UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

Systém evidence a řízení práce ve firmě

Tomáš Rychlík

Diplomová práce
2013

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

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Tomáš Rychlík**
Osobní číslo:	**I11408**
Studijní program:	**N2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Systém evidence a řízení práce ve firmě**
Zadávající katedra:	**Katedra softwarových technologií**

Zásady pro vypracování:

Cílem této práce je vytvoření funkční aplikace s optimalizovanými SQL dotazy v databázovém systému, která bude obsahovat nástroje sloužící pro evidenci a řízení práce ve firmě.
Aplikace bude umožňovat tyto funkcionality:
1. Registraci uživatelů systému.
2. Přístup uživatelů do systému podle jejich práv.
3. Evidenci řešitelů a spoluřešitelů s možností znovuotevření již vyřešeného úkolu.
4. Evidenci vnitropodnikových dokumentů, příkazů a směrnic.
5. Evidenci pracovních úkolů, jejich termínů, časovou náročnost a evidenci pracovníků zodpovědných za úkol.
6. Generování harmonogramu průběhu řešení.
7. Automatická kontrola důležitých termínů dle zadaných parametrů.
8. Generování sestav dle zadaných kritérií pro potřeby řešitelů a vedoucích pracovníků.
9. Komunikace s řešiteli a uživateli systému prostřednictvím mailů a SMS zpráv prostřednictvím bran se sítěmi typu GSM, ICQ, Google Talk, Jabber a další.

V úvodní části je nutno provést rešerši stávajících SW řešení pro evidenci a řízení práce ve firmě. Rešerši je nutné doplnit o porovnání s nově navrhovaným systémem, který bude předmětem této práce. Úvodní část musí obsahovat analýzu navrhovaného řešení, která bude obsahovat popis použitých technologií, návrh databáze a aplikačního řešení. Pro vytvoření aplikace bude využit skriptovací jazyk PHP nebo JAVA a databáze Oracle nebo MySQL. V práci bude navržen postup pro provedení optimalizace vybraných SQL dotazů s následnou analýzou a porovnáním výkonu nalezených alternativ exekučních plánů s původním dotazem.

Rozsah grafických prací:

Rozsah pracovní zprávy:

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

Seznam odborné literatury:

1. LACKO, Luboslav. Oracle. Správa, programování a použití databázového systému. Brno: Computer Press, 2007.
2. GROFF, James R. a WEINBERG, Paul N. SQL kompletní průvodce. Brno: Computer Press, 2005.
3. VRÁNA, Jakub. 1001 tipů a triků pro PHP. Brno: Computer Press a.s., 2010.
4. BRYLA, Bob; LONEY, Kevin. Mistrovství v Oracle Database 11g. Jiří Huf. Vydání první. Brno: Computer Press, a.s., 2009.
5. NOWICKI, Steven D. a LECKY-THOMSON, Ed. PHP 6. Programujeme profesionálně. Brno: Computer Press, 2009.
6. DRUSKA, P. CSS a XHTML - tvorba dokonalých webových stránek krok za krokem, Grada 2006.
7. LONEY, K., THERIAULT, M. Mistrovství v Oracle. Praha, Computer Press, 2002.
8. www.oralce.com

Vedoucí diplomové práce: **Ing. Miloslav Macháček, Ph.D.**
Katedra informačních technologií

Datum zadání diplomové práce: **31. října 2012**
Termín odevzdání diplomové práce: **17. května 2013**

Univerzita Pardubice Česká republika

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

prof. Ing. Antonín Kavička, Ph.D.
vedoucí katedry

V Pardubicích dne 15. listopadu 2012

Prohlášení autora

Prohlašuji, že jsem tuto práci 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 20. 08. 2013 Tomáš Rychlík

Poděkování

Chtěl bych velmi poděkovat vedoucímu mé diplomové práce Ing. Miloslavu Macháčkovi, Ph.D. za veškerou odbornou pomoc a cenné rady při zpracovávání mé diplomové práce.

Děkuji mé ženě Mgr. Alici Rychlíkové a jejím rodičům a také mým rodičům za veškerou psychickou a hmotnou podporu.

Anotace

Cílem diplomové práce je vytvořit plně funkční aplikaci, s optimalizovanými SQL dotazy v databázovém systému, pro evidenci a řízení práce ve firmě. Součástí práce je rešerše stávajících řešení a jejich porovnání s nově vytvořeným systémem. V teoretické části se také zaměřím na popis skriptovacího jazyka PHP a dotazovacího jazyka SQL, které jsem použil pro vytvoření aplikace. Součástí je popis návrhových vzorů použitých v systému. Dále je popsána databázová struktura s ukázkou vybraných SQL dotazů a příkladů exekučních plánů a jejich optimalizace s využitím nástroje Oracle SQL Developer. V poslední části se zaměřím na samotný popis praktické části mé diplomové práce, jejího zpracovávání a zmíním také některé zajímavé problémy, se kterými jsem se během vývoje setkal.

Klíčová slova

PHP, MySQL, systém, evidence, řízení, práce, firma, ERP, NetBeans

Title

System of accounting and control work in the company.

Annotation

The objective of the thesis is to create a fully functional application with optimized SQL queries in a database system for the registration and management work in the company. The thesis includes research of existing solutions and their comparison with the new created system. The theoretical part will also focus on the description of the scripting language PHP and SQL query language which I used to create the application. The thesis also includes the description of the design patterns used in the system. In the next part there are descriptions of the database structure with examples of selected SQL queries and execution plans and optimizing with the use of Oracle SQL Developer. In the final part I will concentrate on the actual description of the practical part of the thesis itself and the processing. I will mention certain interesting problems which I have analyzed.

Keywords

PHP, MySQL, system, accounting, control, work, company, ERP, NetBeans

Obsah

Seznam zkratek

API	Application Programming Interface
CSS	Cascading Style Sheets
CSV	Comma-Separated Values
DBMS	Database Management Systém
DOM	Document Object Model
ERP	Enterprise Resource Planning
GIF	Graphics Interchange Format
GPL	General Public License
GUI	Graphical User Interface
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IDE	Integrated Development Environment
ISAPI	Internet Server Application Programming Interface
JPEG	Joint Photographic Experts Group
JSP	Java Server Pages
OOP	Object Oriented Programming
PHP	Hypertext Preprocessor (Personal Home Page tools)
PNG	Portable Network Graphics
RAC	Real Application Clusters
SGML	Standard Generalized Markup Language
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SSL	Secure Sockets Layer
UML	Unified Modeling Language
URL	Uniform Resource Locator
XHTML	Extensible HyperText Markup Language
XML	Extensible Markup Language
XSL	Extensible Stylesheet Language

Seznam obrázků

Seznam tabulek

Úvod

Cílem práce je vytvoření systému, který řeší problematiku řízení práce ve firmě. Takový systém spadá do oblasti podnikových informačních systémů. Jedná se v postatě o soubor nástrojů pro efektivní správu: zaměstnanců pracujících ve firmě, obdržených zakázek, zpracovávaných projektů a úkolů ve firmě, vedení účetnictví, správu dalších podnikových zdrojů a podobně.

V první části této práce jsou jednotlivé firemní systémy a jejich rozdělení popsány podrobněji. Součástí je detailní popis vybraných stávajících řešení a jejich porovnání s nově navrženým systémem, který je předmětem praktické části této práce.

Ve druhé části jsou popsány jednotlivé použité technologie. Systém je napsán ve skriptovacím jazyce PHP, značkovacím jazyku HTML a využívá databázi MySQL. Uvedené technologie jsou proto popsány podrobně. Součástí kapitoly je i popis vývojových prostředí a softwarových nástrojů. Na závěr kapitoly je uveden popis některých programovacích technik a popis vybraných návrhových vzorů použitých v aplikaci.

Třetí část je zaměřena na popis databáze použité v aplikaci. Součástí je popis hlavních tabulek systému. Dále jsou popsány techniky exekučních plánů a jejich využití pro optimalizaci SQL dotazů použitých v aplikaci.

Poslední část se zabývá detailním popisem samotné aplikace. Popsány jsou její důležité části a grafická podoba. Součástí je také popis realizace případů užití a vytvoření adresářové struktury aplikace. V závěru kapitoly jsou zmíněny také některé zajímavé problémy, které bylo nutné vyřešit.

1 Systémy pro řízení práce ve firmě

Informačních systémů pro firmy existuje celá řada. Z tohoto důvodu se podnikové systémy rozdělují do množství kategorií, například z hlediska zaměření systému nebo velikosti firmy, pro kterou je systém použit. Tyto kategorie je možné různě kombinovat a tak postihnou různé odvětví podnikání. Některé kategorie jsou také určitým způsobem těsně provázané, takže systémy pro řízení projektů obsahují často také modul pro řízení lidských zdrojů nebo modul pro vedení faktur a evidenci účetnictví.

1.1 Rozdělení podnikových informačních systémů

1.1.1 ERP systémy

Jedná se o nejvíce obecnou kategorii podnikových systémů. Spojuje dohromady různá podniková odvětví a jejich správu. ERP tak může zajišťovat řízení činnosti podniku, systémy výroby, správu zásob, skladů a logistiku, prodej, vedení účetnictví, správu personalistiky a lidských zdrojů, evidenci obchodních partnerů, e-business a podobně.

U většiny ERP systémů je rozhodující velikost a typ firmy, pro kterou je systém požadován. Jiné nároky na systém bude mít malá firma zaměřující se na logistiku a dopravu v porovnání s nadnárodní společností prodávající oblečení. Z tohoto hlediska je důležitá také kastomizace, což je proces, při kterém je software upraven podle potřeb konkrétního zákazníka. Ten pak nemusí platit za zbytečné množství modulů v systému, které ve výsledku nikdy nepoužije.

Vybrané příklady ERP systémů:

- ABRA G4[1]
- Altus Vario[2]
- BYZNYS ERP[3]
- IMES ERP[4]
- myWAC[5]
- NET Genium ERP[6]
- QI[7]
- SAP[8]
- Smart4Web ERP[9]

[1] http://www.abra.eu/informacni-systemy/abra-g4
[2] http://www.altus.cz/altus-vario/
[3] http://www.jkr.cz/byznys-erp/popis-systemu
[4] http://www.softok.cz/modul/informacni-system-erp-imes
[5] http://www.mywac.cz/menu/system-mywac
[6] http://www.netgenium.com/cs/produkty/erp-rizeni-podniku
[7] http://www.qi.cz/prohlidka-systemu/
[8] http://www.sap.com/cz/index.epx
[9] http://www.smart4web.cz/aplikace-erp/

1.1.2 Business intelligence

Systémy Business Intelligence (česky také Manažerský informační systém) zpracovávají historická a aktuální podniková data a jejich cílem je zvýšení výkonnosti podniku, podpora plánování a rozhodování a predikce dat. Systémy využívají často datových skladů nebo produkčních systémů k získání informací, které mohou dále upravovat nebo transformovat do podoby vhodné pro vedoucí management podniku. Součástí systému jsou často také nástroje pro modelování a simulaci situací, které by mohly v podniku nastat a jejich následnou analýzu.

Aplikace Business Intelligence jsou většinou rozsáhlé, komplexní systémy, a proto bývají častěji nasazeny až ve středních nebo velkých firmách. Pro malé firmy je investice do takového systému často příliš nákladná.

Vývojem těchto systémů se samozřejmě zabývá řada velkých SW firem včetně Oracle, IBM nebo Microsoft. Příklady těchto systémů jsou následující:

- BI SMART Vario[10]
- Business Intelligence BI[11]
- IBM Cognos Business Intelligence[12]
- Microsoft Power BI[13]
- Oracle Business Intelligence[14]
- POHODA Business Intelligence[15]

1.1.3 Systémy pro řízení projektů

Cílem těchto systémů je především správa a řízení projektů, evidence pracovních úkolů a spolupracovníků přiřazených k projektu. Často se označuje zkratkou PPM – Project and Portfolio Management. Tyto systémy jsou nejčastěji využívány především v oblasti služeb a ve firmách zaměřených na projektové zakázky a servisní služby. Hlavním cílem je efektivní využití lidských a materiálových zdrojů, kterými firma disponuje.

Vybrané příklady systémů pro řízení projektů:

- Atollon Lagoon[16]
- CS Projekt[17]
- Dynamica Ready[18]
- GroupCamp Project[19]

[10] http://www.bismart.cz/
[11] http://web.ortex.cz/produkty/business-intelligence/
[12] http://www-03.ibm.com/software/products/cz/cs/business-intelligence/
[13] http://www.microsoft.com/en-us/bi/default.aspx
[14] http://www.oracle.com/cz/solutions/business-intelligence/index.html
[15] http://www.stormware.cz/pohoda/business-intelligence/
[16] http://www.atollon.cz/lagoon/rizeni-projektu
[17] http://www.ceskysoftware.cz/produkty_a_sluzby/cs_projekt/
[18] http://www.dynamica.cz/dynamica-ready
[19] http://www.groupcamp.cz/online-software-pro-rizeni-projektu

- Instant Team[20]
- Primavera P6 Professional Project Management[21]
- Projet Professional[22]

1.1.4 Systémy pro kontrolu a evidenci docházky

Tyto systémy se nejčastěji kombinují s fyzickými identifikačními médii (bezkontaktní čipy, ID karta, zařízení snímající otisk prstu). Nemusí se tedy vždy jednat o čistě SW řešení, ale o kombinaci HW+SW řešení. Cílem těchto systémů není jen evidence příchodů a odchodů zaměstnanců, ale často také i monitoring v reálném čase, evidence přesčasů nebo zpracování statistických údajů o měsíční odpracované době.

Příklady těchto systémů:

- Aktion.ONE[23]
- CONET[24]
- DOORIS[25]
- ID-WARE[26]
- IKOS D3[27]
- PowerKey[28]

1.1.5 Systémy pro řízení lidských zdrojů

Anglicky se tyto systémy označují zkratkou HRM (Human Resource Management). Cílem je především správa a evidence pracovníků. Aplikace HRM často nabízejí nástroje pro výběr nových pracovníků (generované testy), rozdělování odměn, hodnocení pracovníků nebo řízení vzdělávání a rozvoj stálých zaměstnanců.

Tyto systémy bývají také provázány s jinými typy systémů. Součástí systému HRM může být i evidence pracovních úkolů zaměstnanců nebo napojení na mzdové systémy nebo systémy pro kontrolu a evidenci docházky.

Příklady těchto systémů:

- Eleanor[29]
- HR info[30]

---

[20] http://www.instant-team.com/
[21] http://www.oracle.com/us/products/applications/primavera/p6-professional-project-management/overview/index.html
[22] http://office.microsoft.com/cs-cz/project-help/project-professional-2013-sprava-projektoveho-portfolia-FX103797571.aspx
[23] http://www.dochazkaonline.cz/
[24] http://www.dochazkovysystem.cz/
[25] http://www.gacc.cz/dochazkovy-system/
[26] http://www.id-karta.cz/aplikace-1/dochazkovy-system-4/
[27] http://www.ikos.cz/cz/dochazkove-systemy.html
[28] http://www.advent.cz/produkty/software-powerkey/dochazkovy-system/
[29] http://www.elanor.cz/index.php?id=software
[30] http://www.mmi.cz/cs/produkty-detail/hr-info

- HR Vema[31]
- myWORK[32]
- PERM 3[33]
- PERMIS[34]
- TARGET 2100[35]

1.1.6 Systémy pro řízení a plánování výroby

Do této kategorie spadají systémy určené pro výrobní podniky. Zaměření těchto systémů je především na optimalizaci výrobních procesů, nákladů, plánování kapacity a vytíženosti strojů a skladů, plnění pracovních plánů a jejich optimalizace, evidence výrobků a podobně.

Tyto systémy bývají často propojeny s ekonomickými systémy a systémy pro vedení projektů a zakázek. Často tyto systémy poskytují také modelovací nástroje pro odhad budoucích plánů výroby nebo reakce na nepředvídatelné události ve výrobním procesu a jejich dopadu na produkci.

Vybrané příklady systémů pro řízení a plánování výroby:

- AROP[36]
- Dynamic Manufacturing[37]
- HERAKLES[38]
- Patriot[39]
- Platune[40]
- RIIS+[41]
- RSV – Řízení stavební výroby[42]
- SG Enterprise[43]

1.1.7 Systémy pro řízení vztahu se zákazníky

Tyto systémy jsou označovány také zkratkou CRM (Customer Relationship Management) a jedná se o proces získávání, zpracovávání a využívání informací o zákaznících firmy. Součástí jsou také nástroje pro podporu marketingu a prodeje, PR a vztahu k zákazníkům, správu prodejních kampaní, podporu zákaznických služeb a nástroje pro samotnou komunikaci se zákazníky.

---

[31] http://www.vema.cz/default.aspx?categoryID=Produkty.2
[32] http://www.kvados.cz/Content/myWORK-Work-Management
[33] http://www.kvasar.cz/24729-personalistika-mzdy-perm3
[34] http://www.altec.cz/podnikove-informacni-systemy/permis-mzdy-a-personalistika/
[35] http://www.m-pro.cz/index.php?s=target-2100
[36] http://www.arsiqa.cz/arop/
[37] http://www.allium.cz/CZ/reseni/Stranky/manufacturing.aspx
[38] http://www.drings.cz/software/herakles/
[39] http://www.datapartner.cz/podpora-vyroby/
[40] http://www.insophy.cz/cz/aps
[41] http://www.molo.cz/asw/
[42] http://www.firstis.cz/index.php/cs/rsv-rizeni-stavebni-vyroby/seznameni-s-rsv
[43] http://www.synergit.cz/software-pro-rizeni-vyroby/

Cílem těchto systémů je často získávání informací o chování, potřebách a přáních zákazníků firmy. Díky tomu lze vytvářet další obchodní příležitosti zaměřené na aktuální potřeby těchto zákazníků.

Pomocí detailních informací o zákaznících a jejich chování lze v systémech CRM podrobně určit jeho solventnost, rizikovost nebo odhadnout sekvence nákupů. Systémy umožňují také rozdělení zákazníků podle různých hledisek, například podle demografických údajů nebo sociálního hlediska.

Je patrné, že problematika CRM systémů je velmi rozsáhlá. Proto se tyto systémy často rozdělují na další, více specifické systémy, například:

- **Operativní CRM** („front office“) nejčastěji určené pro řízení marketingu a tvorbu marketingových kampaní.
- **Kooperační CRM** zaměřená především na komunikaci se zákazníky s cílem zvýšení kvality poskytovaných služeb.
- **Analytické CRM**, které slouží k predikci (například při tvorbě cen), tvorbě analýz, podpoře rozhodování a hledání nových prodejních kanálů.

Příklady CRM systémů jsou následující:

- BLUEJET[44]
- CRM2Host[45]
- CRM Compsale[46]
- CRM Leonardo[47]
- eWay-CRM[48]
- Microsoft Dynamics CRM[49]
- Oracle Siebel CRM[50]

1.1.8 Systémy pro řízení podnikového obsahu

Jedná se o systémy ECM (Enterprise Content Management) a jejich cílem je efektivně spravovat veškerá podniková data v elektronické podobě (dokumenty, obrázky, video, audio, fotografie, apod.). Tyto systémy zajišťují sběr a získávání dat, jejich organizaci a třídění, zabezpečení a případnou likvidaci, archivaci, verzování, tisk nebo předpřipravení pro jejich prezentaci. Přístup k dokumentům je často určován podle uživatelských rolí nebo nastavených oprávnění.

[44] http://www.bluejet.cz/
[45] http://www.crm2host.cz/crm.php
[46] http://www.timone.cz/cz/crm-compsale
[47] http://www.d3soft.cz/crm-system.html
[48] http://www.eway-crm.cz/co-je-eway-crm/o-eway-crm
[49] http://crm.dynamics.com/cs-cz/home
[50] http://www.oracle.com/us/products/applications/siebel/overview/index.html

1.1.9 Systémy pro vedení účetnictví a financí

Tyto systémy jsou většinou tvořeny funkčně specifickými podsystémy. Obsahují tedy modul pro vedení účetnictví a k tomu podle potřeby další z následujících systémů:

- Systémy daňové evidence.
- Systémy pro vystavování a evidenci faktur.
- Systémy pro evidenci knih jízd a správu vozového parku.
- Systémy podvojného účetnictví.
- Evidence smluv, norem a zákonů.
- Správa a přehled investic.

Příklady těchto systémů:

- ARBES FEIS[51]
- ATMA[52]
- AZ.PRO[53]
- CubiQ[54]
- DIALOG 3000S[55]
- Fakturka[56]
- Fakturace 3[57]
- PREMIER[58]
- WinFas[59]

1.2 Zaměření navrhovaného systému

Cílem nově navrhovaného systému je:

- evidence práce, projektů, úkolů,
- evidence a správa zaměstnanců a spolupracovníků,
- evidence podnikových dokumentů.

Z těchto tří hledisek je patrné, že navrhovaný systém částečně spadá do těchto kategorií:

- Systémy pro řízení projektů.
- Systémy pro řízení lidských zdrojů.
- Systémy pro řízení podnikového obsahu.

[51] http://www.arbes.com/uvod/produkty/arbes-feis/
[52] http://www.atma.cz/default.htm
[53] http://www.itp.cz/is-az-pro.html
[54] http://www.hsf-software.cz/index.php/cz/cubiq-produkty
[55] http://www.control.cz/index.php?path=produkty/dialog3000
[56] http://www.fakturka.cz/
[57] http://www.faktury.cz/
[58] http://www.premier.cz/cs/produkty_popis.asp
[59] http://www.winfas.cz/index.php?clanek=772-informacni-system-prezentace-o_systemu

Systém je navržen jako on-line webová aplikace a určená je především pro menší a střední firmy. Z hlediska funkcionality existují v systému následující (nejdůležitější) moduly a části:

- Správa projektů,
- Správa úkolů,
- Generování sestav,
- Harmonogramy průběhu,
- Správa uživatelů,
- Interní komunikace,
- E-mailová komunikace,
- Správa dokumentů,
- Základní správa účetnictví,
- Evidence obchodních kontaktů,
- Kalendář akcí,
- Modul statistik.

Tyto vybrané moduly a vlastnosti budou sloužit pro porovnání s existujícími systémy což je předmětem následující kapitoly. Porovnávané existující aplikace, které se navrhovanému systému nejvíce podobají, jsou:

- GroupCamp Project,
- Atollon Lagoon,
- WorkSheetPro,
- WorkWatch,
- Instant Team.

1.3 Porovnání systému se stávajícím řešením

1.3.1 GroupCamp Project

Jedná se o on-line webovou aplikaci pro řízení projektů zaměřenou na malé a střední firmy. Jde o aplikaci soukromé firmy původem z Francie. Zakladatelé projektu jsou Dickel Sooriah a Eric Hassid. Aplikace je nabízena ve více jazykových variantách, včetně češtiny.

V systému je možné naplánovat a vytvářet projekty a u nich spravovat jednotlivé milníky, klíčové vlastnosti a projektové aktivity. V projektu je možné podle potřeby přidávat úkoly. U úkolů lze nastavit prioritu, evidovat odhadovaný čas trvání úkolu i čas reálně strávený u řešení úkolu. Pro vedoucí pracovníky poskytuje systém nástroje pro vyhodnocování úkolů, vytváření hlášení a výkazů za všechny členy týmu. Systém je hodně orientovaný na týmovou spolupráci. Aplikace také poskytuje nástroje pro správu a sdílení souborů. U těch je možná správa verzí nebo sledování komentářů na sdílených souborech. Součástí systému je také možnost archivace, psaní poznámek nebo synchronizovat systém s některými aplikacemi třetích stran.

Systém je objednáván variantou měsíčního předplatného. Plnou verzi systému (základní varianta) je možné pořídit za 589 Kč / měsíc. Nejlepší varianta systému je nabízena za 4 314 Kč / měsíc.

Výhody systému

- Neomezený počet spolupracovníků v systému.
- Třicetidenní zkušební přístup zdarma pro všechny verze systému.
- Velmi dobrá synchronizace s aplikacemi Google Apps a Gmail, Microsoft Office a Apple Mail.
- Export do Excelu a CSV.
- Všechny datové přenosy chráněny 256 bitovým SSL šifrováním.
- On-line uživatelská podpora, webové konference a školení.

Nevýhody systému

- Vysoké měsíční poplatky za používání systému (systém je pouze pronajímán za měsíční poplatky).
- Pokročilá nastavení a manažerské role pouze u dvou nejdražších verzí systému.
- Omezené počty projektů a úkolů.
- Podmínky používání zatím pouze v angličtině.

Následující tabulka (Tabulka 1) porovnává vlastnosti a moduly navrhovaného systému s podobnými moduly v systému GroupCamp Project.

Vlastnosti a moduly	**GroupCamp Project**
Správa projektů	Ano, omezeno na 15 – 200 projektů (podle verze)
Správa úkolů	Ano, určitá omezení u levnějších verzí
Generování sestav	Řešeno exportem do Excelu a CSV
Harmonogramy průběhu	Ne
Správa uživatelů	Ano, neomezený počet
Interní komunikace	Možnost interní diskuze
E-mailová komunikace	Ano
Správa dokumentů	Ano, navíc oddělená správa odkazů
Správa účetnictví	Ne
Evidence obchodních kontaktů	Ne
Kalendář akcí	Ano, součástí každého projektu
Modul statistik	Ano

Tabulka 1 - Porovnání systému GroupCamp

1.3.2 Atollon Lagoon

Tento systém je tvořen základním jádrem (Lagoon Core) a to je možné rozšířit o konkrétní implementaci podle potřeb firmy. Součástí je také sada nástrojů pro kompletní řízení projektů (Lagoon Project Management). Systém pochází od firmy ATOLLON LIMITED

se sídlem v Londýně. Aplikace je nabízena v několika jazykových variantách, včetně češtiny.

Lagoon Project Management nabízí nástroje pro plánování rozsáhlých projektů, evidenci a rozdělování úkolů a vedení výkazů práce vzhledem k danému projektu. Úkoly je možné připomínkovat a také je možné nastavovat neomezená upozornění na úkoly. Součástí systému je také modul pro řízení zdrojů. Komunikace je zajištěna formou e-mailů a interních komunikačních nástrojů. Systém je také zaměřen na tvorbu výkazů práce, reportů a generování sestav. Součástí je také správa a organizace firemních dokumentů.

Systém je možné objednat jako službu (on-line) nebo implementovat v rámci firmy, která systém poptává. Cena při tomto řešení je sestavena vždy až podle konkrétních požadavků klienta. On-line varianta je tvořena měsíčním poplatkem za provoz služby (včetně všech funkcí v systému) ve výši 1 200 Kč / měsíc.

Výhody systému

- Vyzkoušení systému na 31 dní zdarma.
- Komfortní grafické uživatelské rozhraní (intuitivní, kontextová menu, podpora drag&drop ovládání).
- Systém je plně multiplatformní.
- Možnost vytvořit si vlastní balíček funkcí.
- Důraz na bezpečnost (pokročilé zabezpečení aplikačními právy, SSL přenos).

Nevýhody systému

- V ceně systému není zahrnuta podpora produktu (nutné doobjednat zvlášť) a konzultační služby.

Následující tabulka (Tabulka 2) porovnává vlastnosti a moduly navrhovaného systému s podobnými moduly v systému Atollon Lagoon Project Management.

Vlastnosti a moduly	**Atollon Lagoon Project Management**
Správa projektů	Ano, možnost využít šablony
Správa úkolů	Ano, možnost neomezených upozornění na úkoly
Generování sestav	Ano
Harmonogramy průběhu	Řešeno výkazy práce ve vztahu k plánu
Správa uživatelů	Ano, neomezeno
Interní komunikace	Částečná
E-mailová komunikace	Ano
Správa dokumentů	Ano, evidence podnikových i projektových dokumentů
Správa účetnictví	Částečná, lze vytvářet z projektových plánů cenové kalkulace, pro detailní správu účetnictví nutné dokoupit další modul

Evidence obchodních kontaktů	Ne
Kalendář akcí	Řešeno modulem pro sledování času
Modul statistik	Ano

Tabulka 2 - Porovnání systému Atollon Lagoon Project Management

1.3.3 WorkSheetPro

Jedná se o projekt firmy FoxCom s.r.o. Internetová aplikace WorkSheetPro je zaměřená především na evidenci vykonané práce. Je určená pro malé a střední firmy zabývající se zakázkovou výrobou nebo poskytováním služeb.

V systému je možné vkládat záznamy o provedené činnosti pro zákazníka, vytvářet a filtrovat výkazy práce zaměstnanců a evidovat jejich odpracované hodiny. Systém obsahuje adresář obchodních kontaktů ve formě vizitek. Další vlastností je možnost vytvoření podkladů pro fakturaci a evidence samotných faktur. Případný export dat je prováděn do PDF, CSV nebo XLS.

Výhody systému

- Funkce pro automatické vytváření podkladů pro fakturaci.
- Jednoduché účtování za zvolené období.
- Možnost rozlišení různých hodinových sazeb a měn.

Nevýhody systému

- Jednoduché prostředí není příliš dobře graficky zpracované.
- Systém není možné bezplatně vyzkoušet.

Následující tabulka (Tabulka 3) porovnává vlastnosti a moduly navrhovaného systému s podobnými moduly v systému WorkSheetPro.

Vlastnosti a moduly	**WorkSheetPro**
Správa projektů	Ano, v systému nahrazuje pojem „projekt“ konkrétní zakázka
Správa úkolů	Ano, součástí jsou funkce pro tvorbu výkazů práce
Generování sestav	Ne, nahrazeno možností vytvořit podklady pro fakturaci a výkazy práce
Harmonogramy průběhu	Ne
Správa uživatelů	Ano
Interní komunikace	Ne
E-mailová komunikace	Ne
Správa dokumentů	Ne
Správa účetnictví	Částečná, pouze evidence faktur
Evidence obchodních kontaktů	Ano
Kalendář akcí	Ano
Modul statistik	Ne

Tabulka 3 - Porovnání systému WorkSheetPro

1.3.4 WorkWatch

WorkWatch je on-line systém pro evidenci práce, zakázek a nákladů s nimi spojených. Aplikaci vyvinula a spravuje Brněnská firma Web Facies s.r.o.

Aplikaci je možné objednat ve dvou variantách. Varianta FREE je k dispozici zcela zdarma, je ale omezena pouze na jednoho uživatele a chybějí ji některé funkce (například správa pracovníků nebo adresář kontaktů). Z této verze lze kdykoliv přejít na placenou variantu. Placená verze STANDARD je nabízena za 180 Kč / měsíc. Tato verze obsahuje všechny funkce a počet uživatelů v systému není nijak omezen.

V systému je možné evidovat zakázky v zakázkovém listě, měnit jejich stav, sledovat náklady a odpracované hodiny. K zakázkám je možné vkládat neomezený počet souborů a dokumentů. Dále je možné spravovat jednotlivé uživatele a nastavovat jim různé uživatelské role a nastavovat víceúrovňový přístup k firemním informacím. Součástí systému jsou i detailní přehledy a výkazy práce. Součástí je také adresář kontaktů ve formě vizitek. Systém umožňuje archivaci dat. Případný export je prováděn do univerzálního formátu XML nebo PDF.

Výhody systému

- Verze zdarma pro osobní použití.
- Možnost využít demoverzi (k dispozici všechny funkce) přímo na stránkách produktu.
- Bezpečnost komunikace je zajištěna pomocí protokolu HTTPS.
- Zálohování dat v obou verzích.

Nevýhody systému

- Ovládání systému je místy nepřehledné.
- Není možné generovat harmonogramy průběhů.
- Chybí přehled detailních statistických údajů.
- Systém nedisponuje žádnou interní komunikací.

Následující tabulka (Tabulka 4Tabulka 3) porovnává vlastnosti a moduly navrhovaného systému s podobnými moduly v systému WorkWatch.

Vlastnosti a moduly	**WorkWatch**
Správa projektů	V systému řešeno formou zakázky, omezené možnosti
Správa úkolů	Ano, omezené možnosti
Generování sestav	Ne
Harmonogramy průběhu	Ne
Správa uživatelů	Ano
Interní komunikace	Ne

E-mailová komunikace	Ne
Správa dokumentů	Pouze u jednotlivých zakázek
Správa účetnictví	Ne
Evidence obchodních kontaktů	Ano, formou vizitek
Kalendář akcí	Ano, řešeno formou „Osobního panelu“
Modul statistik	Ano, velmi zjednodušené

Tabulka 4 - Porovnání systému WorkWatch

1.3.5 Instant Team

Aplikace Instant Team je produkt pražské společnosti Heaven Industries s.r.o. Na rozdíl od ostatních popisovaných SW se jedná o nativní aplikaci určenou pro Windows, Mac OS a Linux. Systém komunikuje se vzdáleným serverem přístupným přes Internet. Hlavní zaměření systému je řízení projektů.

Aplikaci je možné získat ve dvou variantách. Hostovaná varianta s plnou funkcionalitou je k dispozici za měsíční poplatek 199 Kč / měsíc za každého uživatele. Druhou možností je zakoupit samostatný systém a nainstalovat jej na vlastní firemní server. Toto řešení má tři cenové programy, které se liší množstvím funkcí a licenčními úrovněmi. Základní varianta Standard je nabízena za 1 990 Kč. Verze Professional s rozšířenou funkcionalitou je za 7 990 Kč. Plná funkcionalita je nabízena ve verzi Architect za 59 990 Kč.

Systém je navržen pro řízení portfolia projektů. Těch je možné vytvořit neomezené množství a hierarchicky je členit. Je také možné využít šablony projektů. V systému je také možné vytvářet úkoly, odhadovat jejich náročnost a nastavit jejich periodické opakování. Součástí aplikace je finanční řízení. Dále zpracování statistik, analýz a pracovních výkazů. Import a export je možný do CSV formátu. Dokumenty je možné nahrávat k projektům a mezi uživateli je libovolně sdílet. Systém je k dispozici v češtině a angličtině včetně kompletní dokumentace a podpory.

Výhody systému

- Zkušební verze až pro pět uživatelů na tři měsíce zdarma.
- Systém je velmi modulární a nabízí rozsáhlé možnosti přizpůsobení.
- Přehledné průběhy řešení projektů ve formě Ganttova grafu.
- Aplikace automaticky vypočítává optimální cestu pro řešení úkolu.
- Komunikace se serverem je zabezpečena 128bitovým šifrováním.
- Automatické načítání a ukládání dat z/na server.

Nevýhody systému

- Klientská aplikace vyžaduje nepřetržité připojení k internetu.
- V systému není možná žádná interní komunikace mezi uživateli.

Následující tabulka (Tabulka 5Tabulka 3) porovnává vlastnosti a moduly navrhovaného systému s podobnými moduly v systému Instant Team.

Vlastnosti a moduly	Instant Team
Správa projektů	Ano, lze hierarchicky členit
Správa úkolů	Ano
Generování sestav	Ano
Harmonogramy průběhu	Ano, řešeno pomocí Ganttova grafu
Správa uživatelů	Ano, varianta pro neomezený počet uživatelů je ale velmi nákladná
Interní komunikace	Ne
E-mailová komunikace	Ne
Správa dokumentů	Ano, v rámci projektu
Správa účetnictví	Částečně
Evidence obchodních kontaktů	Ne
Kalendář akcí	Ano
Modul statistik	Ano

Tabulka 5 - Porovnání systému Instant Team

2 Použité technologie

2.1 Vývojové prostředí a databáze

Vývojové prostředí, anglicky označované jako IDE (Integrated Development Environment), je sada specializovaných nástrojů pro vývoj různorodých aplikací. Součástí IDE je především editor zdrojového kódu, dále také kompilátor (překladač do strojového kódu), debugger (slouží k hledání chyb), prostředí pro běh aplikace (interpret) a další nástroje v závislosti na typu IDE.

Databáze je software, který slouží k ukládání dat a umožňuje oprávněným uživatelům k těmto datům přistupovat a nějakým způsobem s nimi manipulovat. Existují různé typy databází, v práci byl využit relační typ databáze.

2.1.1 NetBeans IDE

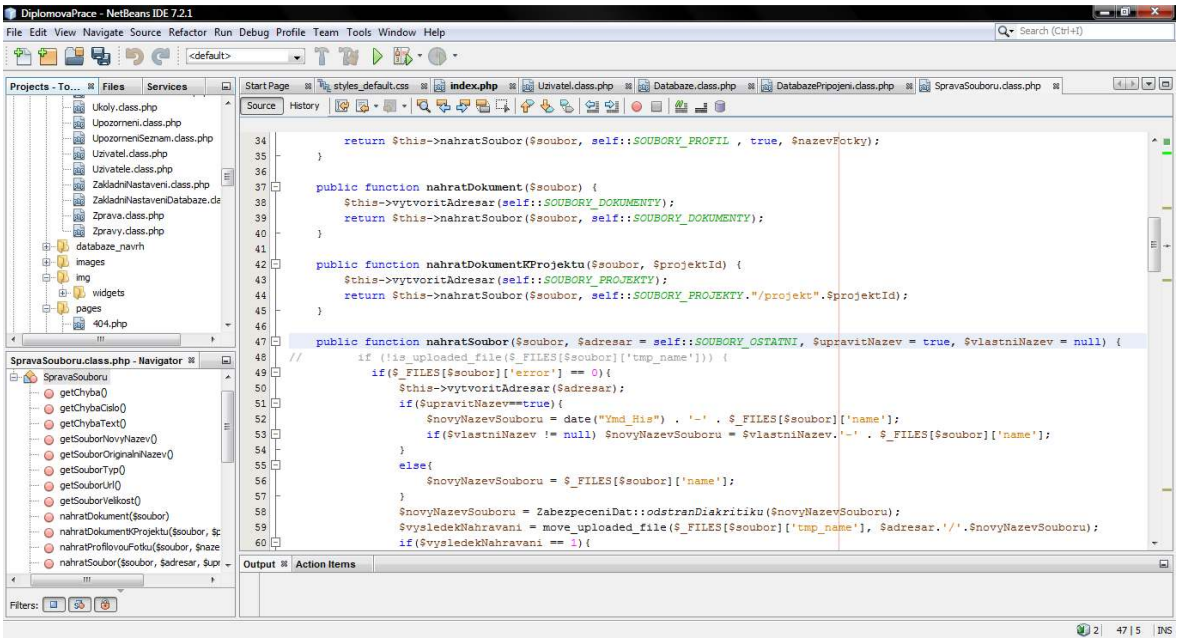
Celá praktická část diplomové práce byla napsána v prostředí **NetBeans IDE**.

Jedná se původem o české vývojové prostředí. Začátek vývoje tohoto software spadá do roku 1996, kdy vznikl ještě jako studentský projekt pod názvem Xelfi. Následovala změna na název NetBeans a odkoupení produktu společností Sun Microsystems. Současným vlastníkem práv na toto vývojové prostředí je firma Oracle Corporation, která získala tento produkt společně s koupí firmy Sun Microsystems v roce 2010.

NetBeans IDE je napsané v Javě, je tedy multiplatformní a hodí se na programování nejenom v Javě, C, C++, ale také pro tvorbu HTML a psaní PHP nebo JavaScriptu. Nejnovější verze, která mimo jiné podporuje i HTML5, má číslo 7.3.1. a vydaná byla 12. června 2013. NetBeans lze bezplatně stáhnout z oficiálních stránek produktu[60].

Následující obrázek (Obrázek 1) ukazuje verzi NetBeans 7.2.1.

[60] https://netbeans.org/



Obrázek 1 - Ukázka NetBeans IDE verze 7.2.1

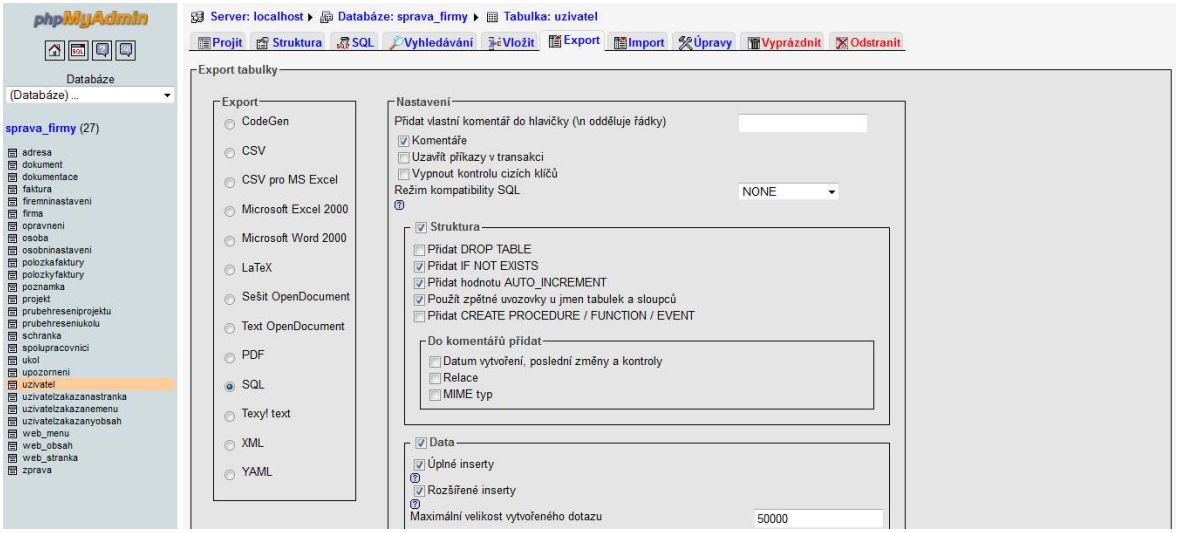
2.1.2 PhpMyAdmin

Pro operace s databází MySQL byl během vývoje aplikace využit software **phpMyAdmin**. Jedná se o nástroj pro správu databází, který je napsaný v jazyce PHP. Jeho výhodou je multiplatformnost a jednoduché webové rozhraní. Nabízí základní funkcionality pro správu databáze MySQL, například vytváření, upravování a mazání tabulek, import a export dat nebo možnost provádět zadané SQL příkazy. Lze také jednoduše spravovat přístupy a oprávnění jednotlivých uživatelů. Software také podporuje vzdálené připojení pomocí SSL.

Vývoj produktu phpMyAdmin začal v roce 1998. Zatím poslední verze 4.0.4.2 byla vydána 28. června 2013. Produkt je dostupný pod licencí GPL a je k dispozici v 62 jazykových mutacích (včetně češtiny). Stáhnout jej lze na oficiálních stránkách produktu[61].

Ukázka starší verze je na následujícím obrázku (Obrázek 2).

[61] http://www.phpmyadmin.net



Obrázek 2 - PhpMyAdmin verze 3.2.1

2.1.3 Databáze MySQL

V praktické části diplomové práce byla pro uchovávání dat použita databáze MySQL.

MySQL je relační databázový server (DBMS). Původně se jednalo pouze o soukromý projekt švédské firmy TCxDataKonsult. Pro veřejnost byla MySQL databáze uvolněna v roce 1996 a v současné době se jedná o jednu z nejvíce populárních open source databází. V době psaní této diplomové práce je aktuální verze 5.6.12 z 3. června roku 2013. Současným vlastníkem je firma Oracle.

MySQL je platformě nezávislý typ databázového systému. Lze ho tedy použít jak na operačním systému Windows, tak i na unixových systémech, kterým je například Linux. Velkou předností databáze MySQL je flexibilita, důraz na výkon a rychlost a také různé možnosti licenčních podmínek při využití této databáze. Často se MySQL používá na webových serverech v kombinaci s Linuxem, Apache a PHP. Tato kombinace softwarových nástrojů se pak označuje jako „LAMP platforma“ nebo „LAMP technologie“.

Důraz na výkon a rychlost databáze MySQL byl, je a pravděpodobně i bude stále hlavní prioritou vývojářů této databáze. Pro starší verze to byla z určitého hlediska nevýhoda, protože výkon se upřednostňoval i na úkor některých funkcí a nabízených možností databáze. Starší verze proto neumožňovali využití například triggerů, uložených procedur, poddotazů, chyběla úplná podpora cizích klíčů nebo pohledy. Současná verze však již většinu těchto nevýhod nemá a jedná se tak o plnohodnotný RDBMS.

Přehled vybraných postupných vylepšení chybějících funkcionalit v závislosti na verzi MySQL zachycují následující seznamy.

MySQL verze 3

- kompletní podpora transakcí,

- zavedeny cizí klíče,
- zamykání na úrovni řádků.

MySQL verze 4

- innoDB součástí standardní distribuce,
- možnost využít poddotazy (od verze 4.1),
- podpora bezpečného připojení pomocí SSL,
- využití cache paměti (k ukládání dotazů) pro zvýšení výkonu,
- podpora pro prostorová data, časová pásma, různé znakové sady,
- integrace MySQL serveru do aplikací.

MySQL verze 5

- zavedeny triggery spouštěné před i po výskytu události,
- možnost využít uložené procedury,
- zavedeny pohledy,
- zavedeno INFORMATION_SCHEMA a rozšířena práce s metadaty,
- kompletní podpora cizích klíčů pro identifikování relace mezi tabulkami.

MySQL je k dispozici ke stažení zdarma na oficiálních stránkách produktu[62].

2.1.4 Databáze Oracle 10g

Pro testovací účely a pro tvorbu exekučních plánů byla využita Oracle Database 10g.

Tato databáze byla uvedena na trh v roce 2003. Verze 10g Release 2 byla uvolněna o dva roky později. Dodává se v několika edicích:

- Standard Edition
- Standard Edition One
- Enterprise Edition
- Express Edition

Poslední zmiňovaná edice je k dispozici zdarma s určitými omezeními. Ostatní verze jsou placené. Všechny verze jsou vzájemně kompatibilní a nezávislé na použitém operačním systému (lze tedy získat databázi pro Windows i pro Linux).

Největší předností databáze Oracle je mimořádně vysoký výkon. Od verze 10g je k dispozici zpracování v gridu (odtud písmeno „g“ v názvu verze). V podstatě se jedná o rozložení výpočetní zátěže. Další výhodou je dostupnost a zabezpečení databáze. Verze 10g Release 2 poskytuje další možnosti šifrování dat, což je důležité v případě krádeže. Vylepšena byla i technologie RAC při práci více počítačů nad jednou velkou databází.

[62] http://www.mysql.com/downloads/

2.2 Skriptovací a programovací jazyky

Hlavní programovací jazyk práce je PHP. Protože se jedná o webovou aplikaci, tak výstup všech stránek odpovídá standardu HTML5. V některých případech byl také využit jazyk Javascript. Při práci s databází byl použit dotazovací jazyk SQL. Všechny zmíněné jazyky budou nyní popsány podrobněji.

2.2.1 PHP – Hypertext Preprocessor

Serverový skriptovací jazyk PHP je objektově orientovaný programovací jazyk podobný jazyku C a Perl pro tvorbu dynamických webových aplikací.

Historie tohoto jazyka sahá až do roku 1995. Autorem je Rasmus Lerdorf, který poskytoval první sadu skriptovacích nástrojů pro webové stránky ještě pod zkratkou *Personal Home Page*.

V roce 1997 byla vydána druhá verze PHP 2.0 (také označovaná jako *Personal Home Page – Form Interpreter*). V té době už PHP začalo nabírat na popularitě a na různých vylepšeních jazyka spolupracovala rozvíjející se komunita z celého světa. Ve stejné době také došlo k přejmenování akronymu z Personal Home Page na Hypertext Preprocesor a zároveň se začala připravovat další verze.

Třetí verze s přepracovaným parsing enginem byla vydána v roce 1998. Verze byla označována jako PHP 3.0 (nebo jen zkráceně PHP 3) a oproti původní verzi byla mnohem rychlejší, obsahovala mnohem více funkcí a samozřejmě byla k dispozici pod operačním systémem Windows. Hlavní iniciativu nad PHP převzali izraelští vývojáři Andi Gutmans a Zeev Suraski, kteří vytvořili základ PHP 3. V roce 1999 bylo opět přepsáno jádro systému a pojmenováno Zend Engine (jedná se o složeninu z křestních jmen Zeev a Andi).
V současné době oba vývojáři založili firmu Zend Technologies[63] se sídlem v Izraeli a připravovali vydání čtvrté verze.

Dne 22. května 2000 byla oficiálně vydána verze PHP 4. Krom přepracovaného jádra obsahovala tato verze řadu novinek a dalších vylepšení, například:

- Podpora objektově orientovaného programování včetně přetěžování objektů a získávání informací o třídách při běhu aplikace.
- Podpora HTTP sessions (zpracování sezení) a zavedení superglobálních proměnných (`$_GET, $_POST, $_SESSION`).
- Začlenění knihovny MCrypt[64] a podpora šifrovacích algoritmů a hashovacích funkcí (MD5, TripleDES).
- Output buffering a vylepšená správa prostředků.
- Podpora ISAPI.
- Začlenění knihovny PCRE[65] pro použití regulárních výrazů.

[63] http://www.zend.com
[64] http://mcrypt.sourceforge.net/
[65] http://www.pcre.org/

- Nové jazykové konstanty.
- Přidání mnoha dalších funkcí pro rozšíření základní výbavy a funkcionality jazyka.

Verze PHP 4 přinesla řadu novinek a znamenala velký krok vpřed. Současně ale trpěla některými nedostatky, například omezenými schopnostmi OOP (absence rozhraní, či destruktorů, nemožnost klonování objektů, chybějící obory viditelnosti atributů a metod).

Zatím poslední verze PHP 5 byla oficiálně vydána 13. června 2004. Přinesla řadu zdokonalení a vylepšení funkcionalit. PHP 4 se od té doby již nevyvíjí a PHP 5 je proto jedinou stabilní verzí. Následující seznam uvádí některá hlavní vylepšení a zdokonalení v této verzi:

- Nové zdokonalené jádro Zend Engine 2.0.
- Vylepšení architektury objektového modelu, odstranění některých chyb a přidání množství funkcionalit s tím souvisejících. Přidány byly abstraktní třídy, explicitní konstruktory a destruktory, statické atributy a metody, byly zavedeny obory viditelnosti členských proměnných a metod, zavedena podpora rozhraní a klonování samotných objektů.
- Nové zpracování výjimek pomocí bloku try/catch a protokolování chyb vzniklých při běhu aplikace.
- Vylepšena podpora webových služeb a rozšíření knihovny libxml pro podporu XML, DOM, SOAP, XSL a dalších formátů.
- Kompletně přepsané rozšíření pro MySQL včetně připravených příkazů (prepared statements) a SSL připojení.
- Podpora databázového serveru SQLLite[66].
- Vylepšená práce s řetězci a zavedení specializované ofsetové syntaxe.

PHP je v současné době stále jedním z nejvíce populárních skriptovacích jazyků pro psaní webových aplikací. Podle stránek společnosti Netcraft bylo k lednu 2013 PHP použito na čtvrt miliardě webových stránek[67]. Podpora PHP ze strany poskytovatelů webhostingu je nesrovnatelná s podporou jiných jazyků, například JSP.

Přestože je PHP určeno spíše pro menší a střední projekty, existuje celá řada robustních aplikací (včetně množství velkých e-shopů, redakčních systémů, ERP systémů), které jsou postaveny na PHP:

- Adminer[68]
- Drupal[69]
- Moodle[70]
- Nette Framework[71]

[66] http://www.sqlite.org/
[67] http://news.netcraft.com/archives/2013/01/31/php-just-grows-grows.html
[68] http://www.adminer.org/
[69] https://drupal.org/
[70] https://moodle.org/

- phpBB[72]
- Wikipedia[73]
- WordPress[74]
- Zend Framework[75]

Všeobecné rysy a největší výhody jazyka:

- Upotřebitelnost v praxi, orientace na webové aplikace.
- Vyspělost a možnosti jazyka, podpora velkého množství formátů a standardů.
- Syntaxe podobná Javě nebo C je jednoduchá a rychlá na pochopení.
- Rozsáhlá knihovna základních funkcí a možnost jejího dalšího rozšíření.
- Nezávislost na platformě.
- Široké možnosti a funkce při práci s databázemi.
- Rozsáhlá podpora ze strany poskytovatelů webhostingových služeb (ať už se jedná o placené služby nebo free hostingy). Z tohoto hlediska je PHP považováno prakticky za standard.
- Prakticky žádná licenční omezení.
- Široká komunita vývojářů a rozsáhlá dokumentace.

Další vlastnosti jazyka:

- Již zmíněná podpora OOP.
- Vágně typový jazyk, datový typ je určen za běhu a je svázán s hodnotou. Jako datový typ je vždy použit ten nejlépe vyhovující.
- Podpora předávání parametrů metod jak hodnotou, tak odkazem.
- Flexibilní práce s řetězci a regulárními výrazy.
- Automatická likvidace proměnných a vrácení prostředků při ukončení skriptu.

Výše uvedený seznam není konečný. Vlastností jazyka je opravdu velké množství a mnoho věcí je stále upravováno a některé chybějící vlastnosti jsou přidávány až s další verzí. Například chybějící jmenné prostory (namespace) přišly s verzí 5.3.0. Některé vlastnosti také pochopitelně PHP nepodporuje. Příkladem je přetěžování operátorů a metod (overloading), které pravděpodobně nikdy ani podporováno nebude.

Žádný programovací jazyk není dokonalý a PHP není výjimkou. Existují zjevné nevýhody jazyka, pro příklad:

- Nedodržování konvencí pojmenování funkcí. Některé funkce jsou pojmenovávány s podtržítky (`str_word_count($string)`), jiné používají názvy bez podtržítek (`strtolower($str)`). Stejně tak názvy parametrů nejsou jednotné.

[71] http://nette.org/
[72] https://www.phpbb.com/
[73] http://www.wikipedia.org/
[74] http://wordpress.org/
[75] http://framework.zend.com/

- Z podstaty vágně typového jazyka nebude nikdy možné plně využít například polymorfismus metod, protože nelze určit typ předávané hodnoty. Existuje sice doporučení pro typ (type hinting), ale to platí pouze u objektů.
- Neúplná implementace výjimek. Samotné knihovny jazyka výjimky téměř nepoužívají.
- Chybí úplná podpora některých řetězcových typů a jejich kódování (lze vyřešit přes dodatečné připojení knihovny).
- Protože je PHP interpretovaný a ne kompilovaný jazyk, tak veškeré skripty a jejich obsah si mohou osoby s přístupem na server přímo přečíst (v libovolném textovém editoru).

I přes tyto nedostatky je PHP stále častou volbou při tvorbě webových stránek a webových aplikací.

2.2.2 HTML – HyperText Markup Language

HTML je v doslovném překladu „značkovací jazyk pro hypertext“ a jedná se o hlavní jazyk pro vytváření webových stránek. Původně vycházející z jazyka SGML. Protože jde o značkovací jazyk, tak jeho hlavní účel je popsání jednotlivých částí dokumentu značkami (anglicky tagy), které definují strukturu a části dokumentu ve webovém prostředí.

Historie HTML sahá až do šedesátých let dvacátého století. Douglas Engelbert v té době formuloval myšlenku „určitým způsobem provázaných dokumentů“. Tehdy se však ještě nejednalo o HTML v takové podobě, v jaké ho známe dnes a jeho myšlenka v té době byla jen těžko realizovatelná. První verze a specifikace jazyka HTML vznikaly až v letech 1991 – 1993 během projektu na informačním systému pro CERN (Evropské centrum jaderného výzkumu) v Ženevě. Autorem je Sir Timothy John Berners-Lee, který je dnes ředitelem mezinárodního konsorcia W3C[76], které vydává standardy pro World Wide Web.

V roce 1995 byla vydána verze HTML 2.0, jejíž specifikace je v RFC dokumentu číslo 1866[77].

Další verzí byla HTML 3.2, která byla oficiálně vydána v lednu 1997. Tato verze byla rozšířena o podporu Java apletů, rozšířeny byly možnosti formátování, přibyly tabulky a další styly pro ovlivnění vzhledu textu.

Další standard HTML byl přijat v prosinci roku 1997. Součástí verze HTML 4.0 byla podpora rámů (frames), skriptů a vícejazyčných dokumentů. Upravena byla specifikace pro tabulky a formuláře. Vzhled dokumentu zajišťuje CSS dokument, který je k HTML dokumentu připojen. Specifikace také definovala tři typy dokumentu:

- Strict – v dokumentu je zakázáno používat některé starší prvky (například značky `font`, `strike`, `center`), veškeré formátování pouze pomocí CSS.
- Transitional – v dokumentu mohou být některé starší prvky.

[76] http://www.w3.org/
[77] http://tools.ietf.org/html/rfc1866

- Frameset – stejné vlastnosti jako typ Transitional, navíc s možností použít rámy ve stránce.

O dva roky později, v prosinci 1999, byla vydána upravená verze HTML 4.01. Tento standard pouze opravoval některé chyby v předchozí specifikaci. Vývoj dalších verzí byl následně pozastaven a dokonce měla být tato verze i tou poslední. Vývoj se měl ubírat směrem XHTML v kombinaci s XML.

V roce 2004 byla založena skupina WHATWG (Web Hypertext Application Technology Working Group). Její členové jsou například Google, Mozilla Foundation nebo Apple. Tato skupina vytvořila specifikaci pro HTML5. V roce 2006 se do vývoje HTML5 zapojilo i konsorcium W3C.

V současné době je HTML5 stále ve vývoji, nicméně specifikace jazyka je zveřejněna jako „pracovní verze“ a vývojáři ji dnes již mohou naplno využívat. Finální verze specifikace by měla být zveřejněna v roce 2014. Aktuálně na specifikaci HTML5 pracují tyto organizace:

- Web Hypertext Application Technology Working Group
- World Wide Web Consortium
- Internet Engineering Task Force

HTML5 přichází s novou vizí, množstvím rozšíření a řadou novinek oproti původním verzím. Přesto je hlavním cílem především jednoduchost a odstranění zbytečných složitostí. Další velké cíle jsou zajištění kompatibility, správné oddělení formy od obsahu, bezpečnost a přínos pro uživatele. Největší změny a novinky v HTML5 jsou následující:

- **Nové elementy pro strukturování dokumentu.** Společnost Google analyzovala milióny webových stránek a objevila množství shod v pojmenování jednotlivých částí dokumentu. Proto byly vytvořeny nové sémantické elementy pro popis těchto sekcí, například element `<header>` pro definování hlavičky dokumentu nebo element `<footer>` pro definici patičky stránky. Použití nových elementů činí kód více srozumitelným. Následuje seznam některých nových elementů:
  - `header` – sekce pro určení hlavičky dokumentu, nejčastěji se používá pro zobrazení loga nebo názvu stránky,
  - `nav` – označuje sekci, ve které je umístěna základní navigace a menu stránky,
  - `section` – určuje část webové stránky,
  - `article` – sekce pro definici obsahu článku nebo komentáře,
  - `aside` – sekce pro související obsah ke článku, poznámka nebo citace,
  - `footer` – určuje část pro zápatí dokumentu, typicky používané pro uvedení copyrightu nebo informací o autorovi.

- **Zjednodušené určení typu dokumentu a znakové sady.** Deklarace dokumentu má pouze následující podobu: `<!DOCTYPE html>`. Znaková sada je zjednodušena na toto zadání: `<meta charset="utf-8">`.
- **Zajištění zpětné kompatibility.** V případě, kdy webový prohlížeč nové funkce nebo elementy nepodporuje.
- **Zjednodušená pravidla pro syntax.** Například název třídy prvku může i nemusí být uveden v uvozovkách.
- **Přístupnost a nezávislost na použitém médiu.** HTML5 by mělo být nezávislé na použité platformě a zařízení.
- **Podpora všech světových jazyků.**
- **Bezpečnost.** Na zabezpečení HTML5 je kladen velký důraz. Specifikace určuje nový model zabezpečení pro použití u různých API.
- **Kreslící nástroje a plátno Canvas.** Nové rozhraní pro dynamické vykreslování grafiky a animací. Pro použití tohoto rozhraní je definován nový element `<canvas>`, který přidá do stránky kreslící plátno.
- **Nové rozhraní pro audio a video.** Specifikace HTML5 nabízí nové prostředky pro práci s multimédii a jejich ovládání. Cílem je nahradit množství pluginů a obsahu typu flash. Pro potřeby práce s audiem je zaveden nový element `<audio>`. Pro práci s videem je zaveden element `<video>`.
- **Rozšíření formulářů.** HTML5 nadále používá beze změny prvek input, ale navíc rozšiřuje typy možných atributů, například:
    - `email` – pole pro zadání e-mailu, které automaticky kontroluje, zda adresa obsahuje zavináč a na některých mobilních zařízeních nabídne odlišné zobrazení klávesnice uzpůsobené pro zadání e-mailové adresy,
    - `number` – formulářové pole, do kterého lze zadat pouze číslo,
    - `url` – formulářové pole pro zadání URL adresy,
    - `search` – hodnota se předá vyhledávači,
    - `range` – pole pro zadání hodnoty v určitém rozsahu,
    - `datetime` – formulářové pole pro zadání data,
    - `color` - formulářové pole pro výběr barvy ze vzorníku.

  Formuláře jsou v HTML5 také rozšířeny o nové atributy. Například atribut `placeholder` zobrazuje nápovědní text pole, do kterého nebyla ještě zadána hodnota. Rozšířeny byly také možnosti validace formulářů.

- **Nové geolokační rozhraní.** Obsahuje nástroje pro identifikaci aktuální polohy uživatele.
- **Nové rozhraní Web Workers.** Umožňuje provádět některé operace na pozadí.
- **Možnost tvorby off-line webových aplikací.** Taková aplikace pak může normálně fungovat i bez neustálého připojení k internetu a prostředky pro běh načítá z mezipaměti.

Výše uvedený seznam není konečný. HTML5 obsahuje velké množství změn, které ještě nemají pevnou specifikaci. Stejně tak podpora nových vlastností HTML5 ze strany prohlížečů není úplná. Přestože je kladen velký důraz na zpětnou kompatibilitu, některé nové vlastnosti mohou ve starších prohlížečích způsobovat nestandardní chování.

2.2.3 SQL – Structured Query Language

SQL je neprocedurální dotazovací jazyk pro práci s databází. Jazyk vznik v 70. letech dvacátého století pod hlavičkou firmy IBM, která se v té době věnovala výzkumu relačních databází. Původně se jazyk jmenoval SEQUEL, postupem času byl název zkrácen pouze na SQL. Existuje několik standardů tohoto jazyka, nejnovější má označení SQL 3.

Příkazy jazyka se rozdělují do určitých částí. Příkazy typu **DDL** (Data Definition Language) můžeme vytvářet, mazat nebo upravovat různé databázové struktury (například tabulky). Další skupinou jsou příkazy pro získávání dat a manipulaci s nimi (naříklad jejich aktualizace nebo mazání). Tato skupina má označení **DML** (Data Manipulation Language). Další skupinou jsou příkazy pro řízení provozu a přístupu k databázi, souhrnně označované jako **DCL** (Data Control Language). Poslední skupinou jsou příkazy pro řízení transakcí **TCC** (Transaction Control Commands).

Problém jazyka SQL je v přenositelnosti mezi databázemi. Ne všechny totiž implementují veškeré normy pro jazyk SQL a některé konstrukce naopak přidávají navíc. Příkladem může být omezení výsledného počtu řádků v závislosti na typu databáze. V databázi MySQL se používá klauzule `LIMIT`, v databázi Oracle se lze využít k získání počtu řádků `ROWNUM` a u Microsoft SQL se zase využívá klauzule `TOP`, jak ukazují následující příklady.

```
// příklad pro MySQL
SELECT jmeno FROM uzivatel LIMIT 5

// příklad pro Oracle
SELECT * FROM ( SELECT jmeno FROM uzivatel ) WHERE ROWNUM <=5

// příklad pro Microsoft SQL
SELECT TOP(5) jmeno FROM uzivatel
```

2.2.4 JavaScript

JavaScript je interpretovaný, objektově orientovaný skriptovací jazyk. Je multiplatformní a používá se pro tvorbu dynamického obsahu HTML stránek. Syntaxe je podobná jazyku C, C++ nebo Javě. JavaScript se nejčastěji spouští na straně klienta (klientský JavaScript).

V praktické části diplomové práce je JavaScript použit jen okrajově. Nejčastěji zpracovává události vyvolané uživatelem. Například při přejetí myši nad určitým prvkem ve stránce je spuštěn JavaScriptový kód, který nejčastěji upravuje některé CSS vlastnosti vybraných prvků ve stránce. Další využití je u kontroly formulářových polí. Například při zadávání hesla je pomocí tohoto jazyka zobrazena uživateli zpětná vazba a informace o tom, zda zadávaná hesla jsou nebo nejsou stejná.

2.3 Návrhové vzory

Návrhový vzor je doporučený návod při řešení některých obecných problémů, se kterými se během programování a návrhu software můžeme setkat. Návrhových vzorů existuje celá řada a lze je využít při vytváření objektů, při strukturování jednotlivých komponent v systému nebo při řešení problémů tykajících se chování a vlastností objektu nebo systému.

2.3.1 Návrhový vzor Jedináček

Při programování objektově orientovaných aplikací se může vyskytnout požadavek vytvoření pouze jedné instance nějaké třídy. Tento problém řeší návrhový vzor singleton, česky „jedináček“ nebo „unikát“. Princip tohoto vzoru spočívá v znepřístupnění konstruktoru třídy tím, že se konstruktor nastaví jako soukromý. Pro získání instance se pak použije statická veřejná metoda, která v případě prvního použití vytvoří novou soukromou instanci objektu, kterou vrátí anebo je vrácena již vytvořená instance objektu. Odkaz na vytvořenou instanci je uložen v soukromém statickém atributu jedináčka.

Použití tohoto návrhového vzoru je ideální v případě, kdy využíváme připojení k databázi. Takové připojení typicky potřebujeme jen jedno. Pokud bychom měli více různých instancí připojení, tak by to mohlo způsobit odlišné výsledky databázových dotazů v závislosti na nastavení instance.

Následující kód ukazuje implementaci jedináčka v jazyce PHP.

```
<?php
class Singleton{
	// privátní atribut instance třídy
	private static $instance = NULL;

	// privátní konstruktor
	private function __construct(){
	}

	// metoda pro získání instance objektu
	public static function getInstance(){
		// Je-li proměnná $instance NULL, tak se vytvoří objekt
		if(self::$instance == NULL){
			self::$instance = new Singleton();
		}
		// Vrátíme jedináčka
		return $instance;
	}
}
?>
```

2.3.2 Návrhový vzor Lazy loading

Při vytváření složitějších webových aplikací, ve kterých existují vzájemně provázané třídy, jež načítají data z databáze, můžeme narazit na případ, kdy použití jedné třídy způsobí postupné načítání dat do jiných tříd, které ale v daném momentě nepotřebujeme nebo je vůbec nepoužijeme.

Budeme mít například třídu `Person`, která v konstruktoru obsahuje metodu `getDataFromDB()`. Tato metoda získá z databáze data, jimiž inicializuje atributy třídy `Person`. Například jméno a příjmení. Třída `Person` však také obsahuje atribut `adress`, který je v konstruktoru inicializován vytvořením nové instance třídy `Adress`. Třída `Adress` v konstruktoru obsahuje také metodu `getDataFromDB()` a ta také získá data o adrese z databáze. Pokud nyní vytvoříme instanci objektu třídy `Person`, abychom zjistili její jméno, tak zároveň z databáze načteme i data o adrese, která ale nepotřebujeme.

Metoda Lazy loading se používá pro zvýšení výkonu. Nejčastěji právě u takové aplikace, která ke svému fungování používá databázi. Databázová data jsou pak při použití strategie Lazy loading z databáze načtena až v případě jejich skutečné potřeby.

Existuje několik typů Lazy loading a samozřejmě různé typy implementace. V PHP lze pro implementaci využít například přetěžování funkcí.

Strategii Lazy loading není nutné spojovat pouze s přístupem k databázi. Tuto techniku je možné využít i v případě zamezení zbytečného vytváření nepoužívaných objektů. Ve výše uvedeném příkladu by se tedy atribut `adress` v konstruktoru vůbec neinicializoval a jeho inicializace by byla provedena až v případě potřeby pomocí volání metody k tomu určené.

2.3.3 Návrhový vzor MVC

MVC je zkratka pro Model-View-Controller. Striktně vzato se nejedná o návrhový vzor v pravém slova smyslu, ale spíše o typ architektury. Také se používají pro jeho označení názvy „architektonický vzor“ případně „agregační návrhový vzor“. Často v sobě obsahuje jiné návrhové vzory. Důležitý je především princip MVC, kterým je oddělení aplikační logiky od uživatelského rozhraní aplikace. Celá architektura je rozdělena na tři části.

Model zajišťuje veškerou práci s daty. Spravuje také business logiku aplikace. V praktické části práce reprezentují model PHP třídy. Model a samotné třídy jsou také úzce spojeny s databází, odkud třídy získávají data.

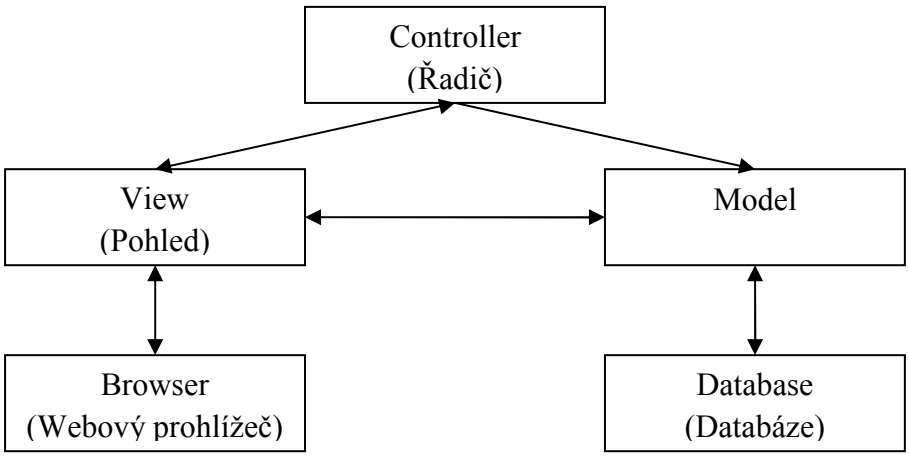
View (Pohled) obstarává části aplikace, které se zobrazují uživateli. Získává data z modelu, které může také nějakým způsobem upravit do lepší podoby pro výslednou prezentaci uživateli. Jedná se tedy především o HTML kód, který je kombinovaný s CSS styly. Součástí pohledu jsou také ostatní grafické prvky, například grafy a nebo JPEG, GIF nebo PNG obrázky.

Controller (Řadič) slouží ke zpracování událostí vyvolaných klientem. Zachycuje HTTP požadavky a odeslaná data a na jejich základě získává nebo upravuje data v modelu nebo zajišťuje změny v pohledu. Ve webovém prostředí je součástí řadiče také část Front Controller.

Front controller tvoří často jen jeden soubor, který slouží jako vstupně výstupní brána, přes kterou procházejí všechny požadavky. V praktické části práce tento soubor také existuje a jmenuje se `index.php`. Součástí tohoto souboru jsou také základní nastavení a

prostředky potřebné pro běh aplikace, takže není nutné je připojovat nebo kopírovat do jiných souborů.

Základní princip MVC architektury použité v praktické části diplomové práce zachycuje následující obrázek (Obrázek 3).



Obrázek 3 - Architektura MVC rozšířená o přístup k databázi a přístup uživatelů

3 Databázová struktura

3.1 Návrh databázové struktury

Návrhu databázové struktury byla věnována značná pozornost. Jedním z požadavků na aplikaci je rychlost. Pokud bude navržený systém pracovat s neefektivním návrhem databáze, může nastat degradace výkonu. Tomu byla snaha zabránit. Databáze je proto navržena tak, aby její tabulky splňovaly bez výjimky druhou normální formu (2NF) a ve většině případů také třetí normální formu (3NF). Zde je však nutné sladit návrh databáze s návrhem aplikace. Návrh tabulek ve třetí normální formě je obecně správný, ale v některých případech je lepší z hlediska výkonu třetí normální formu porušit. Striktní dodržení třetí normální formy by mohlo vést k velkému množství zbytečných, nepřehledných anebo velmi velkých tabulek. Velké množství rozsáhlých tabulek a operace jejich spojování pak mohou snížit rychlost provádění celého dotazu. Příkladem může být tabulka `Adresa` použitá v aplikaci. Tato tabulka eviduje adresy zaměstnanců a obsahuje mimo jiné i neklíčové sloupce `PSČ` a `Město`. Z logického hlediska závislost mezi poštovním směrovacím číslem a městem existuje. Tabulka by se tedy měla rozpadnout na další tabulky. Otázkou pak zůstává, zda má smysl uchovávat tabulku s několika tisíci směrovacími čísly měst a obcí u firmy s desítkou zaměstnanců. Návrh aplikace a s tím spojené databázové struktury je proto zaměřen tak, aby co nejvíce zjednodušoval všechny procesy v rámci systému a poskytoval co nejvyšší výkon a rychlost.

3.2 Nejdůležitější databázové tabulky v systému

Návrh databáze se skládá z velkého množství tabulek, z nichž nejdůležitější jsou popsány níže.

3.2.1 UZIVATEL

Tato tabulka je jednou z nejdůležitějších v systému. Udržuje veškeré informace o uživatelích. Obsahuje například sloupce jméno, příjmení, e-mail, datum narození, sloupec s jednoznačným identifikátorem uživatele (ID) a podobně. Zároveň také uchovává přihlašovací informace, jakými jsou například uživatelské jméno, heslo (v šifrované podobě), aktuální stav přihlášení (jestli je uživatel online nebo ne) a také datum posledního přihlášení. Data získaná z této tabulky jsou pro přihlášeného uživatele v aplikaci udržována v superglobální proměnné `($_SESSION)`, protože jsou často využívána. Stejně tak řada dalších tabulek je s touto tabulkou propojena.

3.2.2 PROJEKT

Tabulka obsahuje důležité informace o projektech systému. Sloupce této tabulky tvoří například ID projektu, název projektu, datum přijetí projektu do systému, datum plánovaného začátku práce na projektu, datum ukončení projektu, prioritu projektu, aktuální stav a podobně.

3.2.3 UKOL

Tabulka udržuje základní informace o úkolech v systému. Každý úkol patří k nějakému projektu, takže součástí tabulky je i sloupec odkazující na tabulku `PROJEKT`. Dalšími informacemi o úkolu, které tabulka uchovává, jsou ID úkolu, název, popis, datum vytvoření úkolu, datum dokončení, odhadovaná a reálná časová náročnost (v minutách), priorita, současný stav a podobně.

3.3 Další tabulky v systému

Výše popsané tabulky jsou hlavními tabulkami v systému. Pro fungování jsou v systémy i další tabulky. Vybrané jsou popsány níže.

`OPRAVNENI` – tato tabulka je spojena s tabulkou `UZIVATEL` a určuje oprávnění daného uživatele. Obsahuje dva sloupce. Název uživatelské role (tvoří zároveň primární klíč) a detailní popis role.

`NASTAVENI` – uchovává informace o individuálním nastavení uživatele. Obsahuje sloupce, které určují například typ výchozího řazení položek na stránkách se seznamy nebo použitý typ stránkování a podobně.

`POZNAMKA` – uživatelé mohou využívat v systému vlastní poznámky, které je možné sdílet přes podnikovou nástěnka. Tabulka `POZNAMKA` proto obsahuje sloupec ID poznámky, datum vložení, samotný text poznámky a také určení, zda je poznámka vložena na nástěnce nebo ne.

`DOKUMENT` – tabulka uchovává informace o dokumentech v systému. Obsahuje sloupec ID, název dokumentu, mime typ, datum nahrání, velikost a podobně. Součástí jsou i sloupec pro určení data, od kterého je možné dokument zobrazit a také data do kterého je dokument přístupný.

`FAKTURA` – obsahuje například sloupec ID, vlastní číslo faktury, datum vystavení, variabilní symbol, způsob platby, datum splatnosti a podobně.

`FIRMA` – v systému je adresář kontaktů firem. Údaje o nich jsou uloženy v této tabulce. Obsahuje informace o ID firmy, název firmy, obor podnikání, identifikační číslo firmy, odkaz na webové stránky a podobně.

`OSOBA` – uchovává informace pro adresář osob. Obsahuje sloupec ID osoby, jméno, příjmení, telefon, e-mail a další. V systému a stejně tak v databázové struktuře jsou striktně odděleny kontaktní osoby a uživatelé systému.

`ADRESA` – tabulka obsahuje například sloupec ulice, město, poštovní směrovací číslo, kraj a podobně. Kvůli zjednodušení nedodržuje tato tabulka 3NF.

ZPRAVA – v systému je interní systém soukromých zpráv. Tato tabulka obsahuje informace o každé konkrétní zprávě v systému. Obsahuje sloupec s ID zprávy, předmět, text zprávy, datum odeslání a podobně.

SCHRANKA – slouží jako poštovní schránka při doručení zpráv. Obsahuje odkazy na zprávy, příjemce a také uchovává informaci o tom, zda byla již zpráva příjemcem přečtena či nikoliv.

UPOZORNENI – v systému si mohou jednotliví uživatelé nastavit upozornění, která jim přijdou například na e-mail nebo formou soukromé zprávy. Tabulka, která to zajišťuje, obsahuje například sloupec ID upozornění, název a text upozornění, datum a také identifikátor, zda se má upozornění odeslat na e-mail nebo do soukromé pošty.

Krom výše uvedených tabulek existují v databázovém návrhu také další. Například propojovací tabulky mezi tabulkami s vazbou M:N, tabulky s rozšiřujícími údaji o systému, tabulky omezující přístup na jednotlivé části stránky nebo zobrazení položky menu, tabulky s historickými údaji (například průběh řešení úkolu) nebo tabulka s globálním firemním nastavením.

3.4 Optimalizace SQL dotazů a exekuční plány

Při získávání dat z databáze můžeme mít řadu rozdílně napsaných SQL dotazů, které však mohou vracet stejná data. Díky optimalizačním technikám a využitím exekučních plánů, lze najít takový dotaz, jehož vykonání stojí databázový server nejméně prostředků (typicky čas procesoru), a tím zvýšit výkon aplikace, ve které je SQL dotaz použit.

Pro tvorbu exekučních plánů byla využita databáze Oracle a použit nástroj Oracle SQL Developer. Ten umožňuje přehledně zobrazit exekuční plány dotazu v grafické podobě. Exekuční plán je v podstatě seznam jednotlivých operací, které databáze provádí při zpracování SQL dotazu. Součástí plánu je i „cena“ jednotlivých operací. Ta určuje, jak efektivně je dotaz vykonáván. Čím nižší cena dotazu, tím efektivněji je dotaz vykonáván. Podle ceny dotazu tak můžeme určit, který SQL dotaz v aplikaci skutečně použijeme.

Využití exekučních plánů je zachyceno na následujícím jednoduchém příkladu.

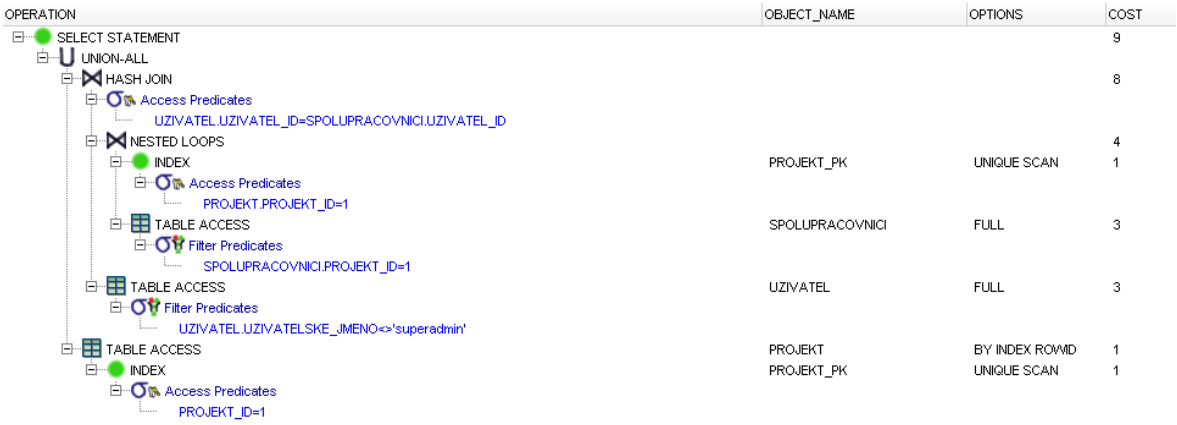
V navrhované aplikaci existuje SQL dotaz na získání všech uživatelů, kteří spolupracují na určitém projektu. Součástí dotazu je také získání uživatele, který je vedoucím projektu. V systému existuje také uživatel „superadmin“, kterého však ve výsledku dotazu mít nechceme.

Možná podoba dotazu je následující (uvažujme jako hledaný projekt ten s identifikačním číslem 1):

```
SELECT uzivatel_id
FROM uzivatel
JOIN spolupracovnici USING(uzivatel_id)
JOIN projekt USING(projekt_id)
WHERE uzivatelske_jmeno<>'superadmin'
```

```
AND projekt_id = 1
UNION ALL
SELECT vedouci_id
FROM projekt
WHERE projekt_id = 1
;
```

Výsledná podoba exekučního plánu uvedeného dotazu je zachycena na následujícím obrázku (Obrázek 4).



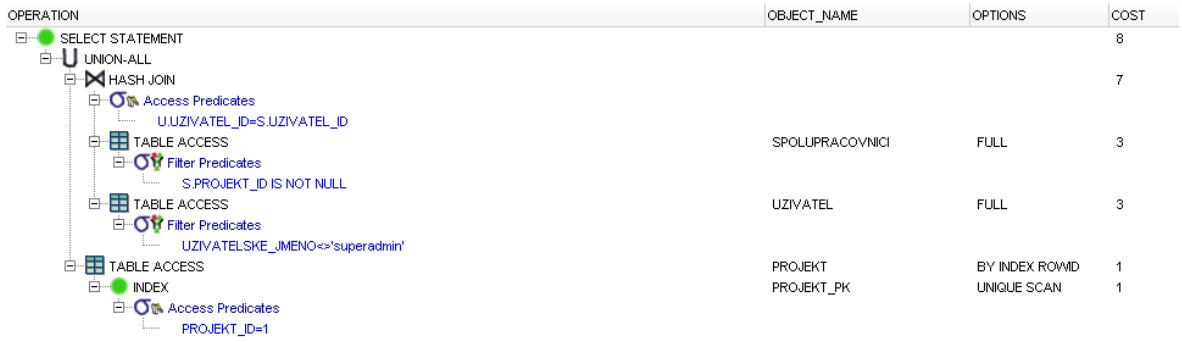
Obrázek 4 - Exekuční plán první varianty SQL dotazu

Výsledná cena dotazu (COST) má hodnotu 9.

Dotaz může být ale zapsán i jiným způsobem. Například bez operací `JOIN`.

```
SELECT u.uzivatel_id
FROM uzivatel u, spolupracovnici s, projekt p
WHERE uzivatelske_jmeno<>'superadmin'
AND u.uzivatel_id = s.uzivatel_id
AND s.projekt_id = p.projekt_id
UNION ALL
SELECT vedouci_id
FROM projekt
WHERE projekt_id = 1
;
```

Exekuční plán pro tento dotaz je následující (Obrázek 5).



Obrázek 5 - Exekuční plán druhé varianty SQL dotazu

V tomto případě jsme ušetřili jednu operaci `NESTED LOOPS` a celková cena dotazu je proto 8.

Přestože obecné doporučení je nepoužívat vnořené dotazy v některých případech jejich použití nejenom zpřehledňuje zadaný SQL dotaz, ale jeho vykonání může být rychlejší a celková cena dotazu je tedy nízká.

```
SELECT uzivatel_id
FROM uzivatel
WHERE uzivatelske_jmeno<>'superadmin'
AND uzivatel_id IN
(SELECT vedouci_id FROM projekt WHERE projekt_id = 1)
OR uzivatel_id IN
(SELECT uzivatel_id FROM spolupracovnici WHERE projekt_id =1)
;
```

Výsledná podoba exekučního plánu s vnořenými dotazy je na následujícím obrázku (Obrázek 6).

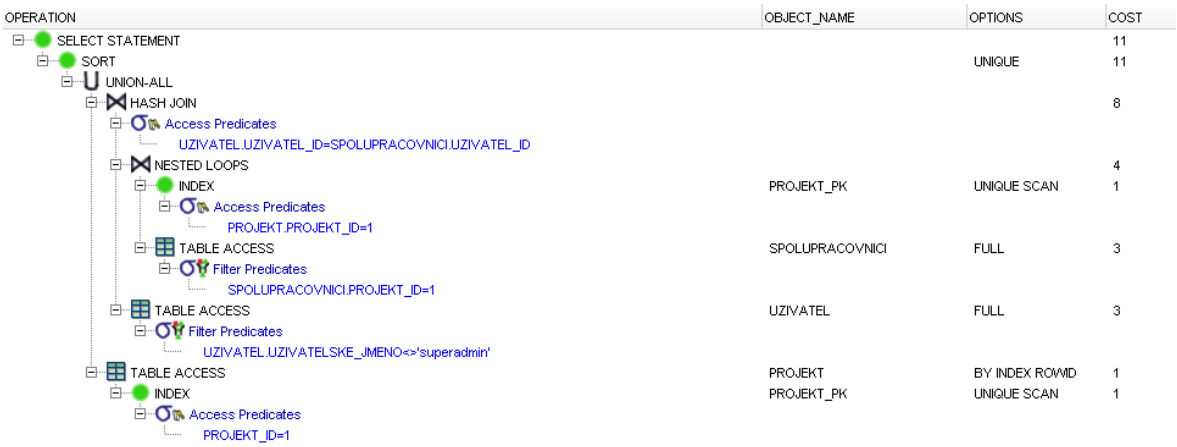
Obrázek 6 - Exekuční plán třetí varianty SQL dotazu

I přes použití vnořeného dotazu je výsledná cena pouze 3. V aplikaci je proto použita tato varianta dotazu.

V uvedeném případu je také dobré věnovat pozornost klauzuli `UNION ALL`. Použití klauzule `UNION ALL` na rozdíl od obyčejného `UNION` neprovádí zbytečné řazení výsledku. Eliminací nadbytečných operací řazení je zvýšena rychlost provedení dotazu. Pokud se vrátíme k první variantě dotazu a nahradíme klauzuli `UNION ALL` variantou `UNION` budeme mít téměř totožný kód.

```
SELECT uzivatel_id
FROM uzivatel
JOIN spolupracovnici USING(uzivatel_id)
JOIN projekt USING(projekt_id)
WHERE uzivatelske_jmeno<>'superadmin'
AND projekt_id = 1
UNION
SELECT vedouci_id
FROM projekt
WHERE projekt_id = 1
;
```

Výsledná cena dotazu je však vyšší než v prvním případě, jak ukazuje následující exekuční plán (Obrázek 7). Ten je téměř stejný jako původní, ale na závěr vykonaná operace `SORT` cenu dotazu zvyšuje.



Obrázek 7 - Exekuční plán odlišné varianty prvního SQL dotazu

Další variantou optimalizace SQL dotazů je použití hintů. Ty mohou ovlivnit výsledný exekuční plán a vynutit jiné přístupové metody optimalizátoru. Hinty slouží jako jakási nápověda nebo doporučení pro optimalizátor. Zadávají se přímo do kódu SQL dotazu. Hintů existuje různé množství variant. Jejich částečnou nevýhodou je fakt, že při jejich použití musíme dobře znát strukturu databázových dat, na které chceme hint u SQL dotazu aplikovat.

Hinty se vkládají do SQL dotazu ve formě komentáře. To je výhodné v případě, kdy kód hintu není zadán správně. Pokud totiž není hint zadán ve správném tvaru, tak je jednoduše ignorován a SQL dotaz tak může být normálně vykonán.

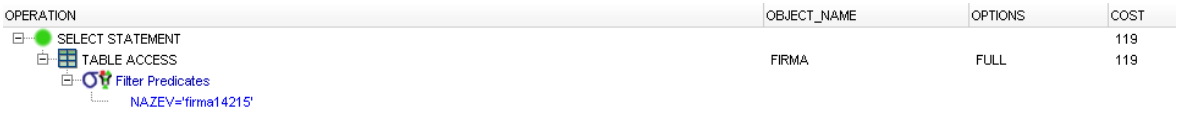
Některé ukázky a výsledky exekučních plánů při použití hintů jsou uvedeny na následujícím příkladu.

V aplikaci existuje tabulka firma, která uchovává kontaktní informace o obchodních partnerech v systému. Pro testovací účely byla tato tabulka naplněna 20 000 řádky. Pro příklad bude uveden SQL dotaz, který hledá všechny informace o firmě s názvem „firma14215“.

Nejjednodušší podoba dotazu je následující:

```
SELECT *
FROM firma
WHERE nazev = 'firma14215';
```

Výsledná cena dotazu je 119. Podoba exekučního plánu uvedeného dotazu je na následujícím obrázku (Obrázek 8).



Obrázek 8 - Exekuční plán bez použití hintu

Optimalizátoru můžeme například doporučit hint pro úplné prohledávání.

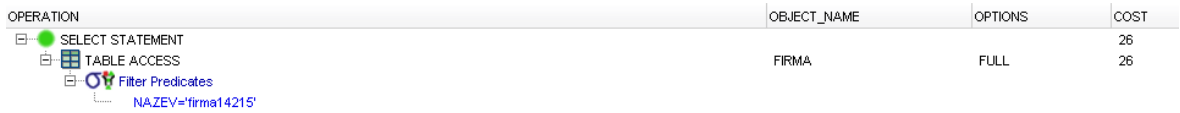
```
SELECT /*+ FULL(f)*/ *
FROM firma f
WHERE nazev = 'firma14215';
```

Z podstaty dotazu je ale jasné, že tato nápověda v daném případě nebude moc užitečná (je zbytečné prohledávat celou tabulku, když potřebujeme jeden řádek). Navíc z výsledku exekučního plánu je patrné, že optimalizátor tuto metodu stejně použil (sloupec OPTIONS).

Předpokládáme, že firma s daným názvem je v tabulce pravděpodobně jediná. Můžeme tedy zkusit nápovědu `FIRST_ROWS(n)`. Optimalizátor poté vrátí prvních n nalezených řádků.

```
SELECT /*+ FIRST_ROWS(1)*/ *
FROM firma
WHERE nazev = 'firma14215';
```

Exekuční plán pro tento případ je následující (Obrázek 9).



Obrázek 9 - Exekuční plán s použitím hintu

Z uvedeného plánu je patrné, že optimalizátor také použil metodu úplného prohledávání, ale protože je použit hint `FIRST_ROWS` tak celková cena dotazu je pouze 26. Různé hodnoty parametru u této nápovědy pak způsobí rozdílné výsledné ceny dotazu. V případě prvních dvou řádků bude výsledná cena 52. V případě třech bude výsledná cena 77. U čtyř bude cena 101. Od pěti a výše již nemá smysl použít tento hint, tabulka je prohledávána celá a výsledná cena bude stejná jako bez použití hintu (119).

4 Detailní popis systému

4.1 Požadavky na aplikaci

Jedním z hlavních požadavků na aplikaci je správa uživatelů a jejich přístup do systému. Jednotliví uživatelé mají v systému určitou roli, která určuje jejich výchozí oprávnění. Systém je navržen tak, aby bylo možné tuto funkcionalitu podle potřeby rozšířit. Práva konkrétního uživatele by pak bylo možné upravit dle aktuální potřeby.

Uživatele vkládá do systému hlavní správce (nebo uživatel s rozšířenými právy – manažer) a jednotlivé role, které jim může při vytváření přidělit, jsou následující:

- Administrátor – hlavní správce s nejvyšším stupněm práv. Uživatel s tímto oprávněním může v systému dělat v podstatě vše. Má povolen vstup na všechny stránky systému a také má vždy zobrazeny všechny dostupné možnosti menu na jednotlivých stránkách. Rozšířené je také menu aplikace o některé položky. Administrátor smí také přidávat, mazat nebo upravovat uživatele v systému. Jako jediný také nastavuje firemní údaje a základní nastavení systému.
- Manažer – z hlediska přístupu má velmi podobná práva jako Administrátor, ale nemůže měnit základní nastavení systému a firemní údaje. Má také částečně omezeno přidávání uživatelů do systému – nemůže vložit uživatele s oprávněním Administrátor. Stejně jako Administrátor může vytvářet nové projekty a přiřazovat jejich vedení libovolným pracovníkům v systému.
- Účetní – tato role poskytuje základní menu aplikace s výjimkou sledování statistik, omezuje také vstup na některé stránky systému. Uživatel s touto rolí nemůže vkládat do systému nové uživatele ani měnit jejích údaje.
- Zaměstnanec – nejnižší stupeň uživatelské role. Uživatel v roli Zaměstnance může efektivně pracovat se systémem, ale pouze z pohledu běžného pracovníka. Nemůže například přidávat další uživatele, vytvářet projekty, prohlížet statistiky ani sledovat informace typické pro účetního.

Jednotlivé uživatelské role jsou navrženy tak, aby v práci se systémem nabídly přesně ty nástroje, které daná role vyžaduje.

V systému jsou ještě další omezení, která jsou daná aktuálním stavem uživatele. Pokud je například uživatel „Vedoucí projektu“, tak má přístup k odlišným nástrojům pro správu projektu než v pozici obyčejného spolupracovníka. Stejně tak nejsou v systému zbytečná omezení z hlediska přístupu k vlastním informacím nebo souborům. Takže například uživatel s oprávněním „Zaměstnanec“ sice nemůže mazat cizí dokumenty, s vlastními dokumenty může ale nakládat podle potřeby (včetně editace a mazání).

Další požadavek na aplikaci je detailní správa projektů a úkolů v systému. Při vytvoření projektu je k němu přiřazen vedoucí projektu. Ten dále může přidávat k projektu další spolupracovníky, kteří jsou v systému a těm zadávat úkoly, které mají plnit.

Součástí systému je i evidence veškerých podnikových dokumentů. V systému je možné nastavit parametry pro jejich zobrazování uživatelům (dokument je například k dispozici jen po omezenou dobu). Stejně tak je možné vést kompletní dokumentaci a přidávání souborů k jednotlivým projektům v systému.

Dalším požadavkem je generování harmonogramu průběhu (více v kapitole 4.6.14.6.1) a sestav průběhu řešení. U sestav je možné nastavit detailní kritéria pro vygenerování.

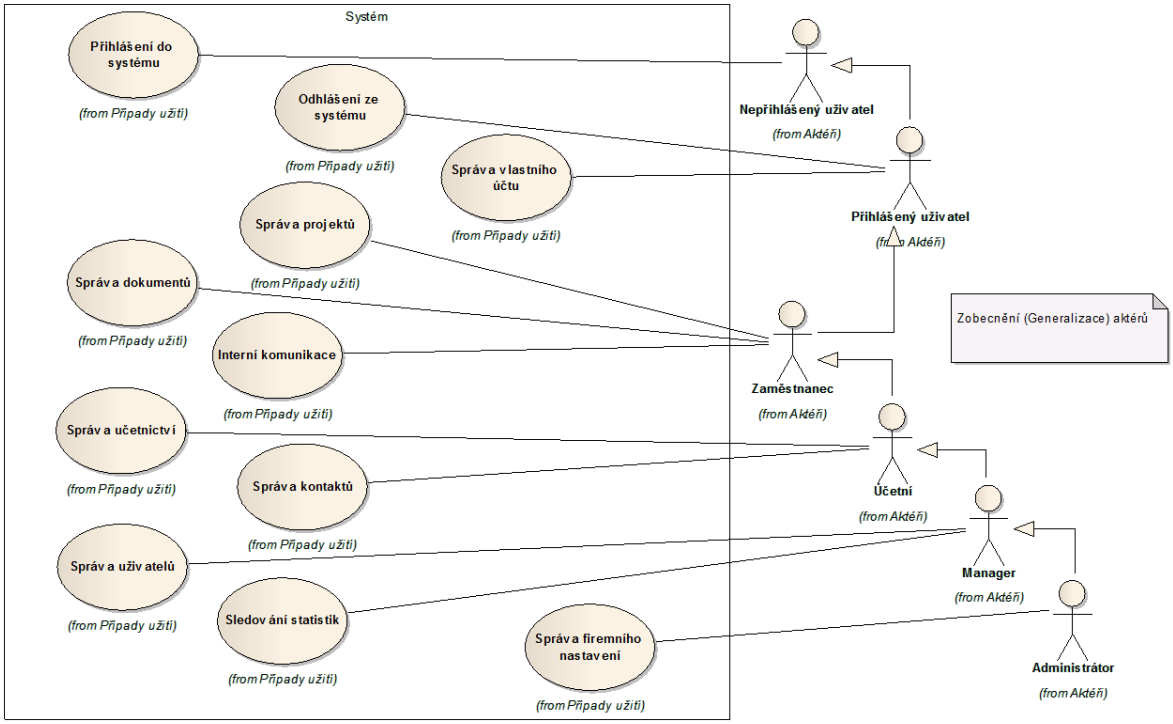
V systému je důležitá také komunikace ať už interní mezi uživateli tak externí. Aplikace proto obsahuje vlastní komunikační mechanizmus na principu soukromých zpráv. Dále je možné komunikovat přes e-maily, které je možné ze systému odesílat. Při povolení dalších komunikačních služeb je možné využít ICQ chat.

4.2 Realizace případů užití

Během zjišťování požadavků na aplikaci a ve fázi analýzy byly využity techniky UML. Například pro hledání tříd byla použita technika rozboru zadávacího dokumentu metodou podstatných jmen a sloves. Pro nalezení dalších tříd bylo využito hledání tříd pomocí metody CRC štítků.

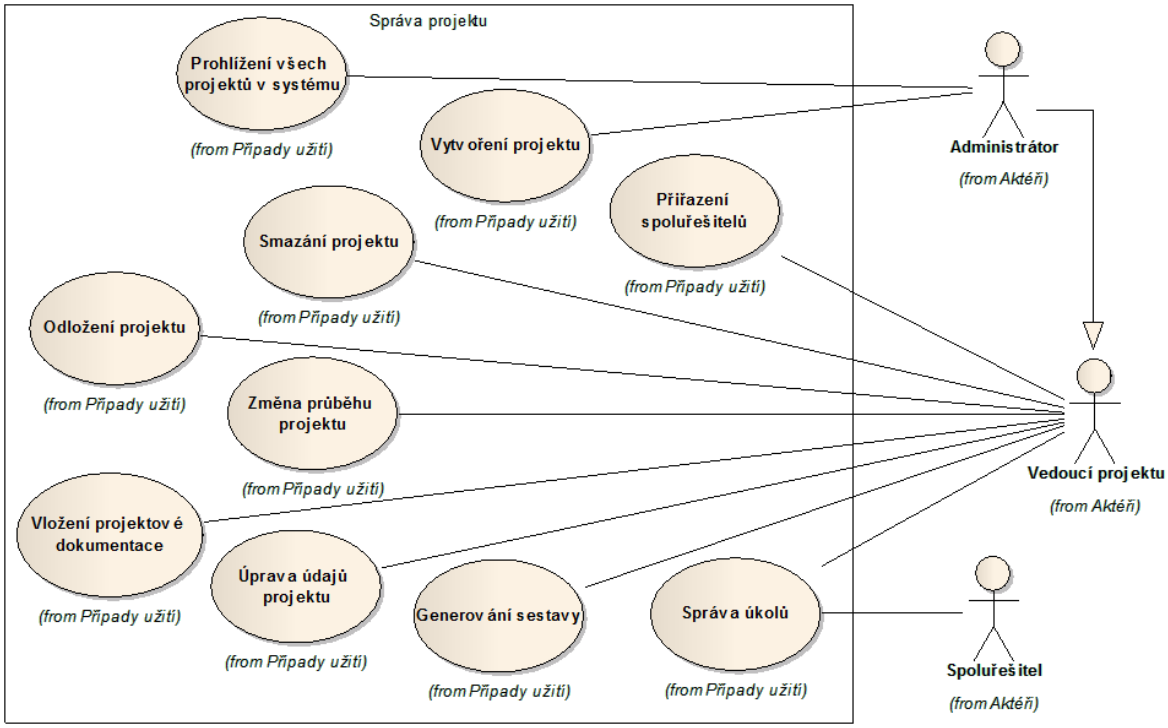
Velice důležité pro správný návrh systému bylo také určení jednotlivých aktérů v systému a jejich případy užití. Jednotlivé případy užití se v průběhu vývoje částečně upravovaly nebo více specifikovaly. Cílem bylo namodelovat jednotlivé případy užití od obecných požadavků až ke konkrétnímu chování a určení scénářů.

Příklad nejobecnějších případů užití systému je na následujícím obrázku (Obrázek 10).



Obrázek 10 - Obecné případy užití systému

Tento obrázek dobře odráží chování jednotlivých aktérů vůči systému. Nicméně se stále jedná o obecné případy užití. Ty je možné více specifikovat. Například při práci s projektem se objeví noví aktéři – vedoucí projektu a spoluřešitel. Níže uvedený obrázek demonstruje detailní případ užití pro práci s projekty (Obrázek 11).



Obrázek 11 - Detail případu užití při práci s projekty v systému

4.3 Popis hlavních tříd v systému

Celá aplikace se skládá z několika desítek tříd, které ale nebudou popsány všechny. Níže jsou detailně popsány pouze ty nejdůležitější. Ukázka zdrojových kódů třídy `Databaze` a `DatabazePripojeni` je uvedena v příloze práce.

- `ADatabazoveOperace` – poskytuje abstraktní metody `vlozitNovyDoDatabaze()`, `editovatVDatabazi()` a `smazatZDatabaze()`. Všechny třídy, které používají základní operace vkládání, editace a mazání jsou potomky této třídy.
- `Databaze` – třída zpracovává všechny dotazy na databázi a poskytuje také přístup k výsledkům zadaného dotazu. Výchozí funkce ve třídě využívají příkazy pro komunikaci s databází MySQL. Pokud bude nutné použít jinou databázi (například Oracle), ke které se přistupuje pomocí jiných funkcí, tak stačí pouze upravit těla metod (použití jiných přístupových funkcí k databázi) v této třídě a systém může dál fungovat bez úpravy jiných tříd.
- `DatabazePripojeni` – zajišťuje a udržuje aktuální připojení k databázi. Třída využívá návrhový vzor „jedináček“ a v systému proto existuje pouze jedna instance

této třídy. Díky tomu je zamezeno opakovanému připojování k databázi v případě potřeby.

- `DatumACas` – pomocná třída, která zajišťuje veškeré operace s daty a časy. Důležitá je především pro převádění data z formátu uloženém v databázi do formátu vhodném pro potřeby některých php funkcí a naopak.
- `Projekt` – obsahuje veškeré informace, které se týkají projektu.
- `Projekty` – třída poskytuje různé seznamy projektů (seznam instancí tříd typu `Projekt`) v závislosti na typu požadavku. Seznamy jsou získávány zavoláním konkrétní metody třídy, například:
  - `ziskatProjekty()` – metoda vrátí všechny projekty v systému.
  - `ziskatProjektyUzivatele()` – metoda vrátí pouze projekty daného uživatele.
  - `ziskatProjektyUzivateleHotove()` – metoda vrátí pouze ty projekty daného uživatele, které mají stav průběhu 100%.
- `SpravaSouboru` – další z pomocných tříd, tato zajišťuje automatické vytvoření adresářů v případě potřeby. Třída je také zodpovědná za manipulaci s nahrávanými soubory na server.
- `StatistickaData` – třída získá z databáze statistické informace a předpřipravuje je pro potřeby zobrazení grafů a harmonogramů průběhů.
- `Ukol` – třída pro zpracování informací týkající se úkolů v projektu.
- `Ukoly` – podobně jako třída `Projekty` umožňuje získat úkoly pomocí předpřipravených metod.
- `Uzivatel` – třída zajišťuje základní uživatelské operace a uchovává informace o uživateli. Po přihlášení uživatele do systému jsou nastavena práva uživatele a vytvořená instance této třídy je uchovávána po celou dobu trvání sezení, nebo dokud se uživatel neodhlásí.
- `ZabezpeceniDat` – třída obsahuje pouze statické metody pro úpravu a zabezpečení formulářových dat (ochrana proti SQL injection, odstranění escape sekvencí, apod.).
- `ZakladniNastaveniDatabaze` – tato třída zajišťuje vytvoření všech databázových objektů, jejich naplnění základními daty a zajišťuje výchozí nastavení systému. Typicky je použita pouze jednou.

4.4 Adresářová struktura

Celá aplikace se stává z několika adresářů. Nejdůležitější adresáře jsou uvedeny níže.

Kořenový adresář obsahuje hlavní soubor `index.php`, který slouží jako výchozí bod pro aplikaci. Dále obsahuje výchozí soubor se styly a také několik php souborů, které tvoří základní kostru stránky (například soubor s údaji o hlavičce, menu a soubor s informacemi uvedenými v patičce každé stránky). Dále obsahuje několik souborů pro potřeby

javascriptové knihovny JAK. Součástí kořenového adresáře jsou i další podadresáře důležité pro běh systému.

Adresář `class` obsahuje veškeré php třídy použité v aplikaci. Ať už se jedná o základní třídy nebo pomocné třídy, které jsou ve vlastním podadresáři.

Adresář `images` obsahuje veškeré grafické prvky, které jsou použity v designu stránek. Obsahuje další podadresáře s ikonami nebo jinými grafickými prvky.

Adresář `pages` obsahuje php soubory, které se týkají jednotlivých stránek v systému (například stránky projektů, úkolů, profily, apod.).

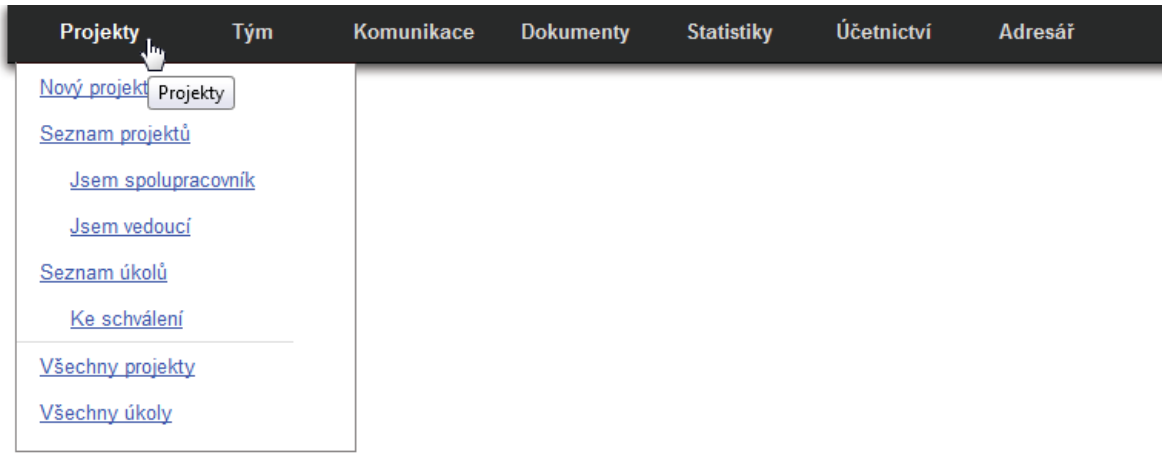
Pro ukládání dat nahraných na server jsou vyhrazeny adresáře `soubory_dokumenty`, `soubory_projekty` a `soubory_profil`. Tyto adresáře jsou v případě neexistence automaticky vytvořeny.

Mimo výše zmíněné adresáře existují ještě další, které jsou určené především pro potřeby javascriptových souborů.

4.5 Návrh systému

Systém je navržen jako on-line webová aplikace, která je nahrána na firemní server. Před prvním spuštěním je nutné vytvořit databázi a zajistit aplikaci přístup do této databáze. Tyto úkony zajistí dodavatel systému. Vytvoření dalších databázových struktur a nastavení výchozích informací je realizováno spuštěním skriptu, který celý proces automatizuje. Součástí je také vytvoření hlavního administrátorského účtu. Pomocí něj je možné se do systému přihlásit a vytvořit další uživatele systému.

Hlavní ovládání je realizováno přes drop down menu. Jeho položky jsou určeny uživatelskou rolí. Následující obrázek (Obrázek 12) ukazuje položky menu u administrátorského účtu.



Obrázek 12 - Ukázka menu aplikace u administrátorského účtu

Položka menu s názvem „Projekty“ poskytuje uživateli možnosti pro vytvoření nového projektu (pokud mu to uživatelská role umožňuje), procházet seznam projektů, ke kterým je uživatel nějakým způsobem přiřazen (ať už jako vedoucí nebo pouze spolupracovník) a seznam úkolů. Administrátor má menu rozšířeno o možnost procházet všechny projekty a všechny úkoly v systému.

Menu „Tým“ obsahuje pro administrátora položku pro vložení nového uživatele. Ostatní uživatelé mají pouze možnost procházet seznam uživatelů v systému.

Menu „Komunikace“ obsahuje položku „Nástěnka“ kde jsou zobrazeny všechny veřejné poznámky. Dále je zde možnost odeslat e-mail a také je zde menu pro interní komunikaci. To obsahuje položku pro přístup k přijatým zprávám, odeslaným zprávám a samozřejmě možnost odeslat novou zprávu.

Položka menu „Dokumenty“ umožňuje přístup k všem veřejným podnikovým dokumentům a možnost nahrát nový podnikový dokument (pokud to uživatelská role umožňuje). Uživatel s rolí administrátora má navíc možnost procházet úplně všechny (tedy i neveřejné) podnikové dokumenty.

Menu „Statistiky“ je k dispozici pouze administrátorovi systému a umožňuje mu přistupovat k detailním statistikám týkajících se projektů a úkolů.

Menu „Účetnictví“ umožňuje procházení faktur, které jsou již v systému vystaveny (přísluší k hotovým projektům).

Menu „Adresář“ obsahuje přístup ke kontaktním údajům buď fyzických, nebo právnických osob v systému.

Po přihlášení je v pravé části také osobní menu uživatele s možností upravit profil, nastavit upozornění či upravit individuální nastavení systému. Součástí je také položka pro bezpečné odhlášení ze systému.

4.5.1 Technické zpracování

Z technického hlediska byl kladen důraz na kvalitu zpracování. Kódování dokumentu je ve formátu UTF-8. Všechny stránky jsou validní podle specifikace na stránkách W3C. Stejně tak zobrazení by mělo být ve všech prohlížečích podobné. Ve starších prohlížečích nebo v prohlížečích nepodporující HTML5 musí vždy dojít k takové degradaci kvality, aby bylo možné systém i nadále používat (byť s omezenými možnostmi).

Chování systému je navrženo tak aby byla vždy uživateli zobrazena nějaká zpětná vazba v případě výskytu určité události. Například při neúspěšném odeslání formuláře je uživateli zobrazena přesná příčina chyby, proč nebylo formulář možné odeslat.

Primárně je systém navržen jako intranetová aplikace, která běží na vlastním serveru firmy. Návrh aplikace však umožňuje vytvořit ze systému také outsourcingové řešení. V takovém případě by aplikace nebyla nahrána na server ve firmě, ale na server poskytovatele tohoto produktu.

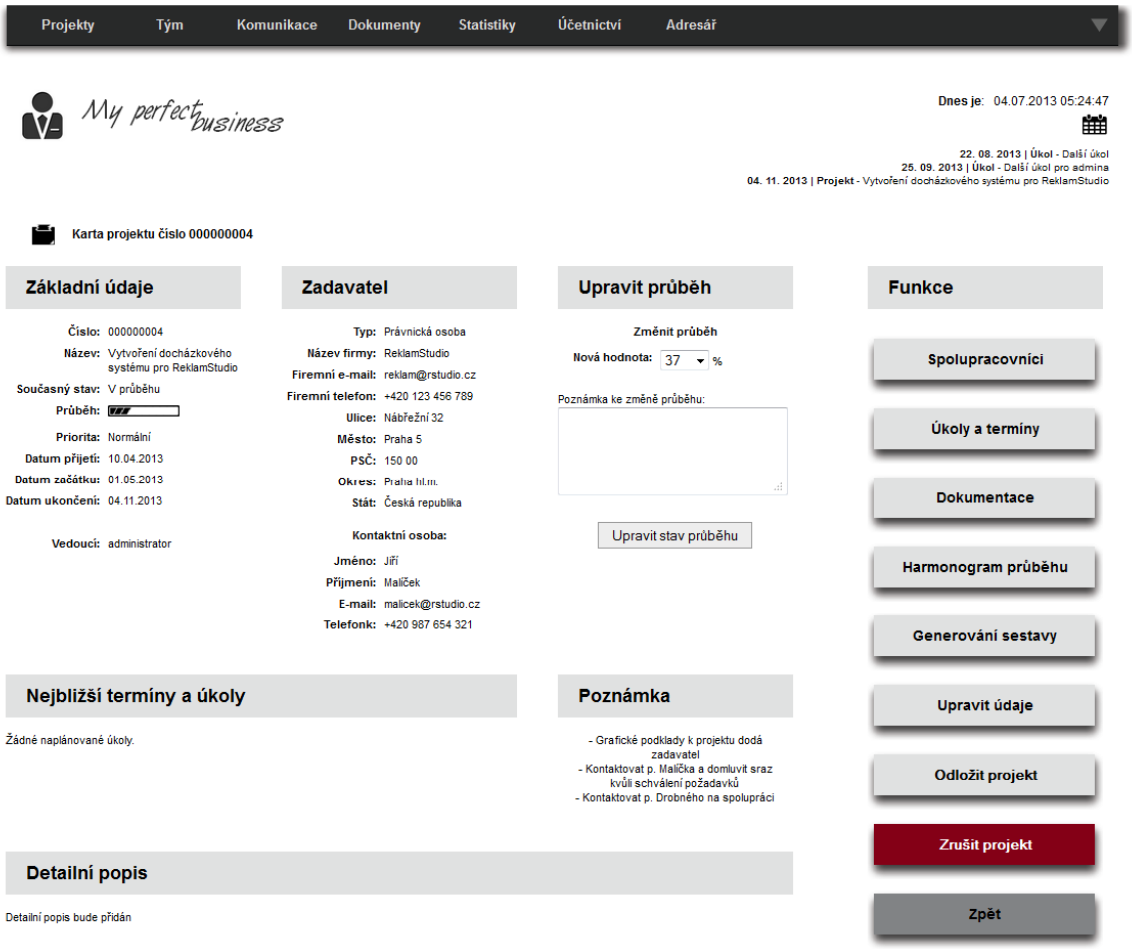
4.6 Grafická podoba a uživatelská přívětivost

Z hlediska uživatelské přívětivosti byla snaha vytvořit aplikaci co nejjednodušší na ovládání. Systém by měl být ovládán intuitivně a bez nutnosti seznámení s detailní nápovědnou.

V systému jsou použity pastelové odstíny barev. Z hlediska psychologie barev byla jako doplňková barva zvolená světle modrá a světle šedá barva. Systém by proto měl být pro uživatele mimo jiné příjemný na pohled.

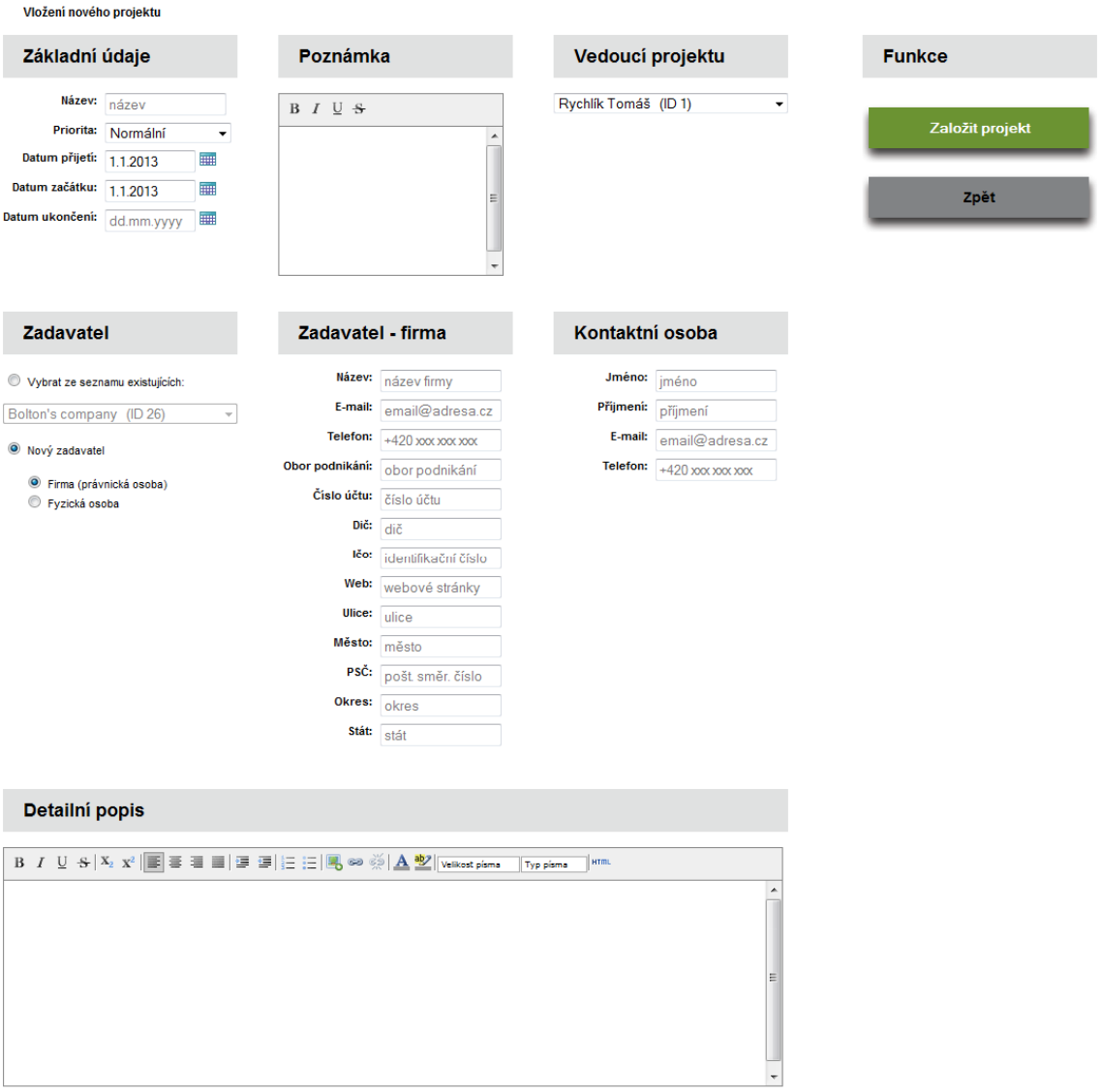
Veškerý grafický styl je určený dokumentem CSS.

Ukázka grafické podoby (stránka s informacemi o vybraném projektu) je na následujícím obrázku (Obrázek 13).



Obrázek 13 - Grafická podoba stránky projektu

Příklad vzhledu části stránky pro vložení nového projektu je zobrazen na následujícím obrázku (Obrázek 14).



Obrázek 14 - Vložení nového projektu

V systému je možné individuálně nastavit různé parametry pro zobrazení položek v seznamech, například typ stránkování, výchozí řazení nebo počet položek na stránku. Samotné seznamy je možné libovolně řadit. Ukázka části stránky se seznamem podnikových dokumentů je na následujícím obrázku (Obrázek 15).

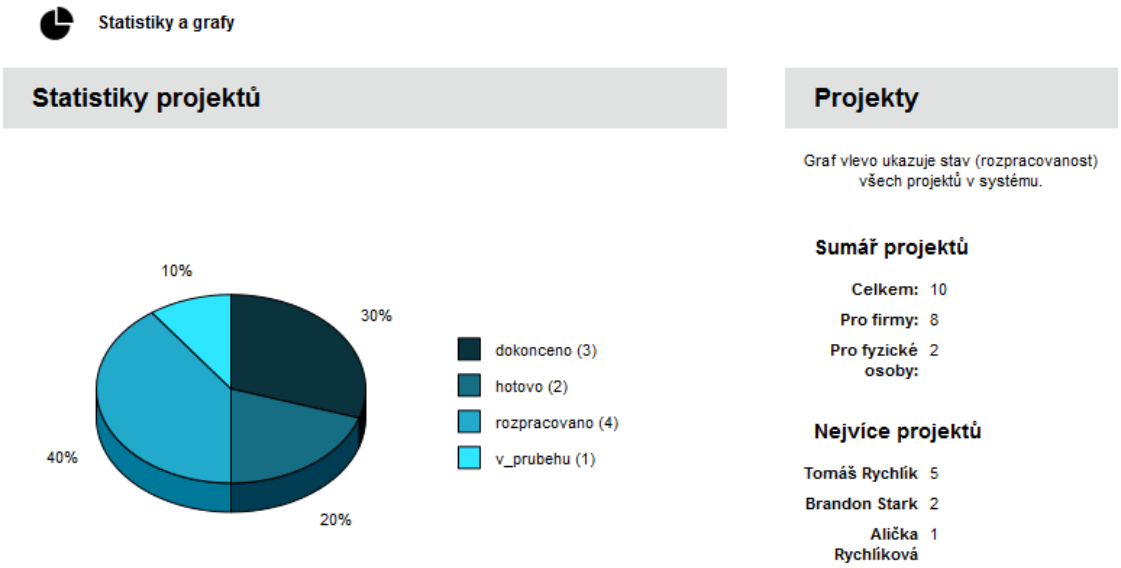
Obrázek 15 - Seznam podnikových dokumentů

4.6.1 Grafy

Jednou z podmínek práce je i generování harmonogramů průběhu řešení, pro větší přehlednost v grafické formě. Další potřeba grafů je v případě zobrazení statistik projektů a úkolů. Pro vyřešení tohoto problému byla využita javascriptová knihovna JAK[78] od společnosti Seznam.cz. Jedná se o framework, který zjednodušuje práci s JavaScriptem. Nabízí množství widgetů pro usnadnění některých obecných problémů při navrhování webových aplikací (prohlížeč obrázků, modální okna, tooltip nápověda, čtení exif dat obrázku a podobně).

Součástí knihovny JAK je i widget určený pro generování grafů, které byly v systému použity.

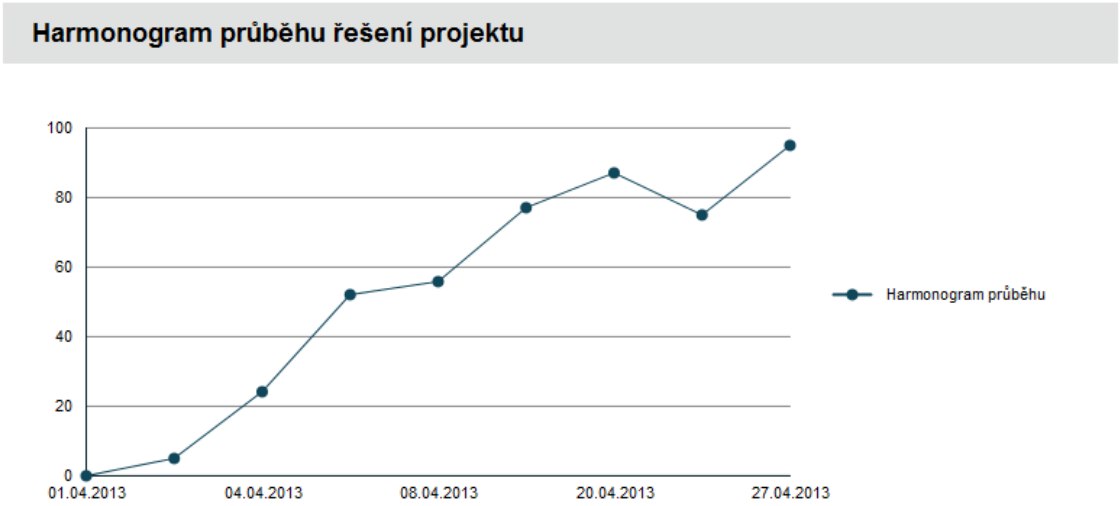
Na následujícím obrázku (Obrázek 16) je zobrazen výstup koláčového grafu s údaji o počtu a stavu aktuálních projektů v systému.



Obrázek 16 - Koláčový graf v aplikaci

[78] http://jak.seznam.cz/

V knihovně JAK je k dispozici i widget pro vykreslení čárového nebo sloupcového grafu. V aplikaci je používaný spojnicový graf. Jeho podoba je zachycena na následujícím obrázku (Obrázek 17). V tomto případě je zachycen průběh řešení projektu.



Obrázek 17 - Spojnicový graf v aplikaci

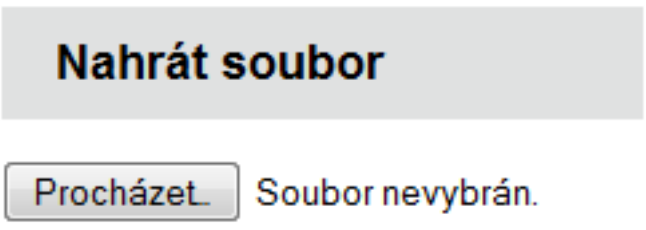
4.7 Problémy

Jedním z problémů pro používání aplikace by mohlo být nastavené rozlišení obrazovky. Aplikace nepoužívá responzivní strukturu zobrazení webové stránky. Velikost stránky je pevně nastavena a u menších rozlišení se tak může její ovládání stát trochu nepraktické.

Protože se jedná o webovou aplikaci, vyskytlo se během zpracovávání práce několik problémů nejčastěji související s rozdílnou interpretací kódu v závislosti na typu prohlížeče.

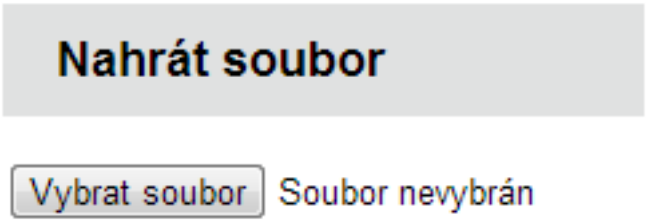
Příkladem je různý vzhled některých standardních prvků HTML v závislosti na typu použitého prohlížeče. U formulářového prvku pro odesílání souboru existují proto následující varianty.

Zobrazení odesílacího tlačítka v prohlížeči Mozilla Firefox verze 22.0 je na následujícím obrázku (Obrázek 18).



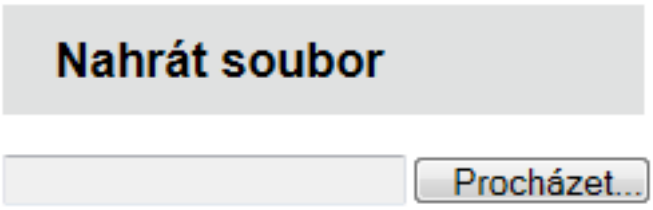
Obrázek 18 - Nahrání souboru v Mozilla Firefox

Mírně rozdílné zobrazení odesílacího tlačítka v prohlížeči Google Chrome verze 28.0.1500.95 ukazuje následující obrázek (Obrázek 19).



Obrázek 19 - Nahrávání souboru v Google Chrome

A úplně odlišné zobrazení má i Internet Explorer verze 9 jak ukazuje následující obrázek (Obrázek 20).

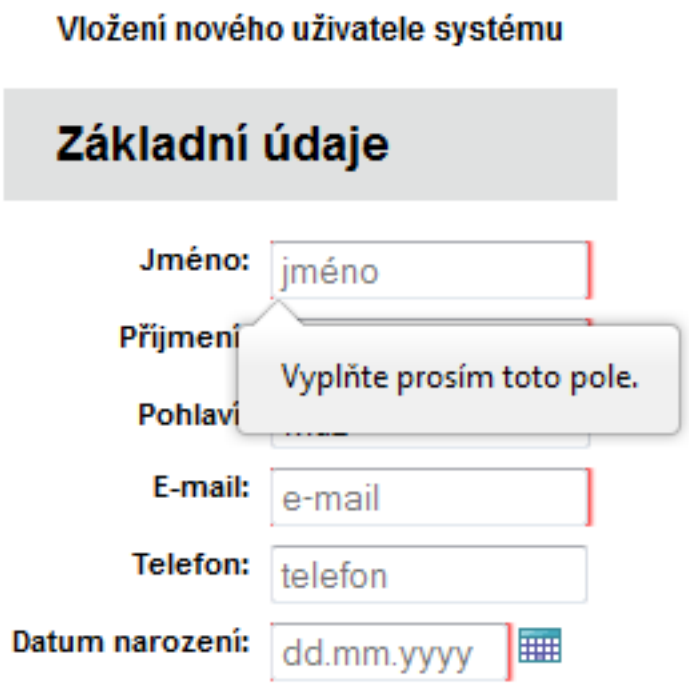


Obrázek 20 - Nahrávání souboru v Internet Explorer

Dalším problémem je nespecifikované chování u formulářových prvků HTML5. Zde je největší překážkou fakt, že HTML5 není plně standardizované a některé chování přenechává plně na prohlížeči. Například při nevyplnění povinného formulářového políčka existuje u každého prohlížeče rozdílný mechanizmus upozornění na chybu.

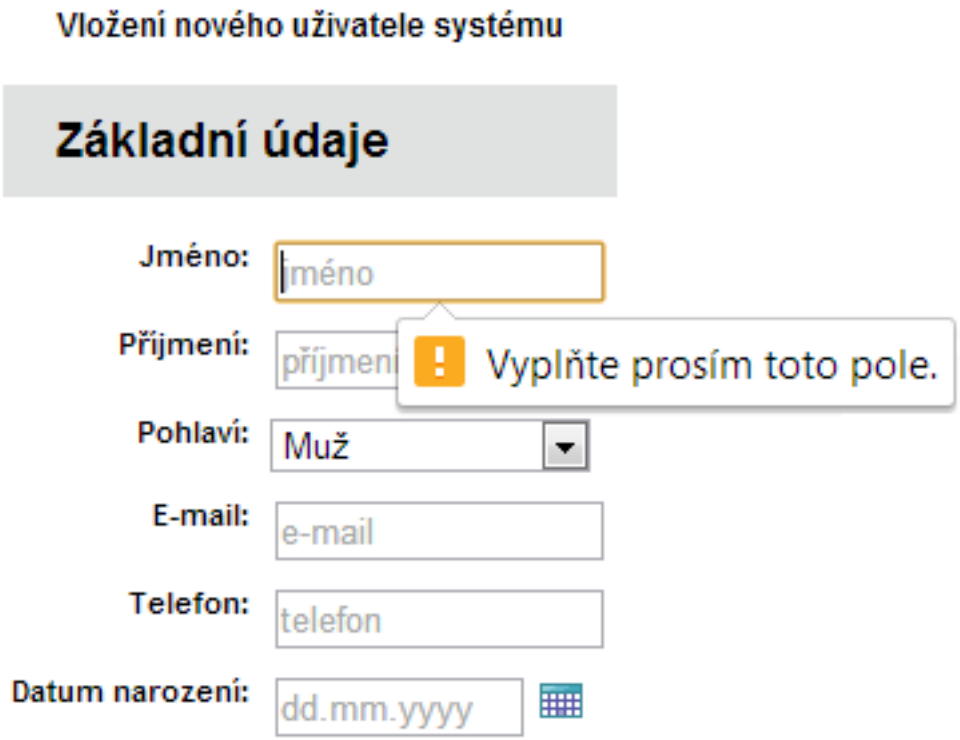
Například u Internet Exploreru verze 9 nejsou nové formulářové prvky podporovány vůbec a při nevyplnění políčka je proto nutné vytvořit vlastní mechanizmus ošetření vstupu.

U prohlížeče Mozilla Firefox verze 22.0 je upozornění na chybu zachyceno na následujícím obrázku (Obrázek 21).



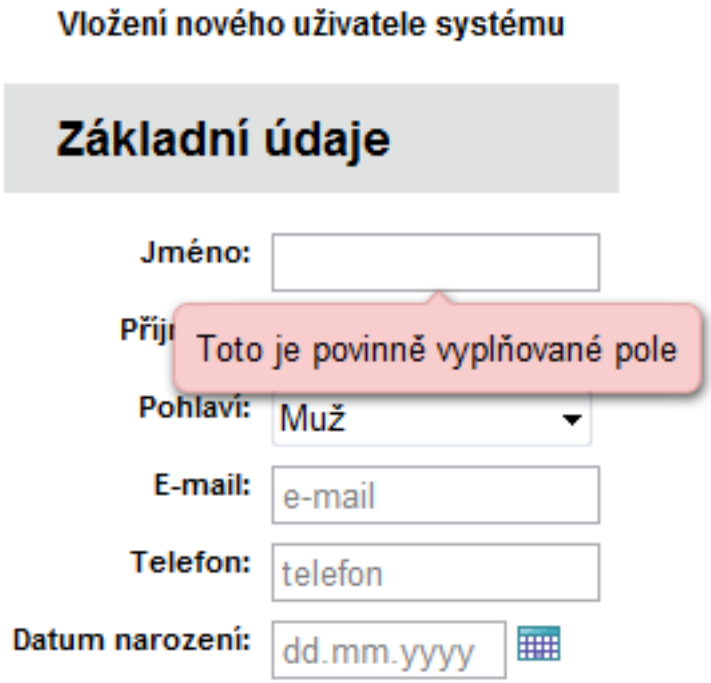
Obrázek 21 - Nevyplnění formuláře v prohlížeči Mozilla Firefox

Podobné je to u prohlížeče Google Chrome ve verzi 28.0.1500.95. Na rozdíl od Firefoxu však neupozorní na všechny povinná pole, ale pouze na první nevyplněné pole (Obrázek 22).



Obrázek 22 - Nevyplnění formuláře v prohlížeči Google Chrome

Podobné chování jako Chrome nabízí i prohlížeč Opera verze 11.51 (Obrázek 23).



Obrázek 23 - Nevyplnění formuláře v prohlížeči Opera

Ve všech případech je nutné zajistit stejné výsledné chování při snaze o odeslání formuláře bez ohledu na to, jak se na případnou chybu upozorní. Úspěšné odeslání prázdného formuláře jednoduše nesmí nikdy nastat.

Závěr

Hlavním cílem práce, krom popisu praktické části, byl také bližší popis ERP systémů. Dále jsem popsal použité technologie, včetně popisu některých použitých programovacích technik. Dále bylo mým cílem popsat návrh databáze a tvorbu exekučních plánů. Teoretické podklady jsem se snažil vhodně doplnit buď pomocí krátkých ukázek zdrojových kódů, nebo obrázky, které lépe přibližovaly danou problematiku.

Vlastní aplikace je plně funkční a připravena k okamžitému použití. Dle mého názoru je ve stavu schopném konkurovat některým podobným systémům. Další vývoj a zdokonalení aplikace však není u konce. Problematika řízení práce je značně složitá a některé podnikové požadavky aplikace zatím neřeší. Z tohoto důvodu bude mojí snahou systém pokud možno nadále vylepšovat, doplnit ještě některé zajímavé funkce a v ideálním případě aplikaci poskytovat jako outsourcingové řešení pro řízení projektů a práce ve firmě.

Literatura

Arlow Jim, Neustadt Ila. 2011. *UML 2 a unifikovaný proces vývoje aplikací.* Brno : Computer Press, a.s., 2011. 978-80-251-1503-9.

Baar, Ondřej. 2008. Historie SQL . *PCWorld.* [Online] 1. Duben 2008. [Citace: 1. Srpen 2013.] http://pcworld.cz/archiv/historie-sql-17391.

Bernard, Borek. 2009. Prezentační vzory z rodiny MVC. *zdroják.cz.* [Online] 11. Květen 2009. [Citace: 8. červen 2013.] http://www.zdrojak.cz/clanky/uvod-do-architektury-mvc/.

Bryla, Bob a Loney, Kevin. 2009. *Mistrovství v Oracle Database 11g.* Brno : Computer Press, a.s., 2009. 978-80-251-2189-4.

Buchalcevová, Alena a Pitka, Lukáš. 2007. *Vývojové prostředí NetBeans.* Praha : Oeconomica, 2007. ISBN 978-80-245-1206-8.

2013. CRM (Customer Relationship Management). *Management Mania.* [Online] 2013. [Citace: 5. Červenec 2013.] https://managementmania.com/cs/customer-relationship-management. 2327-3658.

Dudek, Jan. 2005. Vývoj PHP - cesta správným směrem? *interval.cz.* [Online] 24. Leden 2005. [Citace: 14. Červen 2013.] http://interval.cz/clanky/vyvoj-php-cesta-spravnym-smerem/.

Hlavenka, Jiří, a další. 1999. *Vytváříme WWW stránky.* Praha : Computer Press, 1999. 80-7226-163-0.

Hunt, Lachlan. 2007. Seznámení s HTML 5. *interval.cz.* [Online] 24. Prosinec 2007. [Citace: 30. Červenec 2013.] http://interval.cz/clanky/seznameni-s-html-5/.

Jason, Beaird. 2010. *Principy krásného webdesignu.* Praha : Grada Publishing, a.s., 2010. 978-80-247-2895-7.

Jason, Gilmore W. 2006. *Velká kniha PHP a MySql.* Brno : Apress, 2006. 80-86815-53-6.

Kácha, Pavel. 2004. HTML. *Linuxsoft.cz.* [Online] 3. Červen 2004. [Citace: 29. Červenec 2013.] http://www.linuxsoft.cz/article.php?id_article=185.

Kosek, Jiří. Historie a vývoj HTML. *htmlguru.cz.* [Online] [Citace: 29. Červenec 2013.] http://htmlguru.cz/uvod-historie.html.

Kraval, Ilja. 2002. *Design Patterns v OOP.* [e-kniha] 2002.

Krčmář, Petr. 2009. Sun nakonec kupuje firma Oracle: co bude s MySQL? *ROOT.CZ.* [Online] 21. Duben 2009. [Citace: 15. Červenec 2013.] http://www.root.cz/clanky/sun-nakonec-kupuje-firma-oracle-co-bude-s-mysql/.

Krug, Steve. 2007. *Webdesign: Nenutťe uživatele přemýšlet.* Brno : Computer Press, a.s., 2007. 80-251-1291-8.

Lacko, Luboslav. 2007. *Oracle: Správa, programování a použití databázového systému.* Brno : Computer Press, 2007. 978-80-251-1490-2.

Lubbers Peter, Albers Brian, Salim Frank. 2011. *HTML5: Programujeme moderní webové aplikace.* Brno : Computer Press, a.s., 2011. 978-80-251-3539-6.

Michael, Peacock. 2012. *Programujeme vlastní sociální síť v PHP 5.* Brno : Computer Press, 2012. 978-80-251-3626-3.

Mrozek, Jakub. 2006. OOP v PHP: Vzor Singleton. *interval.cz.* [Online] 15. Únor 2006. [Citace: 17. Červenec 2013.] http://interval.cz/clanky/oop-v-php-vzor-singleton/.

Skřivan, Jaromír. 2000. Databáze a jazyk SQL. *interval.cz.* [Online] 4. Srpen 2000. [Citace: 1. Srpen 2013.] http://interval.cz/clanky/databaze-a-jazyk-sql/.

Zajíc, Petr. 2005. MySQL (1) - pestrý svět databází. *Linuxsoft.cz.* [Online] 1. Březen 2005. [Citace: 29. Červenec 2013.] http://www.linuxsoft.cz/article.php?id_article=731.

—. 2004. PHP (1) - Historie a budoucnost. *Linuxsoft.cz.* [Online] 27. Květen 2004. [Citace: 27. Červenec 2013.] http://www.linuxsoft.cz/article.php?id_article=171.

Příloha A – Zdrojový kód třídy Databaze.class.php

```
<?php
class Databaze {
	private static $HOSTITEL = 'localhost';
	private static $UZIVATELSKEJMENO = 'jmeno';
	private static $UZIVATELSKEHESLO = 'heslo';
	private static $DATABAZE = 'databaze';
	private $pocetProvedenychDotazu = 0;
	private $posledniProvedenyDotaz;
	private $posledniProvedenyDotazVysledek = null;
	private $posledniProvedenyDotazId = null;

	public function __construct() {
		$this->novePripojeni(self::$HOSTITEL,
self::$UZIVATELSKEJMENO, self::$UZIVATELSKEHESLO, self::$DATABAZE);
	}
	public function novePripojeni($hostitel, $uzivatel, $heslo,
$databaze) {
		$DB = DatabazePripojeni::getInstance($hostitel, $uzivatel,
$heslo, $databaze);
		$link = DatabazePripojeni::vytvorSpojeni();
		$database = DatabazePripojeni::vyberDatabazi();
	}
	public function dotaz($dotaz) {
		$this->posledniProvedenyDotaz = $dotaz;
		$this->pocetProvedenychDotazu++;
		$vysledek = mysql_query($dotaz);
		$this->posledniProvedenyDotazId = mysql_insert_id();
		$this->posledniProvedenyDotazVysledek = $vysledek;
		return $vysledek;
	}
	public function dotazPocetRadku($dotaz = null){
		if($this->posledniProvedenyDotazVysledek != null){
			if($dotaz == null){
				return mysql_numrows(
$this->posledniProvedenyDotazVysledek);
			}
			else{
				$this->dotaz($dotaz);
				return mysql_numrows(
$this->posledniProvedenyDotazVysledek);
			}
		}
		else{
			return 0;
		}
	}
	public function dotazRadekVysledku($radekVysledku, $nazevAtributu){
		if($this->posledniProvedenyDotazVysledek != FALSE &&
is_resource($this->posledniProvedenyDotazVysledek)){
			return mysql_result(
$this->posledniProvedenyDotazVysledek, $radekVysledku, $nazevAtributu);
		}
	}
	public function dotazVygenerovaneId() {
		return $this->posledniProvedenyDotazId;
	}
}
?>
```

Příloha B – Zdrojový kód třídy DatabazePripojeni.class.php

```
<?php
class DatabazePripojeni {

	private static $instance = NULL;
	private static $link = NULL;
	private static $database = NULL;

	private function __construct() {
	}

	public static function getInstance(
				$hostitel, $jmeno, $heslo, $databaze) {
		if (self::$instance == NULL) {
			self::$instance = new DatabazePripojeni();
			self::$link = @mysql_connect(
				$hostitel, $jmeno, $heslo);
			self::$database = @mysql_select_db($databaze);
		}
		return self::$instance;
	}

	public static function vytvorSpojeni() {
		if (self::$link == NULL) {
			throw new Exception("Nelze se pripojit k serveru.");
		}
		return self::$link;
	}

	public static function vyberDatabazi() {
		if (self::$database == NULL) {
			throw new Exception("Nelze se pripojit k databazi.");
		}
		return self::$database;
	}

}
?>
```