UNIVERZITA PARDUBICE

Fakulta elektrotechniky a informatiky

Aplikace pro podporu týmové spolupráce uvnitř softwarového týmu

Pavel Pich

Bakalářská práce

2013

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2012/2013**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Pavel Pich**
Osobní číslo:	**I10174**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Aplikace pro podporu týmové spoluprácc uvnitř softwarového týmu**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování:

Teoretická část se bude věnovat analýze existujících komerčních i open-source aplikací, jež slouží pro podporu týmové spolupráce, problematice přístupových práv a zabezpečení v rámci webových aplikací.
Praktická část se bude věnovat návrhu a implementaci dané webové aplikace. Pro aplikaci budou klíčové především následující vlastnosti:
a) Možnost evidovat řadu separátních projektů.
b) Definice uživatelů s různými úrovněmi oprávnění.
c) Definice jednotlivých úkolů, které mohou mít odlišnou prioritu, datum splnění, odlišné řešitele.
d) Možnost tvorby přehledných reportů (prezentace splněných úkolů, generování roadmapy projektu, Ganttovy diagramy). Generování tiskových sestav.
e) Možnost nahrání a stažení souborů vztahujících se k daným projektům.
f) Implementace modulu pro evidenci a správu projektové dokumentace a správu zdrojových kódů.
Použité technologie: HTML, JavaScript, PHP, MySQL nebo Oracle XE.
Odborný konzultant: Ing. Tomáš Váňa

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1) Nowicki, S., Lecky-Thomson, E.: PHP 6 Programujeme profesionálně. Comuputer Press, 2010. ISBN 978-80-251-3127-5.
2) Castro, E.: HTML, XHMTL a CSS. Názorný průvodce tvorbou WWW stránek. Computer Press, 2007. ISBN 978-80-251-1531-2.
3) Lacko, L.: 1001 tipů a triků pro SQL. Computer Press, 2011. ISBN 978-80-251-3010-0

Vedoucí bakalářské práce:	**RNDr. David Žák, Ph.D.** Katedra informačních technologií
Konzultant bakalářské práce:	**Ing. Tomáš Váňa** Katedra informačních technologií
Datum zadání bakalářské práce:	**21. prosince 2012**
Termín odevzdání bakalářské práce:	**10. května 2013**

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

Ing. Lukáš Čegan, Ph.D.
vedoucí katedry

V Pardubicích dne 29. března 2013

Prohlašuji:

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

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

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 16. 8. 2013

Pavel Pich

Poděkování

Mé poděkování patří Ing. Tomáši Váňovi za odborné vedení, cenné rady, trpělivost a ochotu, kterou mi v průběhu zpracování bakalářské práce věnoval.

Dále bych chtěl poděkovat své rodině, přátelům za podporu při studiu a tvorbě této práce.

Anotace

Tato práce se zabývá webovou aplikací pro podporu týmové spolupráce uvnitř softwarové firmy. Teoretická část se zabývá vybranými webovými aplikacemi, které již na trhu existují s jejich popisem. Dále popisuje útoky na webové aplikace a jejich řešení. Praktická část se zabývá návrhem vlastní aplikace, od návrhu databáze až po webovou tvorbu.

Klíčová slova

PHP, Oracle, webová aplikace, týmová spolupráce

Title

Application to support teams collaboration in software team

Annotation

This thesis deals with the web application to support team collaboration within software companies. The theoretical part deals with some web applications that already exist with their descriptions on the market. It also describes the attacks on web applications and solutions. The practical part deals with own applications, from database design to web development.

Keywords

PHP, Oracle, web appliacation, cooperation of team

Obsah

Seznam ilustrací a tabulek

Úvod

V dnešní době zaměstnanci softwarových firem pracují v týmech, ve kterých spolupracují na zadaných úkolech. O bezproblémový chod spolupráce uvnitř týmu se v dnešní době standardně starají aplikace pro jejich podporu.

Tato bakalářská práce se zabývá problematikou spolupráce uvnitř týmu při realizaci jednotlivých zakázek resp. projektů a úkolů v oblasti vývoje informačních systémů. Vycházím z toho, že většinou softwarové firmy mají více zaměstnanců. Pro vyšší efektivnost jsou úkoly řešeny z pravidla více zaměstnanci, kteří spolu spolupracují. To se neobejde bez podpůrného softwaru, který zajistí efektivní a rychlou koordinaci pracovních úkolů mezi jednotlivými členy vývojového týmu. Tyto aplikace většinou využívají společnosti, které se zabývají vývojem softwaru či jiných IT technologií (automatizace technologických procesů, vizualizace realtime technologií atd.). Jejich projekty bývají většinou velmi rozsáhlé a pro jejich velký počet je třeba zajistit jednoduchou orientaci mezi nimi. Vedoucí těchto firem musí koordinovat velký počet zaměstnanců a projektů a právě k tomuto účelu právě slouží aplikace pro podporu týmové spolupráce.

Práce je rozdělena na dvě části. V teoretické části jsou popsány vybrané konkurenční aplikace a problematika přístupových práv jednotlivých uživatelů, týmů a zabezpečení v rámci webových aplikací a databází.

V praktické části popisuji mnou navrženou strukturu databáze, vztahy mezi tabulkami či specifikací datových typů. Dále jsou zde popsány některé zajímavé ukázky ze zdrojových kódů v PHP[1], dotazy, připojení k databázi a práva pro uživatele.

Cílem této práce je navržení a implementace vlastní aplikace, která by mohla být nápomocná vývojářům uvnitř vývojového týmu při řešení pracovních úkolů. Práce dále obsahuje popis celého projektu od návrhu až po samotnou aplikaci.

Protože mám zájem o vylepšení dané aplikace, vybral jsem si právě toto téma mé bakalářské práce a v případě zájmu bych ji mohl i nabídnout firmě, ve které jsem absolvoval praxi.

[1] Hypertext Preprocessor

1. Týmová spolupráce

Slovo **tým** se nejčastěji používá pro označení skupiny lidí, nejčastěji pracovní skupinu, kolektiv, kde každý její účastník má stejný cíl.

„Týmová práce *je dobře koordinovaná a účelně synchronizovaná činnost, která je charakteristická těsným propojením aktivity skupiny. Týmová práce vyžaduje sladění norem, hodnot, stanovisek, vzájemné porozumění. Každý jedinec musí sdílet cíle skupiny a musí se cítit těmito cíli zavázán.“* [1]

Existují tři druhy společné činnosti:

- návaznost činností,
- kooperace (dělba práce, propojení, sladění činností),
- týmová práce.

Rozdíl mezi prvními dvěma body a týmovou prací je takový, že týmová práce je dobře koordinovaná a dobře synchronizuje činnost, která je propojená aktivitami celé skupiny.

1.1 Vedoucí týmu

Vznik týmu začíná určením vedoucího celé skupiny, který je zodpovědný za:

- výběr členů týmu,
- za udržování kázně a dodržování pravidel,
- má na starosti okolní styk,
- rozděluje odpovědnost.

1.2 Sestavení týmu

Tým by se měl skládat z co nejmenšího počtu lidí, kteří jsou schopni daný úkol realizovat. Velká skupina lidí nedosahuje efektivní komunikace při plnění a synchronizaci daného úkolu. Tým by měl být vyvážený. V týmu by neměli být jen návrháři, ale i zdatní analytici, kodéři, testeři.

Členové týmů by se měli doplňovat. Neměli by být stejnostejných rolí, tzn., že by neměli disponovat stejnými znalostmi a schopnostmi. Pro případ založení dalšího týmu by mohl tento člen chybět.

Členy týmu vybíráme podle tří kritérií:

- Odborná nebo profesionální zdatnost - nepotřebujete v týmu člena, který umí všechno. Týmová spolupráce spočívá v tom, že jedinci by se měli navzájem doplňovat.
- Schopnost pracovat jako člen týmu - v týmu by měli být jen ti členové, kteří nenarušují spolupráci a dokáží kolektivně komunikovat a pracovat pro tým.
- Žádoucí osobní vlastnosti - jde především o komunikaci a spolupráci mezi členy týmu. Je třeba, aby každý člen v týmu sdílel potřebné informace, byl schopen a ochoten pomoci svým kolegům.

V tomto směru vše musí probíhat automaticky a koordinovaně, aby se všichni členové v týmu mohli věnovat svým úkolům k dosažení daného cíle.

2. Analýza komerčních a open-source aplikací[2]

Aplikace pro podporu týmové spolupráce umožňuje uživatelům evidovat projekty a informace o postupu práce. Uživatel si vypíše chyby, události na projektech a komunikuje s ostatními členy týmu. Mohou si vyzkoušet jednotlivé funkce na svých projektech vytvořených na webových stránkách společnosti. Uživatel krom základních funkcí, jako tvorbu projektu, zrušení projektu, může používat pokročilejší funkce, např. přidělovat práva ostatním uživatelům, přidělovat úkoly, evidovat chyby a exportovat výpis do počítače. Jediné, co je u všech distributorů nezbytné, je registrace a poté aktivace účtu na adrese, která jim přijde na e-mail jako příloha zprávy, že byli zaregistrováni.

Historie aplikace pro týmovou spolupráci

„Z historických důvodů se často na sdílení informací a spolupráci používají content management systémy, původně určené na programování a správu rozsáhlých webových aplikací. Část z nich je volně dostupná jako open source software s aktivní komunitou vývojářů, kteří vyvíjejí rozšiřující moduly a dokáží systém přizpůsobit na zakázku. Podle dnešních měřítek je ovšem většina těchto systémů pro běžného uživatele příliš složitá a náročná na pochopení. Pokud nějaká firma nasadí takový systém za účelem týmové spolupráce, končí to obvykle tak, že informace publikuje jen hrstka technicky nejzdatnějších členů týmu a ostatní lidé informace jen konzumují. Pokud mají něco publikovat, mají strach to udělat. Buď informace nesdílejí vůbec, nebo musí požádat někoho zkušenějšího o pomoc. Přímá angažovanost a spontánní sdílení zde zcela chybí.“ [2]

Informace, které chtějí uživatelé sdílet na webové stránce, jsou různé obrázky, dokumenty, prezentace, projekty různých formátů (aplikací) apod.

Současné aplikace pro podporu týmové spolupráce

Mnoho dostupných aplikací podporuje jeden typ informace a ostatní zanedbává, nebo jejich podpora je velmi malá. Např. systém Wiki podporuje přidávání textu, ale přidání obrázku nebo Excel dokumentu je zdlouhavé a ne vždy zdárné.

Při výběru systému pro podporu týmové spolupráce bychom měli zvážit, jaké informace chceme sdílet a jak s nimi chceme pracovat.

[2] Počítačový software s otevřeným zdrojovým kódem

Dobrým aplikacím pro podporu týmové spolupráce dnes nestačí pracovat pouze na principu sdílení a stahování informací. I přesto tyto aplikace splňují podstatu práce v týmu a pro tvorbu menších projektů budou postačující.

Vylepšené dnešní aplikace obsahují webová rozhraní, kde je možno dokument, obrázek prohlédnout, aniž by se stahoval. Zdrojové soubory aplikace na webovém úložišti se automaticky synchronizují s naším lokálním diskem, tudíž i námi uložený zdrojový soubor se může automaticky ukládat na webové úložiště.

„V moderních webových prohlížečích podporujících HTML[3] 5 lze také soubory přesouvat pomocí drag and drop mezi prohlížečem a lokálním diskem. Upload a download se tak redukuje na pouhé gesto myší a jedno kliknutí.

Standardem je také historie verzí s možností prohlížet staré revize, sledovat co kdo a kdy změnil a v případě chyb se vracet zpět.“ [2]

V dnešní době mohou aplikace sdílet soubory na stránky firmy v sociálních sítích. Výhodou je sdílení pod vlastním profilem a možnost sdílení souborů více uživatelům (firma může mít více poboček a to i v zahraničí). Uživatelé mohou okamžitě reagovat nejen na projekty, ale i komentáře a „lajky“.

Výhodou této možnosti je nejen možnost sdílení, ale i prohlížení informací a komentářů k úkolům, chybám a to v době, kdy stránky sociálních síti můžeme prohlížet na telefonu.

Možnosti moderních aplikací pro podporu týmové spolupráce jsou však i jiné. Dnešní moderní aplikace mohou běžet i na chytrém telefonu. Jedním z těchto moderních objevů je tzv. BYOD[4], neboli práce na zařízení vlastněná samotným zaměstnancem.

Jedním z těchto vývojářů je firma Apple, která dodává aplikaci pro mobilní telefony na podporu týmové spolupráce. Nevýhodou této aplikace se mi však zdá nutnost vlastnit mobilní telefon či tablet od firmy Apple. Výrobky a technologie jsou od této firmy nákladné a její výrobní politika by nás, vlastníky jiných telefonů, v tomto případě vyloučila z používání aplikace. Pokud by však firma trvala na používání této aplikace, ne každý člověk disponuje

[3] HyperText Markup Language
[4] Bring Your Own Device

dobrou technickou zdatností a obratností na chytrých telefonech, aby mohl okamžitě okomentovat případnou chybu kolegy v této aplikaci.

Závěrem bych chtěl podotknout, že takto chytré a vypracované aplikace, které mohou být propojeny s jakýmkoliv zařízením, jsou pro odborníky zajisté velmi pohodlné s velkou škálou možností. Jsou ale nákladné a každá firma, která by chtěla takto propracovanou aplikaci, bude muset vynaložit nemalé finance. Dle mého názoru jsou tyto aplikace přehnaně složité a z aplikace pro podporu týmové spolupráce se stává multifunkční aplikace, jako je např. obchodní aplikace SAP[5] .

2.1 Nástroje pro týmovou spolupráci

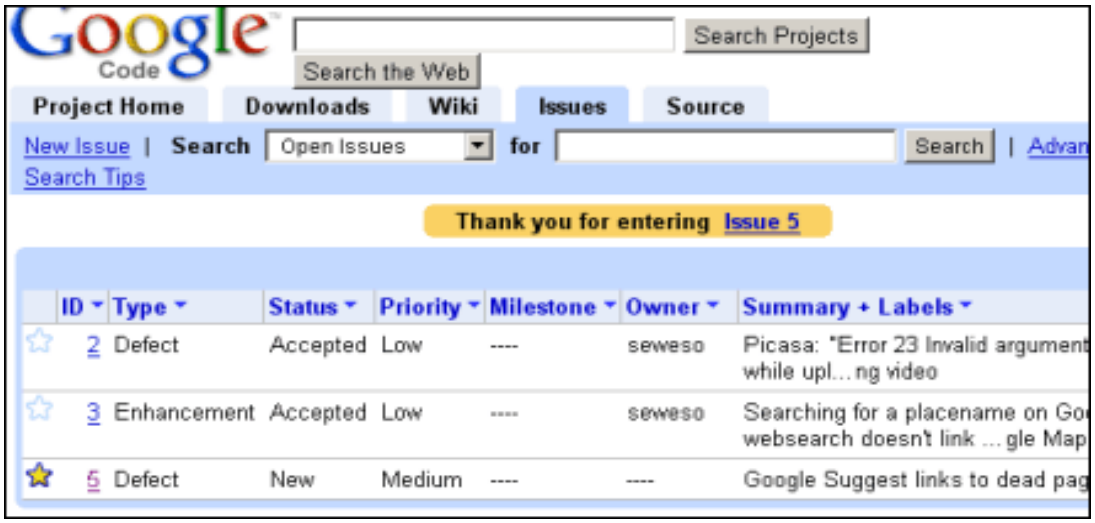
Tyto SW[6] nástroje slouží zaměstnancům pro spolupráci na projektech v reálném čase a omezuje domluvu přes e-maily, telefony. Navíc umožňují zapojit do projektu více zaměstnanců a to i těch externích.

2.1.1 Open source aplikace

MindTouch[7] – „*jedná se o open-source řešení pro spolupráci, které se osvědčí zejména v případě, kdy je potřeba efektivně propojit spolupracovníky pro práci na určitém úkolu a kdy jde primárně o vytvoření statického archivu znalostí ve stylu Wiki. Zúčastnění pak mají nejen přístup k potřebným dokumentům, ale mohou je i editovat a k dispozici je také poměrně propracovaná schopnost vyhledávání, ticketingu*[8] *či možnost zpětné vazby. Architektura řešení MindTouch podporuje rychlý vývoj aplikací bez nutnosti využívat služeb externích vývojářů jako je tomu třeba v případě SharePointu*.“ [4]

Google Code[9] – velký představitel open-source projektů, který je již na trhu 8 let. V této aplikaci se shromažďují různé knihovny a nástroje. Vývojáři webu si mohou z jakéhokoliv místa stáhnout tyto nástroje a připojit do své aplikace.

[5] Systeme, Anwendungen, Produkte in der Datenverarbeitung
[6] Software
[7] http://www.mindtouch.com/
[8] Funguje na principu vydání časově omezeného dokladu
[9] http://code.google.com/intl/cs/



Obrázek 1: Výpis projektů[10]

Důvodem vývoje této aplikace bylo umožnit vývojářům stahovat a používat nástroje a knihovny Googlu. Dnes je zde více než 60 knihoven a API[11] rozhraní. Současný obsah se neskládá jen z open source projektů Googlu, ale také i z obsahu třetích stran.

Ze všech možných knihoven a API rozhraní zde najdeme například napojení na různé javascriptové frameworky (jQuery, jQuery UI atd.), API na Google Search, Google Mapy, Google Earth, nástroje pro vývoj na Androidu a řadu dalších a dalších nástrojů a prostředků….

„Google se u svých projektů drží hesla „v jednoduchosti je síla“, a tak i Google Code je poměrně jednoduchý. Diskový prostor vyhrazený pro projekt je 4GB.“ [3]

Nevýhoda této aplikace spočívá v tom, že Google Code nemá žádné veřejné aplikační rozhraní.

Výhodou je, že vývojáři zde mohou zdarma stahovat, ukládat a případně společně pracovat na zdokonalení existujících projektů.

Assembla[12] – je malá open source aplikace, která poskytuje diskový prostor 2GB[13] pro projekty a neomezený počet repositářů. Tato aplikace však nabízí dvě varianty a to free a professional.

[10] Dostupné z: http://blogoscoped.com/files/google-code-bugtracker.png
[11] Application Programming Interface

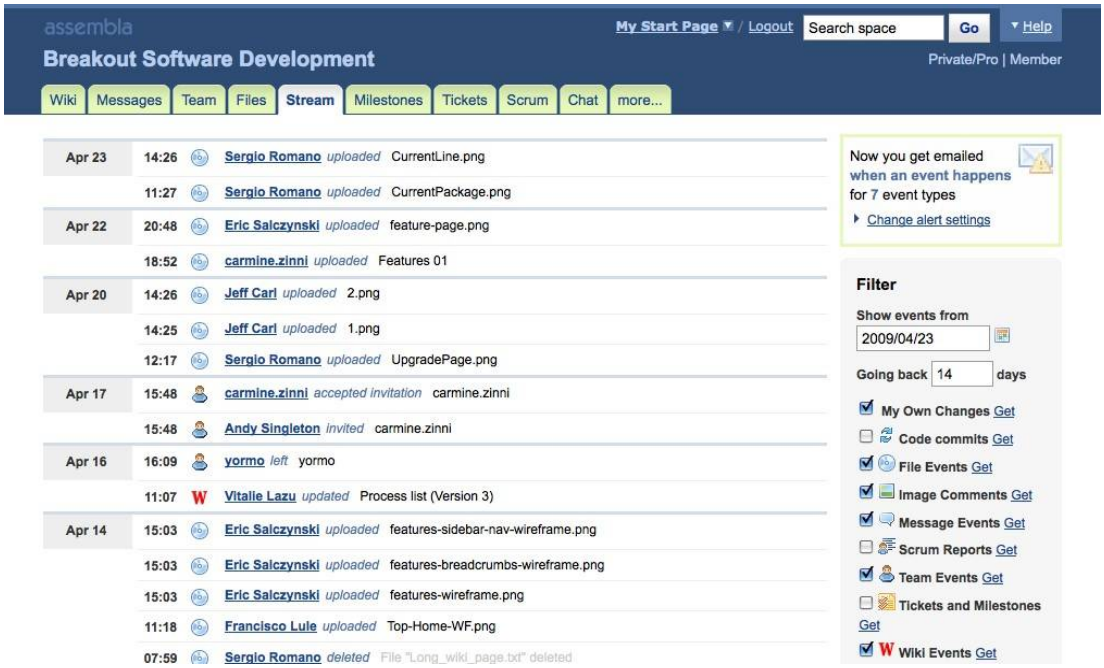
Nevýhodou varianty free je, že projekty (kódy) jsou veřejně přístupné a každou stránku „pronásledují“ nějaké reklamy. Tuto variantu odstraňuje webová aplikace Unfuddle, která však nabízí 200MB pro projekt. Nad některými prioritami musí popřemýšlet právě uživatel.

Celkově Assembla má velice hezký a profesionální design a nabízí velmi zajímavé funkce, jako jsou např.

- nástroje pro sledování času,
- možnost spuštění FTP[14] serveru (to ocení především vývojáři webů),
- nástroj pro zasílání SCRUM reportů a také podpora různých,

komunikačních nástrojů – od zpráv přes chat a Twitter až po Skype.

Na rozdíl od Google Code nabízí veřejné aplikační rozhraní, které je však placené.



Obrázek 2: Výpis událostí v Assemble[15]

[12] https://www.assembla.com/home
[13] Gigabyte
[14] File Transfer Protocol
[15] Dostupné z: http://img1.findthebest.com/sites/default/files/688/media/images/Assembla_1_159908_i0.jpg

BitBucket[16] – *„Tato webová aplikace má jednu velkou výhodu – je zdarma pro týmy, které mají 5 a méně členů a také pro open source projekty, které nejsou omezené počtem členů, ale každý si může tyto projekty prohlížet včetně zdrojových kódů.“* [3]

Velkou nevýhodou této aplikace je nabídka malého množství funkcí, např. nelze nastavit datum nebo sledovat, zda je projekt ve skluzu.

Umožňuje však posílání pozvánek do projektů, což se mi zdá zajímavé. Nejste k projektu přidělen, ale pozván a záleží na Vás, zda tuto nabídku přijmete.

CodePlex[17] **–** je open source aplikace založená v roce 2009 firmou Microsoft, která tímto nabízí bezplatné sdílení projektů. Uživatelé mohou používat nástroje pro sledování požadavků, chyb, podporu RSS[18], statistiky, diskuzní fóra, vlastní Wiki atd. Většina projektů nahraná do této webové aplikace se týká .NET Frameworku, včetně ASP. NET a Microsoft SharePointu, a jsou zde projekty zabývající se SQL, WPF a Windows Forms atd.

Box.net[19] – *„jde spíše o cloudovou úschovnu pro soubory uživatelů, nabízí 5 GB prostoru zdarma pro osobní využití a až neomezeně v podnikové variantě. Jejím úkolem má být možnost zjednodušit práci se soubory, nahradit tradiční FTP přístup a propojit týmy v rámci virtuálního pracovního prostoru.“* [4]

Tato aplikace pro podporu spolupráce pracuje na zastaralé možnosti sdílení dat (FTP). Takovýchto aplikací pro sdílení dat je dnes velké množství (SkyDrive, DropBox atd.) a nejsou určeny k tomuto účelu.

SourceForge.net[20] **–** patří k největším webovým aplikacím sdílející projekty (diskuzní fóra). V této aplikaci si může uživatel svůj projekt zkompilovat, spustit a testovat. Pokud do tohoto projektu vložíte jakoukoliv novou myšlenku, objeví se zpráva o aktivitě na úvodní stránce u všech spolupracujících uživatelů. Čím více je s tímto projektem manipulováno, tím více se posunuje v žebříčku událostí a pokud se stane tzv. projektem měsíce, je projektu vytvořena vlastní vizitka.

[16] https://bitbucket.org/
[17] http://www.codeplex.com/
[18] Formát pro čtení novinek na webových stránkách
[19] https://app.box.com/files
[20] http://sourceforge.net/

U projektu můžete určovat co je to za verzi, stav, programovací jazyk, ve kterém je vyvíjen. Dále můžete určovat, co projektu chybí.

Nevýhodou této aplikace je dle mého názoru opět všude přítomná reklama, která však částečně pokrývá náklady na údržbu serverů apod.

Výhoda je v pohodlném hledání jak podle klíčového slova, tak podle kategorie. Každý projekt je uložen v určité kategorii např. hry, business, media.

Pokud máme projekt velice zajímavý a stahovaný, můžeme si vybrat server, který je v jiném časovém pásmu tzn., že uživatelé v jiném časovém pásmu nebudou tento server v noci tolik zatěžovat.

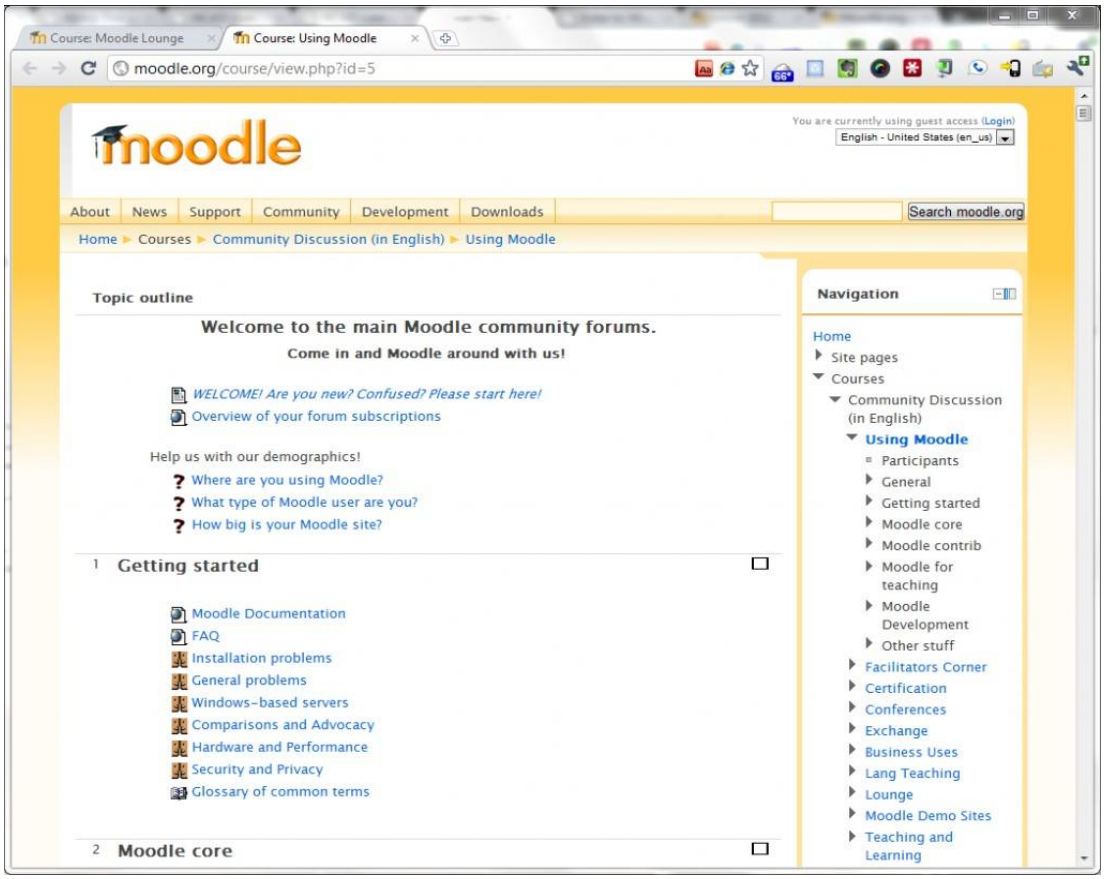
Moodle[21] – *„je open source software určený pro podporu prezenční i distanční výuky prostřednictvím online kurzů dostupných na Internetu. Stal se oblíbeným nástrojem učitelů i studentů z celého světa. Moodle je třeba nainstalovat na webový server, buď na náš vlastní, nebo u našeho poskytovatele webových služeb.“* [5]

Z vlastní zkušenosti si můžeme na Moodlu vytvořit jakýkoliv projekt (kurz) a zde můžeme přidávat texty, obrázky, ale i tvořit testy a hodnocení. Tento projekt (kurz) pak můžeme povolit jednotlivci, skupině anebo nechat veřejný.

Cílem této webové aplikace je však poskytnout platformu pro vzdělávání. Není tudíž prioritně vytvořen pro podporu týmové spolupráce.

Výhodou této služby je překlad většiny důležitých textů, názvů do češtiny.

[21] https://moodle.org/



Obrázek 3: Výpis možností z vybraného kurzu z angličtiny[22]

2.1.2 Komerční aplikace

Cisco WebEx[23] – *„tento on-line nástroj pro podporu virtuálního setkávání, videokonference a podporu mobility je k dispozici za cenu od 19 dolarů za měsíc. Vzdálení zaměstnanci se díky němu mohou na dálku elegantně účastnit různých porad a brainstormingů. WebEx ale není jen o videokonferencích. Podporuje i sdílení adresářů, možnost prohlížení dokumentů ostatními účastníky týmu či možnost vytvoření prostoru pro zpětnou vazbu zúčastněných stran.“* [4]

Microsoft SharePoint[24] – je platforma nástrojů pro podnikovou spolupráci, práci s dokumenty a informacemi, poskytující navíc pokročilé nástroje pro správu související infrastruktury, vysokou flexibilitu.

[22] Dostupné z: http://www.moodlenews.com/wp-content/uploads/moodleorgforums1-1024x841.png
[23] http://www.webex.com/
[24] http://www.ms-sharepoint-portal.net/

Výhodou je i integrace s balíčkem Office. Microsoft má dále přímo pro řízení projektů k dispozici i nástroj MS Project. Poskytuje nástroje pro správu dokumentů a webového obsahu. Umožňuje uživatelům přístup k potřebným informacím, které potřebují.

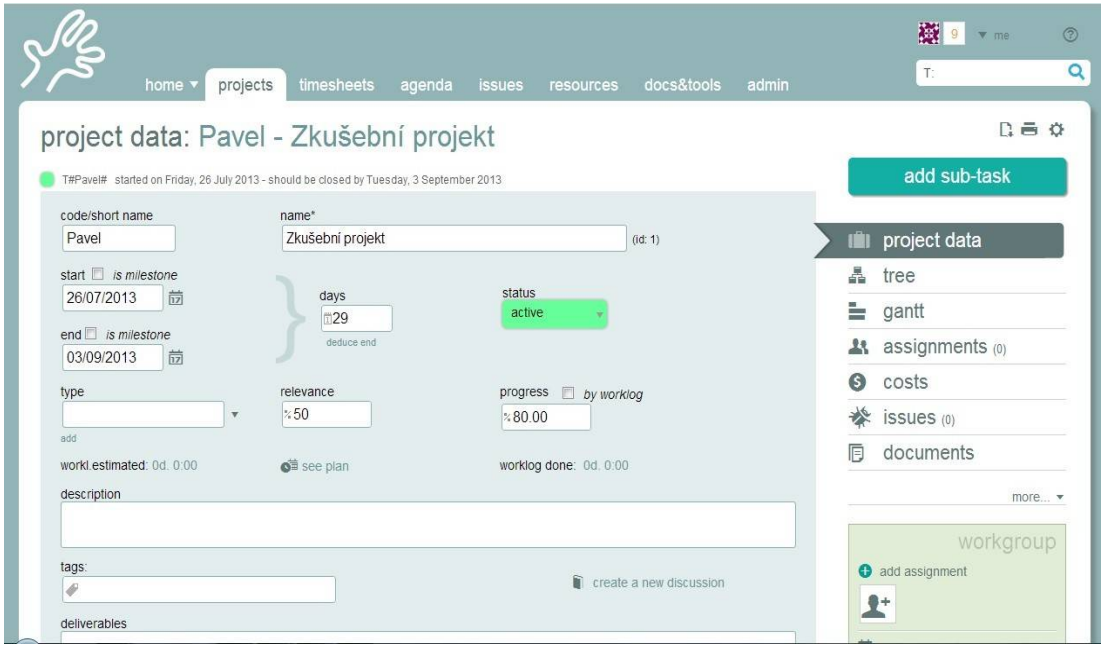
LiquidPlanner[25] – tato aplikace sleduje aktivity zaměstnanců, jejich pokroky a snadno je možné sledovat a ohlídat termíny ukončení projektů. Dále zde lze nastavovat prioritu jednotlivých projektů a to za 24 dolarů měsíčně na zaměstnance.

ConceptShare[26] – tato aplikace nabízí účinný nástroj pro vedení diskuzí, komentování a rozhodování účastníků v týmu. K dispozici jsou nástroje pro zvýraznění důležitých úkolů v rámci určitých projektů, které mají upozorňovat na akce, které jsou důležité k určitému termínu či pro práci kolegů. Velmi propracovaná je práce s obrázky, grafy a možnost vizuálního porovnávání.

Teamwork[27] – tato aplikace je odlišná od všech ostatních. Na stránkách tohoto produktu je nutno stáhnout aplikaci, kterou si můžeme vyzkoušet po dobu 15 dní. Po vypršení této zkušební doby je další používání za 90€ na měsíc.

Aplikaci je však možné nahrát na vlastní server a připojovat se pomocí vlastní domény a portu poté odkudkoliv, nebo používat aplikaci lokálně. Samotná aplikace je velice hezká, přehledná a moderní. Při vytvoření projektu si krom pojmenování musíme určit, odkdy by se mělo na projektu začít pracovat a kdy skončit. Určujeme si status projektu (kompletní, suspendovaný, chybný, aktivní atd.). Při každé práci na projektu můžeme nastavit, kolik toho máme v procentech hotového a jak je projekt důležitý. To je dle mého názoru od ostatních aplikací odlišné, protože v nich nastavujeme důležitost úkolu či obecně celého projektu a to už nezměníme.

[25] http://www.liquidplanner.com/
[26] http://www.conceptshare.com/
[27] http://www.teamworkpm.net/



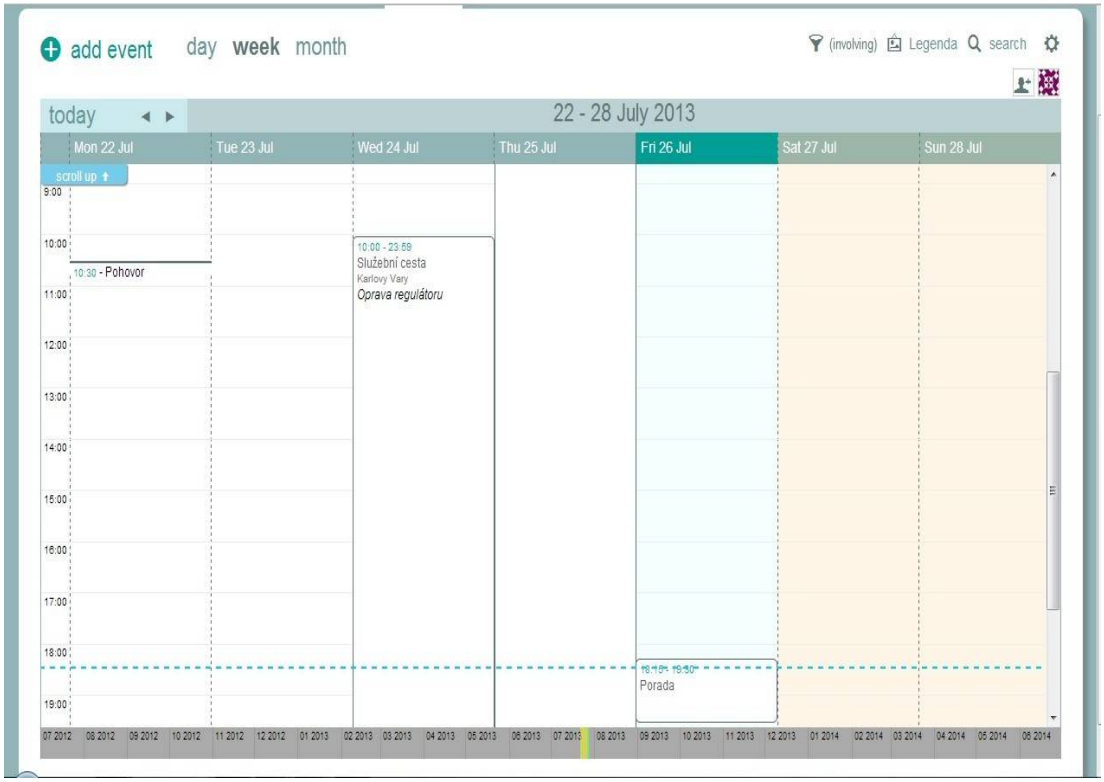
Obrázek 4: Vytvořená projektu v aplikaci Teamwork. Zdroj: vlastní

Zaujal mě malý barevný ukazatel práce na projektu. Když si otevřeme projekty, na kterých pracujeme, objevíme zde u každého projektu množství práce, která byla na něm celkově vykonána. Na hlavní stránce je možné vidět opět poslední aktivity a u svého účtu i kalendář se svými poznámkami.

Obrázek 5: Přehled všech projektů a informace o nich. Zdroj: vlastní

U každého projektu můžeme založit diskuzi a dopisovat si se všemi svými spolupracovníky, kteří pracují na konkrétním projektu.

Aplikace není však dělaná jen na komunikaci, či práci s projekty. Je možné ji používat i jako diář, zapisovat schůzky, úkoly, které se mají v určitý čas udělat.



Obrázek 6: Kalendář s naplánovanými poznámkami. Zdroj: vlastní

Velkou nevýhodou je neexistence přidělení práv jednotlivým uživatelům, což znamená, že každý projekt bude k dispozici všem uživatelům systému.

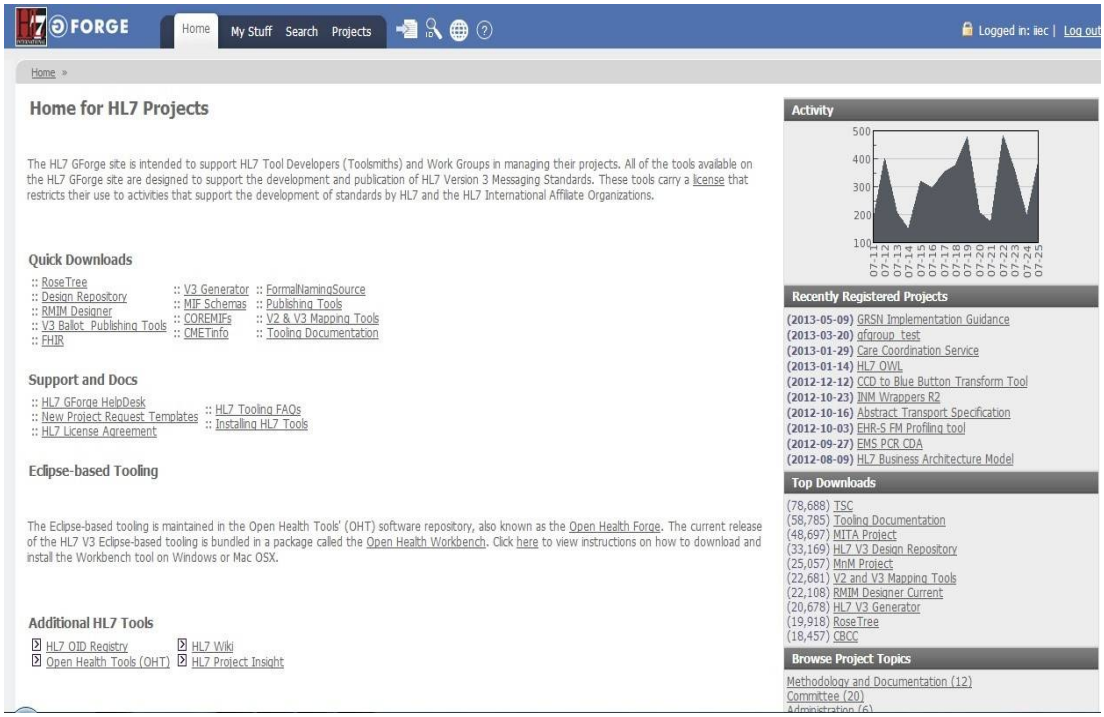
Tato aplikace je jednou z mála, která podporuje alespoň slovenský jazyk. Má však chyby v překladu, takže na jiné jazyky než angličtinu bych nespoléhal.

Gforge – je profesionální aplikace určená vývojářům k podpoře a řízení spolupráce ve svých projektech v pracovní skupině. Tyto nástroje nesou licenci, proto omezují jejich použití.

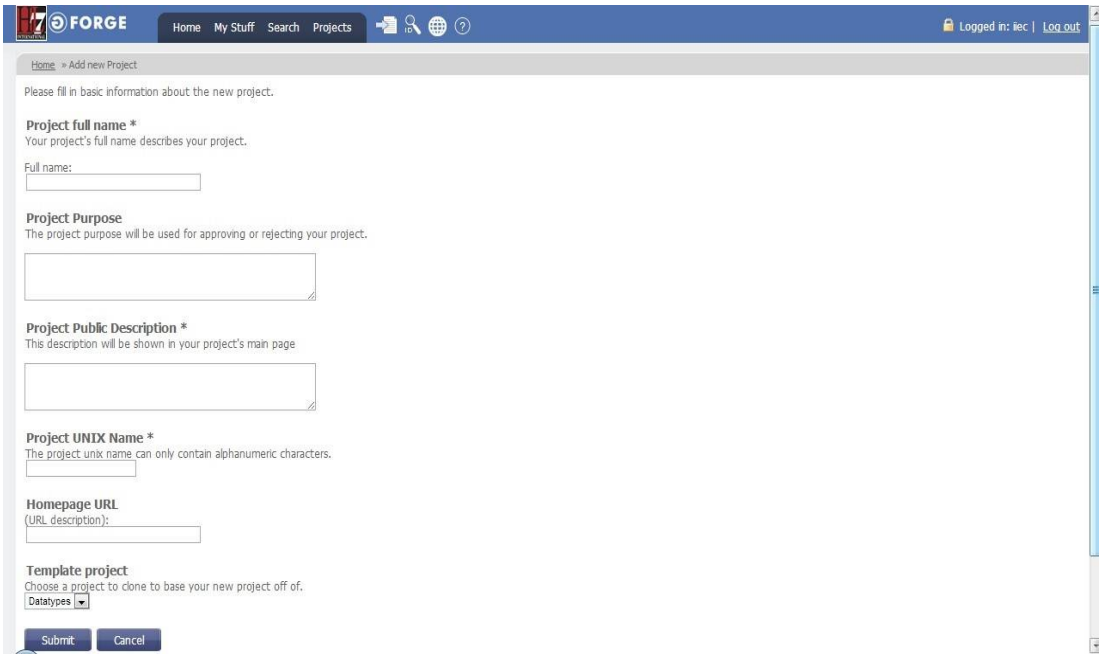
Projektu lze zadat prioritu. Stav, který během každé své účasti můžeme měnit. Jednotlivým uživatelům pracujícím na stejném projektu je možné přidávat poznámky, označovat chyby a přidělovat je konkrétním spolupracovníkům. Mohou tak zaznamenávat čas strávený nad konkrétní chybou. U ukládání souborů jsem nezaznamenal žádný problém. Gforce podporuje

velkou škálu formátů, takže při uložení nevznikal žádný problém. Upozornil bych na tabulku top projektů, které jsou vyhledávány a tabulku pro nejstahovanější projekty. Ty jsou zobrazeny na hlavní stránce v pravé části obrazovky.

Stránky mi přijdou graficky jednodušší s porovnáním např. Assembla. Zde najdeme vše potřebné pro konkrétní projekt. Při založení projektu lze přidělit práva uživatelům, nastavit data akcí, která pak budou znázorněna v kalendáři. Co se týče kalendáře, je velmi přehledný a na rozdíl od jiných aplikací i velký, takže poznámky jsou čitelné. Velmi mě zaujala podpora. Je jednoduchá, ale kvalitní a efektivní. Užitečný je také graf aktivit na serveru. Je možné zde vyčíst čas, kdy byl server zatížen a kdy naopak aktivita byla zanedbatelná.



Obrázek 7: Úvodní stránka aplikace Gforge. Zdroj: vlastní



Obrázek 8: Vytváření projektu v aplikace Gforge. Zdroj: vlastní

Nevýhodou je možnost použití. Tento projekt je veřejný, tudíž vše si můžete pouze vyzkoušet.

2.2 Zhodnocení možných použití popsaných aplikací

Jako uživatel si musím v prvé řadě říci, co od aplikace očekávám a v jakém množství budu aplikaci využívat. Pokud mám omezený počet lidí pracujících na projektech a neočekávám od aplikace takové vymoženosti jako je synchronizace se sociálními sítěmi, využívání aplikace i jako diář apod., určitě bych využil spíše nekomerční aplikaci, nebo aplikaci, kde poplatek za používání je spíše symbolický.

Právě z nekomerčních aplikací se mi líbila Assembla, neboť disponuje velkou škálou funkcí. Nabízí aplikační rozhraní, které je trochu „omezenější“, ale pro jednoduchou správu postačuje. Nemusíme se do této aplikace registrovat, pokud máme účet na Googlu nebo Yahoo.

3. Přístupová práva a zabezpečení webové aplikace

Zabezpečení aplikace je složitý proces. Programátor „bojuje“ proti útočníkům, kteří se snaží z mnoha důvodů nabourat jeho systém.

3.1 Zabezpečení

Zabezpečení aplikace znamená, jak bude infrastruktura aplikace zabezpečena před okolním světem.

Předpokladem zabezpečení a stanovení práv je určení okruhu uživatelů a tím omezení jeho funkčnosti. V mé aplikaci může každý uživatel, který nemá přístupová práva do aplikace, použít pouze prohlídku kalendáře. Zatímco jiní uživatelé, přihlášení do aplikace, ji mohou dle svého oprávnění používat (editovat, číst atd.).

„V prvé řadě je zapotřebí, aby aplikace po ověření uživatele vytvořila novou relaci. Do proměnných relace následně můžeme uložit detaily týkající se identity uživatele. V tomto případě se použije proměnná relace, která bude obsahovat jméno uživatele. Bude tak mít přístup k aktuálním detailům o uživateli kdekoliv v aplikaci.“ [6]

Hlavní slabiny webových aplikací:

3.1.1 Neověřený vstup

Webové aplikace používají vstup z požadavků HTTP pro určení způsobu jak na ně reagovat. Informace HTTP lze zakódovat mnoha různými způsoby. Velice často nejsou webové požadavky ověřeny předtím, než jsou použity webovou aplikací. Útočníci tak mohou nelegálně upravit libovolnou část HTTP požadavku - včetně např. adresy URL, vyhledávacího řetězce nebo záhlaví - a tak obejít zabezpečovací mechanismy webových aplikací.

Pro ověření vstupu jsou používány dva základní způsoby. Jeden představuje firewallová technologie znalá standardů a běžného chování aplikací. Ověřuje shodu s protokolem a nastavením aplikace. Druhým je firewallová technologie na webové aplikace hledající podezřelé struktury v požadavcích HTTP a parametrech. Pro spolehlivé ověření parametru jsou použity oba způsoby.

3.1.2 Narušení kontroly přístupu

Tento problém spočívá v nedostatečném zajištění přístupových práv uživatelů (autorizace, autentizace). Útočníci mohou odhalením nedostatků získat přístup k účtům jiných uživatelů, citlivým souborům nebo kritickým funkcím.

3.1.3 Porušení správy účtů a relací

Slabá ochrana uživatelských údajů, sloužících k přihlášení a údajů identifikujících relaci, představuje další slabé místo popisované skupinou OWASP[28]. Útočníkům je umožněno získat hesla, klíče, cookies[29] relací a další informace, které jim umožní obejít autentizační a restrikční mechanismy a získat cizí identitu.

3.1.4 Zneužití serveru k odesílání skriptů

Webových aplikací lze zneužít jako mechanismu k přenosu útoku na prohlížeč koncového uživatele. Úspěšný útok může vést k vyzrazení přihlašovacích informací uživatele, útoku na jeho počítač či zfalšování obsahu webové stránky a oklamání.

3.1.5 Umístění e-mailové adresy na webu

Toto téma není součástí chyby zabezpečení webové aplikace, ale patří k chybě, která patří k chybám bezpečnosti. Pokud kdokoliv volně publikuje e-mailovou adresu na webu, může být ohrožen např. spamovacími[30] roboty.

Stejně velkým nebezpečím je zveřejnění mobilního čísla.

3.2 Přístupová práva

Důležitým zabezpečením každého systému jsou přístupová práva. K tomu slouží procesy autorizace a autentizace.

[28] Open Web Application Security Project
[29] Malé množství dat, které server pošle webovému prohlížeči
[30] Nevyžádané sdělení (nevyžádané reklamy).

3.2.1 Autentizace

Slouží k jednoznačnému určení uživatele, který je přihlášen k aplikaci. Cílem je zajištění komunikace konkrétního uživatele se systémem.

Každý systém s důležitými daty by měl zajišťovat autentizaci uživatele. Uživatelé jsou vytvořeni a uloženi v databázi s jejich hesly, která by měla být šifrována podle některých ze šifrovacích funkcí (DES[31], MD5[32], atd.).

Tato možnost však není jedinou možností. K autentizaci mohou být použity i jiné speciální aplikace (bezpečnostní aplikace), hardwarová zařízení (čipové karty) či jiné služby operačního systému (biometrie[33]).

Výhodou tohoto procesu je možnost volby počtu neúspěšných pokusů. Při dosažení tohoto počtu neúspěchů si může administrátor nechat zaslat informaci o této aktivitě. Dle této informace se pak může rozhodnout o zajištění větší bezpečnosti sítě, blokace účtu, odebrání práv atd.

3.2.2 Autorizace

Proces, který specifikuje práva, která má jednotlivý uživatel v systému. Tento proces většinou navazuje na proces autentizace. Cílem přidělení autorizačních práv je omezení provádění akcí v systému (čtení, zápis, mazání, apod.).

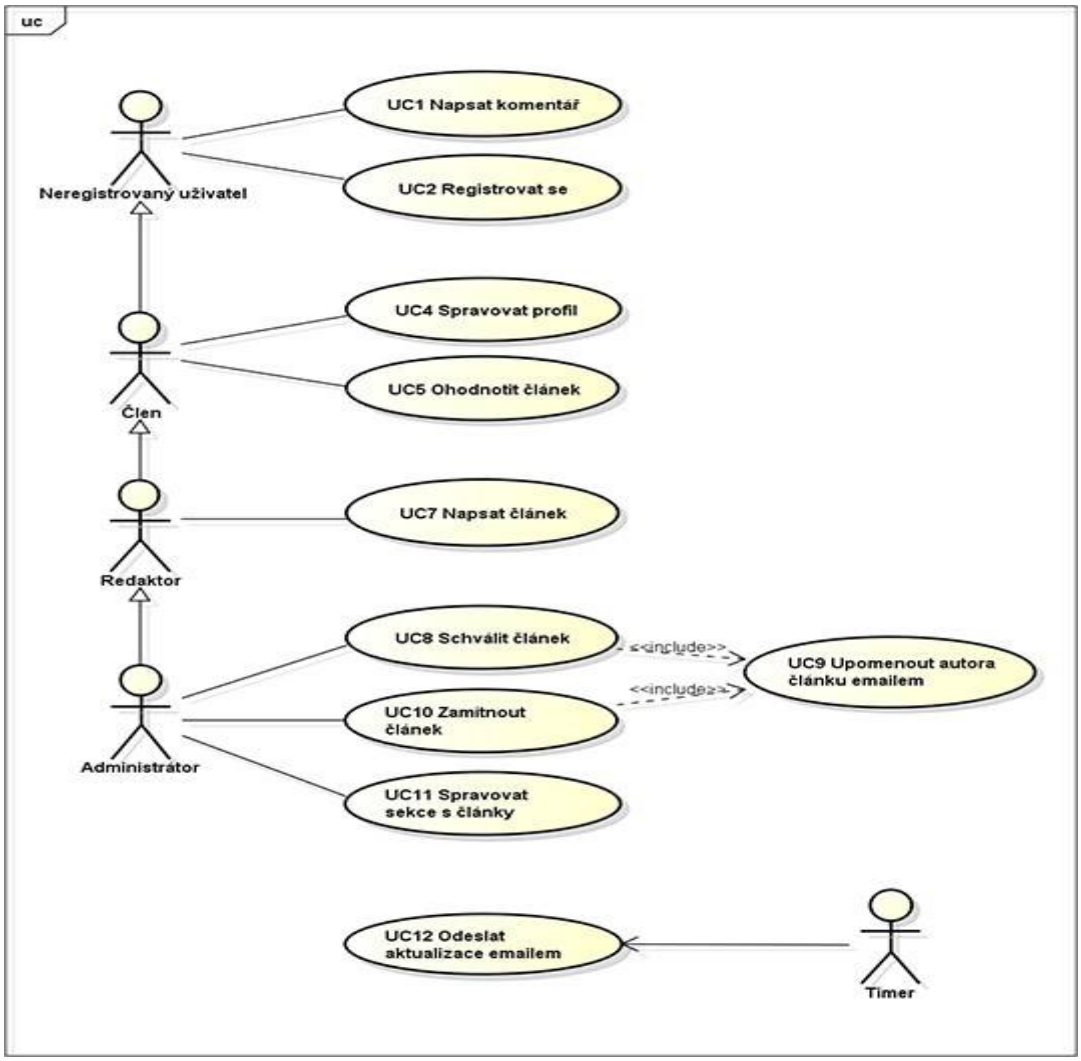
3.2.3 Role

V rámci každé webové aplikace se uživatelé připojují s určitými právy, rolemi. Tato práva, neboli role, slouží aplikaci nebo programu určovat, co vše má uživatel povoleno tzn., co může obsluhovat, modifikovat, číst či přidávat práva.

Typická aplikace by měla mít alespoň dvě role: administrátor a uživatel. V mé aplikaci počítám s tím, že krom rolí uživatel (v mé aplikaci: ostatní) a administrátor, který musí obsluhovat vše a to především kvůli problémům, které mohou v aplikaci nastat, jsou zde role vedoucí oddělení, vedoucí projektu, manager, spoluprojektant atd. Všechny role, které jsou

[31] Data (Digital) Encryption Standard
[32] Message-Digest algorithm
[33] Metoda, založená na rozpoznávání jedinečných biologických charakteristik subjektu (otisk prstu).

v aplikaci použity, neslouží jako přístupová role při přihlášení. Příklad použití rolí předvedu na UML[34] diagramu typu use case[35].



Obrázek 9: Use case diagram[36]

Na obrázku č. 9 figuruje 5 aktérů (uživatelů).

Každý aktér má v systému přidělenu nějakou roli. Neregistrovaný aktér je návštěvník, který aplikaci ještě nepoužíval, nebo nepotřebuje další služby využívat. Jeho právem v systému je psát komentáře nebo se registrovat.

[34] Unified Modeling Language
[35] Zobrazuje chování systému tak, jak ho vidí uživatel.
[36] Dostupné z: http://www.devbook.cz/uml-use-case-diagram

Člen je aktér, který má stejné možnosti jako neregistrovaný uživatel, ale může už hodnotit články a spravovat svůj profil. Propojení mezi uživateli znázorňuje šipka (asociace).

Redaktor je aktér, který má stejná práva jako člen, ale navíc může psát články.

Administrátor je ten, který se stará o funkčnost aplikace, ale má možnosti navíc, se kterými už nikdo nepracuje. Vazba <<*include*>> se používá v případě, že nějaká funkce je natolik důležitá, že ji chceme mít viditelnou v use case diagramu. Tato vazba se spustí vždy, když je spuštěn příkaz, na který je napojen.

Timer jako člen se spouští v určitý čas a odešle aktualizace e-mailem.

Rolím obsaženým v mé webové aplikaci se budu podrobněji zabývat v popisu praktické části.

3.3 SQL[37] injection

Základem každého informačního systému je dobře navržená databáze, od níž očekáváme spolehlivost, stabilitu, rychlost, bezpečnost uložených dat a víceuživatelský systém. Obsahuje uživatelské informace a spoustu citlivých dat (informace o bankovních kontech).

Co se vše může stát, pokud webová aplikace není zabezpečena? Naše bankovní konto bude vybráno, naše osobní údaje budou zneužity.

3.3.1 Co je SQL injection

SQL injection patří mezi kritické chyby webových aplikací. Umožnuje útočníkům (hackerům, crackerům) číst, zapisovat, měnit informace v naši databázi jako např. hesla, čísla účtů, osobní údaje, informace o zakázkách atd. Do databáze se dnes ukládají veškeré informace, se kterými web pracuje.

SQL injection pracuje na bázi vložení SQL příkazu do jiného SQL příkazu, který útočník vloží do napadené webové aplikace, většinou skrze formuláře (přihlašovací, kontaktní, objednávající). Dále může změnit adresu URL[38] , nebo podstrčit tzv. cookie[39]. Jediné, co útočníkovi stačí pro vniknutí do databáze, je webový prohlížeč, znalost jazyka SQL.

[37] Structured Query Language („strukturovaný dotazovací jazyk“).
[38] Uniform Resource Locator („jednotný lokátor zdrojů“).
[39] Malé množství dat, která WWW server pošle prohlížeči.

3.3.2 Ochrana před útoky SQL injection

Ochrana proti útoku typu SQL injection v podobě aplikace žádná neexistuje. Např. běžný firewall, který slouží k zabezpečení síťového provozu, tento problém nevyřeší. Webová aplikace potřebuje otevřený přístup do databáze a to včetně oprávnění (čtení, zápis). Útočníkovi právě z tohoto důvodu stačí znát jazyk SQL.

Ochrana však na tyto útoky přece jen existuje. Musíme zabezpečit vstupy příkazů do databáze.

Nyní bych ukázal pár příkladů, jak SQL injection provést a jak mu předcházet.

Útočník chce vypsat články pomocí `id`[40]:

```
//nezabezpeceny nachylny dotaz

$dotaz = "select * from CLANKY where id = '$_GET["id"]'";
```

Tento kód načítá vše z tabulky `CLANKY`, kde `id` článku se rovná proměnné `id` získané z proměnné `GET` požadavku. Jak vidíte, tento vstup není nijak chráněný a tak útočník může zadat tento dotaz do adresy:

```
clanek.php?id=1 and 1=1/*
```

Tímto dotazem vám upraví data z tabulky `CLANKY`, ale může vám ji také smazat (zrušit), vložit do ní nové údaje.

Řešení tohoto problému spočívá v ošetření vstupu id. Id má být číslo, takže použijeme funkci `is_numeric()`.

```
$id = $_GET["id"];

//kontrola, zda je id cislo

if (!is_numeric($id)):

echo "Toto ne!";

else:
```

[40] Označení klíčovou položku tabulky

```
$dotaz = "select * from clanky where id='$id'";

endif;
```

Co se však může stát, pokud id má být typu `string`[41].

```
if (!is_string($_GET["id"])):

echo "Toto ne!";

else:

//escapovani nebezpecnych znaku

$id = addslashes ($_GET["id"]);

$dotaz = "select * from clanky where id = '$id'";

endif;
```

V těchto dvou případech je použita méně bezpečná metoda `GET`, která předává formulářová data jako součást adresy URL za otazníkem. U parametrů stránky to nevadí, ale u přihlášení je to na opak. U metody `POST` se data předávají v těle dotazu, takže nejsou viděna v adrese URL. To znemožňuje jednoduché zkopírování údajů jiného uživatele.

Pro ošetření vstupu však jsou ještě další možností. Textové pole formuláře bude obsahovat např. 50 znaků. Tím se zamezí vložení škodlivých údajů k provedení útoku.

Nejdůležitější pomůckou pro SQL injection je řetězec: `' or 1=1 --` . Tímto řetězcem se zjistí, zda má administrátor vstupy ošetřené či nikoliv např. u vstupu pro heslo.

Použitím SQL injection v aplikaci, která není nijak ošetřena, si může zkušený uživatel zjišťovat, mazat, přepisovat jakékoliv informace, které jsou v databázi uloženy. Proto bychom měli ty nejdůležitější textové vstupy zabezpečovat a snažit se, aby nám nikdo nezasahoval do bezproblémového chodu aplikace.

[41] Datový typ sloužící pro textový řetězec.

3.4 Další útoky na webové aplikace

Dalšími útoky, které mohou útočníci použít, je napadení klientských skriptovacích jazyků a napodobování existujícího rozhraní pro získání přístupu k citlivým údajům.

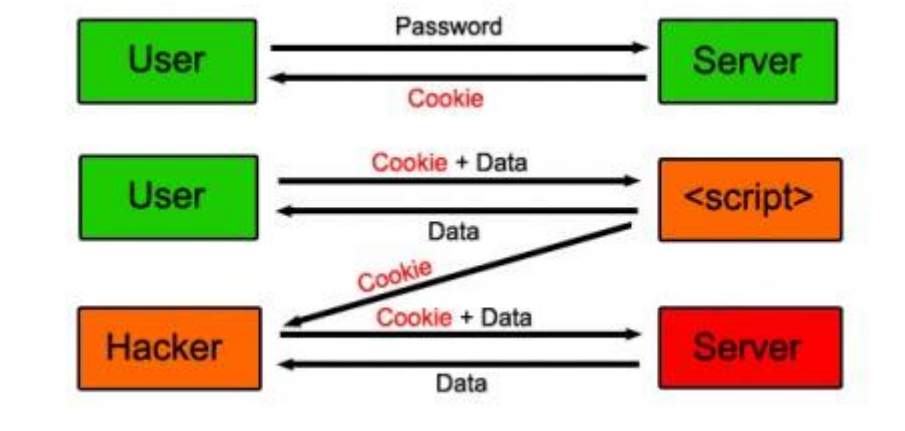
3.4.1 XSS

Pro útoky založené na vkládání nebezpečného skriptu do obsahu webové stránky se používá metoda typu XSS. Spočívá na úmyslném vložení nebezpečného kódu (napsán v klientském skriptovacím jazyku – Javascript) do obsahu webové stránky. K úspěšnému provedení útoku je potřeba vložení skriptu v prohlížeči a někdy i za pomoci klienta.

Rozeznáváme tři typy útoků XSS: okamžitý, perzistentní a lokální.

Okamžitý (non-persistent nebo reflected)

Tento velmi běžný typ XSS útoku se projevuje okamžitým vykonáním útočného skriptu – webová aplikace zobrazí výsledek okamžitě po odeslání požadavku, skript se nikam neukládá. V praxi se běžně využívá parametrů v adrese URL, které se nahradí nebo doplní útočnou konstrukcí skriptu. Pokud na takovou adresu URL oběť klikne, vykoná se útočný skript a útočník může zcizit údaje uložené v uživatelově prohlížeči.



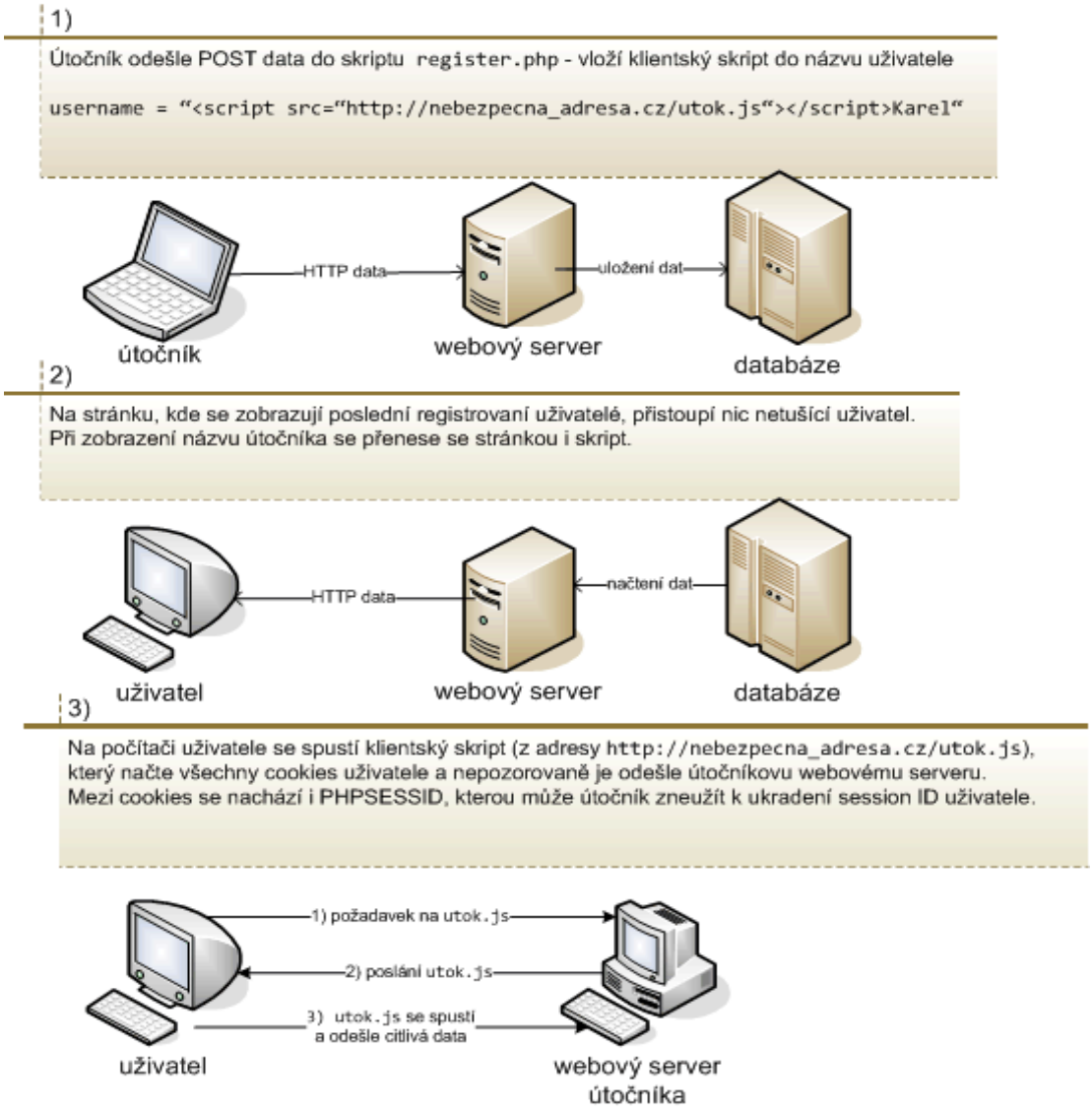
Obrázek 10: Práce útočného scriptu[42]

Perzistentní útok (persistent, stored)

Patří k nejnebezpečnějším typům XSS útokům, kdy oběť nemusí vstupovat na napadané stránky přes upravený link. Skript je generován přímo z databáze a útočník může vložit skript

[42] Dostupné z: http://www.chmag.in/system/files/aug2010/xss.png

do databáze např. přes komentáře v diskusních fórech, ale i přímým vložením. Rozsah poškození může být daleko větší než u předchozího typu útoku, neboť se může týkat všech uživatelů, kteří si danou informaci zobrazí ve svém prohlížeči.



Obrázek 11: Příklad perzistentního útoku[43]

Lokální útok (DOM-based (Document Object Model), local)

Velmi podobný okamžitému útoku typu XSS. K útoku je však využit existující klientský skript – přenesením např. z adresy URL kódu do skriptu stránky.

[43] Dostupné z: http://access.feld.cvut.cz/storage/200709131207_1-xss.gif

Je nutné poznamenat, že slabiny typu XSS nesouvisí pouze s JavaScriptem. Můžeme je nalézt i v technologiích jako jsou ActiveX, Flash, VBScript, HTML, Java.

Zabezpečení proti XSS

Detekce chyb typu XSS (zejména JavaScriptových) je poměrně snadná pomocí testování anebo analýzy kódu. Nejdůležitějším zabezpečením je však nepouštět si ve webové aplikaci žádné skriptovací konstrukce. Protože jazyk HTML je schopný zpracovávat kód klientských skriptů i jako reakci na události jednotlivých značek (např. onchange, onclick) nebo může měnit vzhled prvku z hlediska kaskádových stylů (parametr style), jedno z nejlepších řešení je překódovat HTML ohraničovací značkou < a > jako entity. Tímto způsobem docílíme zobrazení zdrojového kódu přímo v textu, aniž by ho prohlížeč chápal jako značky.

3.4.2 Zneužití oprávněného požadavku (CSRF)

Další možný útok na webovou aplikaci, který je typově podobný útokům typu XSS. Týká se zneužití oprávněných požadavků registrovaných uživatelů na určitý systém. Cílem útočníka je využít přihlášení uživatele do systému k vykonání požadavku, který je v danou chvíli z pohledu vztahu uživatel-systém oprávněný. Podobnost s útokem typu XSS je ve způsobu získání důvěry uživatele. Pomocí klientského skriptu je realizován útok odesláním skrytého formuláře. Při tomto útoku se nepřenáší žádné důvěrné informace, protože vykonavatelem operace je pouze oprávněný uživatel.

Typickým řešením tohoto útoku je systém tokenizace[44], který je založen na generování jednorázových přístupových hesel (tzv. tokenů). Při vykonání požadavku se ověří, zda odeslaný token je identický s očekávaným, operace se vykoná a token se zničí, aby nemohl být použit znovu.

Možností je kontrolování hlavičky HTTP, což je nespolehlivé, neboť některé firewally tuto vlastnost blokují, mažou. Krom toho, ne každý požadavek musí nutně pocházet z předcházející stránky. Další možností je autentizace pomocí aplikací, jako je autorizační kalkulátor či autorizační SMS (např. bankovní převody).

[44] Rozdělení textu na jednotlivé tvary.

4. Vlastní webová aplikace

Při vykonávání povinné praxe v nejmenované firmě jsem se setkal s jednoduchou aplikací, která v této firmě sloužila pro sdílení projektů. Uživatelé v rámci celé aplikace měli svá práva, která byla pro celou aplikaci. Tzn., že zaměstnanec měl v rámci celého systému přístup ke všem složkám se soubory řadových zaměstnanců. Vedoucí oddělení měli přístup neomezený. Já, jako výpomoc v této firmě, jsem měl přístup pouze k jedné složce, do které mi byly zasílány potřebné materiály.

Tato metoda byla zajisté efektivní a v rámci lokální aplikace postačující. Ovšem vnější přístup do sítě nebyl možný. Z mého pohledu byla aplikace trochu nevyhovující, neboť přístup a spolupráce byla podobná protokolu FTP, kdy sdílíte nejen data, ale i informace (potřebné úkoly, chyby v projektu).

Z tohoto důvodu jsem se rozhodl navrhnout vlastní webovou aplikaci, která tyto nedostatky bude řešit.

4.1 Návrh a tvorba databáze pro webovou aplikaci

Standardní webová aplikace, která pracuje s daty, ukládá a získává tato data z databáze. Moje aplikace je tvořena pro firmu zabývající se IT technologiemi. Databáze je tvořena pomocí databázové platformy Oracle, protože je velmi rozšířená – implementace v 6,5 miliardách zařízení.

Pro vlastní implementaci databáze jsem použil Oracle Express Edition (XE), která je omezenou verzí Oracle Database 11g a je plně kompatibilní se všemi edicemi databáze Oracle. Verze XE obsahuje plnou podporu jazyka SQL včetně jeho procedurální nadstavby PL/SQL. Tato verze slouží jako databázový server, na kterém jsem si vytvořil vlastní schéma.

V tomto schématu je potom obsažena celá struktura tabulek s jejich propojením, triggery, sekvence atd.

Pro práci s databází jsem použil následující nástroje: Oracle SQL Developer Data Modeler (verze 3.3.0.747), Oracle Database 11g Express Edition (verze 11.2.0), Oracle SQL Developer (verze 3.2.10.09).

4.1.1 Oracle SQL Developer Data Modeler

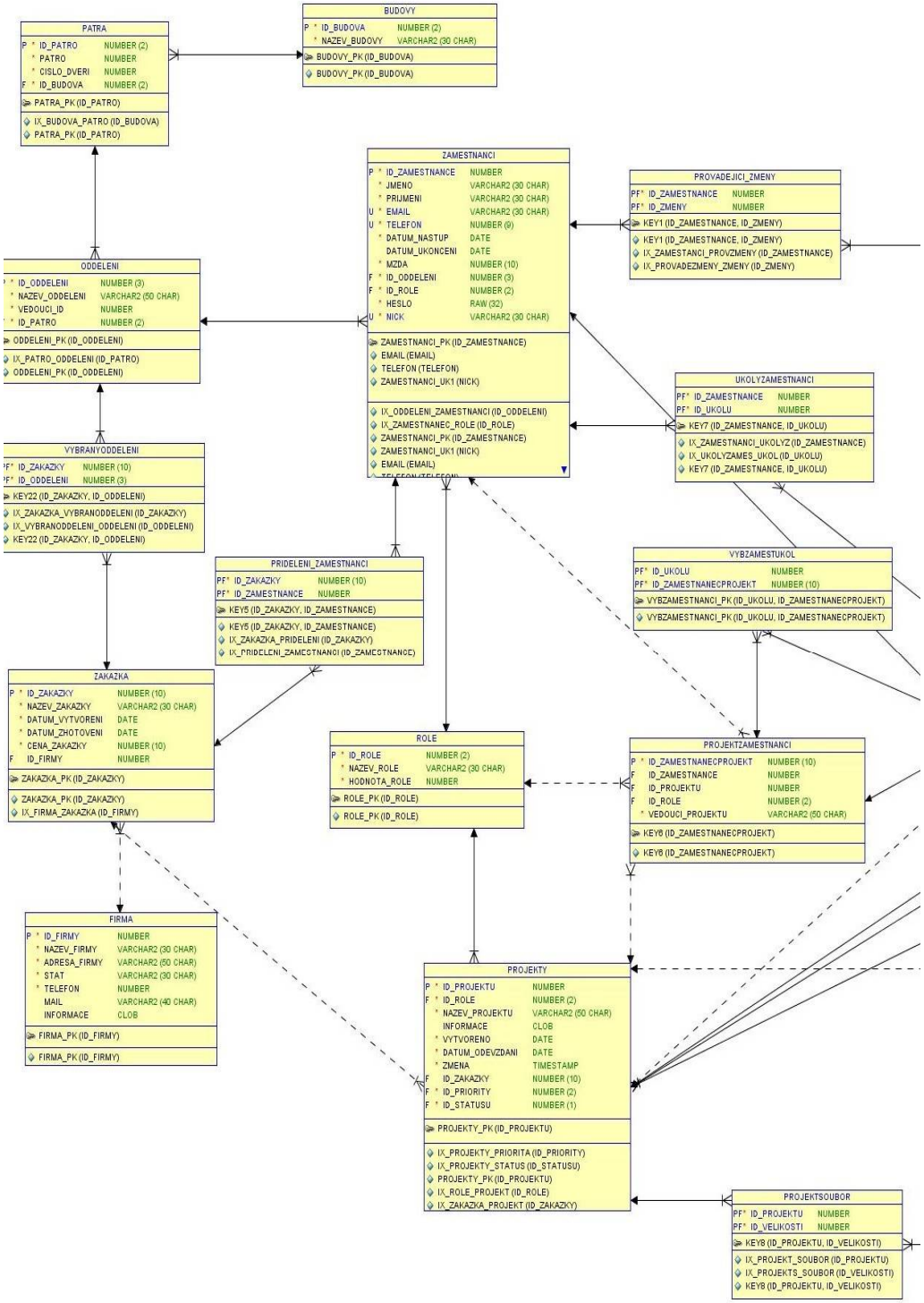
Oracle SQL Developer Data Modeler obsahuje nástroje pro datové modelování na logické i fyzické úrovni. Vedle Oracle Database podporuje i řadu dalších databází, jako je IBM DB2, Microsoft SQL Server a vlastně cokoliv, k čemu máte JDBC drivery.

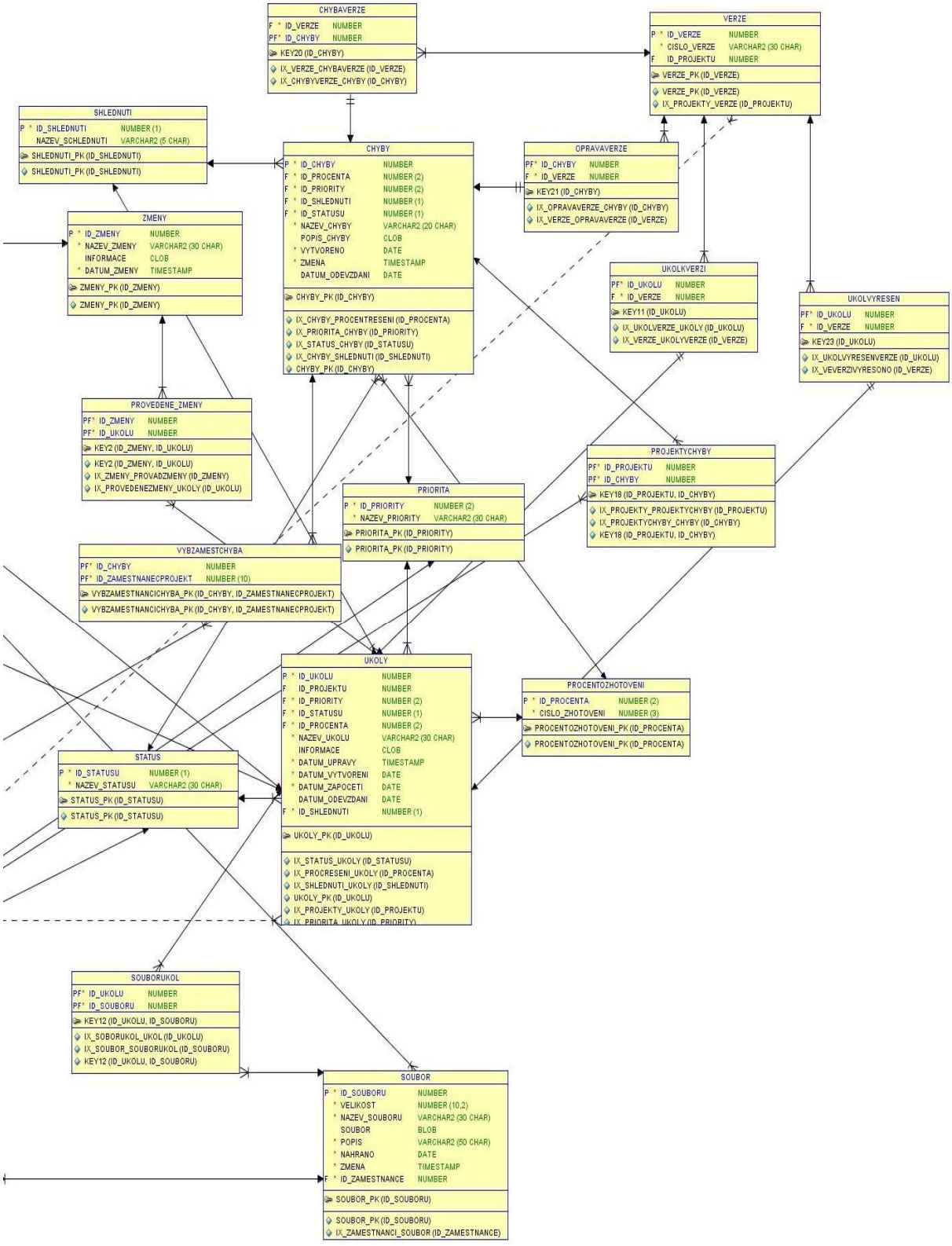
Fyzický datový model

Je model, který popisuje databázový model již v kontextu databázového systému, ve kterém bude implementován. Obsahuje tabulky související s firmou, s jejími zaměstnanci, právy a především s projekty.

„Cílem fyzického datového modelování je navrhnout kvalitní datovou strukturu pro konkrétní aplikaci a databázový systém, který bude naše aplikace využívat k uložení dat. Fyzický datový model zachycuje fyzickou strukturu datové základny aplikace. Jde o popis vlastní realizace systému v konkrétním implementačním prostředí.“ [7]

Datový model a popis tabulek





Obrázek 12: Datový model databáze[45]. Zdroj: vlastní

[45] Nejlepší kvalita obrázku je součástí CD ve složce Datový model.

Nejdůležitějšími tabulkami datového modelu jsou tabulky PROJEKTY a ZAMESTNANCI. Projekt může založit jakýkoliv zaměstnanec (tabulka ZAMESTNANCI). Tyto tabulky jsou k sobě spojeny referencí M:N (jeden projekt může být založen vícekrát - zaměstnanec může založit více projektů). Projekty však mohou ovládat i jiní uživatelé než jen zakladatel projektu a to vybraní zaměstnanci, kteří dostanou přístup k tomuto projektu.

Všechny tyto tabulky jsou spojeny s tabulkou ROLE, kde každý má v systému či k projektu nějakou roli (právo). V mé aplikaci využívám pro aplikaci role: administrátor, vedoucí oddělení, uživatel. V rámci projektu může uživatel nabývat rolí: spoluprojektant, vlastník projektu a všichni (tzn., přidělím projekt všem zaměstnancům v mém oddělení).

Aby se projekt dal založit, musí být připojen k nějaké zakázce (tabulka ZAKAZKA). Každou zakázku vlastní nějaká firma, kterou musí databáze uchovávat, abychom jako uživatelé věděli, s kým naše firma obchoduje. Spojení mezi těmito tabulkami je 1:N (jakékoliv firmě může patřit více zakázek v naší databázi - konkrétní zakázka musí potom patřit jedné firmě).

Jako manager firmy, který dohlíží na obchod ve firmě, přiřazujete zakázku oddělení a konkrétnímu zaměstnanci firmy, který ji dostane na starost. Dále jsou v databázi pro funkčnost aplikace i méně důležité tabulky pro uchování sídla jednotlivých oddělení (budova, patro).

Zaměstnanec, který založí projekt, dále přiděluje práva pro kolegy ve firmě. Ti, pokud mají právo, mohou přistupovat k tomuto projektu, přiřazovat chyby, přidávat data (soubory, obrázky, atd.). Každý soubor je nejenom součástí daného projektu, ale můžeme i zjišťovat kdo jej uložil na server. Jednotlivým nahraným souborům týkajících se projektu, můžeme přidělovat úkoly (např. doplň komentáře v kódu).

Úkoly jsou přiřazeny zaměstnancům, kteří mají na nich pracovat a pro efektivitu spolupráce přidělíme i prioritu tohoto úkolu (jak moc spěcháme na vypracování). Ke kontrole, aniž bychom museli otevírat soubor, slouží tabulka PROCENTOZHOTOVENI, která udává míru práce, která byla na tomto úkolu vypracována.

Tabulka CHYBY je svou funkčností podobná tabulce UKOLY. Rozdíl mezi nimi je takový, že tabulka UKOLY informuje uživatele o úkolech, které se musí vypracovat, aby projekt resp. zakázka byla zhotovena. U všech tabulek (CHYBY, PROJEKTY, UKOLY) slouží k této

informaci o zhotovení tabulka STATUS a její atributy: nerozpracováno, rozpracováno, zhotoveno. Tabulka CHYBY však informuje uživatele, kteří plnili daný úkol, o chybě. Ta může být nalezena kýmkoliv z kolegů v týmu.

Chybu však můžeme nalézt i v souboru, který již byl dávno jako úkol ukončen. Nyní může nad tímto souborem pracovat jiný z kolegů. Pro informaci o chybách, které právě mohou být staršího data, informuje tabulka CHYBAVERZE. Každý projekt, úkol, je řešen v nějaké verzi, na které se pracuje. Co když najdeme chybu ve starší verzi? Proto musíme uživatele o tomto problému informovat a ten se ji pokusí napravit v nějaké další aktuální verzi.

Po vytvoření datového modelu s atributy, triggery, sekvencemi, si vygenerujeme SQL skript, který vložíme do aplikace Oracle SQL Developer a spustíme. Tím se vytvoří struktura v databázi.

4.1.2 Oracle SQL Developer

Ukázka skriptu

Vytvoření tabulky ZAMESTNANCI:

```
CREATE
  TABLE ZAMESTNANCI
  (ID_ZAMESTNANCE NUMBER NOT NULL,
    JMENO          VARCHAR2 (30 CHAR) NOT NULL ,
    PRIJMENI       VARCHAR2 (30 CHAR) NOT NULL,
    EMAIL          VARCHAR2 (30 CHAR) NOT NULL,
    TELEFON        NUMBER (9) NOT NULL,
    DATUM_NASTUP   DATE NOT NULL,
    DATUM_UKONCENI DATE,
    MZDA           NUMBER (10) NOT NULL,
    ID_ODDELENI    NUMBER (3) NOT NULL,
    ID_ROLE        NUMBER (2) NOT NULL,
    HESLO RAW (32) NOT NULL,
    NICK VARCHAR2 (30 CHAR) NOT NULL )
```

Spojení tabulek `ZAMESTNANCI` a `ODDELENI` pomocí primárního (cizího klíče):

```
ALTER TABLE ZAMESTNANCI ADD CONSTRAINT ODDELENI_ZAMESTNANCI
FOREIGN KEY (ID_ODDELENI ) REFERENCES ODDELENI ( ID_ODDELENI )
NOT DEFERRABLE;
```

Přidání triggeru (spouštěče) do tabulky `ZAMESTNANCI`:

```
CREATE OR REPLACE TRIGGER ADMIN.TS_ZAMESTNANCI_PRICTENIZAMES_0
BEFORE INSERT ON ZAMESTNANCI
FOR EACH ROW
BEGIN
:new.Id_zamestnance := pricteniZamestnance.nextval;   END;
```

Sekvence je objekt, který generuje číslo, které se automaticky zvyšuje. Tento objekt je volán triggerem a to vždy, když se přidává nový záznam.

Přidělení sekvence tabulce `ZAMESTNANCI`:

```
CREATE SEQUENCE ADMIN.PRICTENIZAMESTNANCE
INCREMENT BY 1
MAXVALUE 9999999999999999999999999999
MINVALUE 1 CACHE 20;
```

4.2 Webová aplikace

Webová aplikace je tvořena pomocí jazyka HTML[46] a jazyka PHP[47]. Implementace nejdůležitějších částí aplikace je popsána v dalších kapitolách.

4.2.1 Přihlášení a odhlášení od databáze

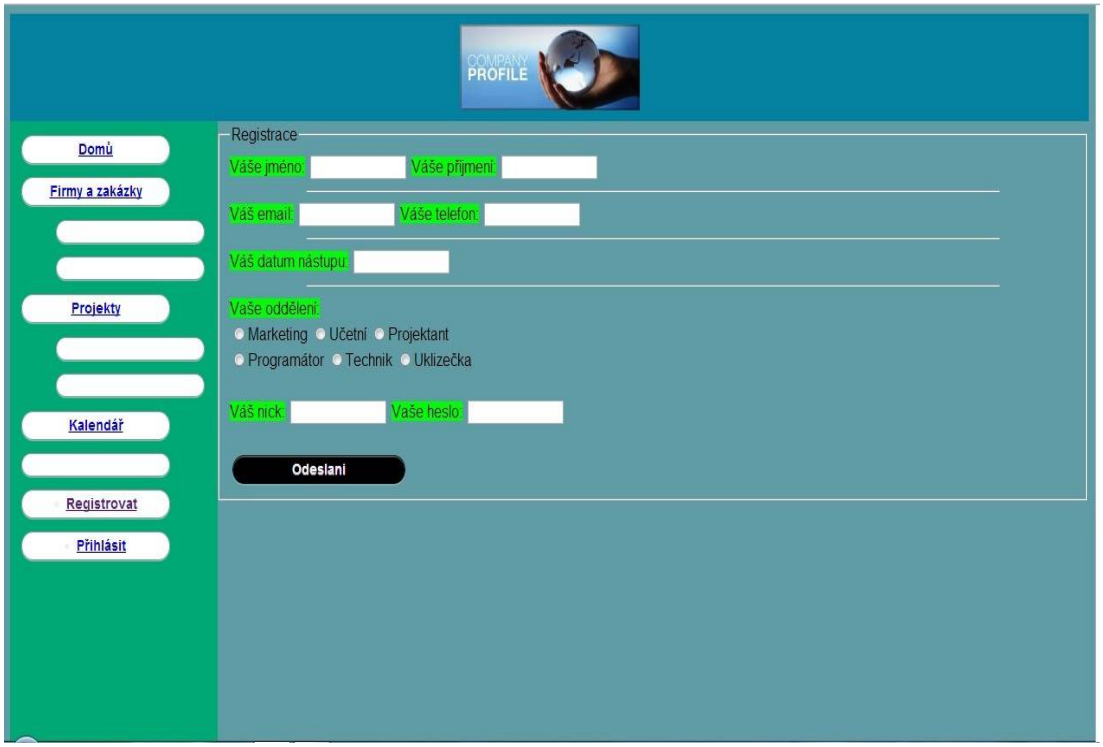
Přihlášení a odhlášení od databáze se provádí vždy, když hodláme pracovat, vypisovat, měnit, či se přihlásit do aplikace. K této funkci sloučí třída `Oracle` s metodami (funkcemi) `connect()` a `close()`.

[46] Hyper Text Markup Language
[47] Hypertext Preprocessor

```
    function connect() {
$c= oci_connect($this->username,
$this->password,  $this->host,
$this->charset);
      if (!$c) {
            $this->e = oci_error();
            return false;
        }
          $this->c = $c;
          return true; }
function close() {
     oci_close($this->c);
     $this->c = null;
     $this->e = null;
     return true;}}
```

4.2.2 Přístupová práva – registrace, přihlášení

V mé webové aplikaci jsem využil dvou přístupových práv. Každý uživatel, který chce používat tuto aplikaci, se musí registrovat. V registračním formuláři je uživatel vyzván k vyplnění osobních dat (jméno, příjmení, atd.), nicku, hesla a oddělení, do kterého bude přiřazen.



Obrázek 13: Obrazovka pro registraci. Zdroj: vlastní

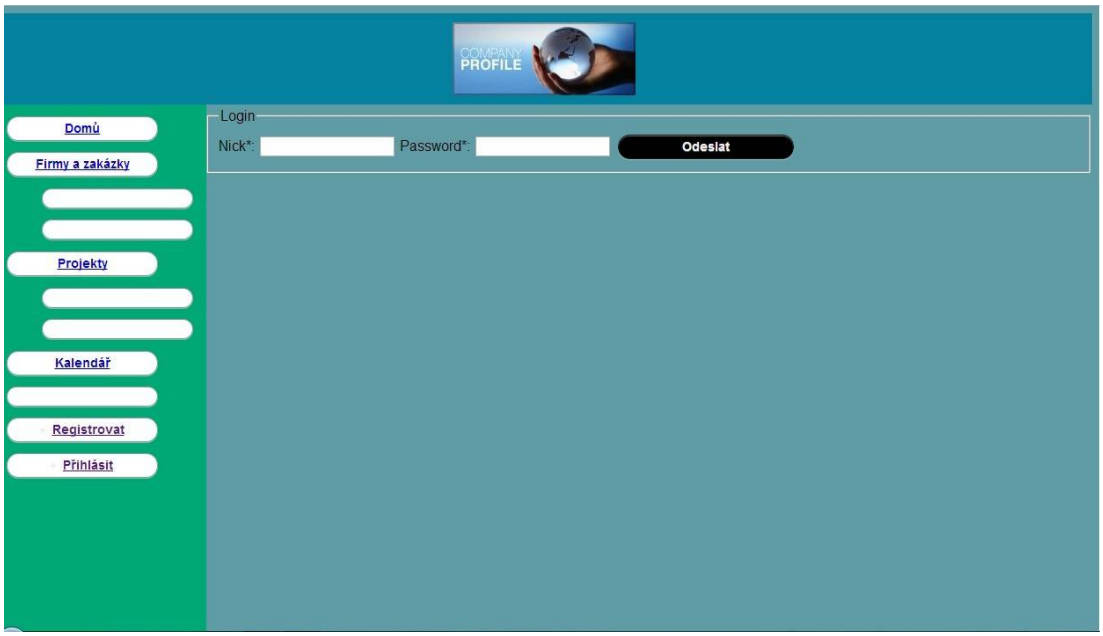
Po úspěšné registraci se může uživatel přihlásit do aplikace. Logika přihlašovací obrazovky je následující. Pokud proměnná specifikující akci formuláře obsahuje hodnotu login, znamená to, že uživatel se snaží přihlásit. Přihlášení se realizuje voláním metody `LOGIN()`, která je implementována ve třídě `Auth`. Pokud je přihlášení úspěšné, zobrazí se hlavní stránka aplikace a v dolní části se změní odkaz přihlášení na odkaz odhlášení. V opačném případě se nastaví proměnná s popisem chyby a přihlášení musíte zopakovat.

Ukázka kódu pro přihlášení:

```
function login($nick, $password) {

        $db = new Oracle ();

        $db->connect();

        $r = $db->sql(

"select * from zamestnanci

join role on role.id_role= zamestnanci.id_role
```

```
where nick ='$nick' and heslo =
rawtohex(sys.dbms_obfuscation_toolkit.md5(input_string =>
'$password'))");

        $_SESSION['uzivatele'] = $nick;

        $db->close();

 if ($r != false) {

    if (count($r) > 0) {

        $profil = array(

        'idecko' => $r[0]['ID_ZAMESTNANCE'],

        'jmeno' => $r[0]['JMENO'],

        'prijmeni' => $r[0]['PRIJMENI'],

        'email' => $r[0]['EMAIL'],

        'telefon' => $r[0]['TELEFON'],

        'nick' => $r[0]['NICK'],

        'datum nástupu' => $r[0]['DATUM_NASTUP'],

       'datum nástupu' => $r[0]['DATUM_UKONCENI'],

   'mzda' => $r[0]['MZDA'], 'id oddělení' =>
$r[0]['ID_ODDELENI'],

       'role' => array('name' => $r[0]['NAZEV_ROLE'],

                        'authority' => $r[0]['HODNOTA_ROLE']

                    ));

                $_SESSION['identity'] = $profil;

                self::$identity = $profil;
```

```
                return true;

            } else {

                self::$identity = NULL;

                return false;

            }} else {

            return false;

        } }
```



Obrázek 14: Obrazovka pro přihlášení do aplikace. Zdroj: vlastní

Ke kontrole přihlášení slouží metoda `isLoggedIn()`, která kontroluje stav přihlášení v rámci relace. V mé aplikaci, pokud tato metoda vrátí odpověď, že nejste přihlášen, máte zablokované některé odkazy pro práci s aplikací. Metoda `hasIdentity()` a její volající metoda `isLoggedIn()` mají tělo následující:

```
function hasIdentity(){

   if (self::$identity == NULL){

   return false;}
```

```
     return true;}

function isLoggedIn() {

   return self::hasIdentity();}
```

Bez přihlášení nemůžete zakládat projekty, prohlížet chyby, události a tím pádem modifikovat jakoukoliv činnost. O udržení spojení (relace), stavu, se stará metoda `session_start()`, která je součástí jazyka PHP.

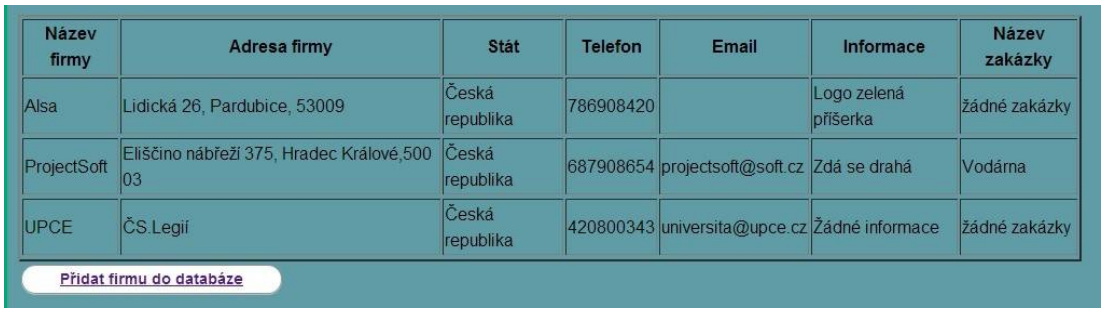
4.2.3 Evidence firem

Každá zakázka, kterou firma v mé aplikaci dostane, je zadána nějakou firmou. Firmu může založit manager, který se stará o zakázky (tedy o ekonomický přínos firmy).

```
function vytvoreniFirmy($nazevFirmy, $adresa, $psc, $mesto,
$stat, $telefon, $email, $informace) {

$db = new Oracle();

$celaAdresa = $adresa . ", " . $mesto . ", " . $psc;

$r = $db->sql("INSERT INTO FIRMA (NAZEV_FIRMY, ADRESA_FIRMY,
STAT, TELEFON, MAIL, INFORMACE)         VALUES ('$nazevFirmy',
'$celaAdresa', '$stat', '$telefon', '$email', '$informace')");

     if (!$r) {

    print_r($db->getErrorMsg());

        }

        $db->close(); }
```

Obrázek 15: Přidání firmy do databáze. Zdroj: vlastní

Přehled všech firem, jejichž zakázky jsou evidovány v databázi, mohou uživatelé vidět bez omezení (musí být přihlášeni do systému).



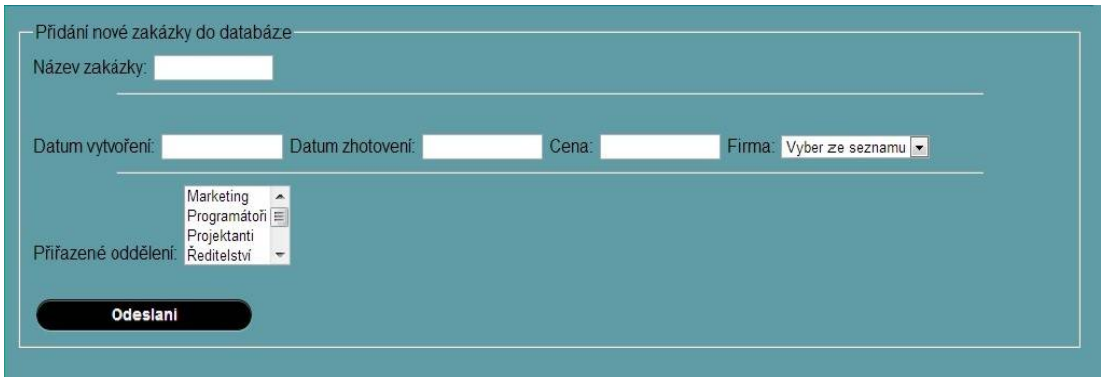
Název firmy	Adresa firmy	Stát	Telefon	Email	Informace	Název zakázky
Alsa	Lidická 26, Pardubice, 53009	Česká republika	786908420		Logo zelená příšerka	žádné zakázky
ProjectSoft	Eliščino nábřeží 375, Hradec Králové,500 03	Česká republika	687908654	projectsoft@soft.cz	Zdá se drahá	Vodárna
UPCE	ČS.Legií	Česká republika	420800343	universita@upce.cz	Žádné informace	žádné zakázky

Přidat firmu do databáze

Obrázek 16: Přehled firem v databázi. Zdroj: vlastní

4.2.4 Evidence zakázek

Manager, který přiděluje zakázky jednotlivým oddělením, je jediný, který může zakázku přiřadit. Výpis však mohou opět prohlížet všichni zaměstnanci všech oddělení, protože mohou zjišťovat, které oddělení plní stejnou zakázku. Podle mě to je praktické třeba pro plánování firemních cest při vyřizování zakázky.



Obrázek 17: Přidání firmy do databáze. Zdroj: vlastní

Manager po vyplnění všech základních údajů týkajících se zakázky, přiděluje oddělení, které bude na této zakázce pracovat. Splnění zakázky může být přiděleno více oddělením. Pro výběr více položek slouží menu `select` s atributem `multiple`. Příklad využití tohoto příkazu implementuji v následujícím kódu:

```
<label>Přiřazené oddělení:</label>

<?php

$dotaz2 = $db->sql("select nazev_oddeleni from oddeleni

order by nazev_oddeleni");

echo "<select name='oddeleni[]'multiple> ";

foreach ($dotaz2 as $user) {

echo "<option value=" . $user['NAZEV_ODDELENI'] . ">" .
$user['NAZEV_ODDELENI'] . "</option>";

}

echo "</select>";

$db->close(); ?>
```

K malému vysvětlení tohoto kódu. Do proměnné `$dotaz2` se uloží výstup z SQL příkazu na vypsání všech oddělení uložených v databázi a seřazených dle názvu.

Příkaz `foreach()` slouží procházení jednotlivých výpisů. K vypsání slouží příkaz `echo`.

Výpis všech zakázek je vidět na obrázku č. 18.

Název zakázky	Datum vytvoření	Datum zhotovení	Cena zakázky	Firma	Zpracovávající oddělení
Vodárna	14.07.13	02.10.13	200000	ProjectSoft	Programátoři
Vodárna	14.07.13	02.10.13	200000	ProjectSoft	Projektanti

Přidat zakázku do databáze

Obrázek 18: Výpis zakázek v databázi. Zdroj: vlastní

4.2.5 Události

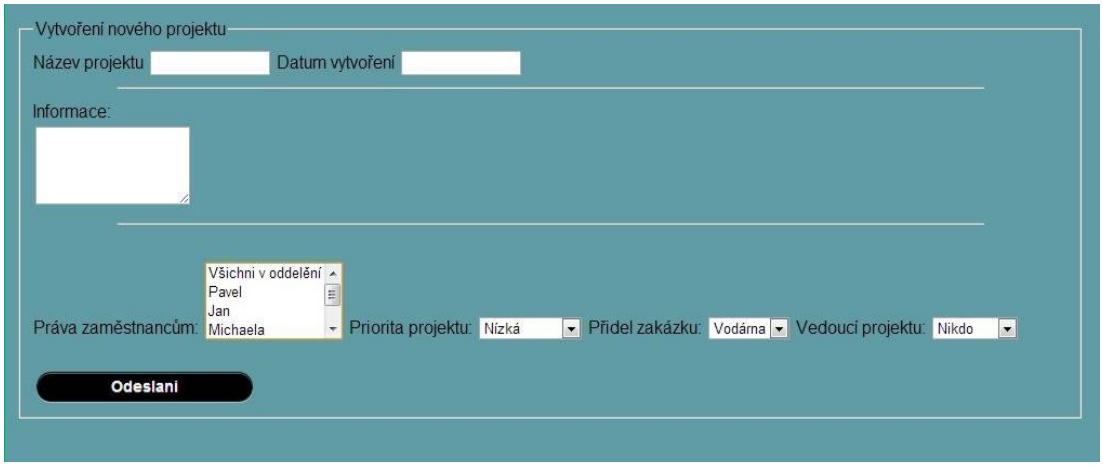
Stránka s událostmi je stránka, která informuje uživatele o posledních změnách, které se prováděly na projektech, na kterých spolupracuje. Tzn., že uživatel by zde měl být informován o úkolech, které mu byly zadány. Také o chybách, kterých se dopustil a samozřejmě by se měl dozvědět o projektu, na kterém bude spolupracovat.

Obrázek 19: Úvodní obrazovka. Zdroj: vlastní

Tato stránka je pouze informativní, proto jsem odkaz na ni nechal jen na hlavní (domovské) stránce.

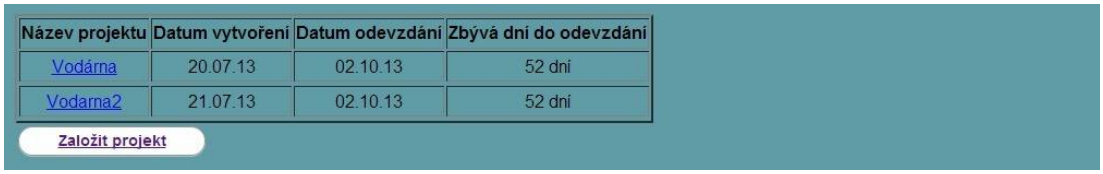
4.2.6 Projekty

Založit projekt může každý uživatel přihlášený do aplikace. Založení projektu obrázek č. 20.



Obrázek 20: Vytvoření nového projektu. Zdroj: vlastní

Uživatel, vytvářející nový projekt, určí název projektu, datum vytvoření a informace, které však nejsou povinné. Dolní část obrázku určuje práva zaměstnancům firmy. Zaměstnanec, kterému není přidělena role `Spoluprojektant`, nemá žádná práva k tomuto projektu. Tento projet neuvidí v seznamu projektů (obrázek č. 21). Dále uživatel, určuje prioritu projektu tzn., důležitost. Každý projekt by měl patřit i nějaké zakázce, která patří do jeho oddělení (zaměstnanec nemůže přiřadit projekt zakázce, která není přidělena jeho oddělení). Vedoucím projektu může být zaměstnanec jiný, než který projekt vytvořil. Zde je přidělena další role k projektu a to `vedoucí projektu`, ten však nemusí být určen.

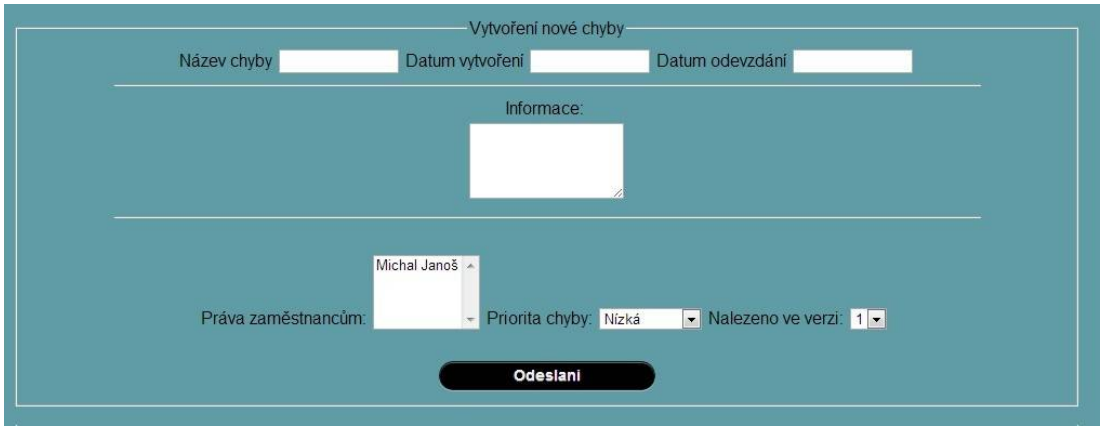


Obrázek 21: Výpis projektů. Zdroj: vlastní

Každý projekt, na kterém uživatel pracuje, je možné vidět ve výpisu projektů. Pokud uživatel hodlá pracovat na nějakém projektu, klikne na odkaz pro další informace.

4.2.7 Chyby v projektu

Při řešení různých projektů a úkolů vznikají chyby, nedostatky. Některé části projektu jsou na sobě závislé, tudíž se i chyby hledají snáze. Uživatel, který má práva k nějakému projektu, může přidělovat chyby k projektům a k úkolům. Chybu přidělí zaměstnanci, který je za ni zodpovědný, nebo je k ní přidělen z jiného důvodu.



Obrázek 22: Vytvoření chyb. Zdroj: vlastní

Chybu zaměstnanec může nalézt i ve starší verzi, než na které momentálně pracuje, takže pro přesné určení chyby může poukázat na verzi, kde byla objevena.

5. Závěr

V této práci byly vysvětleny některé důležité aspekty pro používání aplikací, které slouží pro podporu uživatelů v jednotlivých týmech řešících softwarové projekty.

V teoretické části jsem poukázal na známé či méně známé webové aplikace, které si může každý uživatel vyzkoušet sám nebo v týmu s jinými uživateli. Čtenář se dozví některé důležité informace o open-source či komerčních aplikacích, jejich výhody a nevýhody. Dále práce popisuje možné útoky na databázi aplikace skrze webová rozhraní a snaží se čtenáře nasměrovat k možným řešením těchto bezpečnostních mezer.

V praktické části bylo cílem navrhnout a implementovat vlastní aplikaci, která by mohla být využita vývojářským týmem. Je rozdělena na dvě části. První část popisuje tvorbu databáze s propojením tabulek a objekty. Druhá část se zabývá popisem webové aplikace a její funkčností dle mých funkčních požadavků.

Z teoretické části vyplývá, že firma, či tým o více uživatelích, by měla využívat aplikaci, která je efektivní, přehledná a zjednodušuje spolupráci na projektech v týmu. Je důležité, aby vývojář aplikace zajistil bezpečnost dat na straně databázového systému dříve, než se na databázový systém pokusí útočník zaútočit.

K praktické části bych chtěl dodat, že webová aplikace není dokončena a pro plné použití je třeba dopracovat některá propojení a pro bezpečné použití i některá autorizační a autentizační práva.

6. Literatura

[1] Co je týmová spolupráce. In: Metodický portál [online]. 2012 [cit. 2013-08-07]. ISSN 1802-4785. Dostupné z: http://clanky.rvp.cz/wpcontent/upload/prilohy/2755/kooperace.pdf

[2] Co umí moderní aplikace pro týmovou spolupráci. In: ŠNAJDR, Jaroslav. [online]. 1. vyd. Praha: Dharma Gaia, 1999, 03.01.2012 [cit. 2013-08-07]. ISSN 1805-5486. Dostupné z: http://www.ictmanazer.cz/2012/01/co-umi-moderni-aplikace-pro-tymovou-spolupraci/

[3] ROCHELT, Karel. Integrace systému pro podporu týmové spolupráce s aplikací SWINPRO. Praha, 2011. Dostupné z: https://dip.felk.cvut.cz/browse/pdfcache/rochekar_2011bach.pdf. Bakalářská práce. České vysoké učení technické v Praze. Vedoucí práce Ing. Jiří Mlejnek.

[4] NOSKA, Martin. Tipy na nástroje pro podnikovou týmovou spolupráci [online]. 2012 [cit. 2013-08-12]. ISSN 1805-5486. Dostupné z: http://www.ictmanazer.cz/2012/04/tipy-na-nastroje-pro-podnikovou-tymovou-spolupraci/

[5] ŠÚST, Ján. E-learning pro studenty se specifickými nároky: kompletní průvodce tvorbou a správou elektronických kurzů [online]. 1. vyd. Brno: Computer Press, 2006, s. 1 [cit. 2013-08-12]. ISBN moodle.org.

[6] LECKY-THOMPSON, Ed a Steven D NOWICKI. PHP 6: programujeme profesionálně. Vyd. 1. Brno: Computer Press, 2010, s. 228. Programujeme profesionálně. ISBN 978-80-251-3127-5.

[7] VANČURA, Martin. Webdesigncity. VANČURA, Martin. Http://www.webdesigncity.cz/ [online]. [cit. 2013-08-12]. Dostupné z: http://www.webdesigncity.cz/

[8] CASTRO, Elizabeth. HTML, XHTML a CSS: názorný průvodce tvorbou WWW stránek. Vyd. 1. Brno: Computer Press, 2007, 438 s. ISBN 978-80-251-1531-2.

[9] LACKO, Ľuboslav. 1001 tipů a triků pro SQL. Vyd. 1. Brno: Computer Press, 2011, 416 s. ISBN 978-80-251-3010-0.

[10] HRDINA, Luděk. Deset hlavních slabin webových aplikací. Http://www.systemonline.cz/ [online]. 2005 [cit. 2013-08-12]. Dostupné z: http://www.systemonline.cz/clanky/deset-hlavnich-slabin-webovych-aplikaci.htm

[11] MALÝ, J. a J. KACÁLEK. Zabezpečení webových aplikací I: klientské skriptovací jazyky. Access.feld.cvut.cz[online]. Brno, 2007 [cit. 2013-08-12]. Dostupné z: http://access.feld.cvut.cz/view.php?cisloclanku=2007090001

[12] SQL Injection a zabezpečení. In: VOJÁČEK, Petr. Programujte.com [online]. 2007 [cit. 2013-08-12]. Dostupné z: http://programujte.com/clanek/2007041802-sql-injection-a-zabezpeceni/