskytovat obsah za konkrétních cenových a jiných podmínek nebo možnost správy kontaktů a vizitek uživatele v systému.

Modul by měl využívat moderních internetových technologií a služeb třetích stran, čímž bude dosaženo modularity na úrovni `RIA` (Rich Internet Application) aplikací.

Kapitola 1

Analýza

1.1 Analýza existujících řešení

V této kapitole jsou vybrány některé případy užití a jejich analýza při porovnání s již hotovými řešeními vzdělávacích internetových portálů. Projekt `Mentica` je svojí funkčností a zaměřením na autora i odběratele obsahu výjimečný, a tak se dají porovnat pouze části samostatně, nikoliv aplikace jako celek. V oblasti e-learningu jsou porovnány funkcionality následujících portálů.

open2study.com Projekt je svými poskytovanými službami velice blízký. Poskytuje však pouze možnost studia, nikoliv vytváření obsahu. Kromě kurzů, které jsou zdarma, si po registraci lze vyzkoušet i kurzy dotované. Na stránkách se lze v hojné míře setkat i s prvky gamifikace v podobě badgů nebo maličkostí jako zábavný způsob oznámení nenalezené stránky, tedy chyby s kódem 404. [2]

coursera.org Korporace Coursera Inc. spolupracuje s více jak stovkou vědeckých nebo vzdělávacích organizací z celkově dvaceti pěti zemí světa. [3] Hlavním přispěvatelem je například Stanfordská univerzita. Server nabízí také kurzy s ověřenými certifikáty za jeho dokončení. Hlavním obsahem kurzu jsou videa s probíraným tématem a přiložené slidy se zadanými úkoly pro procvičení látky. Na těchto stránkách také nelze obsah vytvářet.

codeschool.com Portál pro výuku programovacích jazyků, většinou v oblasti webu, jako například `Ruby, JavaScript, HTML a CSS`, ale i `iOS`, jazyk `R` nebo `SQL`. Web je zvláštní svým způsobem výuky, kdy po shlédnutí výukového videa může uživatel svoje vědomosti otestovat pomocí online emulátoru a překladače kódu s okamžitým zobrazením výsledku. [4] Zdarma je jen část obsahu, například pár prvních lekcí, kde se za

pokračování v kurzu musí zaplatit měsíční poplatek. Server ukazuje, že kvalita je důležitější než kvantita.

moodle (school.demo.moodle.net/) `Moodle` je poskytován zdarma jako otevřený softwarový balíček pro tvorbu výukových systémů a elektronických kurzů v prostředí internetu. K dispozici je řada modulů, z nichž se sestavuje obsah pro různé pedagogické situace. Je umožněná evidence studijních výsledků a zaznamenávání činností uživatelů v podrobných protokolech a statistikách. Software je koncipovaný jako pomůcka pro vzdělávací instituce a není tudíž zaměřený na širokou veřejnost. [5]
Charakter tohoto softwaru je odlišný od vyvíjeného portálu `Mentica`, a tak nebude podroben konkrétním bodům testování uvedených v další části.

Zkoumané funkcionality

1. Klasická registrace, požadované vstupy od uživatele
2. Použití sociálních sítí pro přihlašování / registraci
3. Forma zobrazení uživatelského profilu (vlastního i cizího)
4. Notifikace, informovanost uživatele o dění v systému
5. Ochrana osobních údajů
6. Podpora prohlížečů internetu
7. Způsoby placení a jejich technické postupy

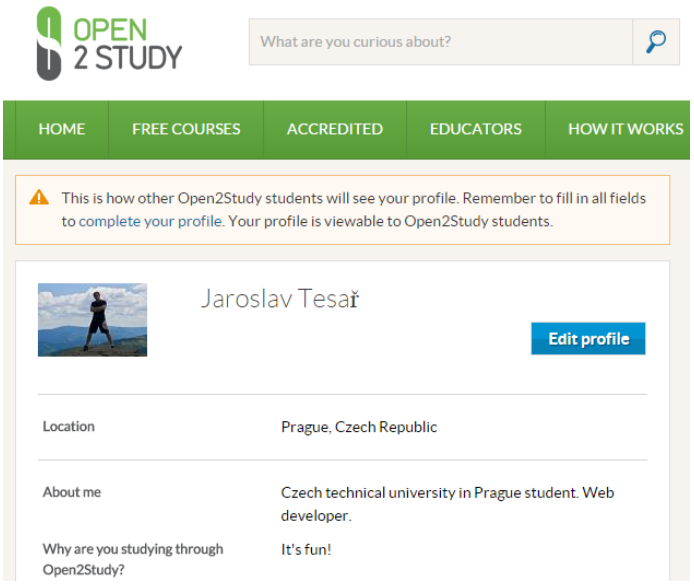
Při testování byly použity prohlížeče v uvedených verzích:

- `Internet Explorer` 11 (11.0.17)
- `Mozilla Firefox` 37.0.1
- `Google Chrome` 41.0.2272.118 m
- Výchozí prohlížeč internetu `Android OS v 4.4.2 KitKat`

1.1.1 open2study.com

Klasická registrace, požadované vstupy od uživatele Křestní jméno, příjmení, email, heslo, viditelnost profilu, přezdívka, souhlas s podmínkami užívání a etickým kodexem

Použití sociálních sítí pro přihlašování / registraci `Facebook`, `Google`, `LinkedIn`



Obrázek 1.1: Ukázka zobrazení profilu uživatele na portálu Open2Study.org

Forma zobrazení uživatelského profilu (vlastního i cizího) Uživatel vidí svůj profil tak, jak jej vidí ostatní uživatelé. 1.1 Emailová adresa použitá při registraci není viditelná. Pro zobrazení profilu je nutná registrace.

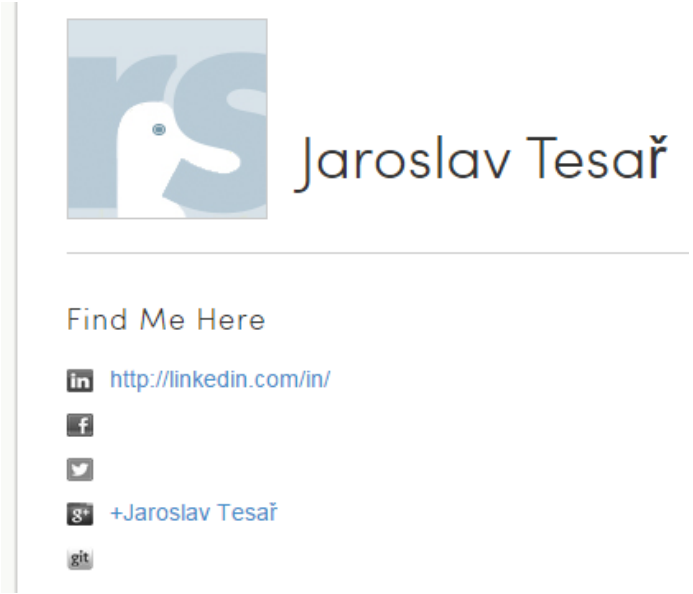
Notifikace, informovanost uživatele o dění v systému Uživatel je na domácí stránce svého profilu informován pouze o aktivitách odebíraného obsahu.

Ochrana osobních údajů Portál obsahuje rozsáhlou část s podmínkami užití. Zajímavá je část o věku registrovaného uživatele. Podmínky uvádějí, že osoby pod 13 let se na portálu nesmí registrovat, osoby pod 18 let pouze pod dohledem rodiče nebo učitele a osoby nad 18 let mohou obsah užívat volně.

Podpora prohlížečů internetu `Internet Explorer` v novějších verzích pouze jinak zobrazuje některé formulářové prvky, ale funkčnost stránek není nijak omezena. V prohlížečích `Firefox` i `Chrome` jsem nezaznamenal žádné problémy.

Způsoby placení a jejich technické postupy Online platby na serveru nejsou k dispozici. Lze se přihlásit k placenému kurzu, ale platbu je poté nutné provést převodem na uvedený australský účet.

Výhody Bezproblémová funkčnost na požadovaných prohlížečích. Propracované zobrazení profilu uživatele.



Obrázek 1.2: Zobrazení následku chybné validace vstupů od uživatele na coursera.org

Nevýhody Chybí online platby. Pro některé činnosti v systému nízká informovanost uživatele. (například chybí informace o zobrazení profilu cizím uživatelem)

1.1.2 coursera.org

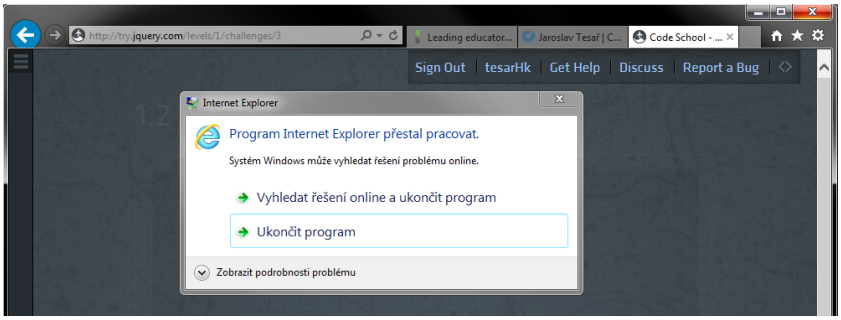
Klasická registrace, požadované vstupy od uživatele Celé jméno, emailová adresa, heslo. Souhlas s podmínkami užití a etickým kodexem je vyjádřen vytvořením profilu.

Použití sociálních sítí pro přihlašování / registraci Pouze `Facebook`. Chyba při odmítnutí získání emailové adresy z účtu `Facebooku`. Systém poté sice vyžaduje její zadání, ale při shodě adres ji odmítne jako nevalidní.

Forma zobrazení uživatelského profilu (vlastního i cizího) Uživatel má k dispozici tři úrovně viditelnosti. Soukromý profil, viditelný v rámci stránek a veřejný po celém webu. Chyba při napojení a následném zobrazení nějaké sociální sítě. (bílé znaky jsou validní vstupy)

Notifikace, informovanost uživatele o dění v systému Uživatel je o dění v systému informován pouze prostřednictvím emailové adresy. V profilu nejsou zaznamenávány žádné notifikace.

Ochrana osobních údajů Opět rozsáhlá část s podmínkami užívání, kde se uvádí mimo jiné informace o tom, že internet není 100% bezpečné pro-



Obrázek 1.3: Fatální chyba při otevření konkrétního kurzu na codeschool.com v prohlížeči Internet Explorer

středí a existuje riziko neautorizovaného přístupu třetích stran k osobním údajům.

Podpora prohlížečů internetu Zobrazení testovaného obsahu bylo na požadovaných prohlížečích totožné.

Způsoby placení a jejich technické postupy Certifikované kurzy jsou placeným obsahem. `Coursera` poskytuje zabezpečenou platební bránu od společnosti `PayPal` nebo lze platit kartou pěti různých značek. Platí se vždy konkrétní kurz.

Výhody Stránky jsou plně responzivní a v pořádku se zobrazují i na zařízení s operačním systémem Android.

Nevýhody Chyba editace profilu - mezery jako validní vstupy. 1.2 Chyba při registraci pomocí sociální sítě `Facebook` a odmítnutí získání emailové adresy. Zcela chybí notifikace o dění v systému.

1.1.3 codeschool.com

Klasická registrace, požadované vstupy od uživatele Emailová adresa, uživatelské jméno a heslo. Zcela chybí podmínky užití.

Použití sociálních sítí pro přihlašování / registraci `GitHub`, `Facebook`, `Google`.

Forma zobrazení uživatelského profilu (vlastního i cizího) Profil je vždy soukromý a obsahuje tedy jen nízké množství atributů. Zajímavý je způsob přístupu k profilovému obrázku, kde je využita služba `Gravatar`, která si za cíl dává sjednocení a jednoduchou správu profilových fotek na internetu.

Notifikace, informovanost uživatele o dění v systému Z pohledu rozsahu portálu je informovanost dostatečná, i když se opět jedná pouze o formu emailu. `Codeschool` totiž poskytuje přibližně 40 kurzů.

Ochrana osobních údajů Server obsahuje středně rozsáhlé podmínky užití inspirované podmínkami portálu `GitHub`. Text o ochraně osobních údajů je kompletní a informuje o způsobu nakládání s poskytnutými údaji a jejich správě.

Podpora prohlížečů internetu Při opakovaných pokusech o otevření interaktivního testu `jQuery` v prohlížeči `Internet Explorer` přestal prohlížeč pracovat. 1.3 Jde o nejnavštěvovanější kurz stránek. Prohlížeče `Mozilla Firefox` a `Google Chrome` pracují se stránkami v pořádku. Procházet interaktivní testy nelze ani ve výchozím prohlížeči internetu `OS Android`.

Způsoby placení a jejich technické postupy Většina obsahu je placená a zpřístupní se po zakoupení členství, které je omezeno časovým obdobím měsíce nebo roku. Platit lze platební kartou nebo pomocí služby `PayPal`.

Výhody Placení za období, možnost zvýhodněné ceny při větším týmu odběratelů.

Nevýhody Při registraci není nikde požadováno nebo oznámeno odsouhlasení podmínek užití. Prohlížeče `Internet Explorer` a prohlížeč na `Android OS` nezobrazí některé kurzy na `Codeschool`.

1.1.4 Shrnutí analýzy konkurenčních aplikací

Při testování uvedených konkurenčních aplikací, které nabízejí podobné služby, se vyskytly chyby obecného i specifického charakteru. Mezi závažnější funkční nedostatky lze zařadit označení prázdných vstupů od uživatele jako validní. Následky mohou být i zásadní pro samotnou funkci aplikace. Častou chybou bylo nedostatečné informování uživatele o dění a interakci v sytému, což může mít za následek jeho ztrátu. Dále se vyskytly chyby v registraci pomocí sociální sítě `Facebook` nebo problém se zobrazením aplikace různými prohlížeči, kterou považuji za chybu nejzávažnější.

1.2 Porovnání PHP Frameworků Nette a Symfony 2, výběr

Tato kapitola se věnuje porovnání dvou frameworků, které staví na scriptovacím programovacím jazyku `PHP`.

Pojem framework lze definovat mnoha způsoby. Nejpřesnější je výrok o uceleném souboru tematicky zaměřených knihoven, nástrojů, funkcí a návyků. Ty svojí znovu použitelností šetří programátorův čas a práci. Použití již hotového a ověřeného řešení (v rozumné míře) navíc dovolí neodvádět pozornost od hlavního cíle vyvíjeného programu nebo aplikace, rozdělit práci mezi větší množství vývojářů (ale i designérů, testerů apod.) a konzultovat již známé problémy s odborníky z celého světa.

Framework pomáhá pracovat lépe (řád v programování) a rychleji (využití obecných modulů). Při dodržení standardů a vývojářských pravidel je s frameworkem snadnější dlouhodobě udržovat a rozšiřovat projekty. V neposlední řadě se také zjednoduší začlenění aplikace do již existujících systémů.

Úvodem o Nette

Zakladatelem, původním autorem a osobností projektu je David Grudl, který roku 2008 vydal první verzi `MVC open-source` frameworku pod `GNU GPL` (původně `BSD`) licencí. O další rozvoj se stará organizace `Nette Foundtation`. Projekt je v České republice velice rozšířený. Autor samotný pořádá školení a neformální setkání příznivců `Nette` Frameworku, která jsou zdarma. Počet přispěvatelů jen lehce přesahuje sto a trend commitů na serveru `GitHub` je velice nestálý. Přesto `Nette` využívají hlavně české weby, například Slevomat, Čsfd, Mladá Fronta nebo Uložto. [6]

Úvodem o Symfony 2

Hlavou `Symfony` je Fabien Potencier, který se od roku 2004 o framework stará. Je `open-source` pod `MIT` licencí a jeho vývoj sponzoruje francouzská firma Sensio Labs. Komunita kolem projektu je na celosvětové úrovni a jen na serveru GitHub je přes tisíc sto přispěvatelů. Symfony 2 se pyšní kvalitní a podrobnou dokumentací a rozsáhlými návody. Na framworku zakládá například redakční systém `Joomla, Drupal`, dále `Yahoo!` nebo `Dailymotion` a z českého internetu pak stránky jazykové školy v Praze - Institut jazykového vzdělávání nebo stránky Katedry Kybernetiky na ČZU. [7]

Hodnotící kritéria

Při výběru frameworku jsme jako tým zohlednili naše zkušenosti s jednotlivými projekty. Zúčastnili jsme se dvoudenního školení Nette pro pokročilé s jeho zakladatelem Davidem Grudlem, v Nette jsme také psali naše bakalářské práce. Se Symfony 2 jsme se seznámili také díky mimoškolním aktivitám. Při výběru jsme se rozhodovali na základě porovnání následujících technických i netechnických aspektů.

Tabulka 1.1: Hodnocení frameworků Nette a Symfony 2 z hlediska kvality komunity a dokumentace

Nette	**Symfony**
Existence dokumentace k jednotlivým vydaným stabilním verzím frameworku.	
1 (ano)	1 (ano)
Průměrné stáří kapitol dokumentace aktuální stabilní verze frameworku. (databáze, formuláře, šablony, presenterer / controller)	
3 (týdny až měsíce)	1 (dny)
Počet přijatých přispěvatelů do vývoje frameworku na serveru `GitHub`.	
3 (105)	1 (1119)
Počet výsledků hledání klíčového slova (název frameworku) s více jak třemi odpověďmi na serveru StackOverflow. (nette answers:3 / symfony answers:3)	
5 (4 výsledky)	2 (1900 výsledků)
Výsledné známky se zohledněním lokalizace frameworků.	
3	**1**

- Komunita, kvalita a aktuálnost dokumentace
- Modelová vrstva pro práci s databází
- Vývoj, ladění a optimalizace kódu
- Správa formulářů (tvorba, užití v projektu, vykreslení)
- Uplatnění v praxi

V každém aspektu bude oznámkováno několik podrobnějších bodů číslem od jedné do pěti, kde číslo pět je, jako ve škole, nejhorší možná známka.

1.2.1 Komunita, dokumentace

Nedostatečná dokumentace je známým a často poukazovaným problémem frameworku Nette. `QuickStart` je neaktualizovaný i několik měsíců, některé části až rok. Při přechodu na novou verzi je popis technických novinek a změn prakticky nedohledatelný. Komunita i dokumentace je z většiny pouze česká, což

klade bariéry při expanzi frameworku do světa a případná pomoc od profesionálů ze zahraničí je tak téměř nemožná. Stav dokumentace se na druhou stranu v poslední době zlepšuje a brzy bude v přijatelné formě.

V tomto ohledu tak vítězí Symfony, které nabízí uživatelům podrobnou a často aktualizovanou knihu ke stažení a detailní dokumentaci. Framework je v angličtině, a tak se problémy vývojářů ve velkém množství probírají i na serverech jako je `StackOverflow`. Komunita je velká, vstřícná a nevnucuje, že Symfony 2 je nejlepším řešením mezi PHP frameworky. Daný `release model` a přesně stanovené období podpory jednotlivých verzí napomáhá udržování zpětné kompatability.

Pro aplikaci rozsahu diplomové práce, která se vyvíjí v týmu třech lidí, byl požadavek na kvalitní dokumentaci zásadní. Polehčující okolností stavu dokumentace a rozsahu komunity kolem frameworku Nette by mohla být česká lokalizace, ale stav projektu na serverech `GitHub` a `StackOverflow`, které se v oblasti správy a vydávání verzí a řešení problematiky stávají výchozím bodem, dává jasná čísla 1.1.

1.2.2 Model

Většinu času a úsilí nechávají programátoři na straně modelu. Jde o práci s daty aplikace, většinou míněno daty uživatelů, která je potřeba ošetřit, ukládat, spravovat a dále s nimi nakládat. Framework by měl nabízet bezpečné, ověřené, ale také pohodlné řešení přístupu do databázového uložiště. Vývojář může sice sáhnout po řešeních třetích stran, jako například `dibi, PDO, NotORM` Jakuba Vrány, `Doctrine` a dalších, ale každá knihovna řeší důležité detaily po svém a v tom může být kámen úrazu při snaze o stoprocentní spolupráci s frameworkem. Knihovny modelových vrstev se liší například v přístupu k datu a času, fulltextovému vyhledávání, práci s `autoincrementem` nebo ukládání velkého množství textu, kdy není možné obsah jen tak přiřadit do běžného SQL dotazu.

Symfony 2 používá v základu již zmiňovanou `ORM` knihovnu Doctrine 2, která zajišťuje mapování objektů na relační databázi. Knihovna Doctrine 2 je logicky postavená, systémová, průhledná a čistá. Klasické `SQL` dotazy lze nahradit `DQL` (`Doctrine Query Language`) notací, kde se místo názvů tabulek a atributů používají výhradně názvy entit a jejich členských proměnných. Výhoda spolupráce s frameworkem je právě ta, že napojení bylo provedeno samotnými autory Symfony 2 a je tak ověřené a konzistentní.

Framework Nette v čistém balíku nepoužívá žádnou knihovnu, ale staví na vlastním řešení přístupu k datům z databáze. Třída Connection tvoří vrstvu nad `PDO` a reprezentuje připojení k databázi. Pro práci s tabulkami slouží

Tabulka 1.2: Hodnocení frameworků Nette a Symfony 2 z hlediska modelové vrstvy

Nette	**Symfony**
Problematika získávání objektu (entity) a jeho dat.	
2 (`getter/setter` - méně možností získání konkrétního záznamu na objektu `Nette/Table`)	1 (`getter/setter` - metoda `findBy`)
Tvorba a správa schématu databáze, relací a omezení.	
5 (žádná - nutno ručně)	2 (notace ve třídě entity)
Počet přijatých přispěvatelů do vývoje frameworku na serveru `GitHub`.	
3 (105)	1 (1119)
Způsob práce s databázovou tabulkou.	
2- (API Selection Filtrace)	1 (`DQL`, vazby entit Doctrine 2)
Výsledné známky.	
3-	**1**

dvojice tříd `Context a Table`, které pomocí databázových dotazů voláním metody `query` mění obsah a strukturu dat. Databázová vrstva v Nette řeší i vazby mezi tabulkami, kde je například možné vypsat data z vazební tabulky `OneToMany` nebo `ManyToMany` a podobně. Jde o zajímavou mutaci dibi a některých funkcí `NotORM`. 1.1

Listing 1.1: Příklad dotazu do DB využitím metody třídy Nette/Database/-Context

```
$database->query(UPDATE users SET ? WHERE id=?, $data,
    $id);
```

Před porovnáním obou řešení je potřeba zdůraznit, že použitelnost a vhodnost nasazení té či oné modelové vrstvy - knihovny (ostatně jako frameworku obecně) závistí na velikosti a rozsahu vytvářeného projektu.

Samotná práce s existujícími daty je u obou řešení obdobná. Zásadní rozdíl je ve vytvoření a správě schématu, kde knihovna Doctrine 2 při použití notací a Symfony console dokáže tabulky na pozadí vytvořit zcela samostatně na rozdíl od databázové vrstvy Nette. V té je potřeba tabulky, jejich vazby a omezení,

Tabulka 1.3: Hodnocení frameworků Nette a Symfony 2 z hlediska uplatnění v praxi

Nette	**Symfony**
Počet pracovních nabídek na portálu `jobs.cz`. (březen 2015)	
25 nabídek	4 nabídky
Počet pracovních nabídek na portálu `cz.indeed.com`.	
420 nabídek	27 nabídek
Počet pracovních nabídek na portálu `careerjet.cz`.	
192 nabídek	30 nabídek
Výsledná známka nabídek práce PHP framework vývojáře pro **Českou republiku.**	
1	**4**

kromě dalších, vytvořit ručně. Knihovny, které jsem zmínil, se dají používat i ve spojení s Nette, ale jejich nasazení již vyžaduje další složitou práci a studii návodů či dokumentací. Porovnání z hlediska modelu je zachyceno v tabulce 1.2

1.2.3 Uplatnění v praxi

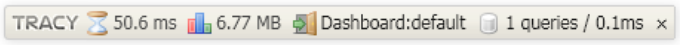
Toto hledisko rozhodně nebylo při výběru frameworku klíčové, nicméně sledování trendu nabídek považuji za zajímavé. V České republice stále vítězí Nette, což plyne i z jeho českých kořenů, ačkoliv v posledních letech se stále častěji objevují nabídky na práci v Symfony, které dokonce na svých stránkách projektu nabízí přihlášky k testům pro získání certifikátu. Je možné si vybrat až ze čtyřech tisíc testovacích center. V Praze je možné si vybrat z deseti. Lze tak zvýšit svoji cenu na mezinárodním trhu práce. 1.3

1.2.4 Vývoj a ladění

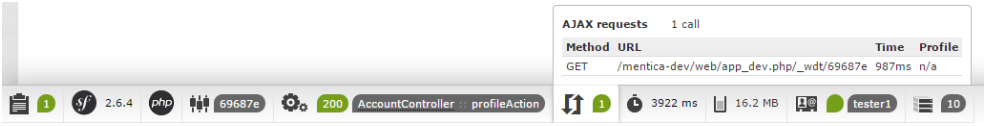
Oba frameworky poskytují vývojářům informace o průběhu akcí na pozadí požadavku. 1.4 Lze zobrazit dobu trvání zpracování požadavku a sestavení odpovědi, velikosti zpracované odpovědi v `megabytech`, název akce `presenteru` (respektive `controlleru`) a použité šablony pro vykreslení nebo počet a trvání dotazů do databáze. Tyto detaily jsou zobrazeny jak v `Tracy Nette` 1.4 tak v Symfony `web debug toolbaru` 1.5, který navíc zobrazuje podrobnosti

Tabulka 1.4: Hodnocení frameworků Nette a Symfony 2 z hlediska kvality vývojových a ladících nástrojů

Nette	**Symfony**
Odchytávání a zobrazení `AJAX` požadavků za běhu.	
ne	ano
Výpis a vysvětlení požadavku a odpovědi serveru.	
ne (nutnost instalace rozšíření)	ano
Identifikace místa volání akce v presenteru / controlleru a šabloně.	
ano	ano
Výsledná známka porovnání obou frameworků bez přidaných rozšíření. (další v textu)	
3	**1**



Obrázek 1.4: Ukázka Laděnky frameworku Nette - Tracy



Obrázek 1.5: Vývojový nástroj PHP frameworku Symfony 2 - bez Symfony profileru

o přihlášeném uživateli v rámci sezení, `AJAX` požadavky, název `route`, verzi PHP a samotného frameworku.

V detailu ladění Symfony 2 lze zobrazit kompletní data z formulářů a požadavku, nastavení a konfigurace projektu, zaslané emaily (v případě, že nějaké existují) nebo detaily práce knihovny Doctrine včetně vysvětlení použitých dotazů a spoustu dalších pomocníků. Samotný `Symfony profiler`, jak se ladící prostředí výsledku konkrétního požadavku nazývá, je rozsáhlý tak, že vývojář většinu nástrojů ani nemusí použít, ale nabízí náhled na problém ze všech různých směrů, takže si programátor může vybrat vlastní efektivní postupy.

Tabulka 1.5: Hodnocení frameworků Nette a Symfony 2 z hlediska tvorby a správy formulářů

Nette	**Symfony**
Počet podporovaných typů vstupních polí (input).	
bez známky - 15 typů	bez známky - 32 typů
Forma nastavení validace vstupů od uživatele.	
2 (`addCondition` přímo u definice vstupu)	1 (u definice vstupu, odděleně nebo u entity)
Vykreslování formulářů v šablonovacím systému. Výchozí / manuální.	
1 (Latte)	1 (Twig)
Napojení formuláře na datový model - mapování vstupů na data objektů.	
není možné	možné díky Entitám Doctrine 2
Výsledná známka správy formulářů frameworků bez dalších rozšíření.	
2	**1**

Ladící prostředí Nette se zaměřuje hlavně na kód jako takový. Zobrazuje detailní výpis chyb a výjimek s vizualizací konkrétního řádku, kde byla chyba vyvolána. Nette nabízí rozšíření do ladícího doplňku `Firebug` a `FireLogger` internetového prohlížeče Firefox v podobě komunikace Tracy s Firebugem přes HTTP hlavičky. Vývojář je ale omezený v testování a ladění pouze na jeden prohlížeč internetu, což se pro nasazení do produkčního režimu jeví jako nedostatečné.

1.2.5 Formuláře

Rozdíl v přístupu k práci s formuláři mezi frameworky spočívá v lehce rozdílné architektuře projektů. Kde je v Nette (MVP) použitý presenter, který komunikuje s modelem a prezentuje jej v lidsky přívětivé formě, v Symfony 2 (MVC) nastupuje známý `Controller`. Podstata obou je stejná, a to zpracovat požadavek a jeho data, provést potřebné operace s modelem a nakonec vytvořit odpověď, kterou předá šablonovacímu systému pro zobrazení. Rozdíl je v samotném vytváření a správě `presenterů` nebo-li `Controllerů`.

Symfony 2 je založeno na takzvaných `bundles`, kde každý reprezentuje modulárně oddělenou část schopnou samostatné činnosti v jiném projektu. Každý tento samostatný soubor může obsahovat až několik formulářových typů, které mohou být použity i ze všech ostatních balíků respektive Controllerů. Validace vstupů od uživatele lze nastavit i přímo u atributu entity, takže v případě použití více formulářů obsahujících stejný vstup není třeba definovat validační pravidlo vícekrát.

Framework Nette používá presentery, ale nedělí svou strukturu na jednotlivé balíky, a tak je používání formulářů na více místech omezeno na společného předka nebo takzvané továrničky, které ale musí obsahovat i pracování formuláře. Je tak sice možné formulář použít na více místech, ale vždy jen se stejným nebo jen složitě upravitelným zpracováním požadavku. Nette nabízí i méně výchozích validačních pravidel a typů vstupů, jako jsou číslo nebo string. Je ochuzeno o možnost správy entity ve formuláři, což je důsledek architektury modelové vrstvy.

Když pominu vykreslování, které je v obou řešeních velice podobné, intuitivní a praktické, tak s formuláři lépe pracuje framework Symfony 2, kde lze díky integraci knihovny Doctrine docílit flexibilnějšího a přímočařejšího sestavení vstupního bodu pro uživatele. Porovnání frameworků z hlediska tvorby a správy formulářů je zachyceno v tabulce 1.5.

1.2.6 Výsledky porovnání, zhodnocení

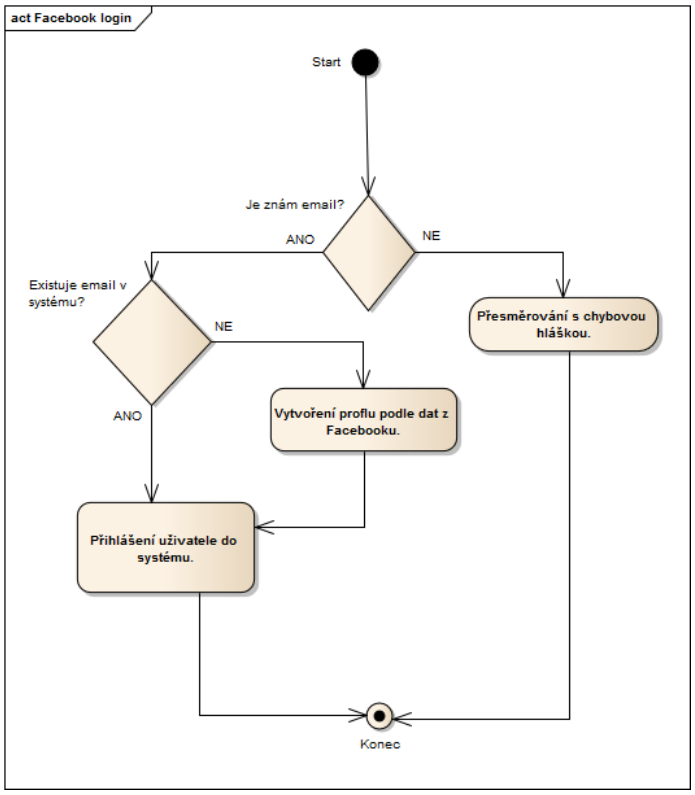
Známkování jednotlivých problémů a funkcionalit bylo prováděno na míru našeho týmového projektu. V případě, že by se jednalo o aplikaci cílenou na menší počet uživatelů nebo s řádově menší rozsahem, bylo by známkování k Nette frameworku přívětivější.

Pro náš projekt je tedy lepší použít framework Symfony 2 od Sensio Labs. Jedním z hlavních důvodů pro výběr tohoto projektu je pojetí modelové vrstvy v podobě Doctrine 2, která zjednodušuje i přechod z návrhu na samotnou implementaci.

Výčet porovnání a kompletní zhodnocení frameworků Nette a Symfony 2 je zachyceno v tabulce 1.6.

Tabulka 1.6: Kompletní shrnutí porovnání frameworků Nette a Symfony 2

Nette	**Symfony**
Komunita, kvalita a aktuálnost dokumentace	
3	**1**
Symfony 2 (i starší verze) je kvalitně a aktualizovaně dokumentový projekt se silnou komunitou.	
Modelová vrstva pro práci s databází	
3-	**1**
Hlavním nedostatkem Nette je vytváření a správa schématu databázového modelu.	
Uplatnění v praxi	
1	**4**
Programátor Symfony není na rozdíl od Nette vývojáře v ČR hojně poptávaný.	
Vývoj, ladění a optimalizace kódu	
3	**1**
Nette nabízí jen základní funkčnost ladění a pro získání dalších možností je třeba použít doplňků.	
Správa formulářů (tvorba, užití v projektu, vykreslení)	
2	**1**
Přístup k formulářům je obdobný. Nette nabízí pouze omezenější škálu možností přístupu k objektům.	
Výsledná známka správy formulářů frameworků bez dalších rozšíření.	
3+	**1-**



Obrázek 1.6: Postup přihlášení uživatele pomocí Facebooku

1.3 Integrace se sociální sítí Facebook

Systém má uživateli umožnit registraci pomocí sociální sítě `Facebook`. Celý proces samotné registrace, respektive přihlášení do systému, je zobrazen na následujícím diagramu. 1.6 Předpokladem je, že uživatel klikl na odkaz pro přihlášení pomocí brány `Facebooku` a vyplnil v ní správné přihlašovací údaje.

Diagram 1.6 zachycuje mimo jiné situaci, kdy po přihlášení uživatele pomocí brány sociální sítě `Facebook` není známa emailová adresa. (scénář je popsán v kapitole scénáře případů užití 1.9) Může se tak stát v případě, že se na `Facebooku` uživatel registroval pouze pomocí telefonního čísla a hesla, což je podle oficiální dokumentace možné. [8]

Existuje několik menších i větších knihoven, určených přímo pro PHP framework Symfony 2, které si za cíl kladou řešení integrace funkcí sociální sítě `Facebook`, jako je přihlášení, přidání tlačítka `like` nebo připojení komentářů na konkrétní stránku aplikace. Jednou ze známějších knihoven s větší komunitou přímo pro framework Symfony je `HWIOAuthBundle`, který podporuje autentikaci pro více než dvacet poskytovatelů, například právě `Facebook` nebo

`Google, GitHub, Instagram, Bitbucket` a další. Knihovna ale funkčně závisí na další a poskytuje jen funkci pro přihlašování pomocí služby `Facebook`.

`Facebook Developers`, služba pro kompletní správu propojených aplikací, poskytuje vývojový software pro `iOS, Android, JavaScript, PHP` a další. Kvalitní dokumentace s podrobnými návody a ukázkami kódu potvrdila rozhodnutí integrovat služby, které `Facebook` nabízí, vlastním způsobem. Pro implementaci bude použito `SKD` pro jazyk `JavaScript`, které také nabízí propojení s knihovou `jQuery`. [9]

1.4 Způsob obnovy hesla

Funkcí aplikací současného internetu je možnost obnovy hesla uživatele konkrétní aplikace. Důležitým prvkem je už samotná registrace účtu, kdy je uživatel vyzván k zadání hesla a opakovanému zadání, kdy se oba vstupy porovnají a je tak zajištěno, že nedošlo k překlepu na klávesnici. Daleko důležitější je však fakt, že heslo není zasláno na email a případný útočník, který získá do účtu pošty přístup, nemůže jednoduše vyčíst citlivé údaje. Samotné heslo je navíc šifrováno, viz sekce výběr šifrovacího algoritmu, takže jeho skutečnou podobu ani emailem zaslat nelze.

Následující kapitoly zachycují možné způsoby obnovy hesla a jejich rizika. Nejpřínosnějším zdrojem informací v této oblasti byl online článek od autora software `Pluralsight`, jménem Troy Hunt. [10]

Zaslání současného hesla emailem. Tento postup vyžaduje uložení hesla v databázi tak, jak jej zadal uživatel při registraci, tedy v "plain textu"nebo užitím chybně zvoleného šifrovacího algoritmu, kde lze získat původní obsah. Případné proniknutí do databáze útočníkem s následkem získání obsahu tabulky s přihlašovacími údaji tak nese nezměrné riziko, kdy si lze údaje buď rovnou přečíst nebo pomocí jednoduchých technik rozšifrovat.
Nepřijatelné, vysoké bezpečnostní riziko

Podání informace o existenci emailu v systému. Případ, kdy uživatel při obnově hesla zadá chybnou emailovou adresu a systém ho informuje o její neexistenci v systému, může být podle obsahu aplikace i nežádoucí. Lze totiž pak zjišťovat, zda je nějaká emailová adresa na serveru zaregistrována. Informace o registraci na webu s citlivým obsahem by mohla být dále zneužitelná. Řešením je zaslat zprávu i na v systému neexistující email, ale pouze s informací, že byla emailová adresa zadána na serveru při pokusu o obnovu hesla.
Pro aplikaci Mentica přijatelné, nehrozí poškození uživatele (nepoužito)

Zaslání nově vygenerovaného hesla. Tento bod je obdobou bodu prvního, ač jde o vyšší bezpečnost, protože heslo může být v databázi uloženo v šifrované nebo `zahashované` podobě, stále jde o bezpečnostní riziko. Emailová schránka by neměla být považována za dlouhodobě bezpečné uložiště citlivých dat, navíc v dnešní době synchronizace mezi více zařízeními připojených na internet ani z hlediska fyzického přístupu k účtu.
Nepřijatelné, nižší bezpečnostní riziko

Zaslání odkazu pro vytvoření nového hesla. Místo hesla samotného se na email uvedený ve formuláři pro obnovu hesla (pokud existuje v systému) zašle unikátní odkaz obsahující bezpečností token, který se dále kontroluje na serveru při zadávání nového hesla uživatelem. Nastavením nějaké časové platnosti `tokenu` se bezpečnost zvyšuje.
Vysoká bezpečnost, použito v aplikaci Mentica

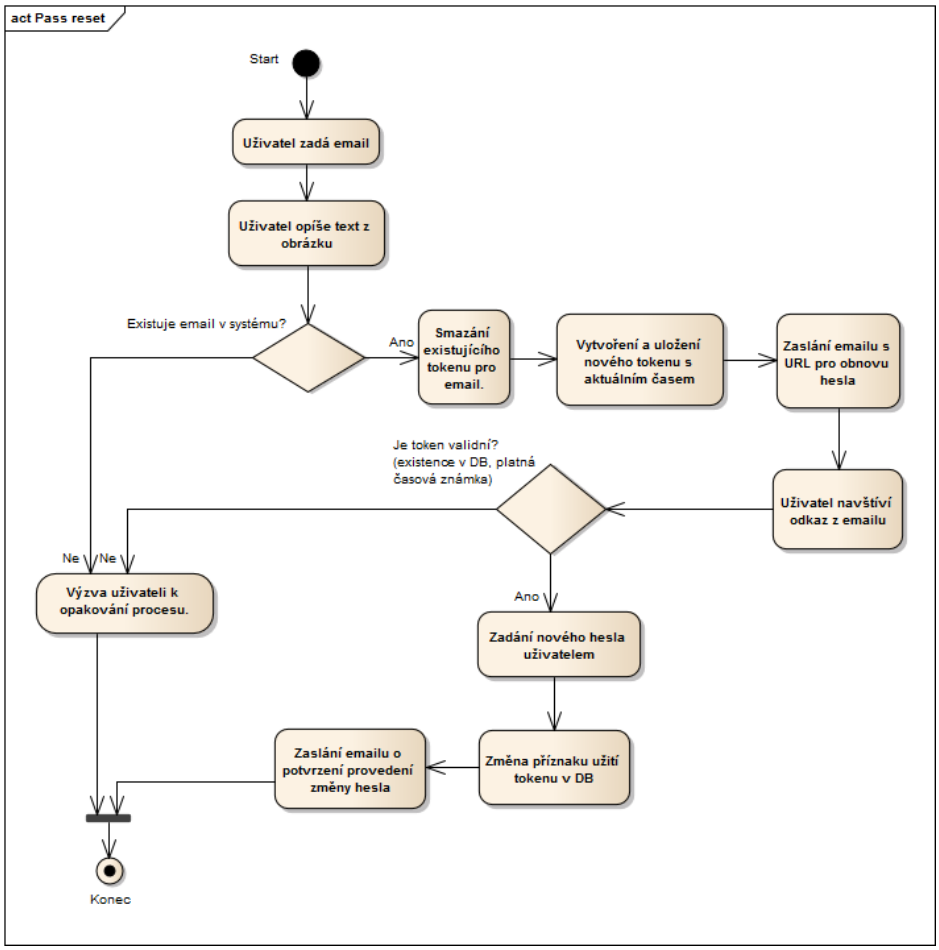
Nutnost opsání textu z obrázku při zadávání emailové adresy. Tento prvek zvyšuje zabezpečení proti opakovaným požadavkům robotů, nebo scriptů, které se tak chovají a které v konečném důsledku lze použít k prolomení bezpečnosti aplikace. Na druhou stranu `CAPTCHA` (jak je tento prvek nazýván) není perfektní a ani uživatelsky přívětivá. Zvyšuje však o několik desítek procent zabezpečení proti opakovaným požadavkům.
Zvýšení bezpečnosti, použito v aplikaci Mentica

Zaslání informativního emailu o změně hesla v systému. Je dobrým zvykem uživatele informovat o veškerém dění kolem jeho účtu. Uživatel tak má možnost zjistit, co se dělo v samotné aplikaci po přihlášení, ale i v účtu svého emailu. Při `logování` událostí v databázi je zaslání informativního emailu to nejmenší. Nesmí se ale opakovat chyba z již zmíněných bodů. Nové heslo v emailové schránce nemá co dělat.
Dobrý zvyk, komunikace s uživatelem, použito v aplikaci Mentica

Obnovy hesla uživatele aplikace se týká mnoho dalších funkcí, jako například vytváření a správa bezpečnostních otázek, delegace na jiné poskytovatele služby nebo více-faktorová `autentikace`. Jejich rozbor, popis i nasazení je nad rámec této práce. Z analýzy plyne vytvoření funkce obnovy hesla podle následujícího diagramu. 1.7 (V první verzi aplikace je doba, po kterou je `token` aktivní, nastavena na 60 minut. Po nasazení v reálném provozu se doba může změnit.)

1.5 Osobní údaje uživatele

Už při registraci do aplikace `Mentica` jsou požadovány některé osobní údaje a je tedy potřeba řešit právní ochranu jak uživatele tak i poskytovatele služby



Obrázek 1.7: Diagram - návrh zachycující proces obnovy hesla uživatele aplikace Mentica

samotné. Podobně, jako na dalších řešeních konkurenčních aplikací jsou údaje přístupny ostatním uživatelům. Jinými slovy pouze těm, kteří registrací prošli taktéž. Je tedy na místě mluvit i o zveřejňování osobních údajů na internetu. Touto problematikou se zabývá zákon č. 101/2000 Sb., o ochraně osobních údajů a o změně některých zákonů (dále jen "zákon o ochraně osobních údajů"), který nabyl platnost a účinnost v roce 2000. [11]

Ze zákona vyplývá, že zpřístupnění osobních údajů prostřednictvím webových stránek je jejich zpracováním podle § 4 písm. e) zákona o ochraně osobních údajů, a subjekt, který osobní data tímto způsobem zpřístupňuje (nemusí se jednat o tentýž subjekt, který předmětné stránky spravuje a provozuje) je z pohledu zákona o ochraně osobních údajů správcem osobních údajů ve smyslu

§ 4 písm. j) tohoto zákona. Jednou ze základních povinností správce, podle § 5 odst. 2 zákona o ochraně osobních údajů, je povinnost zpracovávat osobní údaje pouze se souhlasem subjektu údajů. Další ze základních zásad zpracování osobních údajů, kterou se musí každý správce osobních údajů řídit, je zásada účelnosti zpracování vyjádřená v § 5 odst. 1 písm. f) zákona o ochraně osobních údajů, tj. využívání údajů pouze k tomu účelu, k němuž byly shromážděny. [12]

Dále je třeba informovat o skutečnosti, že provozovatel služby nezpracovává osobní údaje uživatelů webových služeb společnosti jinak, než k zákonným účelům, popř. k účelům, s nimiž uživatel vyslovil souhlas. Přičemž data jsou zpracovávána pouze v rozsahu nutném pro naplnění účelů, které je třeba vyspecifikovat a rovněž pouze po dobu nutnou pro dosažení těchto účelů, nejdéle ale po dobu stanovenou příslušnými právními předpisy či v souladu s nimi.

Prakticky je tedy nutné, aby při registraci vyjádřil uživatel svůj souhlas s podmínkami užití, které budou vyspecifikovány. Existuje několik možných realizací, ale nejběžnější je přidání pole do registračního formuláře, který pak lze odeslat pouze v případě, že bylo ono pole uživatelem zaškrtnuto. Veškeré podmínky a zásady nakládání s osobními údaji uživatelů musí být vypsány na volně přístupné stránce. Technicky se nejedná o zásadní problémy, a tak bude naimplementována pouze příprava na naplnění obsahu zadavatelem, konkrétně tedy zmíněné pole v registračním formuláři a samostatná statická stránka, vždy však bez obsahu.

1.6 Hashovací funkce a metody šifrování

Tato část se věnuje výběru metody `hashování` uživatelského hesla a některým ověřeným postupům. Mezi zásadní `hashovací` funkce jazyka `PHP` patří algoritmus `MD5` vytvořený už v roce 1991, který se dříve často pro ukládání hesel používal. Už v roce 1996 byla ohlášena první kolize kompresní funkce, tedy shodných výstupů na dvou různých vstupech a v nedávné době se na internetu objevily předpočítané tabulky (`rainbow tables`) `hashování` často používaných řetězců pro vytváření hesel. Tato metoda tedy nesplňuje potřebnou úroveň zabezpečení.

Další známý algoritmus vytvářející otisky fixní délky považovaný za nástupce `MD5` a používaný pro `hashování` hesel je `SHA-1`. I když byl v roce 2005 nalezen útok na tento algoritmus a bezpečnost byla zpochybněna kryptografickými odborníky nebo i firmou `Google`, tuto metodu stále využívá nezane-

dbatelné procento webových aplikací. Algoritmus se obecně již nedoporučuje nasazovat, čehož se při implementaci budu držet. [13]

Algoritmus symetrické blokové šifry `Blowfish` položil základy vytvoření adaptivní `hashovací` funkci `Bcrypt`, která je mezi předními PHP programátory považována za nejvíce bezpečnou. (v současné době) Například podle slov pana Jakuba Vrány [14] jde o funkci nejdostupnější. Algoritmus využívá takzvaného solení, kdy je náhodný řetězec připojen k původnímu heslu, až poté se `hashuje`. (dále jen sůl) Využívá se také klíčového faktoru ceny, který jednoduše řečeno projde algoritmus šifrování vícekrát, čímž se stane výsledek stoprocentně odolným vůči zmíněným předpočítaným tabulkám. `Bcrypt` je jeden z pomalých algoritmů, ale v případě `hashování` hesel není čekání nežádoucí. Algoritmus je navíc podporovaný frameworkem Symfony 2, takže jeho použití se více než nabízí. V tomto případě ani není nutné nějak spravovat a ukládat v databázi zmíněnou sůl zvlášť, protože je součástí výsledného řetězce. Ukázka nastavení `hashování` hesel ve frameworku Symfony 2 je v kódu 1.2.

Listing 1.2: Příklad nastavení hashování hesel pomocí funkce Bcrypt s cenou 15 v konfiguračním souboru frameworku Symfony užitím notace YAML.

```
# app/config/security.yml
security:
# ...
  encoders:
  Symfony\Component\Security\Core\User\User:
    algorithm: bcrypt
    cost:      15
```

Bezpečnost a síla `hashovací` funkce závisí na několika faktorech. Při užití dostatečně velké výpočetní síly není žádný `hash` neprolomitelný. Otázka však je, za jak dlouho se tak může povést. Každým cyklem `hashování` vzroste možný počet výsledků exponenciálně. Například každou z 64 fází algoritmu `MD5` vzroste počet možností 10^{77} krát. Pro úspěšné zpětné zjištění hodnoty je tak potřeba nepředstavitelně velký počet možných permutací. Odhady uvádějí nutný výpočetní čas přibližně 10^{4908} let při použití superpočítače, který je schopen pracovat rychlostí 8 `PFLOPS`. (floatingpoint operations per second) [15] Zpětné vypočítání je možné technicky, ale čísla naznačují, že prakticky nikoliv. Při prolamování `hashů` nebo šifer se proto nepoužívá zpětného vypočítání, ale v textu již zmíněných kolizí, jejichž nalezení není tak výpočetně náročné.

Při použití předpočítaných tabulek je zjištění otázkou několika sekund nebo i méně. Z důvodu snížení rizika výskytu řetězce v tabulce se připojují náhodné znaky (solení). Při využití dostatečně náhodného a dlouhého řetězce soli, se pravděpodobnost úspěšného užití předpočítaných tabulek blíží k nule. Stále je však ve hře hrubá síla nalezení kolize. Zde už zmíněný parametr času výpočtu hraje roli zásadní. Funkce jako `MD5, SHA1` nebo `SHA256` jsou velice rychlé, což je také důvod, proč se tradičně používají k podpisům zpráv nebo klíčů. V oblasti zabezpečení hesel je rychlost nežádoucí a tyto funkce se pak používají v cyklech o milionech iterací, 1.3 aby se dosáhlo vyšší složitosti a náročnosti výpočtu.

Listing 1.3: Jednoduchý příklad základního zvýšení bezpečnosti užití funkce MD5 v cyklu.

```
$result = $password + $salt;
for ($i = 0; $i < 10000000; $i++) {
  $result = md5($result + $salt);
}
```

Tyto všechny kroky nutné k zabezpečení logicky vedou k užití funkce, která byla pro `hashování` hesel přímo stvořená a splňuje veškerá bezpečností kritéria. `Hashovací` algoritmus `Bcrypt` je charakteristický tím, že žádný stav není možné vypočítat bez znalosti soli (náhodně vygenerovaný připojený řetězec) a klíče (uživatelova hesla) a stává se tak jednocestným. Při jeho použití je třeba se vyvarovat několika chyb. Sůl musí vždy být náhodná, aby se zajistila obrana proti útokům hrubou silou, a generátor soli nesmí být odhadnutelný. Parametr ceny nesmí být nízký, aby se zajistila klíčová vlastnost - vyšší čas výpočtu. Algoritmus je také závislý na verzi PHP. Konkrétně od verze 5.3.0 se s funkcí počítá a je podporována. Problém by nastal při použití ve verzi nižší, protože by se automaticky, místo nahlášení chyby, použil algoritmus `DES`.

Celý proces nasazení `hashovací` funkce výrazně zjednodušuje užití PHP frameworku Symfony 2, který všechny zmíněné problémy řeší. Zajištění vysoké bezpečnosti je jedním z hlavních bodů nasazení frameworku.

1.7 Požadavky

V této kapitole jsou popsány a rozebrány požadavky zadavatele na funkčnost systému i systém samotný z pohledu uživatelského modulu aplikace `Mentica`.

1.7.1 Funkční požadavky

FP1 *Uživatel bude v systému vystupovat jako mentor i jako student.*
Tento požadavek skrývá definování uživatelských rolí. Představa zadavatele je taková, že uživatel, který v aplikaci vytváří nějaký odebíratelný obsah, může sám odebírat obsah ostatních uživatelů. Tyto role budou nazývány mentor a student.

FP2 *Systém bude s uživatelem komunikovat pomocí emailu.*
Každý uživatel musí mít ověřenou emailovou adresu, pomocí které se do aplikace přihlašuje. Systém musí být schopný generovat a zasílat odkazy pro aktivaci účtu nebo obnovu hesla.

FP3 *Uživatel bude mít právo upravovat svůj účet. Součástí základních informací o uživateli bude celé jméno, získaný titul, pohlaví, město a heslo.*

FP4 *Uživatel (mentor) může v aplikaci zobrazovat své různé kontaktní údaje.*
Mentor může v aplikaci publikovat například jako vysokoškolský profesor nebo autor kurzu pro veřejnost. Musí mít možnost zobrazit různé kontaktní údaje na různých místech v aplikaci.

FP5 *Uživatel může místo klasické registrace využít přihlášení přes Facebook.*
Registrace a přihlašování do aplikace pomocí bezpečnostní brány sociální služby `Facebook` má zjednodušit celý proces vytváření účtu.

FP6 *Mentor může seskupovat další uživatele a tvořit tak třídy studentů.*
Modelovou situací je reálná třída studentů například základní školy, pro které se vytvoří skupina v aplikaci.

FP7 *Mentor může ve třídě vyučovat své kurzy s různou cenou.*
Pro jednu skupinu (třídu) může ve skupině v aplikaci existovat více kurzů, respektive testů, kterých se žáci zúčastní. Cena je zde požadována, aby mohl mentor svůj produkt nabízet rozdílným skupinám odběratelů s různými podmínkami. Například svým studentům na základní škole zpřístupní kurz ve skupině se stoprocentní slevou a v jiné skupině, do které je volný přístup, se slevou deseti procent.

FP8 *Student může vyhledat třídu, spravovat své členství a pozvánky ke kurzům. Mentor může tuto funkci nastavit.*
Skupina, respektive třída, musí být vyhledatelná podle nějakého identifikátoru, který si mentor určí sám. Mentor také při vytváření skupiny určí, zda-li bude veřejně přístupná nebo bude členy spravovat sám. V takovém případě se zasílají pozvánky, které uživatel nemusí přijmout, nebo-li nevyjádří svůj souhlas s cenou.

FP9 *Uživatel může zobrazit profil jiného uživatele.*
Součástí profilu bude fotka, krátký text, který o sobě uživatel napsal, a

další podrobnosti vyplněné při registraci. Zobrazí se také statistické podrobnosti o procházení a odebírání obsahu jako počet úspěšně splněných testů, odebíraných kurzů a podobně.

FP10 *Uživatel může v systému provádět platby*
Jednou z předností systému `Mentica` má být možnost získávání finančního obnosu za odebírání vytvořeného obsahu studentem. Výběr způsobu a typu placení je popsán v analytické části této práce a rozebrán je i v části analýzy hotových řešení. Systém plateb není součástí první verze aplikace, ale z důvodu konzistence řešení byla provedena jeho analýza.

FP11 *Student bude mít k dispozici statistiku procházení obsahu jako úspěšnosti při testech a stavy dokončení kurzů.*
Na uživatelská data je v informačních technologiích kladen velký důraz. Na základě jejich analýzy se vytvářejí modely chování uživatelů v systému a vytváří se byznys modely sledující aktuální stav interakce. Uživatelský modul aplikace `Mentica` má poskytovat podrobnou statistiku plnění zadaných úkolů, testů a průchodů odebíraným obsahem. Ze statistik by měly být jasné závěry, jako například křivka zlepšování v určitém oboru a podobně. Implementace tohoto požadavku bude vyžadovat analýzu reálných dat po nasazení první verze aplikace.

1.7.2 Nefunkční požadavky

NP1 *Přístupnost přes www na požadované adrese*
Zadavatel na vlastní náklady pořídí zvolenou doménu a webový `hosting`.

NP2 *Kompletní podpora prohlížeči Google Chrome, Mozilla Firefox, výchozím prohlížečem internetu systému Android a Internet Explorer, konkrétně verze od:*

- `Chrome` 24+
- `Firefox` 18+
- `IE` 9+
- `Android` 2.1+

NP3 *Intuitivní a uživatelsky přívětivé ovládání aplikace*
Ovládání by mělo být dostatečně intuitivní a pohodlné pro všechny cílové skupiny, tedy jak studenty školního věku, tak i profesory nebo učitele věku vyššího.

NP4 *Použití ověřených a levně dostupných technologií*
Požadavek se týká snížení nákladů na vývoj a správu, viz další nefunkční požadavky.

NP5 *Používání technologií pro minimální zátěž a zrychlení komunikace se serverem*
Při použití technologie PHP jde hlavně o vytváření `AJAX` požadavků a `JS` u klienta.

NP6 *Použití PHP frameworku Symfony 2*
Požadavek vyplynul z analýzy této diplomové práce.

NP7 *Použití knihovny Bootstrap pro zjednodušení definice stylů*
Tento požadavek vyplynul z analýzy kolegyně ve vývoji diplomové práce, Bc. Olgy Budnik

NP8 *Použití relačního databázového systému MySQL*
Tento požadavek vyplynul z analýzy kolegy ve vývoji diplomové práce, Bc. Jiřího Matějky

NP9 *Použití souborového systému PHP knihovny Gaufrette od KNP Labs.*
Tento požadavek také vyplynul z analýzy kolegy ve vývoji diplomové práce, Bc. Jiřího Matějky

NP10 *Použití vlastního řešení integrace se sociální sítí Facebook pomocí Facebook JS SDK*
Tento požadavek úzce souvisí s funkčním požadavkem číslo pět a vyplynul z analýzy této diplomové práce.

1.8 Případy užití

V této kapitole jsou popsány případy užití (Use case, dále UC), které plynou z funkčních požadavků zadavatele. Jsou tedy detailní specifikací a popisem požadavku a podrobným zobrazením toho, jaké služby bude výsledná aplikace poskytovat.

Na diagramech 1.8 a 1.9 jsou zobrazeny případy užití. Z důvodu většího počtu případů užití byly použity dva diagramy, kde první se věnuje funkčním požadavkům FP1 až FP5 a druhý zbývajícím, tedy do FP11 včetně.

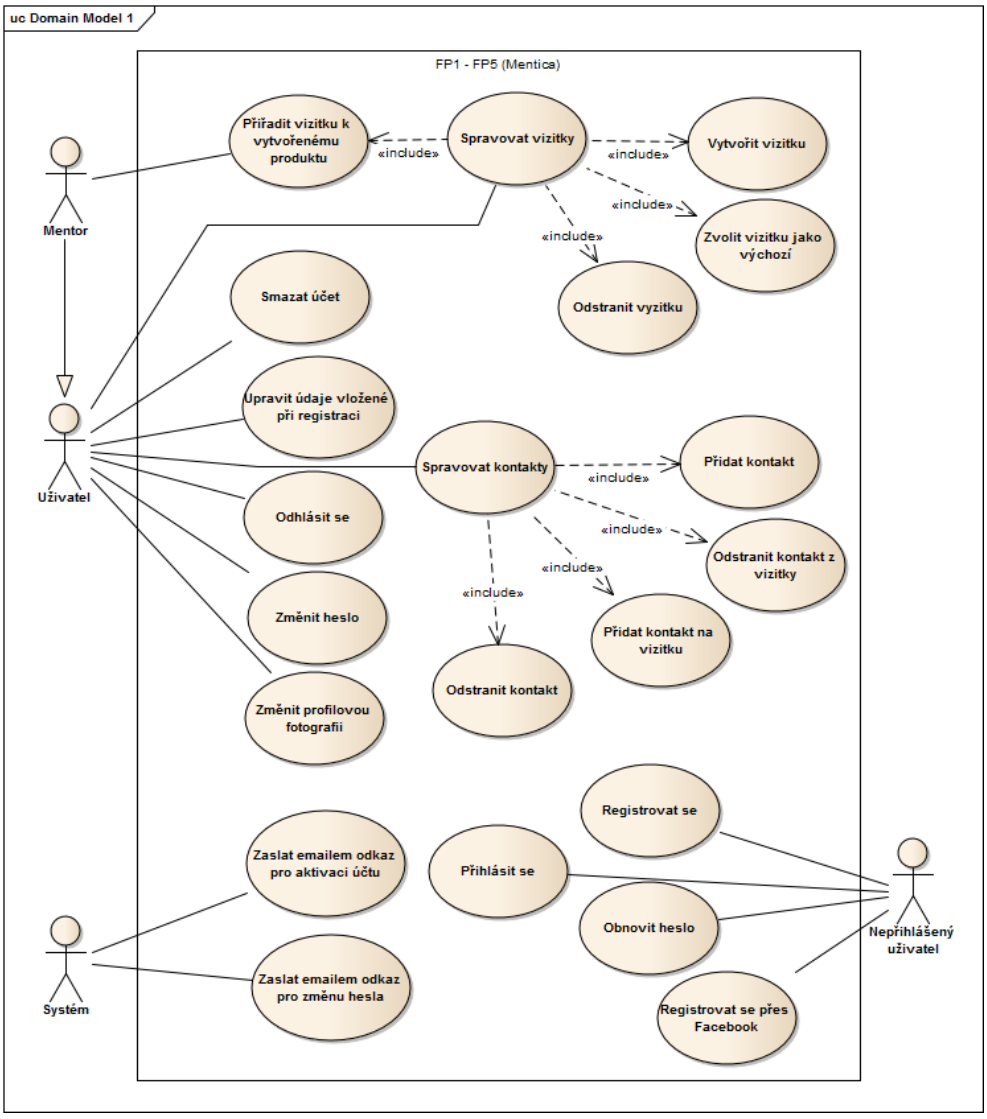
Slovník a definice pojmů

Mentor Role registrovaného uživatele v rámci vytvořeného obsahu (kurz, test)

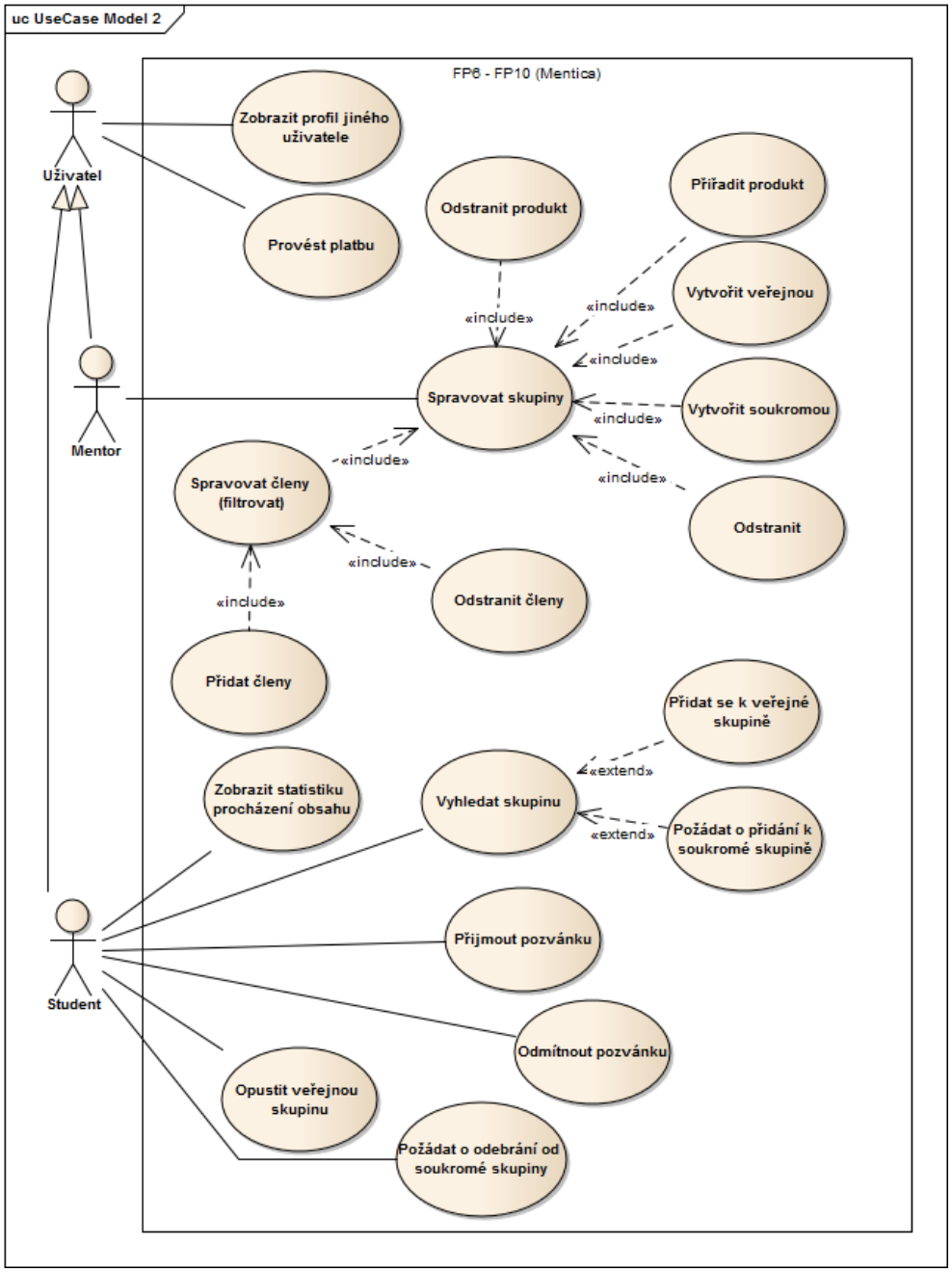
Student Role registrovaného uživatele v rámci odebíraného obsahu, vytvořeného mentorem

uživatel Mentor i student některé případy užití sdílí, pak se mluví o uživateli

Zakladatel skupiny Uživatel, který vytvořil skupinu



Obrázek 1.8: První část diagramu případů užití uživatelského modulu systému Mentica



Obrázek 1.9: Druhá část diagramu případů užití uživatelského modulu systému Mentica

Zakladatel produktu Uživatel, který vytvořil kurz nebo test, obecně produkt

Soukromá skupina Skupina, kterou lze vyhledat, ale členy spravuje pouze její zakladatel. K soukromé skupině se nelze přidat, ale požádat o přidání zakladatele skupiny

Veřejná skupina Skupina, kterou lze vyhledat i se do ní přidat. Existují-li v rámci skupiny nějaké produkty, přidaný uživatel se automaticky stává jejich studentem

Drag and drop Funkce přemístění obsahu jeho tažením s následkem akce systému

SDK z anglického `Software development kit`, je soubor nástrojů pro vývoj software

Facebook Rozsáhlá sociální síť na internetu, která poskytuje `SDK`

FP1 Zde nevyplývají žádné případy užití. Požadavek je pouze definicí umožnění chování uživatele v systému. Uživatel určuje svoji aktuální roli konkrétními činnostmi a akcemi v aplikaci, jak bude uvedeno dále.

FP2 Ani z FP2 se přímo nedají vyvodit UC. Jde opět o definici na obecné úrovni, kde je řečeno, že systém jako hlavní komunikační kanál použije elektronickou poštu. UC jako zaslání aktivačního odkazu a další budou popsány u odpovídajících požadavků.

FP3 Uživatel bude mít právo upravovat svůj účet. Součástí základních informací o uživateli bude celé jméno, získaný titul, pohlaví, město a heslo.

a) Nepřihlášený uživatel se může registrovat vyplněním příslušného formuláře.
b) Systém bude zasílat email s potvrzovacím odkazem k aktivaci účtu.
c) Nepřihlášený uživatel se může přihlásit vyplněním přihlašovacích údajů, emailu a hesla.
d) Přihlášený uživatel si může upravit údaje vložené při registraci.
e) Změnit svoje heslo, kterým se přihlašuje do aplikace.
f) Změnit (přidat) profilovou fotografii.
g) **Nepřihlášený uživatel si může nechat zaslat odkaz pro obnovu hesla. (funkce zapomenuté heslo)**
h) Přihlášený uživatel se může z aplikace odhlásit.
i) Uživatel může svůj účet smazat.

FP4 Uživatel (mentor) může v aplikaci zobrazovat své různé kontaktní údaje. Zadavatelem požadované typy kontaktů jsou telefonní číslo, adresa, osobní www stránky, email, `Skype`, odkaz na `Twitter`, odkaz na `LinkedIn` a odkaz na `Facebook`. Uživatel v systému může provádět následující akce:

a) Přidávat kontakty.

b) Odstranit kontakt.

c) Vytvářet vizitky.

d) Odstranit vizitku.

e) Zvolit vizitku jako výchozí.

f) **Přidávat kontakty na vizitku.**

g) Odstranit kontakt z vizitky.

h) Přiřadit vizitku k vytvořenému produktu.

FP5 Uživatel může místo klasické registrace využít přihlášení přes Facebook
Tento požadavek je sám o sobě případem užití, který bude rozepsán podrobněji.

FP6 Mentor může seskupovat další uživatele a tvořit tak třídy studentů.

a) Uživatel může vytvořit veřejnou skupinu. Skupina bude popsána vyhledávacím identifikátorem, statutem veřejná, popisem a názvem.

b) Může vytvořit soukromou skupinu. Skupina bude popsána vyhledávacím identifikátorem, statutem soukromá, popisem a názvem.

c) **Zakladatel skupiny může přidávat studenty vyhledané v seznamu pomocí zadaných filtračních pravidel.**

d) Zakladatel skupiny může odstraňovat studenty ze seznamu členů, od skupiny vyhledané pomocí zadaných filtračních pravidel.

e) Zakladatel může vytvořenou skupinu odstranit.

FP7 Mentor může ve třídě vyučovat své kurzy s různou cenou.

a) **Autor (kurzu, testu) může produkt přiřadit ke skupině soukromé i veřejné s určitou procentuální slevou oproti plné ceně.**

b) Autor produktu, který je zároveň zakladatel skupiny, může produkt od skupiny odstranit.

FP8 Student může vyhledat třídu, spravovat své členství a pozvánky ke kurzům. Mentor může tuto funkci nastavit.

a) Uživatel může vyhledat veřejné nebo soukromé skupiny podle vyhledávacího identifikátoru.

b) Může se přidat k veřejným skupinám.

c) Může požádat o členství v soukromé skupině.

d) Student může opustit veřejnou skupinu.

e) Student může požádat o odebrání od soukromé skupiny.

f) **Uživatel může přijmout pozvánku ke kurzu přes členství v soukromé skupině.**

g) Uživatel může odmítnou pozvánku ke kurzu.

FP9 Uživatel může zobrazit profil jiného uživatele.
Tento požadavek je případem užití.

FP10 Uživatel může v systému provádět platby. (pouze analýza, bez implementace)
Uživatel může provést platbu, po jejímž uskutečnění získá kredit určený pro otevírání obsahu v systému Mentica.

FP11 Student bude mít k dispozici statistiku procházení obsahu jako úspěšnosti při testech a stavy dokončení kurzů.

1.9 Scénáře případů užití

Případy užití se skládají z jednotlivých kroků a jsou vykonávány uživatelem s určitou rolí, který je pak nazýván aktérem. Zároveň jsou pro případy užití definovány určité vstupní podmínky, které musí být splněny, aby mohl konkrétní případ nastat. Ze všech nalezených případů jsou vybrány a rozepsány komplexnější scénáře.

FP3.g Nepřihlášený uživatel si může nechat zaslat odkaz pro obnovu hesla. (funkce zapomenuté heslo)

Stručný popis systém umožní aktérovi zaslat odkaz pro obnovu hesla pro přihlášení

Hlavní aktéři registrovaný uživatel, systém

Vstupní podmínky aktér je odhlášený

Hlavní scénář
1. Případ užití je spuštěn kliknutím na odkaz "Zapomněl jsem heslo"na stránce s přihlašovacím formulářem.
2. Systém aktérovi zobrazí formulář, kde se po aktérovi vyžaduje zadání ověřené emailové adresy vložené při registraci účtu a opsání vygenerovaného bezpečnostního kódu z obrázku.

3. Aktér vloží svou emailovou adresu použitou při registraci, opíše kód z obrázku a klikne na tlačítko "Obnovit heslo".
4. Systém validuje vstupy, a když jsou v pořádku, tak kontroluje existenci uživatelského učtu s uvedenou emailovou adresou.
   a) Když je emailová adresa v systému zaregistrovaná, systém na ni zašle jedinečný odkaz s platností jedné hodiny.
   b) Když není emailová adresa v systému zaregistrována, systém o této skutečnosti informuje aktéra hláškou.
5. Aktér navštíví odkaz obdržený emailem.
6. Systém validuje platnost bezpečnostního kódu obsaženého v odkazu a jeho časovou platnost. V případě, že je odkaz v pořádku, zobrazí aktérovi formulář pro změnu hesla. (vstupy jsou nové heslo a nové heslo znovu pro kontrolu)
7. Aktér vyplní nové heslo a pokračuje kliknutím na tlačítko "Uložit".
8. Systém přesměruje na stránku pro přihlášení uživatele do systému.

Výstup Změněné heslo aktéra (uživatele v systému).

FP4.f Přihlášený uživatel může na svoji vizitku přiřazovat své kontakty.

Stručný popis systém umožní aktérovi slučovat jeho kontakty a tvořit tak vizitky

Hlavní aktéři registrovaný uživatel, systém

Vstupní podmínky aktér (přihlášený uživatel) má vytvořený alespoň jeden kontakt

Hlavní scénář
1. Případ užití je spuštěn kliknutím na odkaz "Vytvořit vizitku"v sekci editace profilu.
2. Systém zobrazí formulář pro vložení názvu vizitky a přiřazení jednoho nebo více kontaktů.
3. Uživatel vyplní formulář a pomocí funkce `drag and drop` přiřadí kontakty. Formulář potvrdí kliknutím na tlačítko "Vytvořit vizitku".
4. Systém kontroluje vstupy od uživatele a v případě úspěchu vytvoří a uloží vizitku, kterou následně zobrazí na stejné stránce v seznamu.

Výstup Vytvořená vizitka aktéra s přiřazenými kontakty.

FP5 Uživatel může místo klasické registrace využít přihlášení přes `Facebook`

Stručný popis systém umožní registraci a přihlašování do systému pomocí služby sociální sítě `Facebook`

Hlavní aktéři zaregistrovaný nebo nezaregistrovaný odhlášený uživatel

Vstupní podmínky aktér má vytvořený účet na sociální síti Facebook

Hlavní scénář
1. Případ užití je spuštěn kliknutím na odkaz "Přihlásit přes `Facebook`".
2. Systém zobrazí přihlašovací formulář služby Facebook ve vyskakovacím okně.
3. Uživatel vyplní své údaje (nezávislé na systému Mentica) a potvrdí je kliknutím na tlačítko "Přihlásit".
4. Systém (proběhne-li autentikace uživatele v pořádku) může vyžadovat souhlas se získáním některých údajů z uživatelova účtu sítě `Facebook`.
5. Systém kontroluje, zda bylo do formuláře vloženo telefonní číslo (viz. kapitola analýza, část Integrace se sociální sítí `Facebook`).
   a) Pokud ano, systém vytvoří chybovou hlášku, která informuje o nutnosti zadání emailu a přesměruje na stránku s přihlašovacím formulářem a případ užití tím končí.
   b) Pokud ne, tak byla vložena emailová adresa a případ užití pokračuje následujícím bodem.
6. Systém kontroluje, zda existuje zaregistrovaný uživatel s emailovou adresou shodnou se zadanou adresou.
   a) Pokud ano, systém uživatele přihlásí a přesměruje na jeho profil.
   b) Pokud ne, systém vytvoří aktivní účet s ověřenou emailovou adresou pomocí dat získaných ze sítě `Facebook` (email, křestní jméno, příjmení a pohlaví), nově vytvořeného uživatele přihlásí a přesměruje na jeho profil.

Výstup Přihlášení uživatele do systému (popřípadě vytvoření účtu)

FP6.c Zakladatel skupiny může přidávat studenty vyhledané v seznamu pomocí zadaných filtračních pravidel.

Stručný popis Systém umožní vyhledávat a přidávat uživatele (studenty) do vytvořené skupiny

Hlavní aktéři zakladatel skupiny

Vstupní podmínky Aktér vytvořil skupinu a existují uživatelé, které do ní lze přidat

Hlavní scénář
1. Případ užití je spuštěn kliknutím na odkaz “Spravovat členy” z detailu vytvořené skupiny v sekci Správa skupin.

2. Systém zobrazí stránku s formulářem pro podrobnější vyhledávání uživatelů (v tomto případě studentů). V první verzi aplikace `Mentica` je umožněno vyhledávat pomocí příjmení, emailu a města.
3. Uživatel vyplní formulář, buď všechna pole nebo jen některá, a klikne na tlačítko "Filtrovat".
4. Systém validuje vstupy od uživatele a jsou-li v pořádku, tak aplikuje požadovaná filtrační pravidla pomocí SQL funkce LIKE. Výsledek zobrazí. (Systém nezobrazuje členy a správce skupiny)
5. Aktér vybere studenty, které chce ke skupině přidat, zaškrtnutím požadovaných polí a klikne na tlačítko "Přidat ke skupině".
6. Systém přidá požadované studenty ke skupině. Každému novému studentovi je vytvořena notifikace a navíc se kontroluje následující:
   a) Obsahuje-li skupina nějaký placený kurz nebo test, studentovi se vytvoří a zašle pozvánka pro odebírání obsahu.
   b) Obsahuje-li skupina nějaký placený kurz nebo test, který již student odebírá v nějaké jiné skupině nebo přímo, pozvánka se nevytváří.

Výstup Noví studenti přiřazení ke skupině (popřípadě i pozvánky)

FP7.a Autor (kurzu, testu) může produkt přiřadit ke skupině soukromé i veřejné s určitou procentuální slevou oproti plné ceně.

Stručný popis Systém umožní zakladateli skupiny přiřadit vytvořený produkt s možností nastavení slevy v rámci skupiny, do které se produkt přiřazuje

Hlavní aktéři Zakladatel skupiny - autor produktu

Vstupní podmínky Zakladatel skupiny vytvořil nějaký produkt s cenou

Hlavní scénář
1. Případ užití je spuštěn kliknutím na odkaz "Správa produktů"ze sekce detailu konkrétní skupiny.
2. Systém zobrazí všechny produkty, které aktér vytvořil a které lze připojit k aktuální skupině. K těmto produktům zobrazí i aktuální plnou cenu. Systém také zobrazí již připojené produkty.
3. Aktér kliknutím na odkaz "připojit ke skupině"provede výběr produktu.
4. Systém zobrazí formulář, jehož jediným vstupem je procento slevy. Validní je celé číslo od nuly do sta včetně.

5. Aktér nastaví slevu pro vybraný produkt a potvrdí formulář kliknutím na odkaz "Připojit".
6. Systém vypočte zvýhodněnou cenu pro vybraný produkt ve skupině a záznam uloží.
7. Systém přesměruje zpět na detail skupiny, kde se v seznamu vyučovaných produktů již napojení projeví.

Výstup Vytvořené spojení mezi skupinou studentů a mentorem (zakladatelem skupiny) založeným produktem s možností zvýhodnění ceny.

Poznámka Cena produktu se může měnit v čase. Proto je nutné ukládat výsledek zvýhodnění v konkrétní skupině, aby zůstala cena neměnná.

FP8.f Uživatel může přijmout pozvánku (nabídku) ke kurzu přes členství v soukromé skupině.

Stručný popis Po připojení produktu mentorem ke skupině vzniknou pozvánky (nabídky), kde studenti vyjádří svůj souhlas (nebo nesouhlas) s cenou. Systém umožní přijmout nebo odmítnout pozvánku ke kurzu nebo testu v konkrétní skupině.

Hlavní aktéři člen skupiny, ve které existuje připojený produkt - Student

Vstupní podmínky Existence napojení produktu ke skupině se zvýhodněnou cenou

Hlavní scénář
1. Případ užití je spuštěn kliknutím na odkaz “Nabídky” ze sekce detailu profilu uživatele.
2. Systém zobrazí otevřené nabídky potenciálnímu studentovi. Z této sekce si lze odkazy zobrazit i nabídky přijaté nebo odmítnuté. U každé nabídky je zobrazena zvýhodněná cena.
3. Aktér provede rozhodnutí o přijetí nebo nepřijetí kliknutím na odkaz "Přijmout"nebo "Odmítnout". Pro tento případ užití uvažuji přijetí nabídky.
4. Systém provede potřebné operace pro umožnění přístupu do procesu průchodu produktem, tedy odečtení požadovaného kreditu, zařazení aktéra na seznam studentů a vytvoření tabulky se záznamem o průchodu studenta kurzem, respektive produktem.
5. Systém přesměruje aktéra na seznam přijatých nabídek, kde od každé vede odkaz do detailu kurzu.

Výstup Vytvořená vazba studenta na produkt (kurz, test) se zvýhodněnou cenou přes připojení ve skupině po přijetí automaticky vytvořené nabídky.

Kapitola 2

Návrh

Tato kapitola využívá poznatky z analytické části k návrhu byznys procesů a architektury aplikace, kterou bude realizována, tedy doménového a databázového modelu. Je potřeba popsat jednotlivé technologie, které budou při tvorbě použity a popsat entity doménového modelu a vztahy mezi nimi. Součástí bude i návrh uživatelského rozhraní, který se stejně jako další části z důvodu zachování konzistence týmové práce musí konzultovat s dalšími členy projektu.

2.1 Použité technologie a architektura

Z analýzy vyplynulo užití následujících technologií, softwaru nebo postupů.

PHP Framework Symfony 2.6 v kombinaci s ORM Doctrine 2.4.7 (stable)

Framework Symfony v kombinaci se systémem objektově relačního mapování Doctrine poskytne silnou základnu v podobě otestovaných knihoven, přístupů a široké uživatelské základny, dokumentace a návodů.

Composer 1.0.0-alpha9

Při užívání kódu a knihoven třetích stran, které stále vydávají nové verze v podobě aktualizací, je velice složité udržet vše pod kontrolou. Composer [16] je manažer závislostí PHP softwaru a vyžadovaných knihoven na aplikační úrovni, který je úzce spojený s webovou aplikací `Packagist` [17], kde jsou udržovány repozitáře dostupných balíčků.

Twitter Bootstrap 2.3.2

Jde o volně dostupný soubor nástrojů pro tvorbu webu a webových aplikací, který zajišťuje použitelnost aplikace na různých webových prohlížečích. Pod-

poruje technologie HTML, CSS a JavaScript, respektive jQuery, na které je Bootstrap [18] funkčně závislý. Hlavním důvodem volby tohoto nástroje bylo zvýšení komplexnosti vzhledu uživatelského rozhraní mezi vývojáři aplikace Mentica. Použití plyne z nefunkčního požadavku NP7.

JS knihovna jQuery v2.1.3

Jedna z nejznámějších knihoven nad jazykem JavaScript v interakci s HTML, která si klade za cíl stejný výsledek užití různých funkcí na všech prohlížečích internetu. První vydaná stabilní verze byla již v srpnu roku 2006. Jsou nabízeny například funkce pro procházení a změnu DOM, událostí, manipulace s CSS, AJAX a další.

JS knihovna jQueryUI v1.11.3

JQueryUI je rozšířením knihovny jQuery, která mezi základními funkcemi nepodporuje například události přetažení nebo změny velikosti prvku na stránce. Knihovny jsou odděleny z důvodu snížení stahovaného objemu dat při získávání odpovědi serveru klientem, protože většina webů tato rozšíření nepoužívá.

AJAX užitím knihovny jQuery

AJAX je technologie vývoje webových aplikací, kde se mění obsah stránek bez nutnosti kompletního znovu načtení pomocí asynchronního zpracování. K docílení takovéto funkcionality je třeba vytvoření požadavku pomocí JavaScriptu (zde knihovna jQuery), zpracování požadavku XMLHttpRequest na serveru, kde se vytvoří odpověď, například ve formátu HTML nebo JSON. Odpověď je dále zpracována u klienta v prohlížeči internetu. Příklad vytvoření a obsluhy asynchronního volání metody bude dále uveden.

Filesystem PHP knihovny Gaufrette, KNP Labs

Použití plyne z nefunkčního požadavku NP9. V této práci je knihovna nasazena pouze v případě užití změny profilového obrázku uživatele. (případ užití FP3-f) Gaufrette [19] je knihovna v PHP5, která poskytuje abstraktní vrstvu souborového systému s možností ukládání dat užitím Cloudových systémů.

Facebook JS SDK v2.3

Tento soubor nástrojů poskytuje mnoho klientsky založených funkcí pro vytváření sociálních pluginů, získávání dat služby Facebook, autentikace nebo například přidávání tlačítka "Like". Závislost na další knihovně třetí strany je vykoupena jednoduchým užitím, správou a nastavením prostředí.

Enterprise Architect 9.0.907

Desktopová aplikace pro Windows od společnosti Sparx Systems [20] poskytuje nástroje pro vytváření kompletního ULM modelu navrhované aplikace nebo softwaru. Je poskytována podpora pro týmový vývoj a jednotlivé role. Diagramy, které jsou součástí této práce, byly vytvořeny pomocí nástroje Enterprise Architect.

Balsamiq Mockups For Desktop v 2.2.13

Aplikace pro operační systém Windows umožňuje vytváření wireframe uživatelského rozhraní pro prohlížeče internetu a iOS. Na internetových stránkách projektu jsou k dispozici balíky rozšíření od samotných uživatelů. Všechny Lo-Fi prototypy UI v této práci byly vytvořeny pomocí nástroje Balsamiq. [21]

GIT 2.1.0

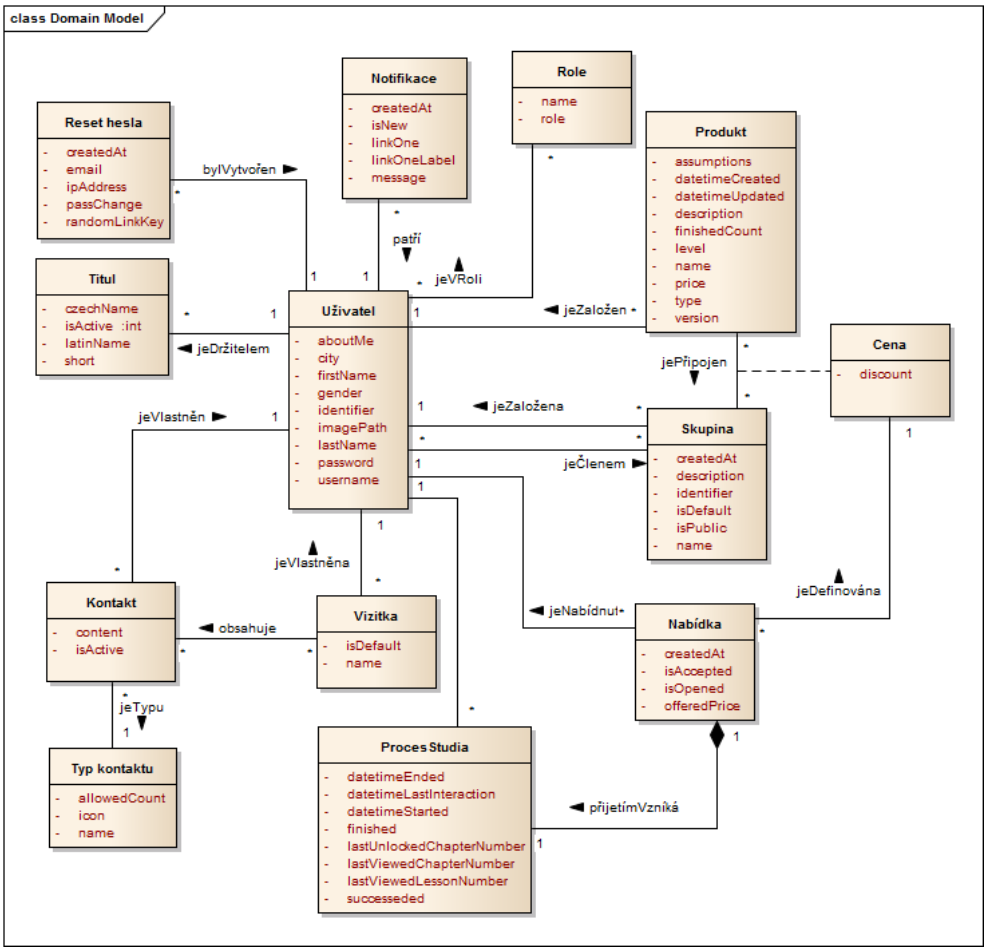
Verzovací systém GIT původně pro vývoj jádra Linuxu, je mocnou podporou pro nelineární a týmový vývoj projektu. GIT [22] poskytuje každému vývojáři lokální kopii celé historie implementace v takzvaných commitech. Lze vytvářet a slučovat jednotlivé vývojové větve, což umožňuje pohodlné oddělení programátorských činností mezi členy týmu. Pro správu commitů, větví a dalších, jsme si v týmu zvolili online nástroj Bitbucket, který obsahuje i nástroje pro koordinaci týmu, jako je delegování úkolů, schvalování změn v kódu a další. [23]

Lokální server XAMPP v3.2.1 (server Apache, databáze MySQL)

Desktopová aplikace poskytující serverovou i databázovou část pro účely testování vyvíjené aplikace. Nejdůležitější částí je interpret PHP skriptu a Apache HTTP server, který slouží k lokálnímu spuštění webové aplikace. Multiplatformní databázové uložiště MySQL je provozováno pod bezplatnou licencí. [24]

2.2 Návrh databáze

Tato kapitola se věnuje návrhu modelu databáze respektive doménovému modelu, který je založen na výsledcích analýzy. Vyplynuly z ní entity a jejich vzájemné vazby, které se budou v aplikaci vyskytovat. Při navrhování doménového modelu bylo potřeba brát v potaz návrhy ostatních vývojářů aplikace, aby bylo dosaženo co nejmenšího provázání na modulární úrovni.



Obrázek 2.1: Doménový model uživatelského modulu aplikace Mentica

Hlavní entity doménového modelu

Na diagramu 2.1 jsou vyznačeny entity, vyskytující se v modelu aplikace, a jejich vzájemné vazby. V následující kapitole jsou popsány pro vývoj ty nejzásadnější. Jako podklady pro tvorbu modelu byly použity přednášky předmětu softwarové inženýrství, zejména pak přednáška čtvrtá, kterou uváděl pan Ing. Jiří Mlejnek. [25] V doménovém modelu je na rozdíl od modelu databázového, který reflektuje plánované skutečné názvy tabulek databáze, použito české názvosloví.

Uživatel je pro tuto práci entita zásadní. Udržuje všechny informace o uživateli, ať už fyzicky, nebo v podobě vazby na entitu jinou. S touto entitou pracují nějakým způsobem téměř všechny ostatní, což je způsobeno nutností zaznamenávání interakce uživatele v systému. Prakticky to zna-

mená udržování vazeb s objektem konkrétního uživatele.

Reset hesla je entitou bezpečnostního charakteru, která má na starosti kontrolu procesu obnovy hesla uživatele. Vždy udržuje bezpečnostní `token`, který kontroluje oproti emailové adrese, času zaznamenávání. Obsahuje také IP adresu, ze které byl požadavek vytvořen, a případná nerovnost s adresou, ze které je navštíven odkaz z emailu, je ukládána do logu.

Kontakt je entita udržující informace o kontaktu uživatele. Kontakt může být různého typu, a proto zde existuje vazba na entitu Typ kontaktu, kde každý typ je v systému reprezentován jinak. V první verzi se počítá s osmi typy, například s telefonním číslem, emailovou adresou nebo odkazem na účet sociální sítě `Facebook`.

Notifikace udržuje jednoduchou informaci o dění v systému pro uživatele. Systém nabízí velkou škálu funkcí, které se uživatele týkají, a přes to o nich nemusí vědět. Proto zde existují notifikační upozornění, která se vytvářejí automaticky, například po přidání studenta do skupiny mentorem.

Skupina představuje entitu sdružení uživatelů ve věci odebírání obsahu vytvořeného mentorem. Vazby s entitou uživatele jsou v celém modelu stěžejní. Udržují totiž informace o zakladateli, ale i o členech skupiny. Další verze aplikace mají poskytovat možnost přidání asistentů při vytváření a správě odebíratelného obsahu, ale i správě členů skupin, kdy bude možné tyto organizační záležitosti delegovat na jiné uživatele systému. Ke skupině se připojují produkty, jejich entita je společným bodem modelu. Spojením vzniká entita Cena.

Cena je vazební entitou mezi skupinou a produktem. Navrženo tak bylo proto, že je nutné nastavovat v různých skupinách různé ceny pro různé produkty mentorů. Vznikem této vazby se automaticky vytváří nabídka uživateli.

Nabídka obsahuje informaci o nabízené ceně produktu v rámci nějaké skupiny. Je na uživateli, jestli přijme mentorem nabízenou cenu produktu ve skupině. Cena se nastavuje procentuálním zvýhodněním oproti ceně původní. Jelikož se cena produktu může kdykoliv změnit, je potřeba udržovat i zvláštní cenu aktuální nabídky. Vznikají automaticky v několika případech:

1. Uživatel se připojí ke skupině, ke které již byl připojený nějaký produkt. Uživateli vzniká a je zaslána nabídka ke každému produktu ve skupině. Studentem se uživatel stane až po přijetí některé z nabídek.

2. Zakladatel skupiny, která již má připojený nějaký produkt, přidá člena. Tomu podobně jako v prvním bodě vznikají nabídky.
3. Mentor přiřadí ke skupině, která již má nějaké členy, své produkty. Všem členům jsou pak zaslány nabídky pro právě připojený produkt.

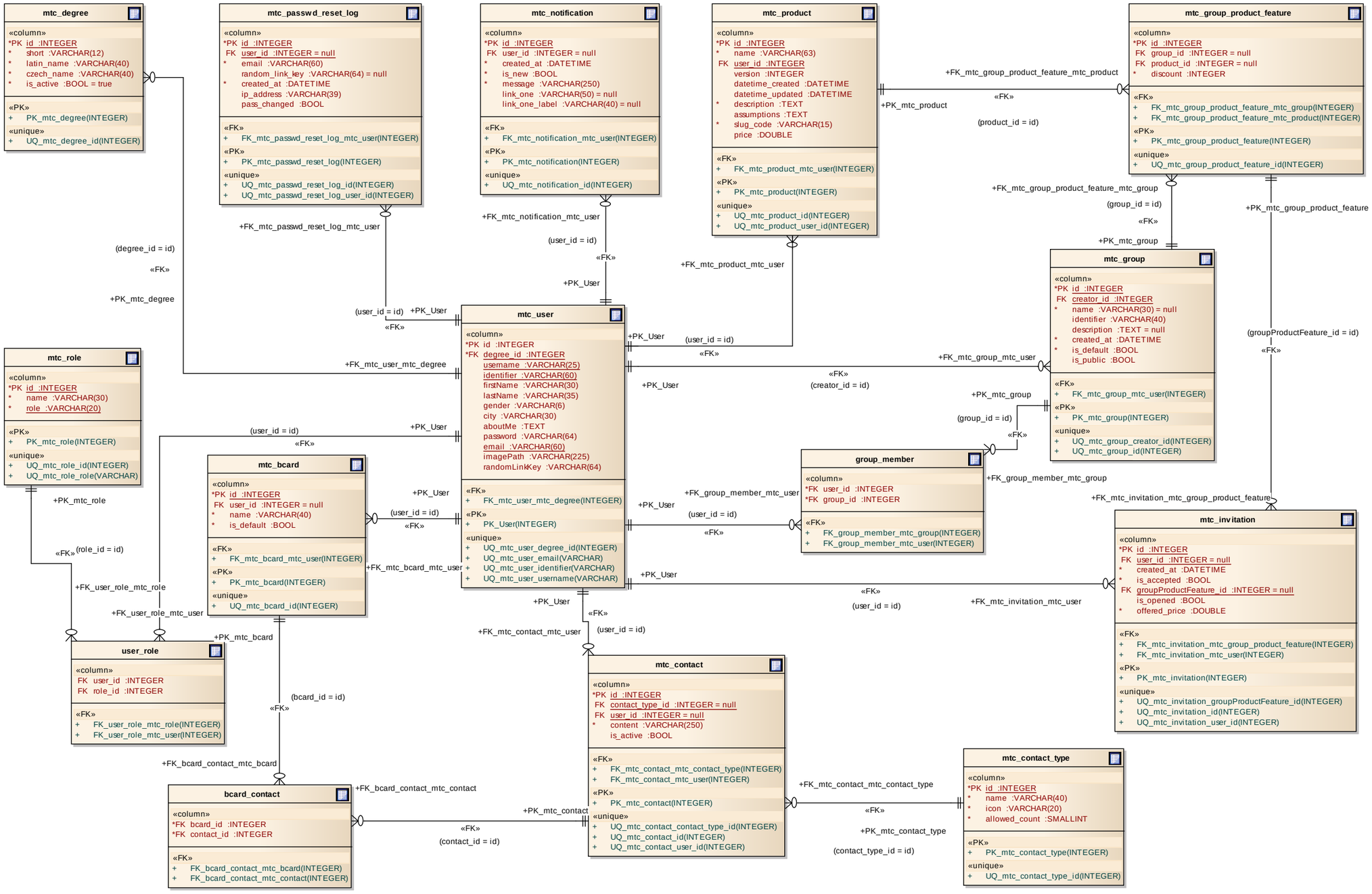
Produkt a Proces studia jsou entity, kterými se vývoj modulu této práce zabýval společně s dalším členem týmu. Bylo potřeba navrhnout architekturu takzvaných `Services` (dále služby), které umožní vzájemnou komunikaci entit. Například přijetím nabídky uživatelem vzniká studijní proces, takže je nutné volat funkci služby, která vytvoří persistentní záznam v databázi. V kapitole diagram komponent 2.4 této práce jsou uvedeny všechny moduly a jejich vzájemné vazby. Protože entity test a kurz dědí od entity produktu, nejsou v doménovém modulu této práce uvedeny. Uživatelský modul má tedy prakticky vazbu pouze s produktem, na které jsou poskytovány zmíněné služby. Některé významné funkce služeb budou popsány v kapitole o implementaci. 3.5

Databázový model

Databázový model sdružuje data do takzvaných relací (tabulek), které tvoří základ relační databáze. Jde o strukturu záznamů s pevně stanovenými položkami, sloupci a atributy. Kolekce více tabulek, jejich funkčních vztahů, indexů a dalších součástí pak tvoří relační databázi.[26]

Při samotném návrhu databázového modelu je nutné mít na paměti pojem normalizace, kdy se dodržováním jistých požadavků předchází možné redundanci, nekonzistenci nebo až ztrátě dat. Pro tyto účely existuje několik kroků, kdy splněním každého z nich se tabulka nachází v určité normální formě.

První normální forma udává požadavek na atribut, který musí obsahovat pouze atomické hodnoty. Příkladem porušení může být například celá adresa bydliště uložená v jednom atributu. Řešením je vytvoření nezávislých atributů město, ulice, poštovní směrovací číslo a podobně. Tabulka může být ve druhé normální formě, pokud splňuje normální formu první a pokud je každý neklíčový atribut plně závislý na každém kandidátním klíči, tedy sloupci nebo kombinaci sloupců, ve kterých mají všechny řádky tabulky své hodnoty unikátní. Pomocí kandidátního klíče lze jednoznačně identifikovat každý řádek tabulky. Na diagramu 2.2 jsou například odděleny tabulky `mtc_contact` a `mtc_contact_type`, jejichž spojení by porušovalo právě druhou normální formu. Třetí normální forma může být splněna, je-li splněna první i druhá normální forma a jsou-li všechny neklíčové atributy vzájemně nezávislé.


mtc_degree
«column»
*PK id :INTEGER
* short :VARCHAR(12)
* latin_name :VARCHAR(40)
* czech_name :VARCHAR(40)
* is_active :BOOL = true
«PK»
+ PK_mtc_degree(INTEGER)
«unique»
+ UQ_mtc_degree_id(INTEGER)
mtc_passwd_reset_log
«column»
*PK id :INTEGER
FK user_id :INTEGER = null
* email :VARCHAR(60)
random_link_key :VARCHAR(64) = null
* created_at :DATETIME
ip_address :VARCHAR(39)
pass_changed :BOOL
«FK»
+ FK_mtc_passwd_reset_log_mtc_user(INTEGER)
«PK»
+ PK_mtc_passwd_reset_log(INTEGER)
«unique»
+ UQ_mtc_passwd_reset_log_id(INTEGER)
+ UQ_mtc_passwd_reset_log_user_id(INTEGER)
mtc_notification
«column»
*PK id :INTEGER
FK user_id :INTEGER = null
* created_at :DATETIME
* is_new :BOOL
* message :VARCHAR(250)
link_one :VARCHAR(50) = null
link_one_label :VARCHAR(40) = null
«FK»
+ FK_mtc_notification_mtc_user(INTEGER)
«PK»
+ PK_mtc_notification(INTEGER)
«unique»
+ UQ_mtc_notification_id(INTEGER)
mtc_product
«column»
*PK id :INTEGER
* name :VARCHAR(63)
FK user_id :INTEGER
version :INTEGER
datetime_created :DATETIME
datetime_updated :DATETIME
* description :TEXT
assumptions :TEXT
* slug_code :VARCHAR(15)
price :DOUBLE
«FK»
+ FK_mtc_product_mtc_user(INTEGER)
«PK»
+ PK_mtc_product(INTEGER)
«unique»
+ UQ_mtc_product_id(INTEGER)
+ UQ_mtc_product_user_id(INTEGER)
mtc_group_product_feature
«column»
*PK id :INTEGER
FK group_id :INTEGER = null
FK product_id :INTEGER = null
* discount :INTEGER
«FK»
+ FK_mtc_group_product_feature_mtc_group(INTEGER)
+ FK_mtc_group_product_feature_mtc_product(INTEGER)
«PK»
+ PK_mtc_group_product_feature(INTEGER)
«unique»
+ UQ_mtc_group_product_feature_id(INTEGER)
+FK_mtc_group_product_feature_mtc_product
«FK»
(product_id = id)
+PK_mtc_product
+FK_mtc_group_product_feature_mtc_group
(group_id = id)
«FK»
+PK_mtc_group
+PK_mtc_group_product_feature
(groupProductFeature_id = id)
«FK»
(degree_id = id)
«FK»
+PK_mtc_degree
+FK_mtc_passwd_reset_log_mtc_user
(user_id = id)
«FK»
+PK_User
+FK_mtc_notification_mtc_user
+FK_mtc_product_mtc_user
(user_id = id) +PK_User
«FK»
mtc_user
«column»
*PK id :INTEGER
*FK degree_id :INTEGER
username :VARCHAR(25)
identifier :VARCHAR(60)
firstName :VARCHAR(30)
lastName :VARCHAR(35)
gender :VARCHAR(6)
city :VARCHAR(30)
aboutMe :TEXT
password :VARCHAR(64)
email :VARCHAR(60)
imagePath :VARCHAR(225)
randomLinkKey :VARCHAR(64)
«FK»
+ FK_mtc_user_mtc_degree(INTEGER)
«PK»
+ PK_User(INTEGER)
«unique»
+ UQ_mtc_user_degree_id(INTEGER)
+ UQ_mtc_user_email(VARCHAR)
+ UQ_mtc_user_identifier(VARCHAR)
+ UQ_mtc_user_username(VARCHAR)
+PK_User
(user_id = id)
«FK»
+FK_mtc_user_mtc_degree
mtc_group
«column»
*PK id :INTEGER
FK creator_id :INTEGER
* name :VARCHAR(30) = null
identifier :VARCHAR(40)
description :TEXT = null
* created_at :DATETIME
* is_default :BOOL
is_public :BOOL
«FK»
+ FK_mtc_group_mtc_user(INTEGER)
«PK»
+ PK_mtc_group(INTEGER)
«unique»
+ UQ_mtc_group_creator_id(INTEGER)
+ UQ_mtc_group_id(INTEGER)
+FK_mtc_group_mtc_user
«FK»
(creator_id = id)
+PK_User
+PK_mtc_group
(group_id = id)
«FK»
mtc_role
«column»
*PK id :INTEGER
* name :VARCHAR(30)
* role :VARCHAR(20)
«PK»
+ PK_mtc_role(INTEGER)
«unique»
+ UQ_mtc_role_id(INTEGER)
+ UQ_mtc_role_role(VARCHAR)
+PK_mtc_role
«FK»
(role_id = id)
+FK_user_role_mtc_role
+FK_user_role_mtc_user
(user_id = id)
«FK»
+PK_User
mtc_bcard
«column»
*PK id :INTEGER
FK user_id :INTEGER = null
* name :VARCHAR(40)
* is_default :BOOL
«FK»
+ FK_mtc_bcard_mtc_user(INTEGER)
«PK»
+ PK_mtc_bcard(INTEGER)
«unique»
+ UQ_mtc_bcard_id(INTEGER)
+PK_User
(user_id = id)
«FK»
+FK_mtc_bcard_mtc_user
+PK_mtc_bcard
group_member
«column»
*FK user_id :INTEGER
*FK group_id :INTEGER
«FK»
+ FK_group_member_mtc_group(INTEGER)
+ FK_group_member_mtc_user(INTEGER)
+FK_group_member_mtc_user
+PK_User
(user_id = id)
«FK»
+FK_group_member_mtc_group
+FK_mtc_invitation_mtc_group_product_feature
mtc_invitation
«column»
*PK id :INTEGER
FK user_id :INTEGER = null
* created_at :DATETIME
* is_accepted :BOOL
FK groupProductFeature_id :INTEGER = null
is_opened :BOOL
* offered_price :DOUBLE
«FK»
+ FK_mtc_invitation_mtc_group_product_feature(INTEGER)
+ FK_mtc_invitation_mtc_user(INTEGER)
«PK»
+ PK_mtc_invitation(INTEGER)
«unique»
+ UQ_mtc_invitation_groupProductFeature_id(INTEGER)
+ UQ_mtc_invitation_id(INTEGER)
+ UQ_mtc_invitation_user_id(INTEGER)
+PK_User
«FK»
(user_id = id)
+FK_mtc_invitation_mtc_user
user_role
«column»
FK user_id :INTEGER
FK role_id :INTEGER
«FK»
+ FK_user_role_mtc_role(INTEGER)
+ FK_user_role_mtc_user(INTEGER)
+PK_mtc_bcard
(bcard_id = id)
«FK»
+FK_bcard_contact_mtc_bcard
+PK_User
+FK_mtc_contact_mtc_user
(user_id = id)
«FK»
mtc_contact
«column»
*PK id :INTEGER
FK contact_type_id :INTEGER = null
FK user_id :INTEGER = null
* content :VARCHAR(250)
is_active :BOOL
«FK»
+ FK_mtc_contact_mtc_contact_type(INTEGER)
+ FK_mtc_contact_mtc_user(INTEGER)
«PK»
+ PK_mtc_contact(INTEGER)
«unique»
+ UQ_mtc_contact_contact_type_id(INTEGER)
+ UQ_mtc_contact_id(INTEGER)
+ UQ_mtc_contact_user_id(INTEGER)
+FK_mtc_contact_mtc_contact_type
«FK»
(contact_type_id = id)
+PK_mtc_contact_type
mtc_contact_type
«column»
*PK id :INTEGER
* name :VARCHAR(40)
* icon :VARCHAR(20)
* allowed_count :SMALLINT
«PK»
+ PK_mtc_contact_type(INTEGER)
«unique»
+ UQ_mtc_contact_type_id(INTEGER)
bcard_contact
«column»
*FK bcard_id :INTEGER
*FK contact_id :INTEGER
«FK»
+ FK_bcard_contact_mtc_bcard(INTEGER)
+ FK_bcard_contact_mtc_contact(INTEGER)
+FK_bcard_contact_mtc_contact
+PK_mtc_contact
«FK»
(contact_id = id)

2.3 Návrh uživatelského rozhraní

Vzhled a podstata uživatelského rozhraní vychází z požadavků zadavatele na funkčnost a analýzy existujících řešení. Hlavním cílem návrhu je vytvoření uživatelsky použitelného prototypu uživatelského modulu aplikace zabývající se vzděláváním na internetu v rozsahu funkčních požadavků zadavatele. Následující kapitoly se věnují nejdůležitějším obrazovkám návrhu.

2.3.1 Přihlášení uživatele do aplikace

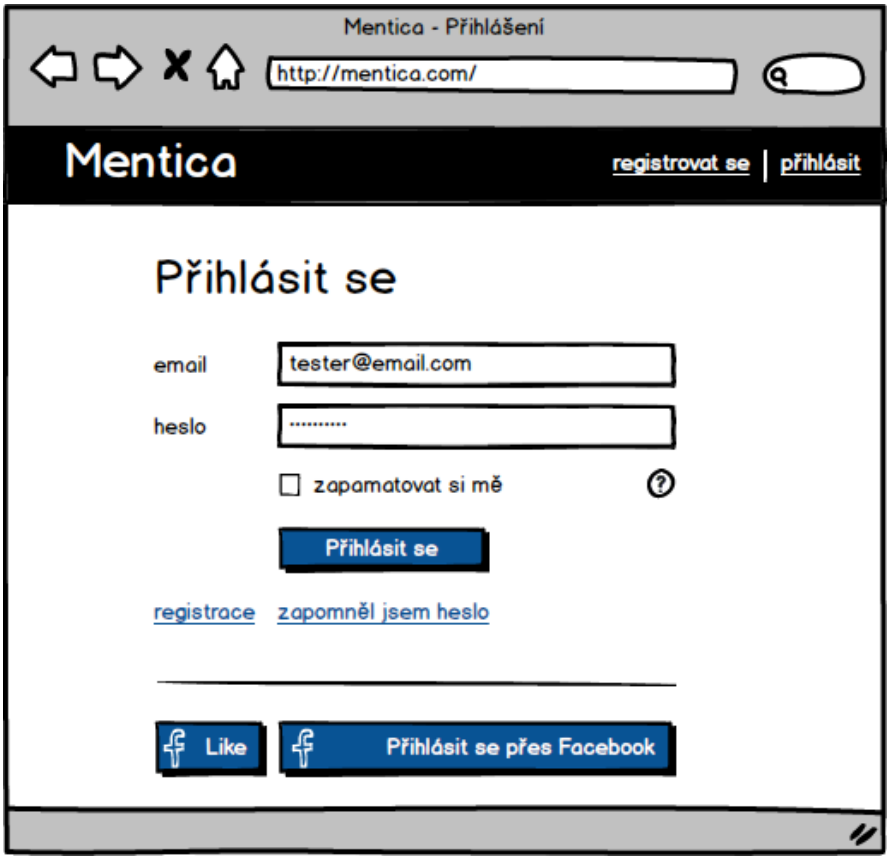
Pro práci v aplikaci se musí uživatel přihlásit, chce-li se podílet na vytváření obsahu nebo nějaký odebírat. Obrazovku s formulářem pro přihlášení zachycuje obrázek 2.3. Formulář požaduje zadání emailu, vloženého při registraci a hesla. Na základě těchto údajů je ověřena existence uživatele v systému a případně přidělena autorizace pro vstup do zabezpečených částí. Uživateli je umožněno se přihlásit, respektive registrovat přes sociální službu `Facebook`, přesměrovat na stránku s registračním formulářem nebo na stránku s formulářem pro obnovu hesla.

2.3.2 Zobrazení uživatelského profilu

Uživatel může zobrazit jak svůj, tak cizí profil. Obrázek 2.4 ukazuje obsah při zobrazení vlastního profilu. V levé části je zobrazena profilová fotografie uživatele, kterou lze libovolně měnit. Má-li uživatel vytvořenou vizitku, kterou si nastavil jako výchozí, je pod fotografií vypsán její obsah, který je kompletně veřejný. Výchozí vizitka se také automaticky připojuje k vytvořenému kurzu nebo testu, v případě, že mentor nezvolil jinak. Uprostřed jsou vypsány detaily profilu, které uživatel zadal při registraci kromě emailové adresy, která ve výchozím stavu viditelná ostatním není. V tabulce je také zobrazen informační `tooltip`, který se zobrazí po najetí kurzorem na ikonu s otazníkem. V pravé části je zobrazena hlavní navigační nabídka, která se zobrazí pouze vlastníkovi profilu. Čísla u některých položek značí počet neotevřených nebo nevyřešených obsahových prvků, jako například počet otevřených nabídek nebo počet nepřečtených notifikací.

2.3.3 Správa nabídek

Jak bylo popsáno v části o entitách doménového modelu, nabídka může vzniknou několika způsoby. Uživatel má mít poté možnost nabídku přijmout nebo odmítnout, a proto je třeba nabídky rozdělit do třech sekcí. Otevřené nabídky jsou ty, na které ještě nebylo reagováno. Když uživatel nějakou přijme, zobrazí se mu informativní hláška a zařadí se do sekce nabídek přijatých. Když naopak nějakou odmítne, tak se zařadí mezi odmítnuté nabídky. Na obrázku 2.5 je zobrazený seznam otevřených nabídek, kde u každé je možnost přijetí či od-



Obrázek 2.3: Návrh přihlašovacího formuláře a stránky

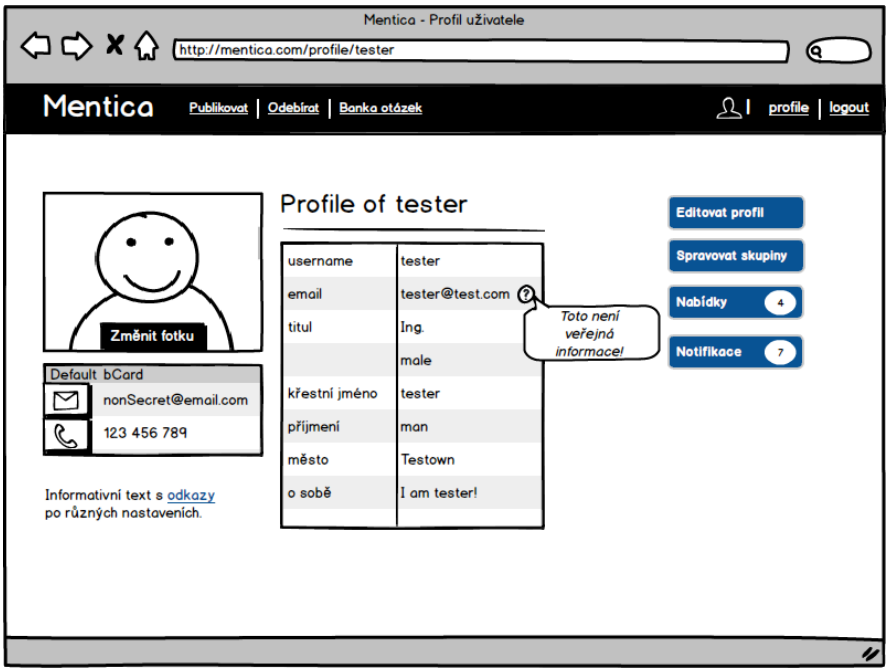
mítnutí, vždy však s nutností potvrzení zvolené akce v dialogu vyskakovacího okna.

2.3.4 Editace profilu

Po kliknutí na odkaz "Editace profilu"zobrazený na obrázku 2.4 je uživatel systémem přesměrován na stránku, kde si může upravit údaje vložené při registraci. Všechny atributy, které lze měnit, jsou vykresleny na obrázku 2.6, kde je i viditelné, že email použitý při registraci měnit nelze. Jde o bezpečností prvek, který zvyšuje ochranu uživatele. Validační kontroly i požadované vstupy jsou stejné jako u registračního formuláře. Tento prototyp z části naznačuje formu registrační stránky.

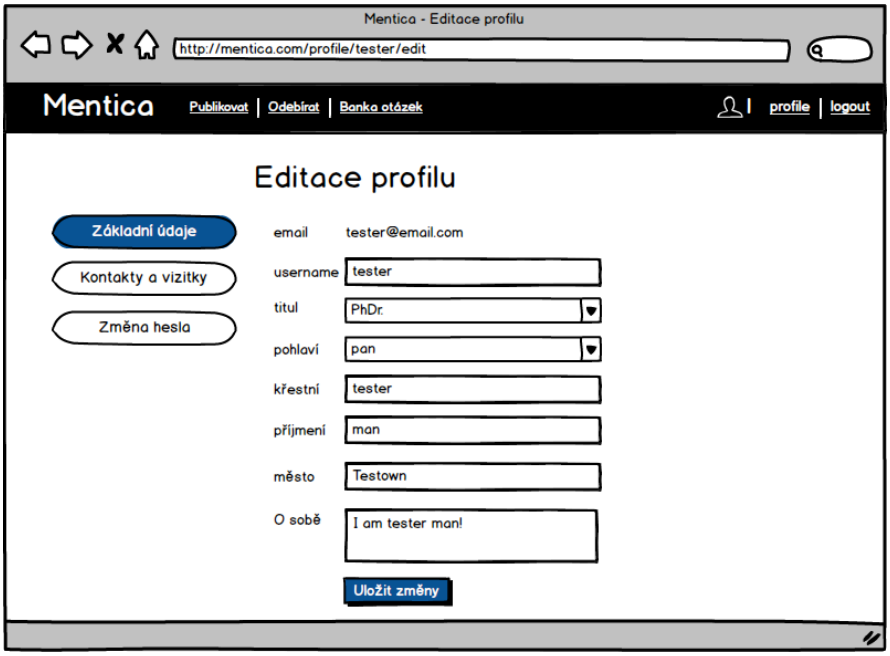
2.3.5 Správa kontaktů a vizitek

Návrh správy kontaktů a vizitek je jedním z rozsáhlejších a výsledek je zobrazen na obrázku 2.7. Jedním z požadavků byla možnost sdružování vytvořených



Obrázek 2.4: Návrh zobrazení profilu uživatele

Obrázek 2.5: Návrh zobrazení otevřených nabídek

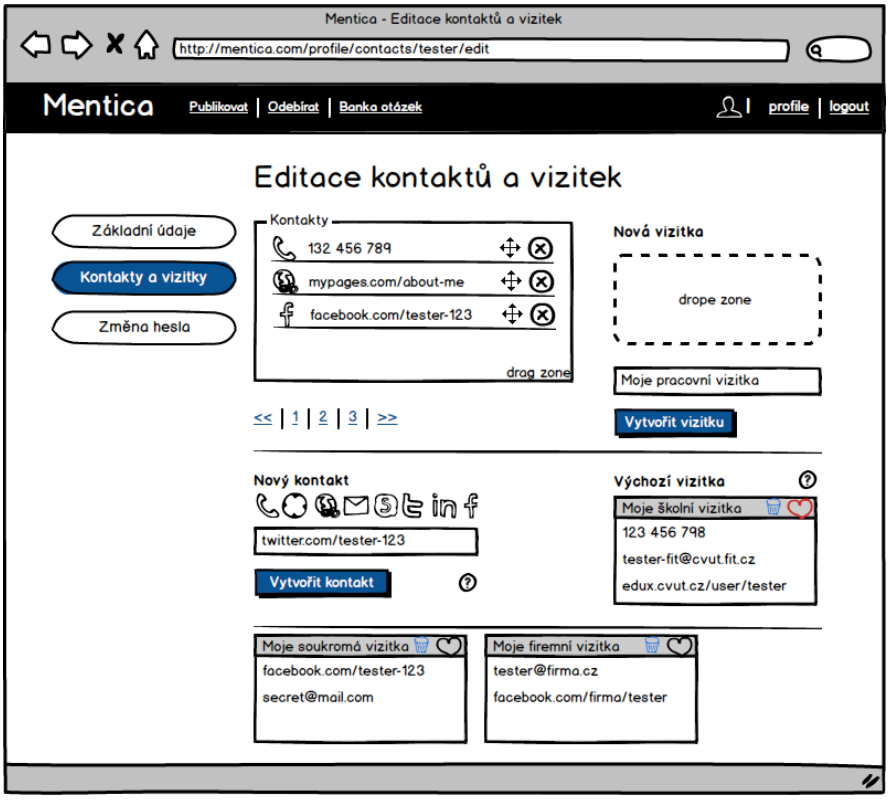


Obrázek 2.6: Návrh stránky editace profilu, údaje vložené při registraci

kontaktů do vizitek, které bude možné dále publikovat v rámci vytvořeného obsahu mentorem, a zároveň označení nějaké vizitky jako výchozí. Prototyp zobrazuje několik částí, které, ač jsou oddělené, spolu úzce souvisí. Horní třetina obsahuje výpis všech kontaktů uživatele s možností mazání a stránkování. Kontakty lze přesouvat do pole určeného pro vytvoření nové vizitky, a tím je sdružovat pod uživatelem zvolený název celku. Druhá třetina ukazuje jednoduchý formulář pro vytvoření kontaktu určitého typu a výchozí vizitku, má-li uživatel nějakou nastavenu. V dolní části jsou vypsány ostatní vizitky, které lze kliknutím na ikonu srdce označit jako výchozí. Taková vizitka může být pouze jedna. Typy kontaktů nejsou finální a po nasazení a ověření užívání v ostrém provozu se jejich počet nebo podoba může změnit.

2.3.6 Správa členů skupiny

Zakladatel skupiny má mít možnost spravovat její členy. Celkový počet uživatelů se může pohybovat až v řádu desetitisíců, a proto je filtrace nezbytná. Je potřeba spravovat členy skupiny, ale i uživatele, kteří členy nejsou, a zakladatel skupiny je chce připojit. Samotný způsob vzniku vazby mezi skupinou a studentem byl probírán velice podrobně. Byl diskutován postup, který vyžaduje akci studenta, tedy vyhledání skupiny na základě nějaké informace a následné připojení do skupiny kliknutím na odkaz. Tento přístup je na místě, pokud se jedná o větší množství studentů. Zde by nutnost přidávání členů mento-

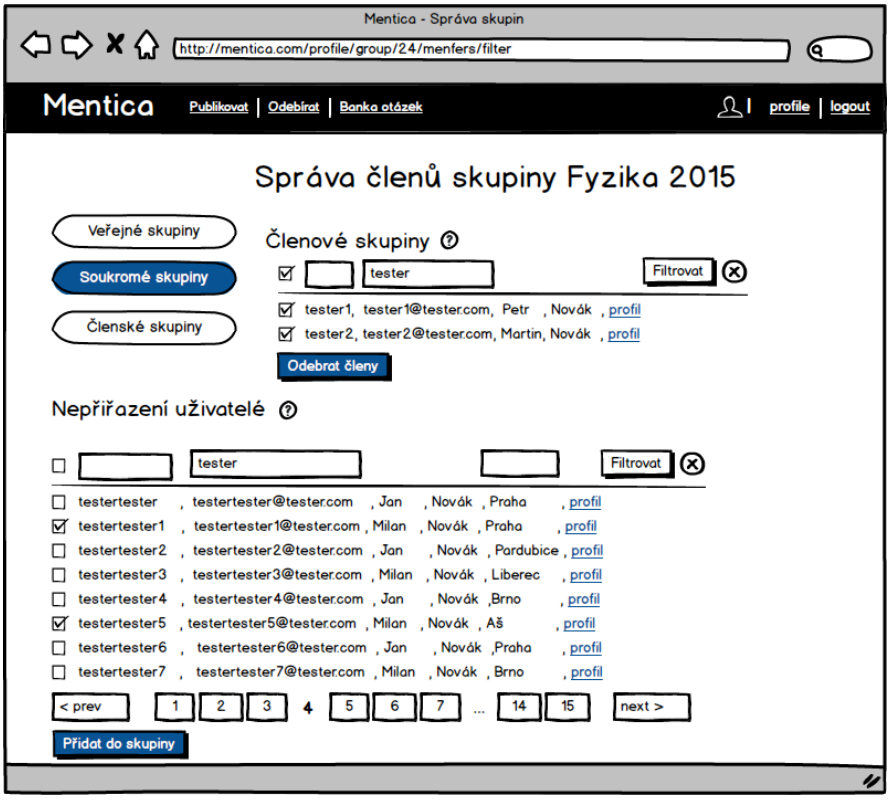


Obrázek 2.7: Návrh stránky, kde lze vytvářet a editovat kontakty a vizitky

rem samotným byla svazující a obtěžující. Skupiny však mohou být veřejné i soukromé. Proto vznikl koncept, kdy má uživatel možnost jednak skupinu vyhledat podle sděleného identifikátoru mentorem, přidat se do ní nebo v případě soukromých skupin o členství požádat, a i zakladatel samotný může potenciální členy vyhledat a přidat je. Následující `wireframe` zobrazuje situaci z pohledu mentora, který má možnost odebírat členy a zároveň přidávat členy nové. 2.8 Jako jeden z hlavních vyhledávacích atributů pro filtraci byl zvolen email z výchozí vizitky uživatele. Vyhledávat lze i podle uživatelského jména a města. Užívání atributů použitelných pro filtraci bude po nasazení zkoumáno, aby bylo dosaženo vysoké uživatelské přívětivosti.

2.4 Diagram komponent

Jak bylo zmíněno, tato práce se týká jednoho modulu, který je součástí většího celku - aplikace `Mentica`. Následující diagram 2.9 zachycuje všechny moduly a vazby mezi nimi. Jde o zjednodušený model, který naznačuje spolupráci všech modulů tvořících celek. Tato práce se zabývá uživatelským modulem. Ten má nejsilnější vazbu na modul kurzu, který vyvíjí student Bc. Jiří Matějka. Modul



Obrázek 2.8: Návrh přiřazování a odebírání studentů od skupiny

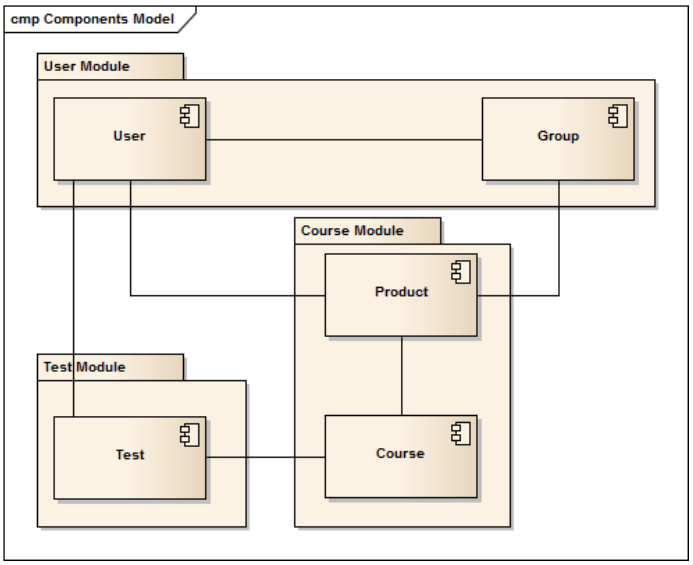
testů má na starosti studentka Bc. Olga Budnik.

2.5 Návrh poskytovaných služeb modulu

Pro společnou integraci je nutné navrhnout služby, které budou moduly poskytovat na základě předem domluvených vstupů určité funkce s konkrétními výstupy.

2.5.1 Získání objektu a jeho dat

Z analýzy vyplynulo nasazení systému `ORM Doctrine`, který umožňuje vytvoření vlastních repozitářů, s jejichž pomocí lze získávat konkrétní objekty a jejich data ve všech na sobě nezávislých `contollerech`. Vytvářený uživatelský modul musí poskytovat službu, pomocí které se získá objekt uživatele. Tato služba je poskytnuta samotným nasazením systému `ORM` automaticky.



Obrázek 2.9: Model komponent zachycující rozdělení práce na modulární úrovni

2.5.2 Vytvoření výchozí skupiny

Uživatel se může stát studentem kurzu, i když není členem v nějaké skupině. Student tím ztratí možnost slevy, ale získá možnost odebírání požadovaného obsahu. Pro naplnění tohoto případu užití vznikl pojem výchozí skupina, do které je uživatel přidán právě v případě, není-li členem skupiny, ke které je produkt připojený. S vytvořením produktu mentorem tedy automaticky vzniká výchozí skupina, která se od založené uživatelem liší pouze v atributu `isDefault`, který je znázorněný v doménovém nebo databázovém modelu. Z toho plyne požadavek na poskytnutí následujících služeb v modulu skupiny.

- vytvoř výchozí skupinu pro zadaný produkt
- přidej zadaného uživatele do výchozí skupiny zadaného produktu
- vrať výchozí skupinu zadaného produktu

2.5.3 Vytvoření a správa nabídky

V kapitole hlavní entity doménového modelu 2.2 jsou popsány akce, které zapříčiní vznik nabídky, se kterou může potenciální student dále nakládat. Z podstaty složitější vazby mezi nabídkou, skupinou a produktem přes další vazební entitu, která udržuje zvýhodněnou cenu, vyplynul požadavek na nasazení následujících služeb. (termíny nabídka a pozvánka jsou zaměnitelné)

- zjisti, zda existuje a zda dostal daný uživatel pozvánku k odebírání obsahu se zvýhodněnou cenou
- vytvoř pozvánku pro studenta ve skupině s vazbou na produkt
- vytvoř a automaticky přijmi pozvánku po přidání studenta do výchozí skupiny
- vytvoř všechny nabídky pro daného studenta po přidání produktu do skupiny
- vytvoř všechny nabídky pro všechny členy skupiny (studenty), do které byl přidán produkt
- zpracuj přijetí nabídky daným uživatelem
- zpracuj odmítnutí nabídky daným uživatelem

Služba přijetí nabídky zavolá službu, která zpracuje a vytvoří studijní proces pro daného uživatele, který nabídku přijal. Tuto službu vytvoří programátor modulu kurzů.

2.5.4 Vytvoření a správa notifikace

Ostatní členové týmu budou implementovat funkce, jejichž vedlejším efektem musí být vytvoření informace o dění v systému, kterou uživatel nalezne na jednom místě. Této problematiky se týkají notifikace, pro které jsou poskytnuty následující služby.

- vytvoř danému uživateli notifikaci s danou zprávou a odkazem a označ ji příznakem nová
- vrať všechny nové, respektive nepřečtené notifikace daného uživatele
- označ dané notifikace příznakem přečteno

Kapitola 3

Implementace

Po analytické části, ve které se zjišťovaly nedostatky a přednosti existujících řešení, sepsaly všechny případy užití a rozebraly další problémy týkající se aplikace `Mentica` a po části "Návrh", jejíž hlavní výsledky jsou databázový model a návrh uživatelského rozhraní, je možné přistoupit k samotné realizaci aplikace. Tato kapitola představuje popis nejdůležitějších částí implementace. V úvodu je popsána práce s frameworkem Symfony 2, jeho architektura a části konfigurace. Dále je uveden příklad vytvoření a používání entity pomocí systému Doctrine a rozebrány některé detaily práce s dokumentem v prohlížeči internetu u klienta.

3.1 Příprava vývojového prostředí

3.1.1 Instalace frameworku Symfony 2.6

Cílem této kapitoly je popsání průběhu instalace frameworku Symfony 2, který byl pro vývoj aplikace zvolen. V době výběru frameworku existovaly dva možné postupy instalace na lokální server nebo i mimo něj. Symfony totiž ve vývojové verzi obsahuje vlastní řešení lokálního serveru, na kterém může pracovat bez nutnosti běhu další aplikace. Jeho spuštění je sice snadné, pomocí jednoho příkazu v `consoli`, ale z důvodu jednodušší nastavitelnosti a správy serveru bylo pro vývoj zvoleno řešení s nutností běhu aplikace `XAMPP`, která obsahuje `Apache` a `MySQL` server.

Samotná instalace se provede pomocí manažeru závislostí Composer, ve kterém se spustí následující příkaz 3.1.

Listing 3.1: Instalace frameworku Symfony 2 příkazem přes Composer

```
$ composer create-project symfony/framework-standard-
    edition my_project_name
```

`Composer` se sám postará o stažení a instalaci veškerých knihoven v aktuální stabilní verzi. Pokud je z nějakého důvodu požadavek na konkrétní verzi frameworku, je ji možné nastavit přidáním čísla verze na konec příkazu. Po instalaci, která může v závislosti na rychlosti připojení trvat až několik minut, lze její správnost ověřit otevřením adresy `localhost` s konkrétním adresářem za lomítkem v prohlížeči. Pokud je spuštěný lokální server a vše proběhlo v pořádku, zobrazí se uvítací obrazovka s informací o úspěchu. Symfony nabízí kontrolní mechanizmus nastavení PHP a serveru pouhým otevřením scriptu `config.php`, jehož výstupem je podrobná tabulka s výpisem úspěšných, respektive neúspěšných kontrol.

3.1.2 Adresářová struktura aplikace

Adresářová struktura Symfony aplikace je velmi flexibilní, ale doporučená je následující.

app Konfigurace, šablony a soubory překladových balíčků. Obsahuje třídu `AppKernel`.

src PHP kódy a scripty projektu.

vendor Závislosti balíčků třetích stran a jejich implementace. Balíčky stažené `Composerem`.

web Kořenový adresář webové aplikace. Adresář pro ukládání veřejného a statického obsahu.

Kód vývojáře se umisťuje do podadresářů složky src do celků tvořících takzvané `bundles`. (dále balík) Každý takovýto celek obsahuje další podadresáře, ve kterých se nacházejí controllery, entity, formulářové typy, konfigurační soubory, šablony a další. Balík se dá vytvořit ručně, kde na vývojáře čeká spousta práce s pojmenováváním adresářů, které je vysoce náchylné k chybám a nebo lze využít automatizovaného postupu pomocí `consolového` příkazu.

3.1.3 Konfigurace aplikace

Symfony poskytuje oddělenou vrstvu konfigurace veškerých součástí aplikace. Kromě samostatného souboru používajícího notaci `YAML`, výhradně určeného k nastavení a konfiguraci balíků třetích stran získaných přes manažera závislostí `Composer`, jsou k dispozici soubory pro vývojový, produkční a testovací režim. Framework sám pozná v jakém režimu spouští aplikaci, a tak se vývojář nemusí složitě starat například o přístupové údaje do databázového uložiště.

Pro modul, kterým se zabývá tato práce, je nejdůležitější soubor `security.yml`, ve kterém se nastavuje například globálně používaná metoda `hashování` (příklad nastavení uveden v kapitole hashovací funkce a metody šifrování), hierarchie rolí užívaných v systému nebo poskytovatel objektu uživatele. Zmíněné role se promítají do nastavení přístupů do jednotlivých sekcí aplikace. Framework umožňuje vytváření takzvaných `firewalls` pro vývojovou, zabezpečenou část, a část administračního rozhraní, kde pro každou mohou být nastavena a použita jiná pravidla. V konfiguračním souboru lze nastavit i výjimky, na které se bezpečností pravidla nebudou vztahovat. Příkladem může být stránka s přihlašovacím formulářem, která nesmí vyžadovat existenci autentikovaného uživatele, jinak by došlo k zacyklení při nekonečném přesměrovávání na přihlašovací formulář. Příklad vytvoření takové výjimky je uveden v kódu 3.2.

Listing 3.2: Vytvoření výjimky routovacího pravidla, kde není vyžadováno přihlášení

```
...
access_control:
...
- { path: ^/login, roles: IS_AUTHENTICATED_ANONYMOUSLY }
...
```

Výjimky lze seskupovat do množin, takže není nutné vypisovat všechny názvy směrovacích pravidel.

3.2 Symfony console

Framework poskytuje několik `consolových` příkazů, které zjednodušují práci a vytváření kódu programátora. Integrace `Doctrine2 ORM` přidává ještě několik dalších příkazů, které umožňují například generovat entity, aktualizovat stav databáze a mapování na objekty nebo vygenerování takzvaných `getters and setters` k existující entitě. Seznam často užívaných příkazů s krátkým popisem je uveden v následující přehledu.

php app/console generate:bundle příkaz spustí průvodce vytvořením nového balíku

php app/console generate:doctrine:entity příkaz spustí průvodce vytvořením entity, kde jsou v dalších krocích pokládány otázky na zjištění názvu balíku, umístění, formátu konfigurace a výchozí struktury

php app/console doctrine:generate:entities TestBundle příkaz vygeneruje zmíněné "gettery a settery"pro entity v balíku TestBundle

php app/console doctrine:schema:update –force příkaz aktualizuje schéma databáze na základě struktury entit v celém projektu, jde o samotné mapování

php app/console cache:clear promazání dočasných souborů aplikace

3.3 MVC architektura v praxi

V této kapitole je uveden příklad zjednodušeného průchodu požadavku aplikací při vytváření odpovědi pro klienta. Zkratka `MVC` skrývá tři klíčová slova - `model, view` a `controller`. Nejprve se obdrží požadavek od uživatele a úkolem systému je napojit obdrženou adresu URL na funkční procesy. Napojení se nazývá `routování` a vytvoření jednoduchého napojení v notaci `YAML` s akcí v controlleru může vypadat například následovně. (kód 3.3)

Listing 3.3: Příklad definování route v konfiguračním souboru frameworku

```
invitations_open_list:
    path:       /list/open/user/{id}
    defaults: { _controller: "TesarInvitationBundle:
        Invitation:listOpenInvitations" }
```

Uvedený kód rezervuje klíčový řetězec názvu pro další použití a volání a URL adresu, která končí hodnotou atributu `path`, namapuje na akci `listOpen InvitationsAction` controlleru `InvitationController` v balíku "TesarInvitationBundle". Důležitý je parametr ve složených závorkách, který se předává jako vstup do funkce controlleru. V tomto případě jde o jednoznačný identifikátor uživatele v systému. Systém tedy zná parametry a akci, kterou má spustit. Obsah zmíněné funkce je zkráceně vypsán v následujícím kódu 3.4.

Listing 3.4: Akce v controlleru vypisující všechny nabídky uživatele

```
public function listOpenInvitationsAction($id){
    // ziskani entity User podle vstupniho parametru
    $em = $this->getDoctrine()->getManager();
    $user = $this->findUserById($id);

    // osetreni nenalezeni zaznamu ....

    //ziskani vsech otevrenych pozvanek uzivatele
    $invitationRepo = $em->getRepository('
        TesarInvitationBundle:Invitation');
    $invitations = $invitationRepo->findBy ....;

    //vytvoreni odpovedi v podobe volani sablony s
        predanim parametru
    return $this->render('TesarInvitationBundle:
        Invitation:list.html.twig',array(
         'invitations' => $invitations,
         'user'         => $user
    ));
}
```

Funkce tedy získá objekt uživatele podle identifikátoru ze vstupního parametru, získá repozitář entity `Invitation` a pomocí funkce `findBy` a vyspecifikování obsahu filtrování, získá všechny otevřené pozvánky uživatele. Struktura a vytvoření entity, tedy modelu, je popsána v následující kapitole. Tato data dále zasílá do šablony pomocí funkce `render` controlleru.

Výchozí šablonovací systém, který framework Symfony používá, má název `Twig`. Cílem užití tohoto systému je zjednodušení vytvoření HTML odpovědi. `Twig` je samostatný projekt od tvůrce Symfony frameworku, který nabízí rozsáhlou škálu funkcí. [27] Jednou z hlavních je procházení mapy všech napojených entit a jejich dat. V případě entity `Invitation` lze jednoduše vypsat data z entity `Group`, se kterou má entita komplexnější vazbu definovanou vztahem `ManyToMany` přes další vazební Entitu `GroupProductFeature`. Důležitou funkcí šablonovacího systému je i procházení datových struktur ve smyčkách. Zde procházení pole `invitations`, které bylo předáno controllerem.

Listing 3.5: Příklad užití šablonovacího systému TWIG pro vypsání dat z controlleru

```
...
{% for invitation in invitations %}
<tr>
<td> </td>
<td>
{% if invitation.createdAt %}
{{ invitation.createdAt|date('Y-m-d H:i:s') }}
{% endif %}
</td>
<td>{{ invitation.groupProductFeature.group.name }}</td>
<td>{{ invitation.groupProductFeature.product.name }}</
    td>
...
```

Uvedený výňatek kódu 3.5 doplňuje trojici controller a model o pojem view. V kódu je mimo jiné zachyceno i použití formátovacího `filteru date` ve struktuře if. Na posledním a předposledním řádku je znázorněno zmíněné získání dat z entity, na kterou existuje vazba. Zde jde o vypsání názvu skupiny a produktu, kterých se pozvánka, nebo-li nabídka, týká.

3.4 Vytvoření ORM entity pomocí systému Doctrine

V předchozí kapitole byl popsán postup vytvoření jednoduchého směrování, základní funkce controlleru a zajímavé funkce šablonovacího systému. Základním kamenem je model, o který se stará systém objektově relačního mapování. Využití systému `Doctrine` poskytuje kompletní objektový přístup k datům zcela automaticky, a tak se o databázi samotnou není příliš nutné starat. V předchozích příkladech byla zmiňována entita `Invitation`, a tak bude popsán zjednodušený postup vytvoření právě této entity. Z pohledu PHP jde o třídu s vyspecifikovanými atributy, kde je každý nějak okomentován. Těmto komentářům se říká notace a právě ty jsou zdrojem akcí pro systém Doctrine. Definice třídy je uvedena v následujícím kódu 3.6.

Listing 3.6: Příklad využití anotací pro definování Entity

```
...
/**
 * @ORM\HasLifecycleCallbacks
 * @ORM\Table(name="mtc_invitation")
 * @ORM\Entity()
 */
class Invitation {
...
```

Uvedený kód nejprve definuje skutečnost, že třída obsahuje události, které je třeba poslouchat. Příklad události, která se provede těsně před uložením záznamu do databáze je uveden v kódu 3.7. Další řádek definuje skutečný název tabulky v databázovém systému a třetí určuje, že následující třída PHP definuje požadovanou entitu. Příklad nastavení primárního klíče a vazby s jinou entitou je uvedený v kódu 3.7.

Listing 3.7: Nastavení primárního klíče, vazby na jinou entitu a příklad události PrePersist

```
...
/**
 * @ORM\Column(name="id", type="integer")
 * @ORM\Id()
 * @ORM\GeneratedValue(strategy="AUTO")
 */
private $id;

/**
 * @ORM\ManyToOne(targetEntity="\Tesar\UserBundle\Entity
    \User", inversedBy="invitations")
 */
private $user;

/**
 * @ORM\PrePersist
 */
public function updatedTimestamps() {
        if($this->getCreatedAt() == null) {
                $this->setCreatedAt(new \DateTime(date('
                    Y-m-d␣H:i:s')));
        }
}
...
```

První část uvedeného kódu definuje primární klíč pomocí funkce na druhém řádku a nastavuje automatické generování hodnot. Důležité je klíčové slovo `type`, kde se určuje datový typ parametru. Doctrine podporuje všechny důležité datové typy relačních databází. Další část ukazuje vytvoření vazby `Many to many` na entitu `User`, ve které je druhý konec vazby definován atributem `invitations`. Poslední část ukazuje způsob vytvoření zmíněné události, která se spustí těsně před uložením nového záznamu do databáze. Zde se jedná o uložení časové známky k atributu `createdAt` pomocí `setteru`, který není součástí příkladu kódu. Po vytvoření kompletní entity včetně definicí všech atributů, událostí, `getterů a setterů` a správného nastavení atributů vazeb na obou stranách, stačí spustit příkaz aktualizace schématu pomocí konzole. (příkaz je uveden v kapitole `Symfony console`)

3.5 Práce se Services na úrovni controlleru

V kapitole návrh poskytovaných služeb modulu byly vyspecifikovány služby, které musí konkrétní moduly poskytovat. Pojem služba se používá k definici objektu, který je vytvořený pro specifický účel. V této práci je to například správa nabídek. Vytvoření samotné služby je téměř totožné s konfigurací controlleru s tím rozdílem, že je nutné ji zaregistrovat v konfiguračním souboru `services.yml`, aby ji bylo možné volat a použít z jakéhokoliv controlleru v aplikaci. Zjednodušený příklad registrace služby je uvedený v kódu 3.8.

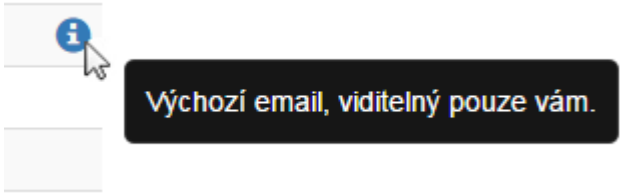
Listing 3.8: Registrace služby v notaci YAML

```
...
services:
    invitation_architect:
        class: ... \Controller\
            InvitationArchitectController
        arguments: [@service_container]
...
```

Službu lze poté volat pomocí funkce `get` s užitím definovaného identifikátoru. Příklad vytvoření instance služby a provedení její funkce je součástí kódu 3.9.

Listing 3.9: Ukázka volání funkce service z controlleru

```
//service na nabidky, prijeti pozvanky
$invitationService = $this->get('invitation_architect');
$invitationService->acceptInvitation($invitation ,true);
```



Obrázek 3.1: Informační tooltip obsahující nápovědu pro uživatele

3.6 Práce s dokumentem u klienta

Při interakci uživatele se systémem je potřeba upravovat prezentační vrstvu klienta a pracovat s objektově orientovanou reprezentací HTML dokumentu (dále `DOM`). Od vzniku první úrovně `W3C DOM`, standardizace specifických rozhraní webových prohlížečů k manipulaci s HTML elementy pomocí `JavaScriptu`, se objevilo několik technologií, které správu dokumentu usnadňují nebo přidávají další funkce, jako například `SAX, XPath, AJAX` nebo `jQuery`. Jako příklad bude popsána implementace dynamického prvku nápovědy (dále `tooltip`), který byl vytvořený použitím `JS` knihovny `jQuery`. (obrázek 3.1)

Nejprve je nutné vytvořit HTML elementy, se kterými se bude pomocí knihovny pracovat. V tomto případě se jedná o statický obsah s atributem data, který se následně získá JavaScriptem. Kód vytvoření HTML obsahu tooltipu je obsahem následující ukázky kódu 3.10. Při implementaci byly použity technologie HTML5 a CSS3. [28]

Listing 3.10: Vytvoření HTML tooltipu pro zpracování Java Scriptem

```
<span class="glyphicon glyphicon-info-sign master-
    tooltip" aria-hidden="true"
       data-tooltip="Vychozi email, viditelny pouze vam."
          >
</span>
```

Uvedený kód je i ukázkou použití souboru nástrojů `Bootstrap` a komponent `Glyphicons`. [18] Element má definovanou třídu, pomocí které se v klientském jazyce prvek vybere a zobrazí obsah atributu data. Tento úkon je zjednodušeně znázorněný v ukázce kódu 3.11.

Listing 3.11: Zpracování tooltipu Java Scriptem

```
$(document).ready(function(){
...
$('.master-tooltip').hover(function () {
// Hover over
    var title = $(this).data('tooltip');
    title = title.replace("EOL", "<br/>");
```

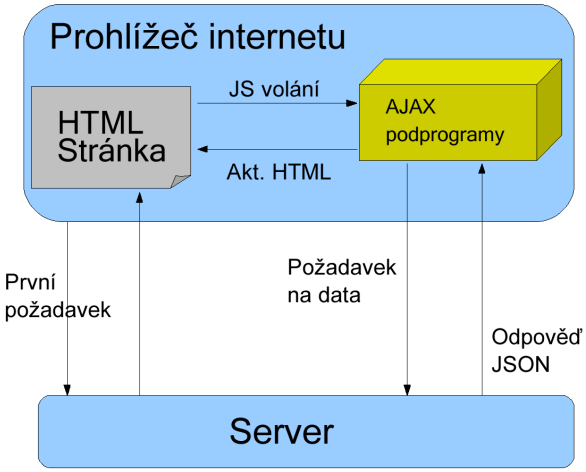
```
    $('<p class="m_tooltip"></p>').html(title).appendTo('
        body').fadeIn('slow');
}, function () {
// Hover out
    ...
    $('.m_tooltip').remove();
}).mousemove(function (e) {
// watch cursor position
    var mousex = e.pageX + 20; //Get X souradnice
    var mousey = e.pageY + 10; //Get Y souradnice
    $('.m_tooltip').css({top: mousey, left: mousex});
});
...
});
```

V první části je obsloužena událost přejetí kurzorem po prvku, který byl vybrán CSS `selectorem` třídy. Získá se obsah HTML atributu data, který se efektem postupného objevení přidá do těla dokumentu pomocí funkcí `appendTo` a `fadeIn`. Druhá část obsluhuje událost opuštění prvku kurzorem, kde se `tooltip` odstraní z DOMu. Poslední část zajišťuje sledování pohybu kurzoru pomocí CSS vlastností top a left pro element `tooltipu` s absolutní pozicí v dokumentu.

3.7 Vytvoření a správa AJAX požadavku

Technologie `AJAX` byla popsána v části návrhu v kapitole o použitých technologiích. Protokol HTTP, na kterém webová aplikace `Mentica` pracuje, je bezestavový, a tak je po nějaké akci, která má dopad na obsah dokumentu u klienta, nutné překreslit celou stránku. Použití technologie `AJAX` tuto nutnost odstraňuje. Proces vytvoření takového požadavku a následných akcí je zobrazen na obrázku 3.2.

Po odpovědi na první požadavek je u klienta v prohlížeči internetu sestavena HTML stránka. Asynchronní komunikace poté probíhá na základě nějaké akce od uživatele, například kliknutí na element nebo i přejetí kurzorem přes nějaký prvek, která je zpracována Java Scriptem, v tomto případě funkcí jQuery, která vytvoří požadavek. Ten je poté na serveru rozpoznán podle typu `XmlHttpRequest`, zpracován, a je vytvořena odpověď ve formátu `JSON`, která je dále zpracována stejnou funkcí jQuery, která požadavek prvně vytvořila a odeslala. Tento postup je použit hned v několika případech.



Obrázek 3.2: Komunikace se serverem při AJAX volání

- nastavení příznaku přečteno u notifikace uživatele
- ověřování existence účtu po přihlášení pomocí brány Facebooku
- odhlášení uživatele ze systému při odhlášení pomocí Facebooku
- smazání kontaktu uživatelem
- smazání vizitky uživatelem
- označení vizitky jako výchozí
- odebrání kontaktu z vizitky (zjednodušeně v kódu 3.12)

Listing 3.12: Vytvoření AJAX požadavku na adresu odkazu z atributu href

```
$.ajax({
  url: $url,
  type: "POST",
  dataType: "json",
  cache: false
  }).done(function(obj){
  $("#bcard-contact-row-"+obj.removedContactFromBcardId)
      .fadeOut(300, function(){
  $(this).remove();
  }).error(function(obj){
  //kod pri nejake chybe
  });
});
```

Listing 3.13: Obsluha AJAX požadavku na straně serveru v controlleru

```
/**
* metoda smaze kontakt z vizitky
* obsluhuje klasicky i AJAX pozadavek
*/
public function removeContactFromBcardAction($bcardId,
    $contactId, Request $request) {
 //kontroly pristupu role, ziskani vizitky atd.
 //smazani kontaktu z vizitky

 //removedContactFromBcard
 //vytvoreni JSON odpovedi pri volani AJAXem
 if($request->isXmlHttpRequest()) {
  $json = json_encode(array(
  'removedContactFromBcardId' => $bcardId
 ));

 $response = new Response($json);
 $response->headers->set('Content-Type', 'application/
     json');
 return $response;
 }

 //akce presmerovani v prizade, ze neslo o AJAX
     pozadavek
}
```

3.8 Validace vstupů od uživatele

Validace je proces kontroly a ošetření vstupních dat od uživatele. Hlavním důvodem validování dat je zabezpečení aplikace, kdy by se nesprávným ošetřením mohl útočník skrz formulář dostat k citlivým datům. Neméně důležitý důvod je navedení uživatele na užití správného formátu požadovaných vstupů za použití co nejméně otravných hlášek nebo načítání stránky. Webové aplikace se dají rozdělit na tři části, a to na klientskou, serverovou a databázovou. Moderní aplikace zapojují validaci dat do všech částí.

Symfony je PHP framework, který obsluhuje serverovou i databázovou část. Ač pro framework existují knihovny třetích stran, které nějak řeší validaci na klientské straně, tak primárním účelem Symfony není validace na straně webového prohlížeče. Symfony nicméně vytváří formuláře v HTML5, které na určité úrovni validaci v prohlížeči uživatele nasazuje. (obrázek 3.3) Nativní validace probíhá i při vypnutém JavaScriptu a další výhodou použití je nabídnutí speciální klávesnice na mobilních zařízeních s dotykovým dis-

Obrázek 3.3: Validace formuláře prohlížečem internetu pomocí HTML5

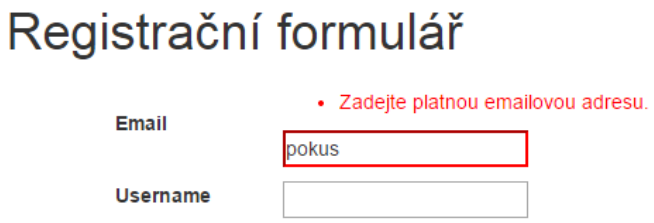
plejem podle užitého typu. (například pro typ email se zobrazí klávesnice se zavináčem)

Na kontrolu HTML5 se ale nelze plně spoléhat, protože je z velké části v režii prohlížeče, které jsou v tomto ohledu velmi nesourodé. Na internetu proto existuje velké množství aplikací, které se zabývají podporou funkcí HTML5 v prohlížečích internetu. Jednou z nich je i aplikace, která zobrazí podrobnou tabulku funkcí a jejich podporu v aktuálně otevřeném prohlížeči. [29] Je tedy nutné použít JS knihovnu nebo nasadit vlastní řešení validace formuláře u klienta pomocí Java Scriptu, které preferuji. Příklad funkce, která validuje emailové adresy pomocí regulárního výrazu, je uvedený v kódu 3.14.

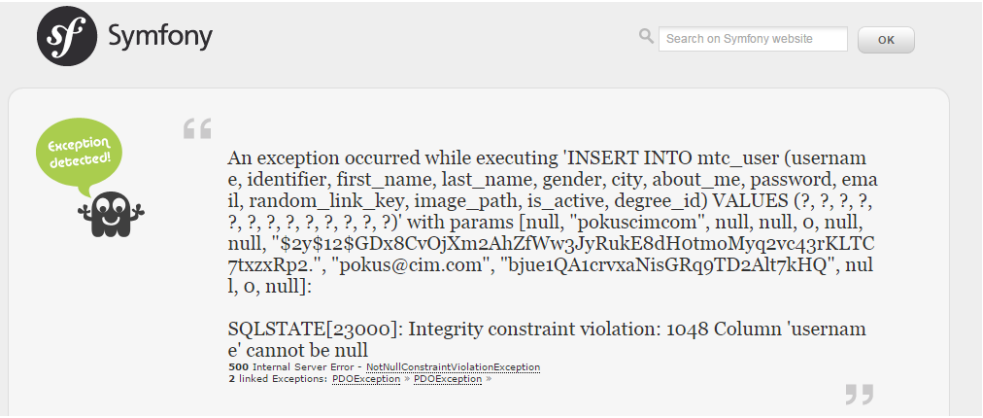
Listing 3.14: Validační funkce v jazyce Java Script

```
function validateEmail($email) {
 var emailReg = /^([\w-\.]+@([\w-]+\.)+[\w-]{2,4})?$/;
 return emailReg.test( $email );
}
```

Validaci na straně serveru zajišťuje důmyslně integrace Doctrine do PHP frameworku, kde se podobně jako při definování atributů využívá anotací. ORM se stará o kontrolu dat před uložením do databáze a v případě nesplnění validačních podmínek vrátí výjimku, se kterou může programátor dále pracovat a nějak ji prezentovat uživateli. Příklad vrácení výjimky, kdy nebylo splněno pravidlo vyžadující nenulový vstup pole uživatelského jména, na vývojovém serveru je zachycen na obrázku 3.4. Kontrola na serverové části se nastavuje na několika místech, je tedy nutné, aby si vývojář nebo tým zvolil jednu cestu, které se bude držet. Při implementaci aplikace `Mentica` bylo rozhodnuto používat zmíněné anotace ve spojení s dodatečnými pravidly v definicích samotných formulářů. Symfony obsahuje mechanismus vytváření znovu použitelných typů formulářů, kde se definují veškeré formulářové prvky. Příkladem je registrační formulář, který sice využívá entity User, ve které jsou pomocí anotací nastavena pravidla, ale součástí formuláře je `checkbox`, jehož zaškrtnutím uživatel vyjádří souhlas s podmínkami užívání. (3.15)



Obrázek 3.4: Ukázka výsledku validace vstupu od uživatele na straně serveru



Obrázek 3.5: Vývojová hláška frameworku Symfony při nedodržení databázového omezení - constrain

Listing 3.15: Přidání checkboxu s podmínkami užití, který musí mít hodnotu true pro splnění validačních pravidel

```
use Symfony\Component\Validator\Constraints as
    constraints;
...
$builder->add('terms','checkbox', array('mapped' =>
    false, 'constraints' => array(new True())));  //
    NotNull()
...
```

Výsledek chybně vloženého vstupu od uživatele a validace na straně serveru může vypadat například jako na obrázku 3.5, kde uživatel nevložil emailovou adresu ve správném tvaru.

3.9 Facebook JS SDK

Jedním z cílů a požadavků na funkčnosti bylo vytvoření integrace sociální služby `Facebook` pro umožnění přihlašování, sdílení obsahu na profilu sociální

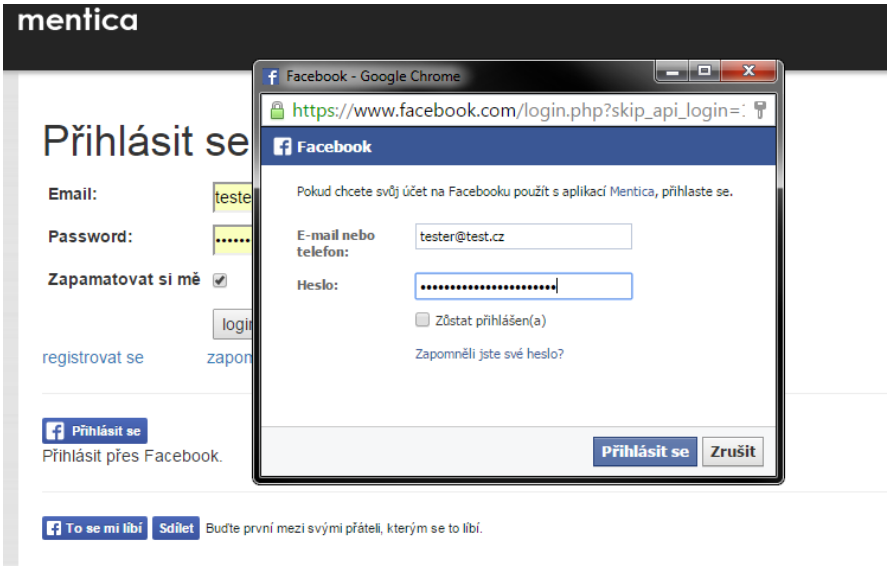
služby a přidávání komentářů. Prvním krokem je vytvoření Facebook Aplikace (dále MenticaFB) ve webovém rozhraní Facebook Developers, což je možné pouze ze soukromého účtu sociální služby a nikoliv z účtu stránky, jak by se mohlo zdát. Tímto omezením je docíleno kontroly nad přístupy do MenticaFB, kde je možné spravovat přístupy ostatních členů sociální sítě. Každá tato aplikace dostane přidělený jednoznačný identifikátor (dále appId), pomocí kterého je docíleno požadované integrace. Při užití SKD vydaného organizací Facebook lze zobrazovat statistiky přístupů Facebook API a podrobný výpis volání jednotlivých funkcí.

Jak bylo popsáno v analytické části této práce, pro vývoj bylo zvoleno vlastní řešení integrace pomocí Facebook JS SDK, které disponuje detailní dokumentací a příklady užití na stránkách projektu. [9] Doporučený postup užití je stažení scriptu v reálném čase pomocí Java Scriptu. Následuje propojení se založenou MenticaFB aplikací pomocí obdrženého identifikátoru appId. Následující kód 3.16 zjednodušeně ukazuje získání scriptu a nastavení propojení s MenticaFB.

Listing 3.16: Vytvoření napojení skutečné aplikace na Facebook Aplication vytvořenou na Facebook Developers

```
(function(d, s, id) {
 var js, fjs = d.getElementsByTagName(s)[0];
 if (d.getElementById(id)) return;
 js = d.createElement(s); js.id = id;
 js.src = "//connect.facebook.net/cs_CZ/sdk.js";
 fjs.parentNode.insertBefore(js, fjs);
}(document, 'script', 'facebook-jssdk'));
...
window.fbAsyncInit = function() {
FB.init({
 appId      : '123456789123456',
 cookie     : true,
 //...
});
};
```

Nyní je vše připraveno pro práci s funkcemi SDK, které podporuje i automatické ostylování tlačítek a ostatních prvků pouhým definováním jejich struktury. Například element fb:login-button se v HTML dokumentu prezentuje jako klasicky modré tlačítko s logem sítě a automaticky reaguje na kliknutí voláním funkce, která kontroluje stav přihlášení uživatele v síti Facebook, tedy otevřením vyskakovacího okna s přihlašovací bránou. (obrázek 3.6) Zmíněná funkce využívá získaného statutu v podobě klíčového slova connected nebo not_authorized. Statutů existuje celá řada, ale pro potřeby integrace jsou dva zmíněné dostačující. Po získání autorizace přes Facebook lze zjistit některá data z profilu sociální sítě, jako email, křestní jméno, příjmení a pohlaví,



Obrázek 3.6: Přihlašovací brána Facebook, integrovaná do aplikace Mentica

což jsou i povinné údaje při registraci do aplikace `Mentica`. Získat lze i další data, jako například seznam přátel a stránek, které uživatel označil tlačítkem "Líbí se mi", ale k tomu je potřeba souhlas uživatele. Jednoduchý postup získání dat od uživatele po jeho autorizaci je uveden v kódu 3.17.

Listing 3.17: Příklad získání dat z profilu uživatele sociální sítě Facebook

```
FB.api('/me', function(response) {
    var email = response.email;
    var firstName = response.first_name;
    var name = response.name;
    //...}
```

Dalším krokem je ověření existence emailové adresy v systému pomocí technologie `AJAX`. Systém na neexistenci adresy při prvním přihlášení pomocí FB reaguje vytvořením profilu uživatele, který si může v editaci později vytvořit heslo pro přihlášení přes klasický formulář. Ten je popsaný například v návrhové části, týkající se uživatelského rozhraní. 2.3.1 Volaná akce v controlleru je uvedena v pseudokódu 3.18.

Listing 3.18: Pseudokód akce controlleru při ověřování existence uživatele v systému

```
public function checkFacebookAction() {
if (existuje email) {
 //najdi uzivatele - user
  if(uzivatel) {
  //prihlasit
  return loginUserJsonResponse(user);
  } else {
  //vytvorit a pak prihlasit
  user = createFacebookUser(email, firstName, lastName,
     gender);
  //prihlasim usera
  return loginUserJsonResponse(user);
 }
}
return null;
}
```

Je potřeba zajistit i správné chování v případě odhlášení ze sytému. Uživatel může být přihlášený pomocí klasického bezpečnostního páru email a heslo nebo výsledkem zadání údajů do přihlašovací brány `Facebooku`, nikdy však oběma způsoby naráz. Když se uživatel odhlásí ze systému, je potřeba zajistit, aby se odhlásil i z `Facebooku`, ale celkově. Pokud má tedy uživatel existující přihlášení v jiném panelu prohlížeče, je odhlášen i zde, což je výchozí chování `FB SDK` pro zvýšení bezpečnosti.

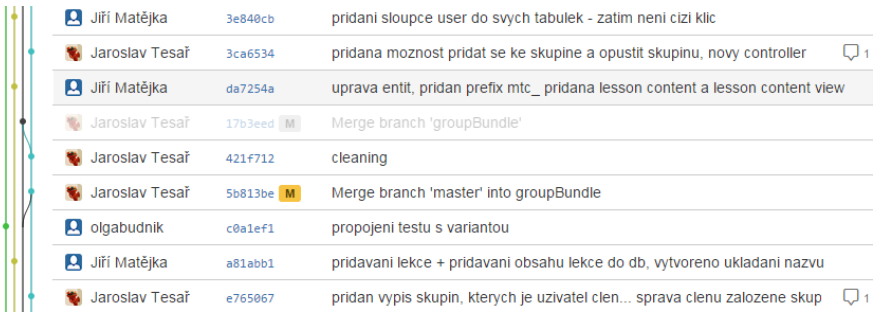
3.10 Verzovací systém GIT

Při vývoji softwaru v tříčlenném týmu je potřeba udržovat implementaci dostatečně přístupnou pro všechny programátory. Aplikace takového rozsahu vyžaduje nasazení verzovacího systému, který by zvýšil koordinaci týmu. Téměř nezáleží, zda-li se jedná o starší `Subversion` nebo rozsáhlejší GIT, ale je nutné nějaký používat. Projekt `Mentica` je kompletně verzovaný pomocí systému GIT ve spojení s webovým správcem a rozhraním `Bitbucket`. Kompletní popis možností systému je nad rámec této práce, budou popsány jen základní a užívané funkce.

git clone repository_name vytvoří lokální kopii existujícího repositáře GIT

git pull automaticky vyzvedne (`fetch`) a začlení (`merge`) vzdálenou větev do aktuální, kterou je třeba mít nastavenou ke sledování vzdálené větve

git checkout správa větví, vytvoření nové s načtením nebo zvolení větve jako aktuální



Obrázek 3.7: Ukázka části výpisu commitů na serveru Bitbucket

git add zahájí sledování nových souborů (je doplněn uvedením cesty k souboru nebo adresáři)

git commit vytvoření `commitu` (je doplněno o komentář za příznakem -m)

git push odeslání větve na vzdálený server, sdílení

Pro jednodušší správu a možnost nahlížení do kódu jiného vývojáře byla použita webová aplikace `Bitbucket`, která nabízí kompletní seznam commitů rozdělených podle větví, možnost vytváření delegovatelných problémů s diskusí, vytváření `wiki` stránek nebo dokumentace a možnost procházení scriptů s rozdílovými řádky mezi verzemi. Obrázek 3.7 ukazuje některé commity z celkového počtu tři sta sedm. (počet commitů ke dni 21.4.2015)

3.11 Uživatelská příručka

V této kapitole budou popsány nejpodstatnější průchody uživatelského modulu aplikace `Mentica`. Některé části budou v upravené podobě použity jako obsah nápovědy pro uživatele. Hlavní cíle uživatele lze rozdělit následovně:

- registrovat se, nebo-li vytvořit účet vyplněním formuláře
- registrace / přihlášení pomocí služby `Facebook`
- vytvoření vizitky s kontakty
- vytvoření skupiny uživatelů
- přiřazení produktů (kurzů, testů) k vytvořené skupině
- odebírání obsahu bez příslušnosti ke skupině
- odebírání obsahu přiřazeného ke skupině

registrovat se, nebo-li vytvořit účet vyplněním formuláře 1. Uživatel klikne na odkaz "registrovat se"v pravém horním rohu.

2. Uživatel vyplní požadovaná vstupní pole označená červenou hvězdičkou, zaškrtnutím vyjádří svůj souhlas s podmínkami užívání, opíše text z obrázku a odešle formulář. Případné chyby ve formuláři, na které bude upozorněno, je třeba opravit a formulář odeslat znovu kliknutím na tlačítko "registrovat se".
3. Uživatel je vyzván k navštívení své schránky emailové adresy, kterou při registraci zadal a ke kliknutí na zaslaný odkaz, čímž emailovou adresu potvrdí. Do této akce nelze účet používat.
4. Uživatel bude při bezchybném postupu informovaný hláškou o úspěchu.

přihlášení pomocí služby Facebook 1. Uživatel klikne na odkaz "přihlásit se"v pravém horním rohu.

2. Uživatel klikne na tlačítko "Přihlásit se přes Facebook".
3. Uživatel je vyzván k zadání přihlašovacích údajů na svůj profil `Facebooku`.
4. Po autorizaci ve službě `Facebook` je uživatel přihlášen do systému a přesměrován na svůj buď již existující nebo právě vytvořený profil.
5. Uživatel může být informován o neexistenci hesla ke svému účtu, jehož vytvořením by mu přibyla možnost přihlásit se klasicky pomocí emailu a hesla.

vytvoření vizitky s kontakty 1. Uživatel klikne na tlačítko "editovat profil"z detailu svého uživatelského profilu.

2. V levém menu klikne uživatel na tlačítko "kontakty, vizitky".
3. Uživatel klikne na tlačítko "přidat vizitku", zobrazí se mu pole, do kterého přesune požadované kontakty z výpisu nalevo. Uživatel zadá název vizitky a vytvoří ji kliknutím na odkaz "uložit vizitku".
4. Nově vytvořená vizitka se zobrazí na stejné stránce ve výpisu v dolní části. Uživatel může některou ze svých vizitek označit za výchozí kliknutím na ikonu srdce.

vytvoření skupiny uživatelů 1. Uživatel klikne na tlačítko "spravovat skupiny"z detailu svého uživatelského profilu.

2. Uživatel klikne na tlačítko "vytvořit skupinu"nad výpisem již vytvořených skupin.
3. Uživatel vyplní formulář s názvem, identifikátorem pro vyhledávání a popisem skupiny. Uživatel si vybere, zda má skupina být veřejná či soukromá kliknutím na odpovídající formulářové pole.

4. Uživatel klikne na tlačítko "Vytvořit", a když proběhlo vše, včetně validací vstupů, v pořádku, stává se zakladatelem skupiny.
5. Z detailu skupiny nebo seznamu všech vytvořených uživatel klikne na "spravovat členy".
6. Uživatel vyplní požadované filtrační specifikace a potvrdí je kliknutím na tlačítko "Filtrovat". Ve vypsaném seznamu si vybere požadované studenty zaškrtnutím a výběr potvrdí kliknutím na tlačítko "Přidat do skupiny". K dispozici jsou filtrační pravidla uživatelského jména, emailu a města uživatele. Tato pravidla se mohou měnit v závislosti na aktuálních požadavcích uživatelů po spuštění aplikace.

přiřazení produktů (kurzů, testů) k vytvořené skupině
1. Uživatel zobrazí detail skupiny nebo seznam všech vytvořených kliknutím na odkaz "spravovat skupiny"ze svého uživatelského profilu.
2. Uživatel klikne na odkaz "správa produktů", kde se mu zobrazí již připojené produkty, které lze dále spravovat.
3. Uživatel u vybraného produktu klikne na tlačítko "připojit ke skupině", vyplní procentuální zvýhodnění oproti původní ceně a potvrdí kliknutím na tlačítko "připojit".
4. Uživatel je o úspěchu informován hláškou.

odebírání obsahu bez příslušnosti ke skupině
1. Uživatel zobrazí detail produktu (kurzu, testu) po vyhledávání nebo ze seznamu.
2. Uživatel klikne na tlačítko "studovat kurz"nebo "zkusit test". Bude-li obsah placený, odsouhlasí podmínky a cenu zaškrtnutím příslušného pole, načež se mu odečte potřebný počet kreditů.
3. Uživatel bude přesměrován na první stránku průchodu kurzem, respektive testem.

3.11.1 Popis instalace

Aplikace se z důvodu rozšiřování požadavků zadavatele, které budou analyzovány a navrženy v další iteraci vývoje, nachází ve fázi prototypu. Z tohoto důvodu není nasazená ani dostupná na internetové adrese. Instalace lokální verze ze souborů na přiloženém kompaktním disku je popsána následujícími kroky:

1. Instalace lokálního serveru. V případě operačního sytému windows, na kterém byl modul vyvíjen, lze použít aplikace obsahující lokální verzi serveru a databázové uložiště MySQL. Na operačních systémech Linux je nutné zprovoznit a nakonfigurovat LAMP `http://en.wikipedia.org/wiki/LAMP_%28software_bundle%29`.

2. Zkopírování souborů na přiloženém kompaktním disku do rootu adresáře lokálního serveru se zachováním adresářové struktury.

3. Spuštění příkazu na aktualizování vazeb balíků třetích stran, jejichž data nejsou z důvodu velké datové náročnosti na disku uvedena. Příkaz se spouští v konzoli pro prostředí composer v následující podobě. `php composer.phar update`. Podrobná dokumentace s příkazy pro composer se nachází na následující adrese `https://getcomposer.org/doc/03-cli.md`.

4. Nastavení přístupu do databáze pro uživatele s heslem v konfiguračním souboru aplikace `app/config/parameters.yml` společně s portem, hostem a driverem.

5. Naplnění databáze iniciačními daty. Bez tohoto kroku nebude možné aplikaci používat. Script ve formátu `sql` je přiložen na disku v adresáři instalace.

Kapitola 4

Testování

Testování aplikace `Mentica` se skládá z několika částí. Testování s uživateli se dále dělí podle všech modulů, ze kterých se skládá. Dále je navrženo a provedeno funkční testování a testování databáze.

Testování s uživateli pokryje vzhled a návrh uživatelského rozhraní. Jeho provedením se mohou zjistit zásadní nedostatky v návrhu `UI`, ze kterých se bude vycházet v návrhu v další iteraci vývoje. Při vyšším počtu participantů se mohou vyskytnout i funkční problémy, které jsou ale hlavní doménou funkčního testování. To se zabývá správným chodem a fungováním aplikace hlavně ve smyslu odkazů a struktury. Jsou testovány veškeré odchozí a interní odkazy, formuláře s validacemi a informovanost uživatele o akcích na pozadí. Testování databáze má na starosti vývojář, který zpracuje dávku dat a vytvoří mechanizmus, který otestuje konzistenci. Při vývoji aplikace často nastává problém se samotnými daty. Nejsou k dispozici data reálná, a tak se vytváří testovací, která jsou pouze odhadem, jak bude databáze vypadat.

4.1 Testování s uživateli

Zadavatel aplikace `Mentica` si nepřál provádět testování ve veřejném `usability labu` v prostorách fakulty školy. Tato kapitola navrhuje postup a přípravu testování s uživateli. Ten pak bude prezentovaný jako doporučený. Na samotném testování, které si přeje zadavatel uskutečnit podle svých požadavků, budou vývojáři v roli moderátora testu, vždy však pro modul, na jehož implementaci se přímo podíleli. Tato kapitola se věnuje navržení postupu testování uživatelského modulu aplikace `Mentica`.

4.1.1 Specifikace cílové skupiny

Z důvodu charakteru aplikace, je cílová skupina svými vlastnostmi velice rozsáhlá. V roli studentů se mohou vyskytovat žáci základních, středních a dalších škol, ale i lidé v důchodovém věku, ač právě na ty není vývoj cílen. Naopak v

roli mentora se předpokládá uživatel ve věku vyšším, který bude mít vztah k výpočetním technologiím mírně odlišný. I zde je ale možnost, že se mentorem stane uživatel školního věku. Chování a reálný stav skupiny, která bude aplikaci užívat, se jen těžko odhaduje a po reálném nasazení bude nutné reagovat na okamžitý stav. V tuto chvíli je nutné vybrat participanty z různých věkových skupin. Pro výběr účastníků samotného testu se vytváří `screener`. Jednoduchý formulář, který pomůže vybrat cílového uživatele. Doporučený počet účastníků se velmi liší, avšak je nutné vybrat takový, který pokryje kompromis ceny, času a efektivity testování. U jednotlivých možností ve formuláři jsou uvedena čísla, vyjadřující početní zastoupení skupiny u jednotlivé otázky v procentech. Otázka bez číselného rozdělení nehraje v dotazníku významnou roli.

4.1.2 Testované průchody

Jako nejpodstatnější průchody aplikací z pohledu uživatelského modulu byly zvoleny následující scénáře.

registrace uživatele do systému zde bude vhodné sledovat počet participantů, kteří použijí sociální službu `Facebook`. Dalším zajímavým atributem bude sledování chování uživatele při nutnosti vyjádření souhlasu s podmínkami užívání. V prostředí testovací místnosti se musí zajistit, aby se uživatel choval co nejvíce přirozeně, aby se předešlo situaci, kdy si zobrazí podmínky užívání jen proto, že by měl. Akce musí vycházet z jeho vlastního přesvědčení. Další detaily sledování budou reakce na nevalidní vstupy a nutnost opsání textu z obrázku.

vytvoření skupiny a přiřazení zadaných studentů v tomto scénáři bude zásadní způsob vyhledávání a přiřazování studentů do skupiny. Participant bude mít k dispozici tabulku s uživateli s rozšiřujícími informacemi. (uživatelská jména, emailové adresy a města vyplněná v profilech uživatelů) Se samotným založením se předpokládá nejistota při volbě parametru veřejná nebo soukromá skupina. Tyto formulářové prvky jsou doplněny o informativní `tooltip`, zmíněný například v implementační části. Sledováno bude užití i těchto prvků.

- vytvoření soukromé skupiny s názvem Fyzika pro 3B
- připojení studentů z poskytnutého seznamu (emailové adresy končící na sskh3b.cz)

připojení produktu ke skupině nejdůležitější parametr, který bude nutné sledovat, je pochopení konceptu zakladatele skupiny a tvůrce produktu, který musí být tatáž osoba, aby mohl připojení provést. Testování vytvoření produktu kurzu nebo testu bude navrženo v pracích dalších vývojářů aplikace `Mentica`. Očekávané problémy by se mohly týkat sekce s nastavením speciální ceny u připojování ke konkrétní skupině.

- připojení, v testování jiného modulu vytvořených, produktů ke skupině Fyzika pro 3B
- první s plnou cenou a druhý se 30% slevou z plné ceny

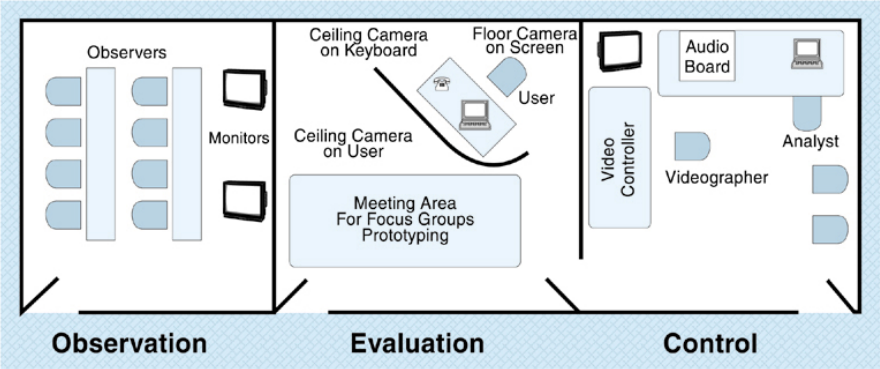
vytvoření vizitky s několika kontakty analýza a navržení toho UC byla při snaze dodržet požadavek zadavatele stěžejní. Zde bude důležité sledovat celkovou orientaci uživatele v navrženém rozhraní. Samotné vytvoření vizitky vyžaduje užití funkce `drag and drop`, a tak bude potřeba sledovat, jak si s problémem aktéři poradí. Funkce není v prostředí internetu příliš častá a problémy se očekávají od participantů vyšších věkových skupin.

- vytvoření telefonního kontaktu se zadaným číslem (nevalidní vstup a následně validní)
- vytvoření kontaktu s vlastní URL adresou
- vytvoření vizitky z vytvořených a předem vložených kontaktů pro potřeby testování
- zvolení jedné vizitky jako výchozí (pro potřeby testování jich bude několik připraveno)

4.1.3 Scénáře testování použitelnosti

Z předchozí kapitoly, která obecně specifikovala oblasti testování modulu s uživateli, plynou konkrétní úkoly. Samotné zadání je však `participantovi` prezentováno obecně, bez uvedení nutných kroků ke splnění požadovaného cíle. Uživatel se tímto dostává do situace, kdy musí sám řešit problém a tím vzniklé úkony, které budou i ze záznamů podrobně analyzovány, jsou předmětem sledování. Seznam a znění navržených scénářů, které budou pro testování doporučeny, je součástí přílohy této diplomové práce. B.2

Testování s uživatelem patří mezi časově i finančně nejnáročnější, ale bývá považováno za nejpřínosnější. Od testování s uživateli se očekává zjištění chování zástupců cílových skupin, konkrétně forma vkládaných dat do formulářů a reakce na chování systému. Výsledky zajistí potvrzení nebo vyvrácení hypotéz z analytické a hlavně návrhové části. Ověřený (zde přibližný) model testovacího zázemí je uveden na následujícím obrázku. 4.1 Obrázek byl stažen ze stránek [30] digitální knihovny a archivu `VirginiaTech`. V levé části se nachází místnost pro potenciální investory (stakeholders) nebo členy vývojářského týmu. Uprostřed je místnost, kde je prováděno samotné testování a poslední část obrázku zachycuje kontrolní stanoviště, ze kterého jsou obsluhována obrazová a textová záznamová zařízení. Výstupem je pak kompletní kamerový záznam minimálně obrazovky, klávesnice a obličeje participanta testu,



Obrázek 4.1: Ukázka možného rozmístění usability laboratoře, určené k testování s uživatelem

se kterým je v místnosti přítomen i moderátor, jehož cílem je provést aktéra celým testováním. Důležitým dokumentem je i post-test dotazník, který je součástí přílohy. B.3

4.2 Testování databáze

Téměř každý požadavek provedený uživatelem musí dostat odezvu od DB severu. Tato komunikace se pak stává kritickým bodem. Byl vytvořený jednoduchý test, který dodržuje validační procesy na serverové straně a do databáze nejprve vkládá větší počet připravených dat. Za tímto účelem byly připraveny funkce, které slouží pouze k tomuto účelu. Především se jednalo o naplnění tabulek:

- `mtc_user`
- `mtc_contact`
- `mtc_group`
- `group_member`

Testování tabulky uživatelů, skupin a vazeb

Při vytváření testovacích dat bylo nutné získat jména, příjmení, emailové adresy a pohlaví. Pro tyto účely byl vytvořen mechanizmus získávání náhodné kombinace jména a příjmení z webové stránky generátor jmen online. [31] Emailová adresa byla vytvořena složením jména a příjemní a přidání náhodné části za zavináčem a české domény. Funkcí `testUsers()` bylo takto vytvořeno 1000 testovacích uživatelských účtů. Tabulka skupin nepotřebovala žádná

speciální data. Zde byla polovina zvolena jako veřejná a další polovina jako soukromá skupina.

Vztahová tabulka `group_member`, která vyjadřuje příslušnost studenta v nějaké skupině, obsahuje pouze cizí klíče. Ta byla naplněna na základě náhodného generování čísel v rozsahu velikostí tabulek uživatelů a skupin, což bylo díky vztahům `many to many` možné.

Testů bylo provedeno několik. První se týkal nastavení velikosti sloupců databáze, tedy například délek typů `varchar` nebo `integer`. Při chybném nastavení by se do databázového uložiště mohly dostat neznámé znaky a nekompletní vstupy. Data musí být v konzistentním stavu, tedy jakýkoliv formát na vstupu musí být uložen v jednotné konzistentní formě. Testovaly se typy datum, desetinná čísla, malá a velká písmena.

Při testování bylo využito dvou přístupů vkládání dat do databázového uložiště. Pozitivní testování má za cíl odhalit zmíněnou nekonzistenci po vložení validních dat. Negativní testování vkládá data nevalidní a je kontrolováno, zda se uloží či jinak nepoškodí stav databáze.

Výsledek testování databáze

Při základním testování databáze byl zjištěn stav, kdy po odstranění člena ze skupiny stále existuje aktivní záznam průchodu produktem, který lze zobrazit. Objevená chyba se promítne pouze do profilu dotyčného uživatele, ale i tak je nepřípustná. Uživatel může stále zobrazit obsah produktu připojeného ke skupině, které již není člen. Záznam průchodu produktem zůstává aktivní proto, že se může uživatel do skupiny opět přihlásit a v odebírání obsahu pokračovat. Řešením je kontrolování stavu přihlášení studenta do skupiny před každým zobrazením obsahu, který přihlášení vyžaduje.

Závěr

Primárním cílem této práce bylo vytvořit modul aplikace `Mentica`, který umožní a zjednoduší uživatelský přístup ke tvorbě a odebírání obsahu. V následujícím soupisu jsou uvedeny všechny funkční požadavky zadavatele na systém a popis jejich zpracování v rámci této diplomové práce.

Uživatel bude v systému vystupovat jako mentor i jako student.
Tento požadavek byl splněn v plném rozsahu. Není požadavkem na modul, který byl vyvinut v rámci této diplomové práce, ale na celou aplikaci. Uživatel, který vytvoří nějaký obsah (Mentor), může stejně tak jiný obsah odebírat a stát se tak studentem.

Systém bude s uživatelem komunikovat pomocí emailu.
Systém zasílá uživateli email v případě registrace, kdy je nutné potvrdit existenci emailové adresy kliknutím na zaslaný odkaz. Dále v případě obnovy hesla, zadání neregistrované emailové adresy do formuláře pro obnovu hesla.

Uživatel bude mít právo upravovat svůj účet. Součástí základních informací o uživateli bude celé jméno, získaný titul, pohlaví, město a heslo.
Uživatel má možnost editovat informace, které zadal při registraci. Lze také měnit heslo, používané pro přihlašování, nebo v případě účtů registrovaných pomocí sociální služby Facebook heslo vytvořit. Součástí editace jsou i kontakty a vizitky uživatele.

Uživatel (mentor) může v aplikaci zobrazovat své různé kontaktní údaje.
Uživatel může vytvářet kontakty několika typů, které lze spojovat a vytvářet tak vizitky. Při vytvoření kurzu lze jednu z vizitek vybrat a prezentovat tak jen některé kontakty. Tímto postupem je zajištěna možnost znovu použití vizitek a odpadá tak nutnost zadávat stejné kontaktní údaje na více místech.

Uživatel může místo klasické registrace využít přihlášení přes Facebook.
Integrace sociální sítě `Facebook` byla podrobně analyzována, navržena a implementována použitím `Facebook JS SDK`. Modul umožňuje uživateli přihlášení, respektive registraci pomocí `Facebooku`, dále přidávání komentářů na stránkách detailu produktu a sdílení některých stránek pomocí tlačítka "To se mi líbí".

Mentor může seskupovat další uživatele a tvořit tak třídy studentů.
Tento požadavek byl zadán až v průběhu prvotní analýzy a bylo tak nutné reagovat na nová data. Modul poskytuje veškeré funkce, které byly specifikovány v kapitole s případy užití. Lze vytvářet soukromé a veřejné skupiny a seskupovat uživatele, kteří podle typu skupiny musí přijmout pozvánky.

Mentor může ve třídě vyučovat své kurzy s různou cenou.
Mentor může k vytvořeným skupinám přiřazovat produkty s různými cenovými zvýhodněními. Lze tak docílit případů, kdy je skupinou nějaká reálná školní třída, která nesmí obsah odebírat za finanční poplatky. U nich se nastaví stoprocentní výhoda oproti ceně produktu. Mentor může v jiných skupinách svoje práce poskytovat i za finanční obnosy.

Student může vyhledat třídu, spravovat své členství a pozvánky ke kurzům. Mentor může tuto funkci nastavit.
Skupina obsahuje atribut identifikátoru pro vyhledávání, který nastaví mentor při jejím vytváření. Student poté může pomocí filtračních pravidel, která jsou popsána například v kapitole návrhu uživatelského rozhraní, vyhledat požadovanou skupinu.

Uživatel může zobrazit profil jiného uživatele. Zobrazit cizí profil je v aplikaci možné. Zobrazení cizího profilu se od vlastního liší výpisem informativních hlášek a položek kontextové nabídky menu. V této verzi aplikace lze profil zobrazit například po kliknutí na odkaz "profil"po vyhledávání potenciálních členů skupin.

Uživatel může v systému provádět platby. Tento požadavek nebyl v rámci uživatelského modulu implementován, ani analyzován.

Student bude mít k dispozici statistiku procházení obsahu jako úspěšnosti při testech a stavy dokončení kurzů.
Implementace modulu statistik by vyžadovala existenci většího objemu reálných dat z ostrého provozu. Data by bylo nutné zanalyzovat. V této iteraci se vytvořil takový databázový systém, který zaznamená veškerá důležitá data pro pozdější zpracování a navržení modulu statistik, který tedy není součástí této verze.

V analytické části bylo provedeno zkoumání některých existujících řešení e-learningových aplikací. Byly porovnány PHP frameworky Nette a Symfon 2 a způsoby integrace sítě Facebook a její funkcionality. Byl proveden průzkum právních záležitostí ohledně nakládání s osobními údaji uživatelů, způsoby `hashování` hesel a postup jeho obnovy.

Na základě analýzy byl vytvořen návrh modulu, včetně `wireframů` uživatelského rozhraní, podrobného modelu databáze a rozpisu poskytovaných služeb mezi všemi moduly aplikace `Mentica`. Součástí návrhu je i popis použitých technologií a software. V kapitole implementace jsou popsány nejzajímavější a stěžejní oblasti vývoje.

Po implementaci prošel modul testováním databáze, které odhalilo pouze menší nekonzistenci, jenž byla opravena, a byl navržen podrobný postup pro testování s uživateli.

Budoucí vývoj

Požadavky naimplementované v rámci této diplomové práce tvoří základ, na kterém se v budoucím vývoji a přidávání funkcí bude stavět. Jednou z možností dalšího vývoje je zavedení modulu statistik a `gamifikace`, která při správném nasazení určitým způsobem obohatí aplikaci. Nutností v dalším vývoji bude zavedení platební brány a systému kreditů, které se stanou virtuálním platidlem za odebíratelný obsah. Další možností rozšíření bude integrace dalších sociálních sítí. S nárůstem uživatelské základny se otevře možnost zavedení emailových odběrů o novinkách a celkovém dění v systému. Už jen z tohoto bodu je jasné, že se nebude jednat pouze o softwarové novinky. Veřejný `hosting` přestane být dostačující a aplikaci bude nutné nasadit na virtuální privátní server nebo samostatný `hardware`.

Literatura

[1] Michael, A.: *Řízení lidských zdrojů: Nejnovější trendy a postupy - 10. vydání.* 2007, ISBN 9788024786322. Dostupné z: `https://books.google.cz/books?id=c5CMAgAAQBAJ`

[2] Ltd, O. U. A. P.: Open2Study, FREE online study for everyone! Dostupné z: `https://www.open2study.com/`

[3] Inc., C.: Take the world's best courses, online, for free. Dostupné z: `https://www.coursera.org/`

[4] Code School LLC, P.: Code Chool, learn by doing! Dostupné z: `https://www.codeschool.com/`

[5] Martin Dougiamas, M. t.: Moodle is used for e-learning projects in universities, schools, corporate training and other sectors. Dostupné z: `https://www.moodle.org/`

[6] David Grudl, N. F.: PHP framework. Dostupné z: `http://nette.org/cs/`

[7] Fabien Potencier, S.: PHP framework. Dostupné z: `http://symfony.com/`

[8] coorporation, F.: Facebook for developers. Dostupné z: `https://developers.facebook.com/`

[9] coorporation, F.: Facebook jQuery integration for developers. Dostupné z: `https://developers.facebook.com/docs/javascript/howto/jquery`

[10] Hunt, T.: Everything you ever wanted to know about building a secure password reset feature. Dostupné z: `http://www.troyhunt.com/2012/05/everything-you-ever-wanted-to-know.html`

[11] České republiky, P.: Zákon o ochraně osobních údajů a o změně některých zákonů, Předpis č. 101/2000 Sb. Dostupné z: `http://www.zakonyprolidi.cz/cs/2000-101`

[12] Pavlát, P. D.: Úřad pro ochranu osobních údajů, zveřejňování osobnívh údajů na internetu. Dostupné z: https://www.uoou.cz/ke-zverejnovani-osobnich-udaju-na-internetu/d-1592

[13] Schneier, B.: Cryptanalysis of SHA-1. Dostupné z: https://www.schneier.com/blog/archives/2005/02/cryptanalysis_o.html

[14] Vrána, J.: Ukládání hesel bezpečně. Dostupné z: http://php.vrana.cz/ukladani-hesel-bezpecne.php

[15] Wikipedie - otevřená encyklopedie, FLOPS. Dostupné z: http://cs.wikipedia.org/wiki/FLOPS

[16] Nils Adermann, J. B.: Composer, Dependency Manager for PHP. Dostupné z: https://getcomposer.org/

[17] Nils Adermann, J. B.: Packagist, The PHP package archivist. Dostupné z: https://packagist.org/

[18] Mark Otto, J. F.: Bootstrap - HTML, CSS, and JS framework for developing responsive, mobile first projects on the web. Dostupné z: http://getbootstrap.com/

[19] Hérault, A.: Gaufrette, PHP5 library that provides a filesystem abstraction layer. Dostupné z: https://github.com/KnpLabs/Gaufrette

[20] Inc., S. S.: UML Modeling and Lifecycle Tool Suite. Dostupné z: http://www.sparxsystems.com.au/

[21] Balsamiq Studios, L.: Balsamiq Mockups. Excruciatingly simple. Filled with hidden powers. Dostupné z: https://balsamiq.com/products/mockups/

[22] Git; Conservancy, S. F.: Git –distributed-is-the-new-centralized. Dostupné z: http://git-scm.com/

[23] org., A.: Bitbucket, Git and Mercurial code management for teams. Dostupné z: https://bitbucket.org/

[24] non-profit project Apache Friends: XAMPP is the most popular PHP development environment. Dostupné z: https://www.apachefriends.org/index.html

[25] Mlejnek, I. J.: Přednáška BI-SI1, LS 2015, Analýza problémové domény, ČVUT, FIT. Dostupné z: https://edux.fit.cvut.cz/courses/BI-SI1/_media/lectures/04/04.prednaska.pdf

[26] Relační databáze, anglická Wikipedia. Dostupné z: http://en.wikipedia.org/wiki/Relational_model

[27] Potencier, F.: The flexible, fast, and secure template engine for PHP. Dostupné z: http://twig.sensiolabs.org/

[28] A. Goldsteinová, L. L. a. E. W.: *Kniha HTML5 a CSS3 pro webové designéry*. 2011, ISBN 978-80-7413-166-0.

[29] Leenheer, N.: HTML 5 Test - how well does your browser support HTML5. Dostupné z: https://html5test.com/

[30] Varga, L. R.: Investigating the Interface, Digital Library and Archives, Virginia Libraries. Dostupné z: http://scholar.lib.vt.edu/ejournals/VALib/v45_n1/newins.html

[31] Jména a příjmení v České republice, kompletní generátor. Dostupné z: http://www.jmenaprijmeni.cz/generator-jmen-online/

Příloha **A**

Seznam použitých zkratek

AJAX Asynchronous JavaScript and XML

BSD Berkeley Software Distribution

CSS Cascading Style Sheet

CSV Comma separated values

ČVUT České vysoké učení technické

ČZU Česká zemědělská univerzita v Praze

DES Data Encryption Standard

DOM Document Object Model

DQL Doctrine Query Language

GUI Graphical user interface

GNU GNU's not Unix

GPL General Public License

HTML HyperText Markup Language

Inc. Incorporation (business)

JS Java Script

JSON JavaScript Object Notation

MVC Model view controller

ORM Object Relational Mapping

PDO PHP Data Objects

PHP Hypertext Preprocessor

SAX Simple API for XML

SDK Software development kit

SHA Secure hash algorithm

SQL Structured query language

UI User Interface

UK Univerzita Karlova v Praze

W3C World Wide Web Consortium

XML Extensible markup language

XPath XML Path Language

YAML YAML Ain't Markup Language

Příloha B

Přílohy

B.1 Screener pro testování s uživateli

1. Vaše věková skupina
   a) 7-15 let (20%)
   b) 15-19 let (20%)
   c) 20-26 let (25%)
   d) 26-32 let (35%)
   e) jiná
2. Jste student/ka ?
   a) ano(50%)
   b) ne (50%)
3. V případě negativní odpovědi na předchozí otázku - vyučujete na nějaké škole? (90% pozitivní odpověď -a/b/c/f)
   a) na základní škole
   b) na střední škole
   c) na vysoké škole
   d) již ne
   e) ne
   f) jiné - doplňte
4. Jakým oborem se zabýváte? (bez ohledu na vztah ke škole)
   a) technickým
   b) humanitním

c) ekonomickým
d) právnickým
e) jiné - doplňte

5. Jste aktivním uživatelem internetu?
   a) ano
   b) ne (maximálně jeden participant)
6. Kolik času trávíte používáním informačních technologií? (včetně chytrých telefonů, notebooků, tabletů a podobně.)
   a) většinu, jejich užívání mne živí (menšina)
   b) několik hodin denně (většina)
   c) hodinu nebo dvě denně (menšina)
   d) nepoužívám
7. Máte zkušenost s užitím vzdělávacích portálů na internetu?
   a) ano - uveďte s jakým a způsob využití
   b) ne

B.2 Scénáře testování použitelnosti

Registrovat se pomocí registračního formuláře

Tabulka B.1

Předpoklad: Není

Registrovat se pomocí sociální služby Facebook

Tabulka B.2

Předpoklad: participant vlastní profil na `Facebooku`

Vytvořit zadané kontakty

Tabulka B.3

Předpoklad: uživatel se nachází na stránce profilu

Vytvořit vizitku z vytvořených kontaktů v předchozím úkolu

Tabulka B.4

Tabulka B.1: Scénář testování použitelnosti - klasická registrace

ID kroku	**Nutná akce**	**Očekávaný následek akce**
1	Kliknout na odkaz "Registrovat se".	Přesměrování na registrační formulář
2	Vyplnění reálných údajů do formuláře.	Informace o zaslání potvrzovacího emailu nebo výzva k opravení zadaných údajů
3	Přečtení emailu a aktivace účtu kliknutím na zaslaný odkaz	Informace o úspěšné registraci na portálu Mentica
Poznámka: Uživateli bude pro potřeby testování založena dočasná emailová schránka. Přihlašovací údaje budou poskytnuty.		

Tabulka B.2: Scénář testování použitelnosti - registrace přes Facebook

ID kroku	**Nutná akce**	**Očekávaný následek akce**
1	Kliknout na odkaz "Přihlásit se přes Facebook".	Zobrazí se vyskakovací okno přihlašovací brány `Facebooku`
2	Vyplnění reálných údajů do formuláře.	Přesměrování uživatele na právě vytvořený profil a zobrazení informace o možnosti vytvoření hesla pro klasické přihlašování do systému
Poznámka: Ihned po testování bude účastníkovi poskytnuta možnost změny hesla na Facebooku na zařízení s mobilním internetem (v místnosti bez kamerových zařízení). O této skutečnosti bude informovaný před testováním.		

Tabulka B.3: Scénář testování použitelnosti - vytvoření kontaktu

ID kroku	Nutná akce	Očekávaný následek akce
1	Kliknout na odkaz "Editace profilu".	Přesměrování na stránku s formulářem na editaci údajů zadaných při registraci a odkazy pro správu skupin a správu kontaktů a vizitek
2	Kliknout na odkaz "Kontakty, vizitky"	Přesměrování na stránku s výpisem všech kontaktů a vizitek
3	Kliknout na odkaz "přidat kontakt"	Zobrazení jednoduchého formuláře pro vytvoření kontaktu (typ, obsah kontaktu)
4	Kliknout na ikonu telefonu, vyplnit číslo a odeslat formulář	První nevalidní vstupy budou mít za následek chybové hlášení, po odeslání validních vstupů bude zobrazena informativní hláška o úspěchu - kontakt se zobrazí v seznamu existujících kontaktů v levé horní části obrazovky
5	Stejné, jako krok ID 3	
6	Kliknout na ikonu URL, vyplnit adresu a odeslat formulář	Po odeslání validních vstupů bude zobrazena informativní hláška o úspěchu - kontakt adresy URL se zobrazí v seznamu existujících kontaktů v levé horní části obrazovky

Poznámka: Účastník bude mít za úkol vytvořit kontakt typu telefonní číslo (bude zadáno nejprve nevalidní - 10 znaků, následně validní) a kontakt typu vlastní URL adresa. (obsah v režii participanta)

Tabulka B.4: Scénář testování použitelnosti - vytvoření vizitky z vytvořených kontaktů

ID kroku	Nutná akce	Očekávaný následek akce
1	Kliknout na odkaz "Editace profilu".	Přesměrování na stránku s formulářem na editaci údajů zadaných při registraci a odkazy pro správu skupin a správu kontaktů a vizitek
2	Kliknout na odkaz "Kontakty, vizitky"	Přesměrování na stránku s výpisem všech kontaktů a vizitek
3	Kliknout na odkaz "vytvořit vizitku"	Zobrazení jednoduchého formuláře pro vytvoření vizitky (pole `dropzone` pro umístění kontaktu a pole pro název vizitky)
4	Přetáhnout kontakty z levého seznamu existujících kontaktů do `dropzone` pomocí funkce `drag and drop`, vložit název a odeslat formulář tlačítkem "Uložit"	Po odeslání validních vstupů bude zobrazena informativní hláška o úspěchu - vizitka se zobrazí v seznamu existujících vizitek v dolní části obrazovky

Poznámka: Pro potřeby testování bude vytvořen účet s několika vytvořenými kontakty, který se použije v případě, že participant bude mít s předchozím úkolem problém. Výsledek tohoto úkolu bude také použit v úkolu testování modulu kurzů - volba vizitky produktu.

Předpoklady: uživatel splnil předchozí úkol a vytvořil kontakt s telefonním číslem a kontakt s URL adresou, uživatel se nachází na stránce profilu

Zvolit jednu z vizitek jako výchozí

Tabulka B.5

Předpoklady: participant má více vytvořených vizitek, nachází se na stránce správy kontaktů a vizitek

Tabulka B.5: Scénář testování použitelnosti - zvolení vizitky jako výchozí

ID kroku	Nutná akce	Očekávaný následek akce
1	Zvolit jednu z vizitek jako výchozí kliknutím na příslušnou ikonu srdce u vizitky.	Bez opětovného načtení stránky bude zobrazena informativní hláška o úspěchu
Poznámka: Následkem akce bude kromě informativní hlášky i dynamické zvýraznění nové výchozí vizitky.		

Tabulka B.6: Scénář testování použitelnosti - vytvoření skupiny

ID kroku	Nutná akce	Očekávaný následek akce
1	Kliknout na odkaz "spravovat skupiny".	Přesměrování na list vytvořených skupin s odkazy na vytvořené soukromé / veřejné skupiny, na seznam, ve kterých je uživatel člen.
2	Kliknout na odkaz "vytvořit skupinu".	Přesměrování na formulář pro vytvoření skupiny. (název, vyhledávací identifikátor, popis, typ - veřejná/soukromá s tooltipy)
3	Vyplnit údaje a odeslat formulář kliknutím na "Vytvořit".	Přesměrování na detail právě vytvořené skupiny - zobrazení informativního hlášení v horní části stránky.
Poznámka: Data pro zadání do formuláře budou poskytnuta.		

Vytvoření skupiny

Tabulka B.6

Předpoklady: účastník testu má zobrazený svůj profil

Tabulka B.7: Scénář testování použitelnosti - přidání členů k vytvořené skupině

ID kroku	Nutná akce	Očekávaný následek akce
1	Kliknout na odkaz "Správa členů".	Přesměrování na seznam všech členů i nečlenů skupiny s filtrováním
2	Zadat filtrační pravidla podle poskytnuté tabulky uživatelů, které je třeba přidat do skupiny.	Zobrazení omezeného seznamu uživatelů. Předpokládané použité pole pro filtraci je email.
3	Zaškrtnout požadované uživatele pomocí prvku `checkbox` a potvrdit volbu kliknutím na odkaz "Přidat ke skupině".	Zobrazení informativní hlášky o úspěchu a přesměrování na detail produktu, kde se zobrazí nově seznam uživatelů, kterým byla poslána informace o přidání do skupiny.

Poznámka: Seznam členů, které je třeba přidat ke skupině bude poskytnut.

Přidání členů k vytvořené skupině

Tabulka B.7

Předpoklady: účastník testu prošel předchozím úkolem testování, je zobrazen detail vytvořené skupiny

Připojení vytvořených produktů k vytvořené skupině

Tabulka B.8

Předpoklady: participant prošel testem modulu kurzů, ve kterém si vytvořil produkt (kurz / test)

B.3 Post-test dotazník pro testování s uživateli

1. Jaký máte dojem z testování webové aplikace Mentica?
2. Jaké stránky pro vás byly při procházení systému nejdůležitější?

Tabulka B.8: Scénář testování použitelnosti - připojení vytvořených produktů k vytvořené skupině

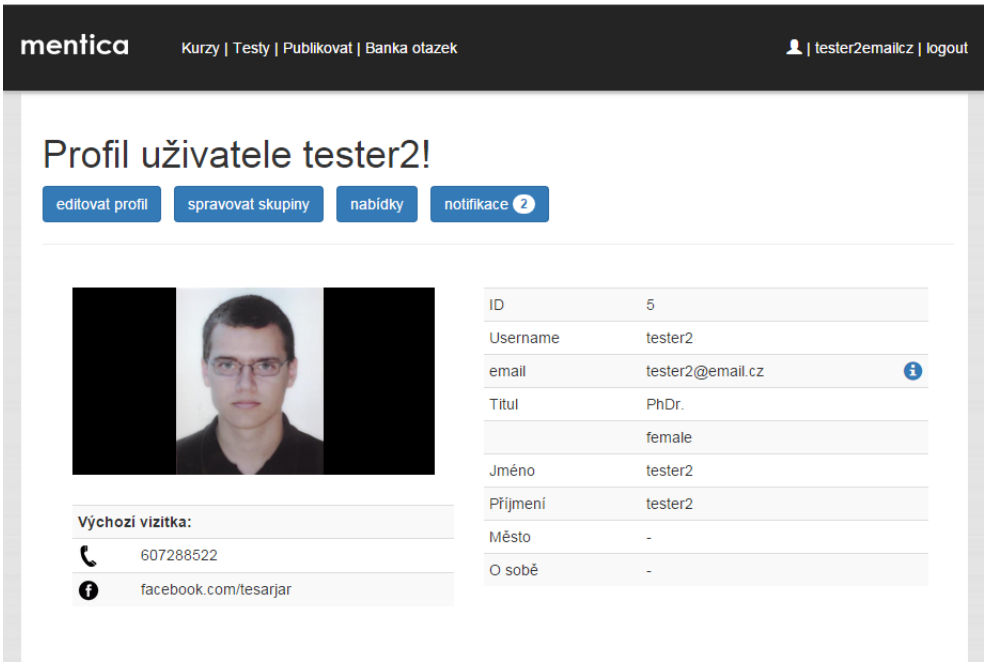
ID kroku	**Nutná akce**	**Očekávaný následek akce**
1	Kliknout na odkaz "Správa produktů".	Přesměrování na seznam všech produktů, které lze ke skupině připojit.
2	Kliknout na odkaz "připojit ke skupině"u požadovaného produktu.	Zobrazení formuláře pro vložení procentuálního zvýhodnění oproti původní ceně.
3	Vyplnit číslo procenta a potvrdit kliknutím na "Připojit".	Zobrazení detailu skupiny, ve kterém se nově zobrazí seznam s připojenými produkty. Uživatel je informován o rozeslaných pozvánkách a úspěchu připojení produktu.

Poznámka: Participant bude mít vytvořené nějaké produkty po úkolu z testování modulu kurzů. V tomto testu bude vyžadováno připojení dvou libovolných produktů.

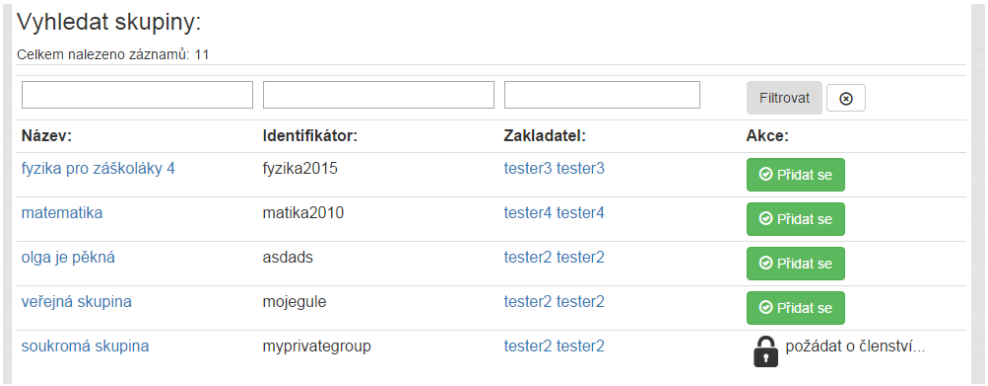
3. Měl/a jste někdy problém s pochopením dané stránky? Jaké a proč?
4. Měl/a jste problém s nalezením nějakého prvku, respektiva očekával/a jste nějaký prvek na jiném místě?
5. Co byste na stránkách e-learningového portálu Mentica zlepšil/a?
6. Využil/a byste aplikaci Mentica pro sebevzdělání v nějakém oboru, který vás zajímá za nějaký poplatek? Jaký a proč?

Příloha C

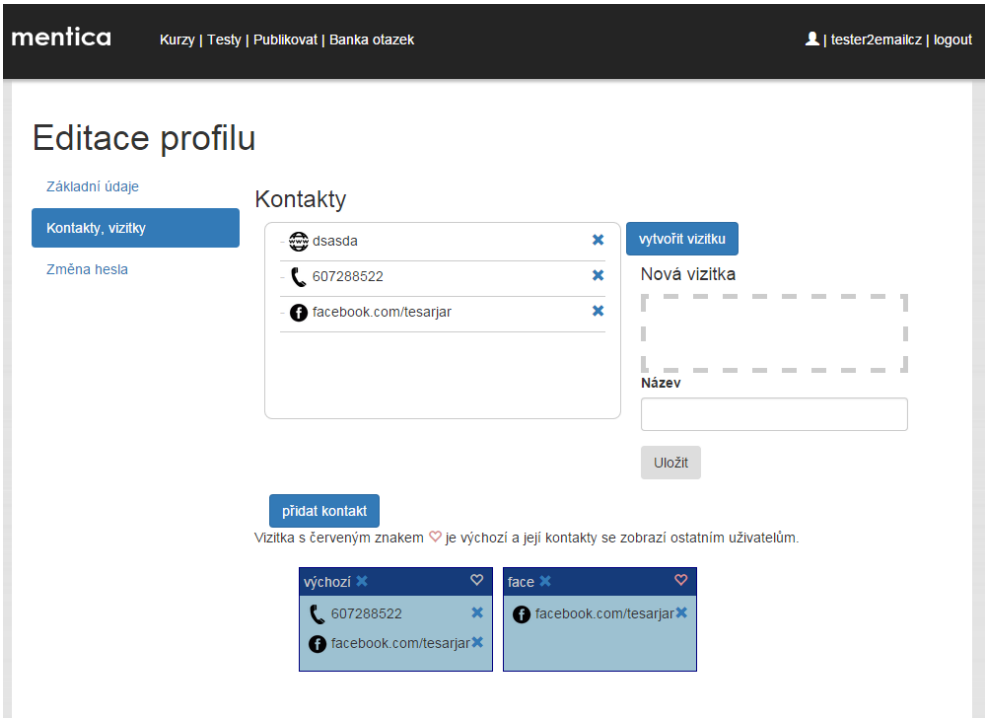
Screenshoty uživatelského modulu aplikace Mentica



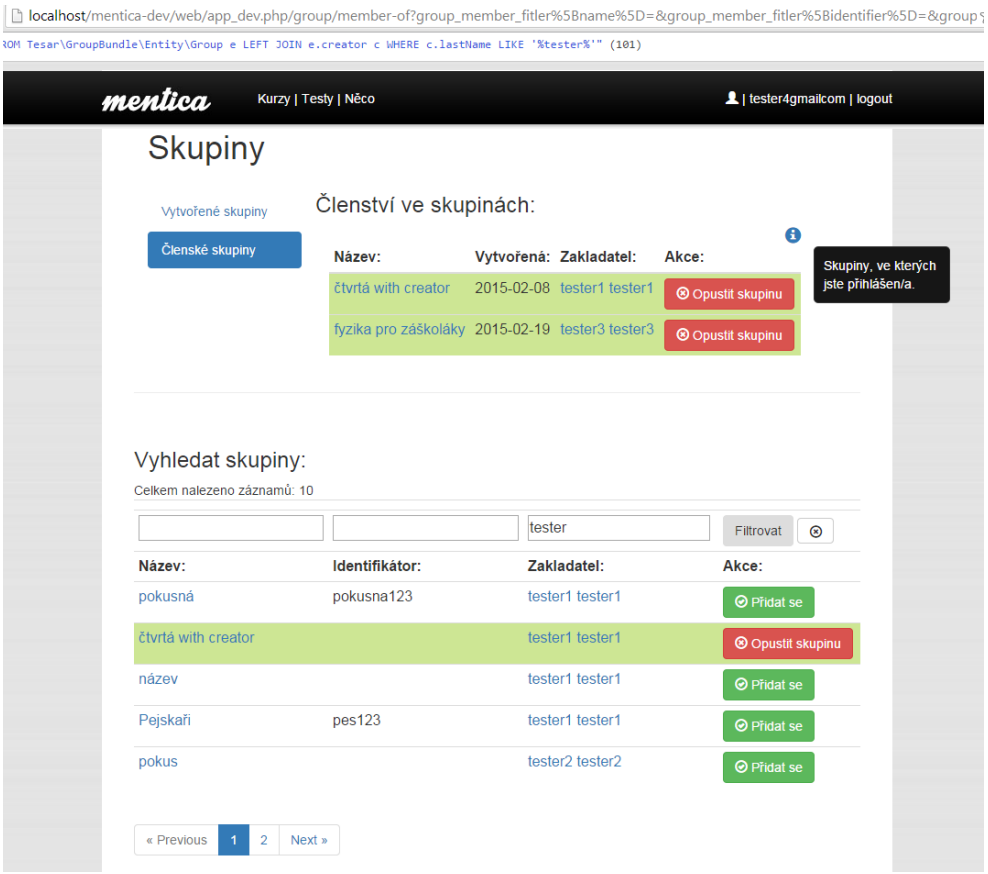
Obrázek C.1: Mentica screenshot - uživatelský profil



Obrázek C.2: Mentica screenshot - ukázka zobrazení soukromé skupiny po vyhledávání



Obrázek C.3: Mentica screenshot - správa kontaktů



Obrázek C.4: Mentica screenshot - vyhledávání veřejných skupin pomocí filtrování, zobrazení databázového dotazu a generované URL adresy ve vývojovém prostředí

Příloha D

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