Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra informatiky a výpočetní techniky

Diplomová práce

Systém pro správu a sledování realizace zakázek pro reklamní agenturu

Plzeň, 2012 Lukáš Kadlec

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 16.5.2012 ……

Lukáš Kadlec

Název práce: Systém pro správu a sledování realizace zakázek pro reklamní agenturu

Autor: Lukáš Kadlec

Katedra: Katedra informatiky a výpočetní techniky

Vedoucí diplomové práce: Ing. Martin Zíma, Ph.D.

Abstrakt: Tento dokument je součástí diplomové práce a popisuje realizaci systému pro správu a sledování průběhu zakázek v reklamní agentuře. Dokument je rozdělen do několika částí: Popis agentury, vysvětlení použitých technologií, specifikace požadavků na systém, popis průběhu práce a popis výsledného systému. Popis systému nepopisuje kompletní implementaci systému, ale pouze výsledné řešení a detailněji některé zajímavé části.

Klíčová slova: sledování zakázek, PHP, JavaScript, informační systém, komunikace se zákazníkem, TrafficSystem, webová aplikace

Title: The system for managing and monitoring the implementation of contracts for advertising agency

Author: Lukáš Kadlec

Department: Department of Computer Science and Engineering

Supervisor: Ing. Martin Zíma, Ph.D.

Abstract: The document is a part of the graduation theses and it describes the realization of a system administrating and observing a course of contracts in an advertising agency. The document is divided into several parts: a description of the agency, an explanation of used technologies, a specification of demands on the system, a description of the work progress and a description of the final system. The system description does not describe the complete system implementation, but only the final solution and some interesting parts in detail.

Keywords: monitoring of contracts, PHP, JavaScript, information system, communication with the customer, TrafficSystem, web application

OBSAH

1 Úvod

Informační systémy jsou dnes součástí každé větší fungující společnosti. Zjednodušují získávání i rozšiřování informací, umožňují udržet přehled nad chodem společnosti, a tím přímo napomáhají jejímu růstu.

Společnost Kristián s.r.o. již dlouhou dobu hledá vhodné řešení pro správu zakázek z pohledu sledování realizace a využití lidských zdrojů. Kvůli jejím specifickým požadavkům byly vyloučeny již mnohé systémy a nakonec bylo rozhodnuto, že si nechají vytvořit řešení na míru jejich potřebám.

Cílem této diplomové práce je vytvoření systému, který bude pokrývat potřeby společnosti v dané oblasti. Bude se jednat o webovou aplikaci umožňující sledování realizace zakázky a distribuci úkolů jednotlivým pracovníkům. Výsledná aplikace bude nasazena do pilotního provozu, tedy spuštěna v cílovém prostředí s reálnými daty a omezenou skupinou uživatelů.

2 SPOLEČNOST A SOUČASNÝ STAV

Poznat strukturu a fungování společnosti se ukázalo jako velmi důležité při vývoji. Systém musí zapadnout do existujícího chodu společnosti, jelikož zásahy do zvyklostí nejsou ve většině případů vítány. Také při sběru požadavků nemusí být vždy zcela jednoduché získat přesné nároky na funkčnost aplikace. Pokud při schůzkách známe cílové prostředí, můžeme při zjišťování informací mnohem lépe řídit diskusi a získat tak potřebná data.

2.1 STRUKTURA SPOLEČNOSTI

Společnost Kristián s.r.o. je nesíťová[1] česká agentura, která poskytuje klientům maximální servis (tzv. fullservis) na co nejvyšší úrovni. Její součástí jsou všechna důležitá oddělení, např. kreativní, mediální, produkční, account (péče o klienta), DTP (desktop publishing), audio-video studio aj. V mediálním oddělení se připravuje účinná strategie, kreativní má za úkol přicházet s neotřelými řešeními jak po stránce formální, tak obsahové. Oddělení péče o klienty se věnuje zákazníkům s maximální péčí. Společnost Kristián spolupracuje se sestřinými společnostmi Estetica (nabízí kompletní řešení v oblasti internetového marketingu) a Alias Consulting (poskytuje služby v oblasti personální inzerce a poradenství), které jsou plně k dispozici v rámci zadaného projektu.

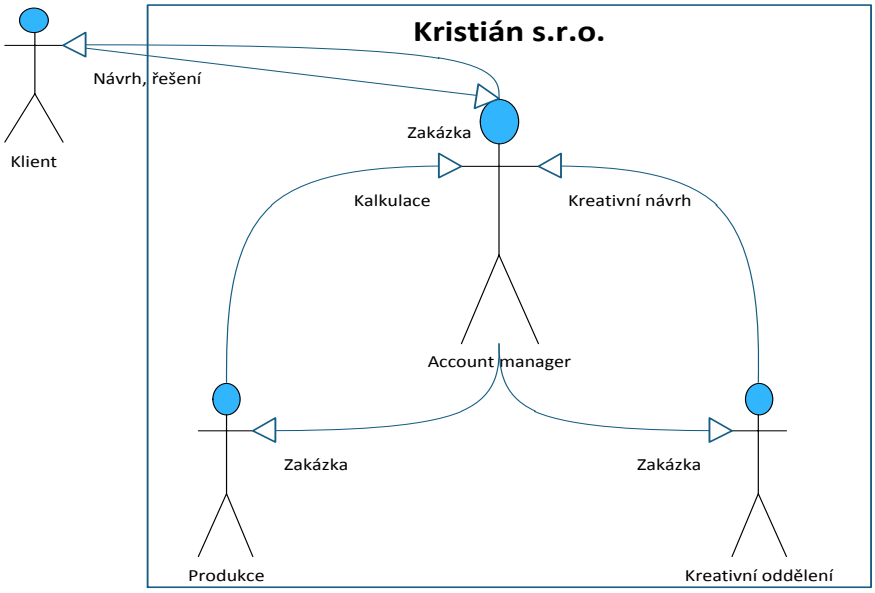
2.2 SOUČASNÝ STAV

Jako v každé firmě řeší i v Kristiánu problém komunikace uvnitř společnosti. Problematika je o to důležitější, že firma podnikající v oblasti komunikace má problémy s komunikací mezi svými zaměstnanci.

Důležité je sledování a výměna informací o stavu a průběhu projektu (zakázky). Na diagramu (viz Obrázek 2.1) je znázorněn průběh zakázky společností. Vše začíná u klienta, který přichází za account manažerem (dále jen ACC) s určitým požadavkem. ACC svolá poradu s produkčním a kreativním oddělením. Produkce připraví kalkulaci, kreativní oddělení vytvoří návrh řešení. Tato data putují do rukou ACC, který je konzultuje s klientem. Případné připomínky jsou přes ACC vráceny

[1] Nesíťová agentura znamená, že nepatří do žádné nadnárodní sítě společností.

příslušným oddělením. ACC má tedy roli zprostředkovatele a současně zodpovídá za termíny, informace apod.



OBRÁZEK 2.1 - ŘEŠENÍ ZAKÁZKY

Jelikož v těchto procesech dochází k oněm problematickým komunikačním ztrátám, řešila společnost, proč tomu tak je. Závěr je, že z důvodů řešení mnoha zakázek najednou, nestíhá ACC sledovat, jaké informace komu předal, a tím dochází ke ztrátám. Možným řešením je zavedení nové pozice tzv. traffic manager. Tuto roli je možné zařadit na stejnou úroveň jako ACC. Ulehčuje práci ACC tím, že přebírá odpovědnost za respektování veškerých termínů, které jsou spojené s prací na zakázce.

Aby se nemusela zavádět nová pozice, a tím např. i další oddělení, bylo rozhodnuto, že se problém sledování průběhu zakázky (termínů) pokusí vyřešit nasazením informačního systému namísto nových zaměstnanců.

Agentura v současné době využívá interní software pro sledování zakázek, který je sleduje pouze z pohledu fakturace. Nová objednávka je zaevidována, po dokončení jsou vyplněny časové údaje na úkolech pod zakázku spadajících a výstupem je faktura pro zákazníka. Cílem poptávané aplikace (dále TrafficSystem) je načíst ze zakázkového systému zakázky týkající se daného oddělení, vytvořit úkoly, umožnit

rozdělení úkolů jednotlivým zaměstnancům a zpět do zakázkového systému reportovat čas strávený na zakázce.

Zároveň se nyní v agentuře používají další dvě aplikace, které nahrazují funkčnost budoucího TrafficSystem-u. Jedná se o webovou aplikaci ShiftPlanning[2], která umožňuje např. rozdělování úkolů zaměstnancům a sledování jejich vytíženosti. Ovšem neumožňuje jednotlivé úkoly sdružovat do projektů a statistiky o práci zaměstnanců (např. jaký byl plánovaný čas na úkol a kolik zaměstnanec na úkolu strávil) jsou detailní, ale nepřehledné. Druhou aplikací je desktopový software Things[3], který pomáhá sledovat plnění celého projektu.

Hlavním cílem bude aplikace spojující funkčnost obou jmenovaných plánovacích programů, umožňující sledování projektů na požadovaných úrovních.

[2] ShiftPlanning - http://www.shiftplanning.com/
[3] Things - http://culturedcode.com/

3 POUŽITÉ TECHNOLOGIE A NÁSTROJE

V této kapitole budou uvedeny použité technologie a nástroje, které byly při vývoji aplikace použity.

3.1 PHP

Serverová část aplikace je psána ve skriptovacím jazyce PHP[4]. Použitá verze jazyka je 5.3, podpora této verze webovým serverem je nutnou podmínkou pro provoz aplikace. PHP bylo součástí požadavků na aplikaci, kvůli budoucí údržbě. A jelikož složitost cílové aplikace nijak nepřekračuje možnosti tohoto jazyka, nebyl důvod se této podmínce bránit.

3.1.1 NETTE FRAMEWORK

Framework-y jsou nepostradatelným pomocníkem většiny programátorů. Ulehčují a zpříjemňují psaní zdrojového kódu. Přebírají často řešené problémy a programátor se tak může soustředit na psaní vlastní logiky programu.

Nette je český PHP Framework (dále jen FW), vzniká od roku 2004 a od roku 2008 je uvolněn jako open source. Nette jsem si zvolil přibližně před dvěma roky, kdy jsem se rozhodoval, který z existujících FW používat. Nad konkurenci jej povýšila jednak velmi aktivní česká komunita, rychlost, FW vyšel z testu [1] jako jeden z nejvýkonnějších, a především kvůli velikosti. Nette není tak rozsáhlé, jako například Zend, což přináší užitek nejen v počátcích, kdy je možné jej rychleji pochopit, ale i následně, kdy není k výsledné aplikaci potřeba připojovat mnoho nevyužívaného kódu.

Je použita aktuální stabilní verze 2.0.

3.2 JAVASCRIPT

Aplikační logika na straně klienta je psána v jazyce JavaScript (dále jen JS). S tímto jazykem jsem před začátkem práce neměl příliš velké zkušenosti a také s ním během vývoje bylo nejvíce potíží.

[4] PHP – PHP: Hytertext Preprocessor

3.2.1 *Framework jQuery*

Výsledná aplikace je webová, přístupná různými webovými prohlížeči na různých platformách. Potlačit rozdíly mezi odlišnými klientskými prostředími pomůže mimo jiné JS knihovna jQuery.

Použitá verze je 1.7.2.

3.2.2 *AJAX (AJAJ)*

AJAX[5] je označení pro skupinu technologií pro vývoj interaktivních webových stránek, které nevyžadují při načítání dat znovunačtení celé stránky. *X* ve zkratce značí XML[6], které je používané pro přenos dat. Pro označení mnou převážně používaných technologií, je lepší méně známá zkratka AJAJ, kde *J* značí JSON (viz kapitola 3.3). Dalšími možnými formáty přenosu dat pomocí AJAX-u je HTML (viz kapitola 3.5) nebo prostý text. Oba jmenované formáty byly při vývoji také použity.

3.3 JSON

JSON[7] je odlehčený formát pro výměnu dat. Je jednoduše čitelný i zapisovatelný člověkem a snadno analyzovatelný i generovatelný strojově. [2]

JSON jsem jako formát pro přenos dat zvolil proto, že má vynikající podporu v JavaScriptu (je založen na jeho podmnožině), tedy na straně klienta se s ním jednoduše pracuje. Současně na straně serveru lze jen jednoduše zakódovat z obyčejného pole pomocí PHP funkce *json_encode()* [3].

3.4 MySQL

Data aplikace je potřeba někde ukládat a v dnešní době je zbytečné uvažovat o jiném úložišti než je databáze. Vzhledem k rozsahu a potřebám vyvíjeného systému nejsou na databázi kladeny žádné vysoké nároky a mohl jsem proto zvolit jakékoli řešení. Vybral jsem si MySQL, především kvůli předešlým zkušenostem.

3.5 HTML, CSS

Přestože finální specifikace značkovacího jazyka HTML[8] verze 5 je plánována až na rok 2014, mohl jsem si jej již dovolit použít, díky specifikaci požadavků,

[5] AJAX - Asynchronous JavaScript and XML
[6] XML - Extensible Markup Language
[7] JSON - JavaScript Object Notation
[8] HTML - HyperText Markup Language

ve které je uvedeno, že systém bude provozován na aktuálních prohlížečích, které potřebnou funkčnost již mají. Novou specifikaci jsem se rozhodl použít, jelikož umožňuje kratší zápis tagů tím, že není nutné uvádět některé atributy. Například JavaScript-ový kód se ohraničuje značkou *<script></script>*. V předchozích verzích jazyka bychom museli uvádět v jakém skriptovacím jazyce obsah je. Jelikož se v dnešní době převážně jedná o JavaScript, pokud není vyplněn, je předpokládán on.

Používáme-li pro strukturování stránek HTML, je dnes samozřejmostí, že pro vzhled stránek použijeme CSS[9]. Zvolená verze 3 není stejně jako HTML5 ještě zcela specifikována. Výhody, ve formě např. kulatých rohů nebo přechodů barev, jsou natolik výrazné, že zcela zastíní případné zápory. A vzhledem k již zmiňovaným požadavkům na prohlížeče jsou nevýhody eliminovány na minimum.

3.6 MySQLWorkbench

Pro správu databáze jsem doposud používal pouze webový nástroj Adminer od Jakuba Vrány [4]. Prvotní impuls pro hledání jiného nástroje vyšel z potřeby vytvořit UML diagram databáze. Zvolený MySQL Workbench mi ovšem ukázal, že s jeho použitím může být práce s databází mnohem jednodušší, a nakonec zcela změnil můj zažitý vývojový proces.

Doposud průběh projektu ve zkratce obnášel vytvoření databáze, vytvoření aplikace, při níž docházelo k postupným úpravám databáze dle potřeb a výsledkem bylo nasazení aplikace. Dokumentace DB (ERA model) nevznikala, jelikož chyběl vhodný nástroj pro údržbu modelu. Upravovat jej ručně je nereálné a neaktualizovaný nemá smysl. Workbench umožňuje *Reverse Engineering*, pomocí kterého bylo možné jej nasadit v průběhu vývoje bez jakéhokoli narušení. Následné úpravy dále probíhaly modifikací modelu a *Forward Engineering* umožnil promítnout změny do DB.

3.7 NetBeans

Jako vývojové prostředí jsem použil IDE[10] NetBeans 7.1. Toto prostředí jsem volil především proto, že v něm už dlouhou dobu pracuji a plně mi vyhovuje. Umožňuje

[9] CSS - Cascading Style Sheets
[10] IDE - Integrated Development Environment, vývojové prostředí

tvorbu PHP projektů a existuje pro něj plugin pro Nette Framework, který velmi zjednoduší vývoj.

3.8 Mozilla Firefox

Jelikož výsledkem je webová aplikace, je potřeba pro zobrazování výsledků používat webový prohlížeč. Pro vývoj se mi nejvíce osvědčil prohlížeč Mozilla Firefox a především jeho plugin Firebug [5]. Tento doplněk umožňuje zobrazovat konzoli, do které můžeme např. JavaScript-em posílat ladící výpisy nebo se zde zobrazují chyby JavaScript-u, a především vypisuje AJAX-ovou komunikaci klienta se serverem. Velmi užitečná je záložka *Síť*, ve které je vidět veškerá HTTP[11] komunikace se serverem při načítání stránky, společně s časem, po který trval daný požadavek na server. To pomůže odhalit slabá místa v případě pomalého načítání aplikace.

Dalším užitečným rozšířením je FireLogger [6]. Ten oceníme ve chvíli, kdy se snažíme odhalit chybu aplikace na straně serveru, při AJAX-ovém požadavku. Do konzole se vypíše pouze informace, že požadavek skončil chybou. Nette obsahuje ladící nástroj, který v případě AJAX-ového požadavku odešle informace právě do popisovaného doplňku.

Aby byl Firefox „dokonalým“ pomocníkem při vývoji, je, krom instalace zmiňovaných doplňků, užitečné nastavit, aby po zobrazení chyby bylo možné rovnou přejít do zdrojového kódu na místo, kde k chybě došlo. Opět nám v tom pomůže ladící nástroj Nette, který popis chyby odesílá jako odkaz na protokol *editor://*. Stačí si vytvořit obslužný soubor, který zpracuje předané parametry, název souboru a číslo řádku s chybou, a do systému tuto obsluhu zaregistrovat. Přesný popis nastavení na [7].

3.9 Git

S verzovacími nástroji jsem se doposud setkal jen při semestrální práci z předmětu KIV/ASWI. Jednalo se o SVN[12]. Vědom si výhod plynoucích z používání těchto nástrojů, rozhodl jsem se vybrat si před začátkem práce vhodný verzovací systém.

[11] HTTP – HyperText Transfer Protocol
[12] SVN - Subversion

Původně jsem se chtěl zaměřit na to, jaký systém si vybrat, ale nikde jsem nenašel žádné rozumné porovnání. Stav sice není takový jako u jiných systémů, kdy někdo používá jeden a ostatní zatracuje, zde jsem se spíše setkal s tím, že jeden používají a o jiných mnoho neví. Z článků a diskusí vyplynulo, že pozornost by měla být směřována k distribuovaným verzovacím systémům (DVCS). Hlavními hráči na poli DVCS jsou Git, Mercurial a Bazaar. Bazaar není příliš populární a tak zbývají první dva zmiňované. Oba jsou na tom velmi podobně. V některých diskuzích byl Mercurial upřednostňován kvůli jednoduchosti a lepší podpoře pro Windows. To se již změnilo. Git již není pouze Linuxovou záležitostí a na Windows je jej možno bez problémů provozovat. Co se týče jednoduchosti, je Git pro základní použití také jednoduchý a naopak dostatečně „složitý“, aby v případě potřeby poskytl požadovanou funkčnost. Že se jedná o obří projekt, by tedy nemuselo být na škodu.

Jednoznačně určit, že jeden je nejlepší, nejde. Zvlášť po přečtení několika článků a diskuzí. Nicméně k určitému rozhodnutí to stačilo. Systémy si jsou opravdu velmi podobné, a když se člověk naučí pracovat s jedním, neměl by být velký problém, např. v případě objevení nějakého neřešitelného problému v jednom systému, přejít k jinému. Princip používání je stejný a dokonce i některé příkazy se shodují.

Pro verzování aplikace vyvíjené v rámci této práce jsem se, dle dostupných podkladů a názorů v diskuzích, rozhodl pro Git. Vlastnosti, které nahrávaly systému Mercurial, především podpora pro Windows, jsou již minulostí a o žádných nevýhodách Gitu jsem se nedočetl. Na rozdíl od Mercurial, který má prý problém s verzováním velkých souborů (nad 10MB).

4 SPECIFIKACE POŽADAVKŮ

Prvním krokem vývoje cílové aplikace bylo vytvoření specifikace požadavků. Výsledný dokument se stává podkladem pro následnou analýzu a na závěr slouží pro kontrolu předávaného řešení. Data obsažená v dokumentu specifikace požadavků jsou popsána v této kapitole.

4.1 ROZSAH PRODUKTU

Cílem TrafficSystem-u je webová aplikace pro distribuci úkolů jednotlivým pracovníkům s možnostmi:

- zadávání deadline
- zadávání časové náročnosti
- alokace úkolu v čase
- timesheet (reálné odpracované hodiny)
- linky na soubory (podklady, brief atp.)
- status úkolu (zadaný, rozpracovaný (stav v %), splněný)
- jednoduchost
- „píchačky“ (přihlášení a odhlášení se k práci)

Žádoucí je propojenost se stávajícím zakázkovým systémem (stačí off-line formou nějaké synchronizace).

TrafficSystem bude načítat ze zakázkového systému specifikace (číslo, název, klient) jednotlivých zakázek. Je nutné mít možnost k zakázkám v TrafficSystem-u přiřadit příznak, že se jedná o grafickou (mediální nebo produkční) zakázku (přičemž mohou být přiděleny všechny současně), aby byla možnost odfiltrovat zakázky, které se oddělení netýkají.

Zakázkový systém načítá hodiny z timesheetu z TrafficSystem-u.

4.2 KOMUNIKACE MEZI SYSTÉMY

Zakázkový systém nemá v současné době žádné API pro komunikaci s jinými systémy. Je potřeba navrhnout vhodné řešení pro výměnu dat, které bude následně implementováno do stávajícího systému.

Pokud automatická výměna nebude realizovatelná, může být prováděna ručně.

4.3 UŽIVATELÉ

Aplikaci budou používat čtyři skupiny uživatelů s rozdílnými požadavky a možnostmi.

4.3.1 ADMINISTRÁTOR

- Nastavuje systém.
- Spravuje uživatele.

4.3.2 ŘEDITEL ODDĚLENÍ

- Má právo nahlížet na stav všech projektů jeho oddělení.
- Vidí kalendáře svých podřízených.
- Přebírá nové zakázky z ekonomického systému (vytváří projekty).
- Vkládá brief.
- Propojení s podklady.
- Vytváří a rozděluje úkoly svým podřízeným.
- Přiděluje zúčastněné osoby (např. dohlížející osobu – obchodního zástupce, pověřeného komunikací s klientem).
- Přebírá a potvrzuje dokončení úkolu a projektu.
- Provádí export dat (hodiny na dokončených projektech) do ekonomického systému.

4.3.3 PRACOVNÍK ODDĚLENÍ

- Přijímá (odmítá) úkoly do svého časového plánu.
- Vytváří si úkoly (možnost bez délky trvání).
- Vidí svůj kalendář a stav projektů, ke kterým je přiřazen
- Mění stav úkolu (rozpracovaný, splněný).
- Uvádí na kolik % je úkol dokončen.
- Přikládá soubory k projektu – výsledek (data do produkce, ke korektuře).

4.3.4 PRACOVNÍK ACCOUNT ODDĚLENÍ

- Má shodné možnosti obsluhy systému jako **Pracovník oddělení**.
- Má právo nahlížet na stav projektů.

4.4 FUNKČNÍ POŽADAVKY

4.4.1 ZADÁNÍ PROJEKTU

Import nových zakázek
Do TrafficSystem-u se nejprve naimportují zakázky ze zakázkového systému.

Označení zakázek týkajících se grafiky[13]
Pokud se mezi zakázkami vyskytují zakázky pro jiné oddělení, ručně se vyseparují jen ty, týkající se grafického oddělení. Každá přijatá zakázka dostane přidělenu barvu.

Připojení zadání (brief)
K novému projektu je možno připojit zadání v podobě 1 - N souborů. Připojení je provedeno nahráním nového souboru, nebo vložením odkazu na již uložený soubor.

Připojení podkladů
K novému projektu je možno připojit podklady v podobě 1 - N souborů. Připojení je provedeno nahráním nového souboru, nebo vložením odkazu na již uložený soubor. Především se budou vkládat odkazy, jelikož poklady bývají již na serveru.

Zvolení odpovědných osob
K projektu je přiřazena odpovědná osoba, obchodní zástupce nebo zástupce produkce.

Určení priority zakázky
Každá zakázka obdrží prioritu v rozmezí 1 (nízká) – 5 (vysoká).

4.4.2 ZADÁNÍ JEDNOTLIVÝCH ÚKOLŮ
Projekt může obsahovat 1 – N úkolů. Každý úkol má totožné atributy.

Název úkolu
Krátké vyjádření cíle úkolu.

Výběr zaměstnance
Na daný úkol je přiřazen zaměstnanec. Je nabídnut výčet zaměstnanců.

Začátek plnění úkolu
Datum a čas, kdy má zaměstnanec na úkolu začít pracovat.

[13] Týká se stavu, kdy bude aplikace automaticky synchronizována se zakázkovým systémem.

Mezní termín splnění úkolu
Zadává se termín za počátečním datem, kdy má být úkol nejpozději dokončen. Uvádí se datum a čas. Předvyplněné je datum, shodné s počátečním datem, a časový údaj o hodinu větší.

Časová náročnost[14]
Plánovaná časová náročnost úkolu. Automaticky se předvyplní hodnota „2 hodiny". Plánovaná časová náročnost může být změněna, ale nemůže být větší než rozdíl mezi mezním termínem a začátkem úkolu.

Textové zadání (poznámky)
Úkol může volitelně obsahovat textové zadání, které zaměstnanci popisuje cíl úkolu nebo vysvětluje nejednoznačné části úkolu.

Odkazy na podklady
Úkol automaticky přebírá všechny odkazy na podklady, které jsou přiřazeny k projektu. Zadavatel určí, které podklady patří k úkolu (může odebrat).

4.4.3 VLOŽENÍ ÚKOLU DO KALENDÁŘE

Úkol se po vytvoření umístí na zadané volné místo v kalendáři odpovědného zaměstnance. V případě, že časové okno nestačí na náročnost úkolu, může nový úkol nahradit stávající, v případě, že má stávající úkol nižší prioritu. Na tuto skutečnost je zadavatel upozorněn a požádán o potvrzení nahrazení. Stávající úkol je, v případě nahrazení, odebrán zaměstnanci a vložen do seznamu nových úkolů (není přiřazeno počáteční datum a přiřazen zaměstnanec).

Pokud nelze úkol umístit, je o tom zadavatel informován a může upravit umístění úkolu.

Ze seznamu nových úkolů může být úkol umístěn do kalendáře editací vlastností nebo tažením myší na požadované místo.

Veškeré úkoly jsou nejprve rozmisťovány u zadavatele a následně hromadně publikovány, aby se zamezilo opakovanému informování o novém úkolu při přesunech. Přesuny úkolů budou umožněny tažením myší.

[14] Tato se nestanoví v případě produkce a accountů – nutnost mít možnost toto vypnout.

4.4.4 PLNĚNÍ ÚKOLU

Zaměstnanec vidí přehled svých úkolů. Přiřazený úkol může odmítnout (kvůli nedostatku času - např. prodloužení předcházejícího úkolu - nebo z důvodu nedostatečných zkušeností). Po začátku práce na úkolu změní statut na *Rozpracovaný*. Poté nelze grafikovi úkol odebrat nebo jej přidělit jinému.

Po dokončení úkolu změní status na *Dokončené.* Potvrzuje nebo mění počet odpracovaných hodin a předává informaci vedoucímu a odpovědné osobě. V případě překročení plánované časové náročnosti jsou informováni všichni zúčastnění.

K dokončenému úkolu může zaměstnanec přiložit soubory.

4.4.5 PŘERUŠENÍ ÚKOLU

V případě, že zaměstnanec obdrží např. úkol s vyšší prioritou, bude možno úkol přerušit a naplánovat jeho pokračování na jiný termín.

4.4.6 KALENDÁŘ

Nastavení pracovního kalendáře provádí vedoucí oddělení.

a) Úkoly se přidělují pouze na pracovní dny a v nastavenou pracovní dobu.
b) Pracovní doba lze nastavit pro každého grafika zvlášť.
c) Kalendáři lze nastavit svátky jako volné dny (správa svátků).
d) Je možné nastavit individuální dovolenou.
e) Je možné nastavit hromadnou dovolenou (nařízená firmou).
f) Lze zablokovat určitý den (víkend, svátek, hromadná dovolená) a hodiny po pracovní době lze na konkrétní den odblokovat a umístit úkoly.

V kalendáři jsou úkoly odlišeny barvami přidělenými projektu a označeny dle priority projektu.

Odpovědná osoba může měnit umístění úkolu tažením myši. Po přetažení je vyžádáno potvrzení nového umístění, po potvrzení odchází zpráva zúčastněným osobám.

Kliknutím na pole s úkolem lze vyvolat dialog s nastavením úkolu (vedoucí oddělení) nebo detailem úkolu (zaměstnanec).

4.4.7 PŘEHLED PROJEKTŮ

Vedoucí může zobrazit přehled všech projektů v textové podobě ve formě tabulky. Zaměstnanec v takovém výpisu vidí pouze úkoly, na kterých pracuje.

Výpis projektů je možné filtrovat podle:

a) stavu (nové, rozpracované, dokončené)
b) klienta
c) projektu
d) pověřených osob
e) dnů

Řazení výpisu projektů je sestupné podle mezního data ukončení.

Každý záznam výpisu odkazuje na detail projektu. V detailu jsou následující informace:

a) klient
b) brief (zadání)
c) podklady
d) seznam úkolů
e) odpovědné osoby
f) soubory od grafika - výsledek

Je možná editace projektu, především lze přidat:

a) úkol
b) podklad
c) soubor
d) odpovědnou osobu

4.4.8 DOKONČENÍ PROJEKTU

Po dokončení projektu je vedoucí vyzván k vytvoření exportu hodin na projektu pro ekonomický systém. Je odesláno oznámení zúčastněným osobám.

4.4.9 SPRÁVA UŽIVATELŮ

Každý uživatel má v systému svůj profil. Ten umožňuje zveřejnit fotografii (avatar), celé jméno, kontaktní telefon, IM (ICQ, jabber).

Uživatel si může měnit své přihlašovací heslo, druh zpráv, které chce dostávat a termín dovolené.

Nové uživatele vytváří vedoucí oddělení.

4.4.10 KOMUNIKACE

Aplikace nemusí realizovat žádné vlastní řešení pro komunikaci s uživateli. Veškeré zprávy, upozornění a hlášení budou posílány na e-mail uživatele.

Uživatel si může nastavit, jaké zprávy chce dostávat.

4.5 POŽADAVKY NA SNADNOST POUŽÍVÁNÍ

Při návrhu bude kladen důraz na jednoduché a intuitivní ovládání. Jako příklad lze použít uživatelské rozhraní aplikace ShiftPlanning.

Údaje o práci zaměstnance, kolik měl odpracovat a kolik odpracoval, bude kromě číselného údaje možno zobrazit v grafu.

4.6 ZABEZPEČENÍ

Aplikace je přístupná pouze po přihlášení platným uživatelským jménem a heslem. Jako uživatelské jméno bude sloužit e-mail uživatele.

Uživatel si po přihlášení může měnit své heslo.

4.7 SPOLEHLIVOST

Aplikace je závislá na dostupnosti serveru, bez něj nelze aplikaci TrafficSystem (webové rozhraní) používat.

4.8 UDRŽOVATELNOST

Vzhledem k budoucí údržbě je pro vývoj aplikace preferován skriptovací jazyk PHP.

4.9 VÝKONNOST

Na jakoukoli reakci systému bude uživatel čekat maximálně 5 sekund.

4.10 PROHLÍŽEČ

Uživatel bude schopen používat aplikaci pomocí internetových prohlížečů Internet Explorer 9, Mozilla Firefox 8, Google Chrome 15, Opera 11.52 a Safari 5[15].

Je předpokládáno defaultní nastavení prohlížeče, především povolený JavaScript a povolené zobrazování obrázků.

[15] Uváděné verze prohlížečů jsou aktuální v době psaní DSP. U všech platí, že verze je minimální požadovaná.

Aplikace bude používána na počítačích Mac i PC.

4.11 Požadavky dokumentaci a nápovědu

Vzhledem k předpokládané intuitivnosti rozhraní nejsou žádné požadavky na uživatelskou dokumentaci ani nápovědu. Postačí tooltip nápověda u prvků rozhraní.

4.12 Uživatelská rozhraní

Aplikaci bude možné obsluhovat pouze přes webové rozhraní.

5 PRŮBĚH PROJEKTU

Jelikož má společnost sídlo v Praze, byly uskutečněny pouze dvě osobní schůzky a zbytek komunikace probíhal prostřednictvím elektronické pošty, případně telefonicky. V této kapitole bude popsáno, jak probíhaly důležité milníky komunikace.

5.1 DEFINICE ZADÁNÍ

První schůzka proběhla s ředitelem kreativního oddělení. Byl to on, který stál za vznikem požadavku na vznik systému pro své oddělení, jelikož nenašel existující řešení, které by splňovalo veškeré jeho požadavky. V současné době používá webovou aplikaci ShiftPlanning a desktopový software Things. Jedná se o řešení, které společnou funkčností splňuje požadavky, a tak cílem diskuse bylo probrat části systémů, které by měl obsahovat výsledný systém.

Současně byla řešena komunikace se stávajícím zakázkovým systémem. Synchronizace mezi systémy by velmi ulehčila práci, kdyby časy, které budou stráveny na projektu, byly automaticky vkládány do zakázkového systému. Zakázkový systém vyvinul před lety správce sítě společnosti. Vzápětí proběhla schůzka také s ním. Bohužel se nepodařilo vysvětlit, že nový systém nemá za cíl nahradit stávající zakázkový systém, a tak diskuse probíhala spíše na téma, že je zbytečné jej vyvíjet. Nicméně výsledkem bylo, že by bylo možné přistupovat přímo k databázi a že vymyslí nějaké řešení.

Závěrem schůzky bylo, že automatická synchronizace není až tolik důležitá. Vytvoří se systém, který bude fungovat samostatně a propojení se zrealizuje následně. Byl vznesen požadavek na dodání harmonogramu projektu a grafického návrhu, jako podkladů na poradu společnosti.

Výsledky schůzky byly sepsány jako dokument specifikace požadavků. Na jeho základě byl vytvořen grafický návrh uživatelského rozhraní, který z velké části přebírá rozložení aplikace ShiftPlanning, na jejíž rozhraní je zadavatel zvyklý. A byl vytvořen harmonogram projektu, obsahující 7 iterací po 1-2 týdnech.

5.2 Porada společnosti

Po dodání grafického návrhu a harmonogramu byl systém diskutován na nejbližší poradě společnosti. Myšlenka ředitele kreativního oddělení vzbudila zájem i u ostatních oddělení. Tato skutečnost samozřejmě přinesla odlišné požadavky na celý systém. Bohužel změna musela být akceptována, jelikož systém se v případě, že jej nebudou využívat všechna oddělení, stával pro společnost nezajímavým.

Komunikace probíhala pomocí e-mailů. Postupnými oboustrannými úpravami specifikace požadavků byly zapracovány změny v požadavcích. Dokument byl schválen a začala se vyvíjet aplikace.

5.3 Ukázka řešení

Druhá schůzka v sídle společnosti se uskutečnila při nasazení aplikace v ukázkovém provozu. Aplikace byla nasazena na testovacím serveru a naplněna ukázkovými daty.

Ukázka systému proběhla za účasti ředitelů oddělení grafiky a produkce. Byly prezentovány jednotlivé funkce systému a sepisovány jednotlivé nedostatky. Jednalo se např. o přiblížení podoby kalendáře, Google Calendar, přizpůsobení určitých částí uživatelského rozhraní zvyklostem uživatelů, nevýrazné (přehlédnutelné) pole pro popis úkolů. Přijaty byly i návrhy na doplnění chybějící funkcionality, např. zobrazení termínu dokončení projektu při vytváření úkolu.

Současně byly vyjasněny části aplikace, které dosud nebyly implementovány z důvodu chybějících informací. Jednalo se především o ujasnění funkce a využívání souborového serveru, jelikož s ním měl systém komunikovat. Bylo domluveno, že není potřeba realizovat nahrávání souborů na server, veškerá data jsou již v úložišti nebo budou nahrána jinou cestou. Ze serveru se budou data pouze číst. Byla zahájena komunikace se správcem sítě ohledně zpřístupnění serveru pro přístup systému a zřízení hostingu pro výslednou aplikaci.

5.4 Pilotní provoz

Bohužel správce sítě nevytvořil hosting pro systém na serveru společnosti a pilotní provoz byl spuštěn na testovacím serveru. Tato skutečnost příliš nevadí, pouze se při přechodu na finální nasazení změní adresa aplikace.

V systému byli vytvořeni ředitelé jednotlivých oddělení, kterým byly odeslány přístupové údaje. V současné době je systém používán všemi odděleními s omezeným počtem zaměstnanců.

6 STRUKTURA APLIKACE

Aplikace je založena na model-view-controllerstruktuře. Přesněji řečeno MVP, která je používána Nette Framework-em, a kde písmeno P značí presenter. Jejímu popisu je věnována samostatná kapitola 8 (Popis systému) na straně 31. Tato kapitola se zaměřuje na strukturu databáze a adresářovou strukturu.

6.1 STRUKTURA DATABÁZE

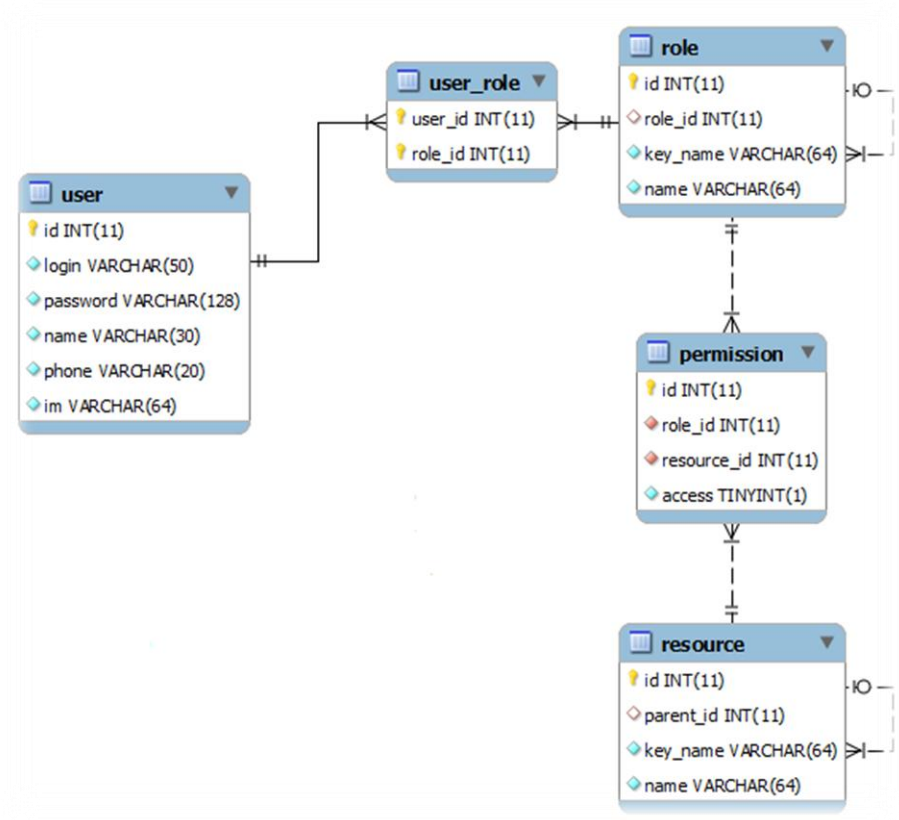
Databázi tvoří 14 tabulek a dva pohledy. Všechny tabulky používají úložiště InnoDB. Typ úložiště je zvolen především z důvodu podpory cizích klíčů. Databáze kompletně pokrývá potřeby aplikace. V této části bude rozložena na segmenty, ve kterých spolu jednotlivé tabulky nějak souvisí, a bude popsán jejich význam. Struktura celé databáze je vidět v příloze A – Datový ERA model.

6.1.1 UŽIVATELÉ A OPRÁVNĚNÍ

Tabulka **user** slouží k ukládání informací o uživatelích systému. Sloupce tabulky ilustruje Obrázek 6.1. Všechny sloupce jsou *NOT NULL*. Řetězec ve sloupečku *password* je osolený[16] MD5 hash uživatelova hesla.

Uživatel může mít přiřazenu jednu až N rolí. Vzniklou vazbu N:M řeší rozkladová tabulka **user_role**. Tabulka **role** obsahuje sloupec pro název role (*name*), unikátní označení (*key_name*), které je zbaveno diakritiky, mezery jsou nahrazeny podtržítky a všechna písmena jsou převedena na malá. *Key_name* slouží pro odkazování se na roli ve zdrojovém kódu, abychom si nemuseli pamatovat ID role a současně nemuseli používat pro tuto potřebu nevhodný název role. Poslední položkou tabulky je odkaz na nadřazenou roli (*role_id*). Každá role může mít jednu nebo žádnou rodičovskou roli. Slouží k vytvoření hierarchické struktury.

[16] Kryptografická sůl je prostředek pro zvýšení bezpečnosti kódovaných řetězců. K řetězci je přidáno několik znaků (tzv. sůl), unikátních pro daný systém a změní se tak hash pro stejný řetězec v různých systémech.

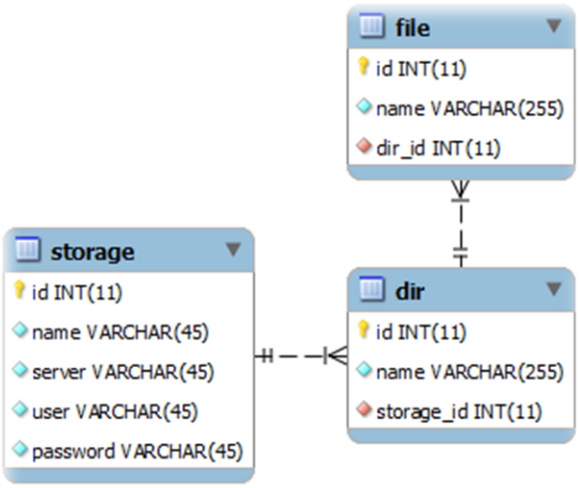


OBRÁZEK 6.1 - TABULKA UŽIVATELŮ A OPRÁVNĚNÍ.

Tabulka zdrojů (**resource**) slouží k ukládání částí systému, ke kterým je řízený přístup. Význam sloupců je totožný s tabulkou **role**. Poslední tabulkou tohoto segmentu je **permission**, která definuje přístup jednotlivých rolí (*role_id*) k určitým zdrojům (*resource_id*). Sloupeček *access* má defaultní hodnotu 0 – přístup zakázán, je-li nastaven na „1“, je přístup povolen.

6.1.2 ÚLOŽIŠTĚ SOUBORŮ, PODKLADY

Ze specifikace vychází požadavek na připojování podkladů k projektům a následně k úkolům. Pro tyto potřeby jsou v DB tři tabulky umožňující uložit informace o podkladech (viz Obrázek 6.2).



OBRÁZEK 6.2 - TABULKY PRO PODKLADY K PROJEKTŮM.

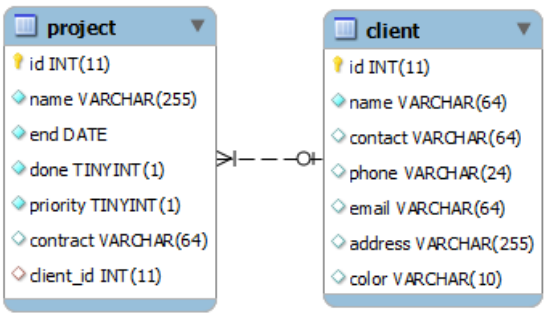
Tabulka **storage** obsahuje informace k prvnímu kroku při výběru podkladů. Jsou zde uloženy údaje pro připojení se k souborovému serveru. Název (*name*) je označení úložiště pro použití v systému, ostatní sloupce jsou potřebné údaje k přihlášení se k serveru.

Úložiště obsahuje složky. Pro každý projekt se na serveru zřizuje složka pro umisťování podkladů. Tabulka **dir** drží názvy složek, se kterými se pracuje v systému.

Ve složkách jsou jednotlivé podklady, potřebné pro vypracování dílčích úkonů na projektu. Názvy souborů, které jsou potřeba pro některé úkoly, jsou vkládány do tabulky **file**.

6.1.3 PROJEKTY

Důležitou vlastností systému je práce s projekty. Pro uložení projektu slouží tabulka **project**. Každý projekt má svůj název (*name*), termín dokončení (*end*), indikátor dokončení (*done*), který má defaultní hodnotu 0 (nedokončen), důležitost projektu (*priority*). Dále pak identifikátor zakázky v zakázkovém systému (*contract*), kvůli synchronizaci systémů, a v neposlední řadě je projekt vykonáván pro nějakého klienta (*client_id*).

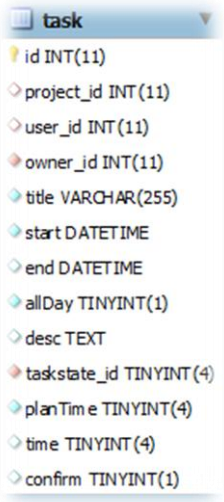


OBRÁZEK 6.3 – TABULKA PROJEKTŮ A KLIENTŮ

Na Obrázek 6.3 můžeme vidět, že sloupeček *client_id* může být NULL. Je to kvůli prvnímu projektu v systému, kterým je projekt „Bez projektu“, určenému pro nezařaditelné úkoly, a který není pro žádného klienta. Tabulka **client** obsahuje pouze jediný povinný sloupec, kterým je jméno klienta (*name*). Ostatní sloupce jsou nepovinné z důvodu importu dat ze zakázkového systému, kde jsou uváděny pouze jména klientů. Sloupeček *color* slouží k určení barvy klienta, na základě této barvy se obarví veškeré projekty pod klienta spadající a stejně tak dílčí úkoly.

6.1.4 Úkoly

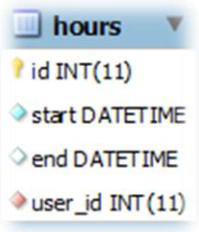
Hlavním cílem systému je přiřazování úkolů pracovníkům. Jednotlivé úkoly jsou ukládány v tabulce **task** (viz Obrázek 6.4). Úkol patří do nějakého projektu (*project_id*), může mít nějakého vykonavatele (*user_id*), ten ovšem nemusí být přiřazen, což nastane v případě odmítnutí úkolu vykonavatelem. Sloupec *owner_id* ukazuje na uživatele, který vytvořil úkol. *Title* je krátký výstižný název úkolu, pokud je potřeba k úkolu připojit více informací, využije se k tomu sloupec *desc*. Úkol je potřeba zaměstnanci naplánovat na určitý den, k čemuž slouží sloupec *start* a *end*, kde *end* může být NULL, ale v takovém případě musí být nastaveno, že se jedná o celodenní úkol (*allDay* nastaven na 1). Dále se úkolu určuje plánovaný čas plnění (*planTime*) a po dokončení je uložena skutečně odpracovaná doba (*time*). *Confirm* určuje, zda je před dokončením úkolu nutná kontrola vlastníkem úkolu, defaultně je nastaveno na 0, pokud ano, nastaví se na 1. A posledním sloupcem je *taskstate_id*, které ukazuje na číselník stavů úkolu.



OBRÁZEK 6.4 - TABULKA ÚKOLY

6.1.5 OSTATNÍ TABULKY A POHLEDY

Tabulka **hours**(viz Obrázek 6.5) vychází z požadavku na sledování docházky (odpracované doby) zaměstnanců. Dovoluje ukládat začátek práce (*start*), pokud zaměstnanec odešel z práce, uloží se čas konce práce (*end*), jinak je NULL, a samozřejmě ukazatel na pracovníka.



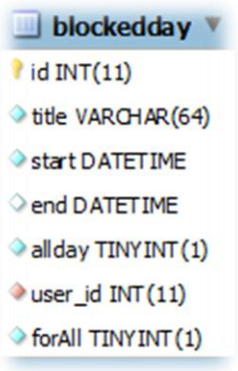
OBRÁZEK 6.5 - TABULKA DOCHÁZKA

Jelikož je pro zobrazování informací o přítomnosti uživatele důležitý celkový čas (rozdíl mezi koncem a začátkem práce), byl vytvořen pohled **hours_total**, který má stejné atributy jako tabulka **hours** a navíc sloupec *total*:

```
SELECTid, user_id,start, end,
(CASE WHEN ISNULL(end) THEN timediff(NOW(), start)
ELSE timediff(end,start) END) AS total FROM hours;
```

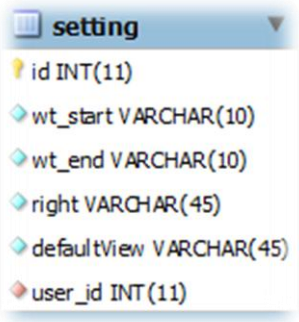
Pokud zaměstnanec ukončil práci, je pomocí MySQL funkce *timediff()*, vypočten rozdíl mezi časy. V případě, že ještě pracuje, je vrácen rozdíl začátku práce a aktuálního času.

Další tabulka (viz Obrázek 6.6) udržuje informace o blokovaných časech (**blockedday**) uživatelů. Většina atributů je shodných s tabulkou **task** (viz 6.1.4). Atribut *user_id* ukazuje na uživatele, který čas blokoval, a *forAll* značí, že je čas blokován pro všechny uživatele, kteří jsou v hierarchii systému pod tvůrcem bloku. Defaultně je nastaven, že není (0).



OBRÁZEK 6.6 - TABULKA BLOKOVANÝCH DNŮ

Poslední tabulkou, plynoucí z potřeby individuálního nastavení systému, je **setting**. Ta umožnuje uchovat informace o začátku a konci pracovní doby (*wt_start, wt_stop*), nastavení prvků kalentáře (*right, defaultView*) a samozřejmě ukazatel na uživatele, který si data nastavil.



OBRÁZEK 6.7 - TABULKA NASTAVENÍ

Ke kompletnímu popisu prvků datového modelu schází ještě pohled *userprojects*:

```
SELECT user.id as user_id, project.* FROM user,user_project,project
WHERE user.id=user_project.user_id AND
user_project.project_id=project.id AND project.done='0' UNION SELECT
user.id, project.* FROM user,task,project WHERE user.id=task.user_id
AND task.project_id=project.id AND project.done='0';
```

Tento pohled nám umožňuje propojit uživatele se všemi projekty, které jej zajímají. Jsou to vždy nedokončené projekty, ke kterým je přiřazen jako zodpovědná osoba nebo na kterých pracuje (má přidělen úkol spadající pod projekt).

6.2 Adresářová struktura

Adresářová struktura aplikace přesně dodržuje strukturu doporučenou použitým Nette Framework-em.

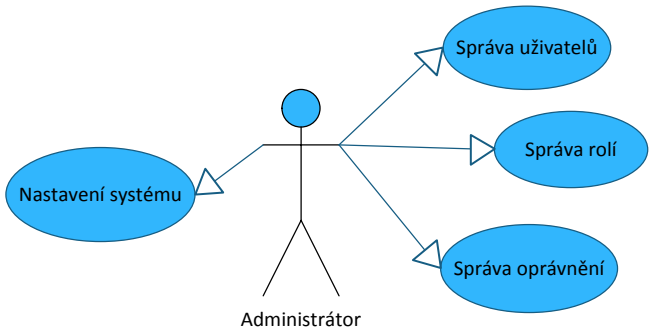
- **app** obsahuje veškeré zdrojové soubory spouštěné na straně serveru a šablony aplikace. Dále se větví na:
  - **config**, configurační soubory aplikace
  - **models**, zdrojové soubory datové vrstvy
  - **presenters**, zdrojové soubory presenční vrstvy
  - **templates**, latte šablony
- **libs** obsahuje knihovny třetích stran použité v aplikaci, především samotný Nette Framework.
- **log, temp** slouží k logování chyb a pro potřeby kešování a session. Musejí mít nastavena oprávnění pro zápis do obou složek.

Všechny tyto složky je z důvodu bezpečnosti doporučováno umístit do neveřejné části webového serveru, tedy o úroveň výše než ukazuje document root serveru. Opačným případem je poslední složka adresářové struktury **www**. Její obsah se naopak musí umístit do veřejné části serveru a obsahuje veškeré obrázky, CSS soubory a JavaScript-ové zdrojové kódy. Více o adresářové struktuře na [8].

7 UŽIVATELSKÉ ROLE A PRÁVA UŽIVATELŮ

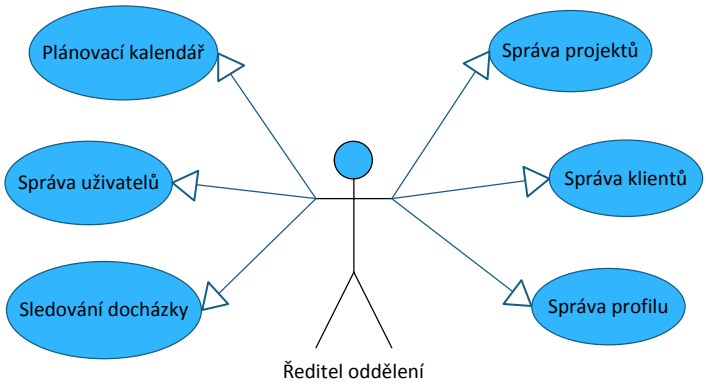
Dle specifikace požadavků je systém připraven na čtyři skupiny uživatelů. Systém omezuje možnosti datového modelu, přiřadit uživateli jednu nebo více rolí, a každý uživatel může být pouze v jedné skupině. Role jsou:

a) **Administrátor** je uživatel mající na starosti nastavení systému, vytvoření prvních uživatelů a správu uživatelských oprávnění. Tento uživatel je automaticky vytvářen při instalaci systému. Případy užití ilustruje Obrázek 7.1.



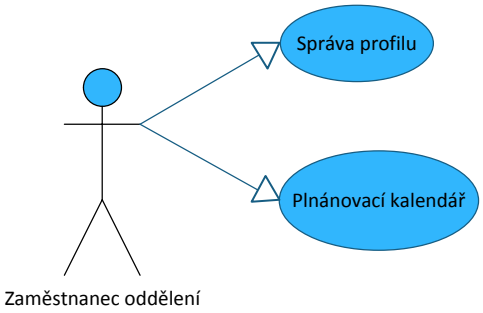
OBRÁZEK 7.1 - DIAGRAM PŘÍPADU UŽITÍ ADMINISTRÁTOR

b) **Ředitel oddělení** může vše nad svým oddělením. Vytváří podřízené, spravuje projekty a přiděluje práci. (viz Obrázek 7.2)



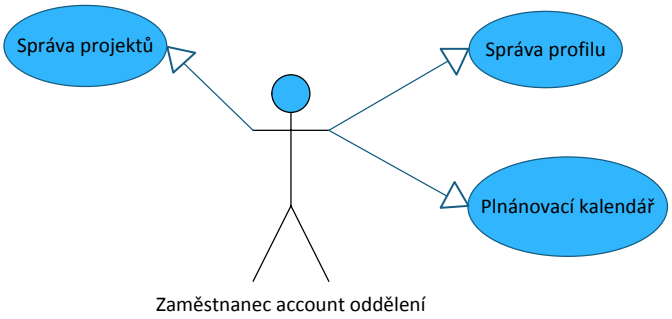
OBRÁZEK 7.2 - DIAGRAM PŘÍPADU UŽITÍ ŘEDITEL ODDĚLENÍ

c) **Zaměstnanec oddělení** vidí přidělenou práci, může si přidělit práci na projektu, na kterém pracuje. (viz Obrázek 7.3)



OBRÁZEK 7.3 - DIAGRAM PŘÍPADU UŽITÍ ZAMĚSTNANEC ODDĚLENÍ

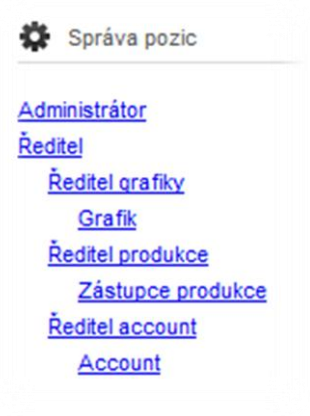
d) **Zaměstnanec oddělení péče o zákazníky** má stejné možnosti jako zaměstnanec jiných oddělení a navíc může spravovat projekty. (viz Obrázek 7.4)



OBRÁZEK 7.4 – DIAGRAM PŘÍPADU UŽITÍ ZAMĚSTNANEC ODDĚLENÍ PÉČE O ZÁKAZNÍKA

Uživatel kterékoli skupiny se může přihlásit do systému, odhlásit se, zobrazit a nastavit své profilové informace.

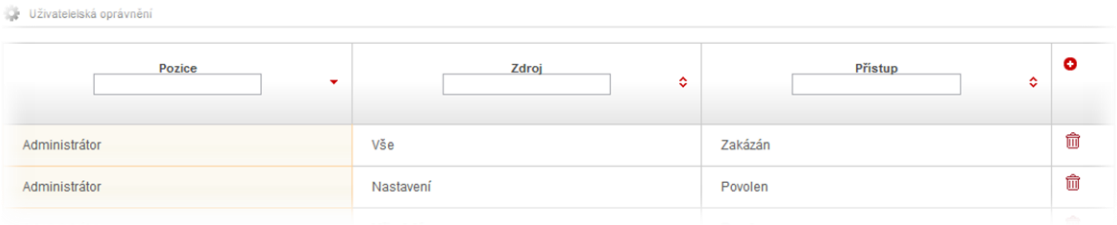
Jelikož jednotlivá oddělení mají různé požadavky, byly role dále rozděleny podle oddělení a docíleno tak možnosti samostatné specifikace oprávnění. Výslednou hierarchii ilustruje Obrázek 7.5. Samostatnou skupiny tvoří **Administrátor**. Všem odděleními je nadřazena role **Ředitel**. Ta slouží k současnému nastavení oprávnění uživatelů všech oddělení, např. zakázání přístupu do globálního nastavení systému, kam má přístup jen administrátor.



OBRÁZEK 7.5 - HIERARCHIE ROLÍ V SYSTÉMU

Systém má definovány tzv. zdroje. Jedná se o části systému, ke kterým je řízený přístup. Může se jednat o celé moduly (kalendář, projekty), nebo jen část funkcionality (správa rolí, oprávnění). Práva uživatelů jsou řízena na základě příslušnosti k určité roli. Defaultně je nastaveno, že skupina má vše zakázáno (aby byl zajištěn přístup jen k definovaným zdrojům a funkcím). Následně se přístup může povolit. Hierarchické uspořádání rolí je čistě účelné a umožňuje dědění oprávnění. Práva, která udělíme roli, se propisují do všech potomků, což například umožňuje nastavit všem ředitelům práva přes roli ředitel, na druhou stranu zároveň přináší nutnost více věcí zakazovat rolím pracovníků.

Nastavení oprávnění obnáší sestavení trojice „*kdo, kam, co*“. Tedy nějaká role má k určitému zdroji přístup povolen nebo zakázán, jak ilustruje Obrázek 7.6.



OBRÁZEK 7.6 - NASTAVENÍ UŽIVATELSKÝCH OPRÁVNĚNÍ V SYSTÉMU

8 POPIS SYSTÉMU

V této kapitole budou popsány jednotlivé části systému. Jak již bylo řečeno, aplikace je postavena na Model-View-Presenter architektuře (dále jen MVP) a využívá návrhový vzor Dependency Injection (dále jen DI).

MVP vychází ze softwarové architektury Model-View-Controller, která rozděluje kód aplikace na tři vrstvy, kód aplikační logiky (Model), kód obsluhy (Controller) a kód mající na starost zobrazování dat (View). Obdobou kontrolérů jsou v Nette Framework presentery. Presenter je objekt, který vezme požadavek přeložený routerem z HTTP požadavku a vygeneruje odpověď. Odpovědí může být HTML stránka, obrázek, XML dokument, soubor na disku, JSON, přesměrování apod. Obvykle se pod pojmem presenter myslí potomek třídy *Nette\Application\UI\Presenter*. Podle příchozích požadavků spouští odpovídající akce a vykresluje šablony. [8]

V aplikaci nedědí třídy presenterů přímo od *Nette\Application\UI\Presenter*. Od té dědí abstraktní třída BasePresenter, která implementuje společnou funkčnost všech presenterů aplikace, např. ověření oprávnění k přístupu.

Podstatou Dependency Injection (DI) je odebrat třídám zodpovědnost za získávání objektů, které potřebují ke své činnosti (tzv. služeb) a místo toho jim služby předávat při vytváření. [8]

Tímto způsobem objekty presenční vrstvy přistupují k objektům vrstvy modelu. Model opět obsahuje základní třídu *BaseModel*, která obsahuje základní funkčnost modelu, např. vkládání, úpravu a mazání dat v DB. Pokud objektu stačí tato základní implementace funkcí modelu, volá tuto třídu. Pokud je vyžadována nějaká specifická funkčnost, volá některého z potomků této třídy.

8.1 UŽIVATELÉ

Tato část systému má na starosti vytváření uživatelů, pozic ve společnosti (uživatelským rolí) a správu uživatelských práv. V této části bude také popsána komponenta editovatelné tabulky, jelikož ji zde prvně zmiňuji.

8.1.1 SPRÁVA UŽIVATELŮ

Do systému se není možné zaregistrovat, nový uživatel může vzniknout pouze tím, že jej přidá již existující uživatel (administrátor nebo ředitel oddělení). Přidání uživatelů do systému je řešeno, obdobně jako zbytek funkcí systému, s ohledem na jednoduché ovládání. Uživatel vyplní jméno, e-mail (sloužící jako přihlašovací jméno) a vybere roli nového uživatele. Pro vytvoření není více informací potřeba, a proto nejsou vyžadovány. Na výběr má možnost *přidat*, která vytvoří uživatele a přesměruje na výpis uživatelů, nebo *přidat a další*, která vytvoří nového uživatele a znovu zobrazí formulář pro přidání dalšího.

Data z odeslaného formuláře jsou nejprve zkontrolována, zda jsou vyplněna a má-li e-mail správný formát. Proběhne-li validace v pořádku, je uživateli vygenerováno osmimístné heslo. Generuje se náhodný řetězec složený z „a-z“ a „0-9“, vynecháno je písmeno „l“ (malé L) a číslice „1“ (jedna), protože by snadno mohlo, při opisování, dojít k záměně. Data se uloží do databáze, heslo je uloženo jako osolený MD5 hash. Následně je odeslán e-mail s adresou systému a přihlašovacími údaji.

8.1.2 SPRÁVA POZIC A UŽIVATELSKÝCH PRÁV

Není žádoucí, aby všichni uživatelé měli přístup ke všem částem systému, a nastavovat práva každému uživateli samostatně by bylo zbytečné. Využita byla struktura společnosti a je možné vytvářet uživatelské role, shodné s pozicemi ve společnosti. To je možné, jelikož zaměstnanci na určitých pozicích mají stejné požadavky na použití systému.

Role tvoří hierarchickou strukturu. Formulář pro přidání a editaci role obsahuje pouze pole pro název role a výběr nadřazené role. Do tabulky *role* se ukládá ještě třetí řetězec (viz kapitola 6.1.1), který je generován z pole *název* a slouží pro identifikaci role ve zdrojovém kódu.

Nastavení práv uživateli se provádí při přihlášení do systému. Rekurzivně jsou načteny role z databáze a vytvořeno n-rozměrné pole nesoucí strukturu potřebných rolí. Stejně je vytvořeno pole zdrojů. Záznamy z polí jsou postupně vkládány do FW, který si je po dobu přihlášení uživatele pamatuje. Následně jsou do FW předány informace, která role má přístup k jakému zdroji. FW po předání těchto

informací umožní, pomocí funkce *isAllowed(zdroj)*, jednoduše určit zda přihlášený uživatel má ke zdroji přístup či nikoli.

Autentizace neboli přihlašování uživatelů, je řešeno plně dle doporučení Nette. Detailní popis používaného postupu je popsán v dokumentaci [8].

Pro definování trojic „role, zdroj, přístup“ je použita editovatelná tabulka, která je využívána i v ostatních částech systému.

8.1.3 EDITOVATELNÁ TABULKA

Jedním z požadavků na systém bylo jednoduché používání. Za tímto účelem byla do systému vložena komponenta umožňující tzv. inline editaci položek. Lze ji využít při tabulkovém výpisu hodnot z databáze.

Jako základ byla použita Nette komponenta Tabella [9]. Toto řešení zcela nesplňovalo požadavky a bylo potřeba jej upravit. Tabella využívá pro komunikaci s databází samostatný Framework. Nebylo žádoucí zbytečně připojovat do systému další knihovny, proto byla komponenta přepsána, aby framework nebyl potřeba.

Komponenta umožňuje definovat sloupce, které se mají zobrazovat z vypisované tabulky. Vstupním parametrem je pouze tabulka, ze které se mají vypsat data. Následně se určí sloupce, které má výpis obsahovat. U sloupců lze nastavit, jakého typu mají být, např. *SELECT* při editaci zobrazí předvolenou množinu možností, *DATE* k editačnímu poli připojí výběrový kalendář. Sloupce se dají řadit podle abecedy a obsahují pole pro filtraci obsahu.

Kliknutím do řádku se sloupce změní na editovatelná pole a hodnoty záznamu mohou být změněny. Změny se aplikují klávesou *Enter* nebo tlačítkem na konci řádky. Při editaci se na konci řádky objeví také tlačítko pro zrušení změn. Při needitovatelném stavu záznamu, je na konci každého řádku jen tlačítko pro smazání záznamu (smazání vyvolá nejdříve dialog pro potvrzení akce).

8.2 PROFIL

Aby si uživatel mohl nastavit systém podle sebe, je v systému část nazvaná *Profil*. Uživatele si může zobrazit svůj profil, na kterém vidí identifikační a kontaktní údaje, odpracovanou dobu za aktuální den a měsíc a nejbližší úkoly. Tentýž výpis si může u každého ze svých podřízených zobrazit ředitel oddělení.

V nastavení je editační formulář, který umožňuje nastavit:

- Jméno
- E-mail (uživatelské jméno pro přihlášení)
- Telefon
- ICQ/Jabber (kontakt na instant messaging)
- Avatar (obrázek uživatele)

Krom jmenovaných vlastností umožňuje formulář změnit přístupové heslo do systému. Pro zadání jsou zde dvě políčka, první pro vložení hesla a druhé pro opakované zadání, kvůli zamezení překlepu (při zadávání se zobrazují místo znaků hvězdičky). Pokud uživatel pole nevyplní, nic se nestane, pokud alespoň jedno vyplní, porovnají se a v případě shody se změní záznam v databázi.

8.3 Docházka

Sledování odpracované doby se dá z uživatelského pohledu rozdělit na dvě části, a to na obsluhu měření času a přehled odpracované doby.

První jmenovanou část vidí v uživatelském rozhraní všichni uživatelé systému. Jedná se o odkaz, který pomocí CSS simuluje kontrolku stavu měření odpracované doby (viz Obrázek 8.1).

OBRÁZEK 8.1 – INDIKÁTOR MĚŘENÍ ODPRACOVANÉ DOBY

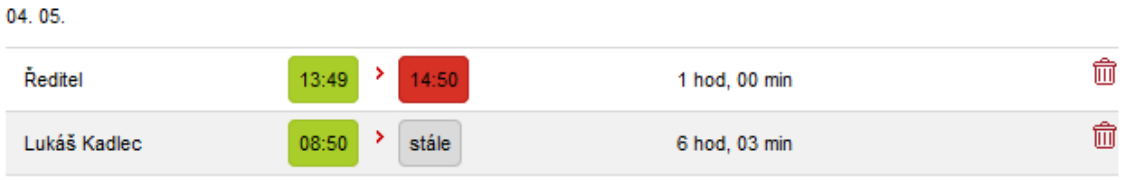
Kliknutí na ikonu vyvolá na základě současného stavu indikátoru příslušný AJAX-ový požadavek. Pokud není ve stavu *on* (červená ikona, nepracuje) je zavolán handler *startWork()*. Ten vytvoří záznam v tabulce **hours** s aktuálním časem a identifikátorem přihlášeného uživatele. Pokud indikátor je ve stavu *on* (zelená ikona, pracuje) je zavolán handler *stopWork()*, který upraví dříve vzniklý záznam vložením aktuálního času do sloupce *end.* Oba hadlery jsou součástí třídy *BasePresenter*[17]. Obslužný JavaScript-ový kód vypadá následovně:

[17] Handlery pro obsluhu měření času jsou umístěny v BasePresenter-u, kvůli snazšímu použití. Může být omezen přístup k celému modulu Docházky, ale tyto části jsou přístupné z každého modulu.

```
if ($("#status a").hasClass("on")) {
  $.get("?do=stopWork", function() {
    $("#status a").removeClass("on");
    $("#status a").attr("title", "Začít pracovat");
  });
} else {
  $.get("?do=startWork", function() {
    $("#status a").addClass("on");
    $("#status a").attr("title", "Ukončit práci");
  });
}
```

Měření času je obsluhováno pouze pomocí indikátoru stavu. Původně bylo plánováno zastavení práce při odhlášení nebo zavření prohlížeče, ovšem tyto varianty nebylo vhodné implementovat. Někteří zaměstnanci tráví převážnou část pracovní doby mezi klienty a není tedy žádoucí, aby museli mít spuštěn prohlížeč.

Druhou částí uživatelského rozhraní *Docházky*, je přehled odpracované doby uživatelů (viz Obrázek 8.2.). Tuto část vidí pouze ředitelé oddělení. Zobrazují se data zprostředkovaná databázovým pohledem **hours_total**. Výstup je seskupován podle dní. Defaultně se zobrazuje posledních sedm dní a připraven je filtr pro zobrazení aktuálního dne.



OBRÁZEK 8.2 - MODUL DOCHÁZKA, PŘEHLED ODPRACOVANÁ DOBY

8.4 KONTAKTY

Každá zakázka je plněna pro nějakého klienta. Krom jeho názvu se v systému uchovávají i kontaktní údaje, a jelikož tyto údaje převažují, byl celý modul pro správu klientů nazván *Kontakty*. Údaje, které lze ke klientovi uložit (pracuje s tabulkou **contact**, viz kapitola 6.1.3, strana 23), jsou:

- název (mimo jiné identifikace při přiřazování k zakázce)
- kontaktní osoba
- telefon

- E-mail
- adresa

Pro správu kontaktů je použita již zmiňovaná editovatelná tabulka (viz kapitola 8.1.3, strana 33). Ta byla doplněna o podporu polí pro editaci barvy. Při výpisu dat se buňce označené jako barva nastaví jako pozadí vložená barva. Pokud se taková buňka edituje, je uživateli nabídnuta paleta předdefinovaných barev. K zobrazení palety byl použit jQuery plugin *Really Simple Color Picker* [10].

Barva klienta se používá k rozlišení úkolů. Každý projekt pro klienta má jeho barvu a stejně tak úkoly na projektech.

8.5 PROJEKTY

Cílem systému je správa a sledování zakázek, které jsou zde evidovány jako projekty. Modul *Projekty* má tři funkcionality:

- výpis projektů
- přidání projektu
- zobrazení detailu

Výpis je tvořen editovatelnou tabulkou (viz kapitola 8.1.3, strana 33), přes dvě databázové tabulky (projekt, client). V přehledu se vypisuje název projektu, název klienta, pro kterého je řešen, termín, do kdy má být projekt dokončen a zaškrtávací políčko označující a umožňující dokončení projektu. Defaultně se zobrazují pouze nedokončené projekty.

8.5.1 PŘIDÁNÍ PROJEKTU

Pro přidání projektu je připraven formulář, který se zobrazí po kliknutí na tlačítko *Přidat projekt* nad výpisem projektů. Políčka *Název* a *ID zakázky* slouží pouze k předání dat do databázové tabulky *project*. Stejně tak políčko priorita, které pouze omezuje vstup na čísla 1 až 5. Výběrové políčko *Klient* zobrazuje názvy klientů (tabulka *client*) a výběr vrací ID klienta. Zajímavější jsou zbývající formulářové prvky.

Podklady

K projektu je potřeba připojit podklady pro realizaci. Tyto soubory jsou dle zvyklostí společnosti ukládány na souborový server do složky projektu. Cílem připojení

podkladů v projektu je propojení s touto složkou. Souborových serverů může být několik (interní server, FTP server pro externisty). Výběrové pole *Server* nabízí předdefinovaná úložiště (tabulka *storage*). Při výběru se odešle AJAX-ový požadavek, který odešle ID zvoleného serveru.

```
$("form").delegate('#select', 'change', function() {
  $.get("?do=selectLoad", {"storage_id": $(this).val()});
});
```

Na straně serveru bylo potřeba vyřešit výpis složek na souborovém serveru. Na vyžádání byl interní souborový server společnosti zpřístupněn přes FTP (pouze pro čtení), aby se sjednotily způsoby přístupu k serverům.

Třída FtpModel

Byla vytvořena třída **FtpModel** pro čtení seznamu souborů ze serveru. Konstruktor třídy přijímá jako parametry adresu serveru, přihlašovací jméno a heslo. Konstruktor se pokusí navázat spojení se serverem a následně se přihlásit, pokud se některá část nezdaří, vypíše se chybová hláška a průběh se ukončí. V případě úspěšného připojení jsou ve třídě implementovány metody pro získání názvů souborů.

První potřebnou metodou je **getDirList()**, která vrací pole názvů složek v předaném místě serveru. Jako parametr přijímá cestu ke složce, ze které má získat seznam adresářů. Pokud není parametr předán, vypisují se složky v rootu serveru. PHP funkcí ftp_rawlist() získáme pole s detailním výpisem, každý prvek v poli má strukturu:

```
drwxr-x---   3 vincent  vincent      4096 Jul 12 12:16 public_ftp
```

Řetězec rozdělíme podle mezer a od deváté části dál bereme vše, jelikož název složky může být víceslovný a je potřeba vzít všechna slova. S tím nám pomůže PHP funkce na dělení řetězce pomocí regulárního výrazu *preg_split("/[\s]+/", $line, 9)*,

- *\s* zastupuje veškeré bílé znaky, tedy i mezeru,
- v $line je vstupní řetězec,
- poslední parametr (9) říká, že řetězec rozdělíme na devět podřetězců, poslední bude zbytek (název souboru).

Počátek řetězce značí, o jaký typ souboru se jedná, chtěné je *d*, které značí složku.

Další metodou je **getList()**, která vrátí opět pole, tentokrát všech souborů a složek pro potřeby výpisu podkladů ve složce přiřazené k projektu. Vstupní parametr je shodný s předchozí metodou. Pro získání informací ze serveru se využívá PHP funkce *ftp_nlist()*, která na rozdíl od předchozí funkce vrací jen názvy (bez ostatních informací).

Objekt třídy FtpModel tedy umožní získat data pro výběrem složky s podklady k projektu. Požadavek je dokončen a odešlou se data.

Odpovědné osoby

Odpovědná osoba může prohlížet detail projektu a přidávat k projektu úkoly. Je tím především myšlen ředitel oddělení nebo account manager, ale aby si mohl přidávat úkoly, může touto osobou být i zaměstnanec. Prvek pro zvolení odpovědné osoby je tedy výběrové pole se všemi uživateli systému (vyjma administrátora).

Odpovědných osob může být přidáno nula až N. Aby toto bylo umožněno, je od počátku potřeba výběrové pole uvažovat jako pole těchto prvků, tedy nazvat jej např. *users[1]*. Indexace od jedné současně signalizuje počet existujících polí. Vedle pole je přidávací prvek (tlačítko), který spouští následující JavaScript-ovou obsluhu:

```
$( "#pridat" ).bind("click", function() {
  var osoby = $( ".osoby" );
  var pocetOsob = $(osoby).length;
  var aktualniCislo = (pocetOsob*1) + 1;
  var dalsi = $( "#frmaddEdit-users-1" ).clone();
  dalsi.attr( "id", "frmaddEdit-users-" + aktualniCislo );
  dalsi.attr( "name", "users[" + aktualniCislo + "]" );
  $( "#frmaddEdit-users-" + pocetOsob ).after(dalsi);
});
```

Zjistí se počet prvků, první prvek se naklonuje, změní se ID prvku a především *name*, čímž je vytvořen nový prvek v poli. Na závěr je vykreslen za poslední prvek v poli.

8.5.2 ZOBRAZENÍ DETAILU

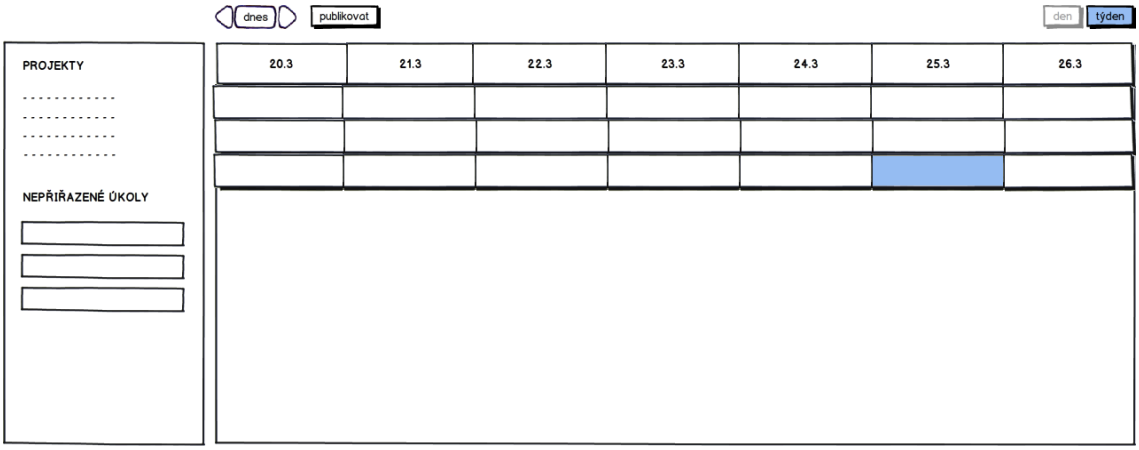
Zobrazit detail projektu má právo každý uživatel odpovědný za projekt. Sdružují se zde veškeré informace k projektu. Základní údaje projektu (název, termín), kontaktní údaje na klienta, umístění složek s podklady. Jsou zde vypsány všechny úkoly na projektu s jejich stavem, vykonavatelem a časem, který na nich byl stráven. Pod úkoly je součet těchto časů jako celková doba strávená na projektu. Poslední informací na detailu je výčet odpovědných osob s možností přidání další. Přidání je řešeno stejně jako při vytváření projektu.

K detailu projektu se uživatel dostane buď z modulu *Projekty*, kde ve výpisu projektů klikne na název projektu, nebo z modulu *Kalendář*, kde se projekty, ke kterým patří, zobrazují v levém panelu (viz kapitola 8.6).

8.6 KALENDÁŘ

Řídící a především plánovací aplikace se nemůže obejít bez kalendáře. A na práci s kalendářem je také směřována většina a u některých uživatelských rolí všechna obsluha systému. Jak ilustruje wireframe[18] (viz Obrázek 8.3), je obrazovka modulu *Kalendář* rozdělena na dvě části. V levém panelu je umístěn výpis projektů, se kterými je uživatel spojen (je odpovědnou osobou nebo na projektu pracuje), a výpis úkolů, které uživatel vytvořil a které byly vykonavatelem odmítnuty (nepřiřazené úkoly). V pravé části okna je kalendář a prvky určené k jeho obsluze. Tlačítka k pohybu vpřed a zpět v kalendáři, tlačítko „dnes“ pro návrat na aktuální datum, „publikovat“, které zapíše provedené změny do databáze a tlačítka „den“ a „týden“ pro přepínání mezi denním a týdenní zobrazením na kalendáři.

[18] Wireframe – drátěný model, se používá pro návrh rozložení webových stránek



OBRÁZEK 8.3 - DRÁTĚNÝ MODEL PLÁNOVACÍHO KALENDÁŘE

Zobrazení kalendáře se liší podle role přihlášeného uživatele. Ředitel vidí kalendáře všech zaměstnanců svého oddělení. Zaměstnanec vidí pouze svůj kalendář. V obou případech mohou přepínat mezi týdenním (pondělí až neděle) a denním (nastavená pracovní doba) zobrazením.

8.6.1 jQUERY PLUGIN FULLCALENDAR

Řešení pro práci s kalendářem již existuje mnoho. Nebylo tedy potřeba vyvíjet vlastní od začátku. Hledání bylo omezeno na aplikace psané v jazyce JavaScript. Jelikož bylo jisté, že při vývoji systému bude použit framework jQuery, dalším omezením byly aplikace postavené na tomto FW. V neposlední řadě muselo řešení obsahovat co nejvíce z funkčnosti známé již při zadání (např. přesouvání objektů kalendáře tažením myši).

Nalezené aplikace byly převážně velmi rozsáhlé a bylo by zbytečné trávit dlouhou dobu jejich pochopením a nevyužít velkou část jejich funkcionality. Výsledným základem pro vznik kalendáře je open source projekt FullCalendar [11], vytvořený jako jQuery plugin. Jelikož byl vybrán na začátku sběru požadavků, je jeho funkční potenciál využit téměř beze zbytku. Funkčnost systému mohla být mírně směřována dle jeho možností.

FullCalendar je velmi dobře přizpůsobitelný. Obsahuje širokou škálu parametrů, kterými jde nastavit vzhled i chování kalendáře. Veškeré možnosti jsou popsány v dokumentaci [11]. Velmi vítaná byla například možnost lokalizace veškerých textových částí kalendáře. Základní konfigurace se provádí při inicializaci komponenty pomocí parametrů v JSON formátu. Nastavení jednotlivých parametrů lze měnit

i při již zobrazené komponentě, ale převážná část nastavení je pohromadě na jednom místě.

Bohužel nic není dokonalé a ani tento plugin nepokryl veškeré požadavky. Jako zásadní nedostatek se ukázala nemožnost zobrazení kalendářů všech podřízených řediteli oddělení. Při řešení tohoto problému se potvrdila správnost volby při výběru komponenty kalendáře. FullCalendar má možnost různých „views“, tedy způsobů jak se zobrazují dny a události. Připravených je pět základních pohledů a další je možno vytvořit. Zde se projevila další výhoda vybraného řešení, a to, že projekt je šířen jako open source s poměrně rozsáhlou komunitou. Již několik členů komunity řešilo obdobný problém a mezi řešeními si tedy bylo možno dokonce vybrat. S drobnou úpravou byl problém vyřešen. Řešení nepodporovalo předání vlastníka kalendáře při přetažení externího objektu.

Jako další bylo do kalendáře potřeba doplnit ukazatel aktuálního času (viz Obrázek 8.4), na který jsou uživatelé zvyklí z Google Calendar. Řešení opět poskytla komunita okolo FullCalendar [12].

OBRÁZEK 8.4 - UKAZATEL AKTUÁLNÍHO ČASU

Řešení není složité. Výhodou diskuse, která vznikla již v roce 2011, je, že byly odhaleny různé problémy, kterými by pravděpodobně bylo nutné projít, a byla tak velmi ulehčena práce.

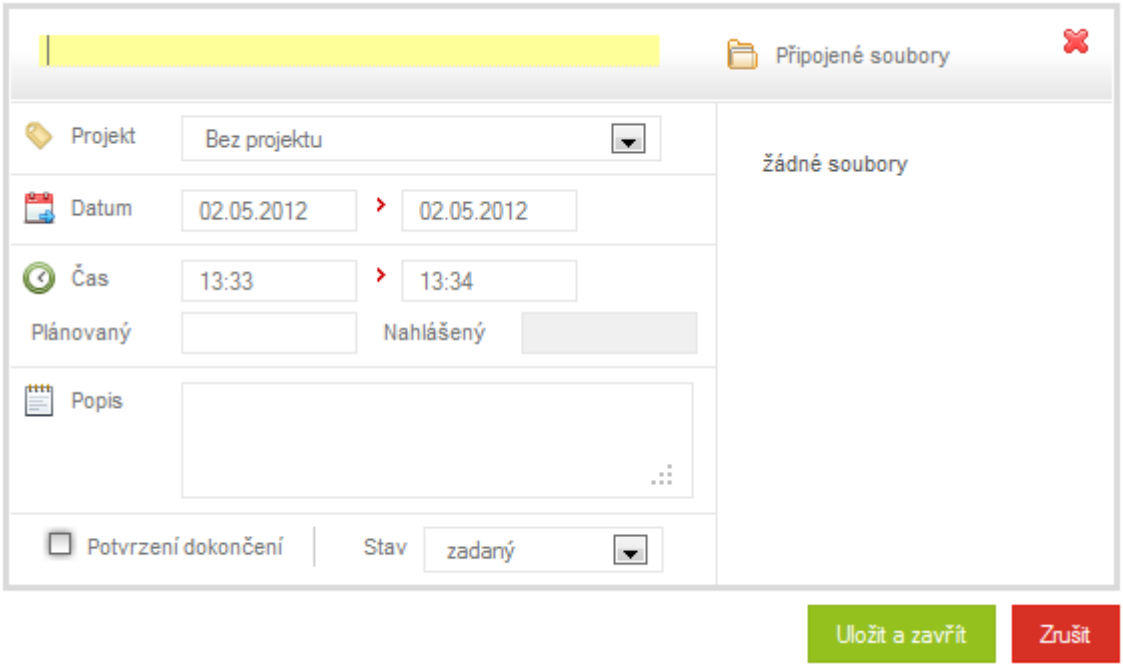
8.6.2 ÚKOLY A UDÁLOSTI

FullCalendar dovoluje načítat objekty z více zdrojů najednou. Toho je využito a jsou mu předány dva zdroje, jeden pro vrácení úkolů na projektech a druhý pro události (používáno zatím jen pro dny pracovního klidu). Kalendář při zobrazení určitého časového intervalu odešle AJAX-ový požadavek na adresu připojených zdrojů. Požadavek má dva GET parametry, které udávají začátek a konec zobrazovaného intervalu ve formátu UNIX timestamp. Server na tento požadavek reaguje

výběrem záznamů z databázea vytvořením a odesláním JSON formátu s daty událostí. V odpovědi na požadavek kalendář přijme data a vykreslí.

Přidání úkolu

Pro vytvoření obsluhy událostí, obsahuje FullCalendar několik callback funkcí, které se volají při různých akcích nad kalendářem. Jednou z těchto callback funkcí je **select**. Ta je volána při výběru nějakého intervalu na kalendáři (může to být jedno pole, např. den, nebo se tažením vybere časový interval). Důležitými vstupními parametry je začátek a konec intervalu. Na tuto událost je navázáno vytvoření dialogového okna při přidání úkolu (viz Obrázek 8.5). Dialog umožňuje nastavit veškerá potřebná data k úkolu. O vytvoření dialogu se stará JavaScript-ová třída **Dialog**. Vstupním parametrem pro vytvoření dialogu je JSON objekt s daty budoucího úkolu. Objekt je před odesláním naplněn známými daty: vlastník úkolu, vykonavatel úkolu (získán jako parametr funkce select), začátek a konec plnění úkolu.



OBRÁZEK 8.5 - DIALOG PRO PŘIDÁNÍ/EDITACI ÚKOLU

Vstupní pole dialogu (datum a čas začátku a konce úkolu) jsou předvyplněné daty ze vstupního parametru. V případě, že bylo kliknuto na den v týdenním kalendáři, nepředává komponenta kalendáře informaci o čase (je předáno 00:00). V takovém případě je pro začátek použit aktuální čas a pro konec o minutu větší.

Pokud je potřeba pole pro datum editovat, po kliknutí se zobrazí kalendář pro výběr data. To jednak zjednoduší výběr data a současně zajistí požadovaný formát. Správný formát je zajištěn i nastavením masky vstupnímu poli (lze zadat pouze čísla ve formátu „xx.xx.xxxx“). Editace času je zjednodušena tím, že formát není pevně stanoven. Vstupní hodnoty „09:00“, „9:00“ a „9“ jsou totožné. Pokud se uživatel pokusí uložit úkol s nesprávným časovým intervalem (konec před začátkem) je na to upozorněn a je požadována náprava. Plánovaný čas nemusí být vyplněn, v takovém případě se dopočte jako rozdíl začátku a konce plnění úkolu.

Pokud je vybrán projekt, do kterého úkol patří, zobrazí se pod výběrovým polem termín dokončení projektu a v části připojené soubory se načtou podklady, které souvisí s projektem. Datum dokončení projektu je informace,která slouží k tomu, aby úkol nebyl plánován za tento termín. Výpis podkladů projektu umožňuje vybrat podklady, které bude vykonavatel potřebovat ke splnění úkolu. Při zobrazení úkolu vidí jen vybrané a zjednoduší se mu tak orientace ve složce s podklady.

Po kliknutí na „uložit a zavřít“ se úkol uloží do paměti kalendáře a zavře se dialog.

Editace/Zobrazení úkolu

Dialog pro editaci (zobrazení) úkolu je téměř shodný s dialogem pro přidání úkolu (viz Obrázek 8.5). Je vyvolán callback funkcí **eventClick** kalendáře, při kliknutí na objekt v kalendáři. Jako jeden z parametrů přijímá data o objektu, na který bylo kliknuto, ve formátu JSON. Rozdíly závisí mimo jiné na vztahu uživatele k úkolu, je-li vlastník nebo vykonavatel. Prvním rozdílem je změna pole pro plánovaný čas na úkolu na needitovatelné (pro vykonavatele) a naopak zaktivnění pole pro zadání času stráveného na úkolu.

Další změna je v červeném tlačítku, které v přidávacím dialogu bylo „zrušit“. Nyní zde vykonavatel úkolu má tlačítko „odmítnout“, kterým vrátí úkol vlastníkovi. Odmítnutí provede v databázi v záznamu úkolu nastavení vykonavatele na *NULL*. Takové úkoly má vlastník zobrazeny v levém panelu a může je tažením myši přiřadit jinému zaměstnanci. Pro vlastníka úkolu slouží červené tlačítko, při editaci s popisem „odstranit“, k odstranění úkolu. Jelikož by mohlo dojít k přehlédnutí, že tlačítko již není nazváno „zrušit“, a omylem smazat úkol, je uživatel před definitivním

provedením akce, upozorněn a dotázán, chce-li opravdu úkol smazat. Odstranění se po potvrzení provede okamžitě AJAX-ovým požadavkem a je nevratné.

Editace pomocí dialogového okna není vždy potřeba a u volby vykonavatele úkolu není ani možná. S ohledem na jednoduché ovládání jsou některé možnosti editace umožněny tažením myši. Jedná se o výběr vykonavatele (přetažení do jeho kalendáře), změna umístění úkolu v čase (přetažení úkolu na jiný interval), změna konce úkolu (roztažení nebo zkrácení objektu). Poslední možnost editace pomocí myši je přetažení odmítnutého úkolu z levého panelu do kalendáře. Na všechny tyto akce měl kalendář připraveny callback funkce.

Publikování změn

Změny, které uživatel provádí v kalendáři, se, vyjma mazání úkolů, nezapisují ihned do databáze. Není to žádoucí, jelikož při rozdělování práce může docházet k častým změnám a ty by se současně projevovaly i u vykonavatelů. Z tohoto důvodu jsou změny ukládány do paměti kalendáře (prohlížeče), a je-li to požadováno, jsou změny hromadně zapsány do databáze. Za tímto účelem bylo do hlavičky kalendáře přidáno tlačítko „Publikovat“. Tlačítko je v defaultním stavu neaktivní (šedé). Dojde-li k nějaké změně v objektech kalendáře (vytvoření, editace) tlačítko zezelená a napojí se na něj událost po kliknutí. Událostí je zavolání funkce, která získá objekty držené v paměti a odešle je k zápisu do databáze.

Jelikož kalendář vrací vždy všechny objekty, které jsou v paměti, bylo potřeba odlišit, které úkoly se budou do databáze přidávat a které upravovat. Úkoly, které jsou načítány z databáze, mají nastaven atribut *id*, dle sloupce *id* v tabulce *task*. Nově vytvořeným úkolům je generován unikátní parametr přidáním timestamp-u k řetězci „tmp_“. Čas zajišťuje unikátnost identifikátoru a řetězec „tmp_“ odliší nové úkoly. Atribut *id* slouží také pro mazání úkolů, pomocí něj se identifikuje úkol k odstranění.

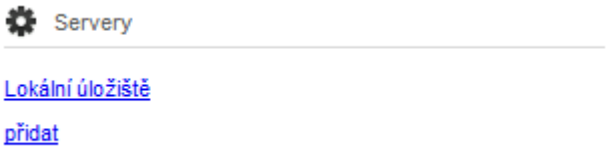
Po úspěšném uložení změn do databáze se znovu načte stránka a tlačítko „Publikovat“ je opět neaktivní.

8.7 Nastavení systému

Aby bylo možné jednoduše přednastavit systém pro jeho používání, nebo během chodu nastavení upravit, byl vytvořen modul *Nastavení*. Implementována byla

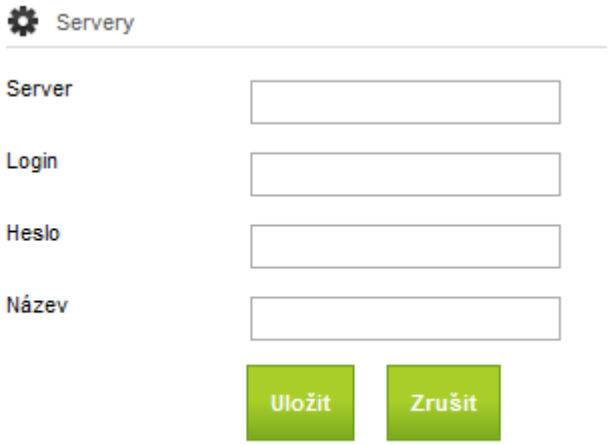
správa svátků a nastavení úložišť pro podklady k projektům. Tato část systému je určena pouze pro administrátora.

Výpis nastavení zobrazuje na úvodní obrazovce všechna nastavení. Jednotlivé konfigurovatelné části jsou v samostatných boxech. Každý tento box je obsluhován AJAX-em a tím pro úpravy konfigurace není nutné znovu načítat stránku. Ve výchozím zobrazení vypisuje box aktuální nastavení (viz Obrázek 8.6).



OBRÁZEK 8.6 - BOX PRO KONFIGURACI ÚLOŽIŠTĚ PŘÍLOH (VÝPIS)

Box obsahuje titulek, výpis nastavení a odkaz na přidání nového záznamu. Při kliknutí na *„přidat“*, se box překreslí a zobrazí vstupní formulář (viz Obrázek 8.7). Nový záznam můžeme uložit nebo zrušit akci. Pokud zvolíme uložit, provede se kontrola vyplněných polí, pokud je vše správně, záznam se uloží, v opačném případě je uživatel požádán o nápravu. Po úspěšném vložení je zobrazen aktuální výpis záznamů.



OBRÁZEK 8.7 - BOX PRO KONFIGURACI ÚLOŽIŠTĚ PŘÍLOH (PŘIDÁNÍ/EDITACE)

Každý vypisovaný záznam je editovatelný. Při kliknutí na něj se v boxu vykreslí tentýž formulář jako při přidávání, ale v polích budou data záznamu.

Překreslování jen částí stránky je řešeno pomocí Nette makra *snippet*, která zajistí většinu práce za nás. Pokud je povolen JavaScript, odesílají se veškeré požadavky na server AJAX-em. Na serveru se obslouží požadavek, sestaví se stránka a na závěr se z ní „vystřihne“ pouze část, která je ohraničená jako snippet. Ta se odešle jako odpověď na AJAX-ový požadavek a vykreslí se. Pokud JavaScript nelze použít, použije se klasické volání s obnovením celé stránky.

Svátky (dny pracovního klidu) jsou řešeny uchováváním vlastní databáze svátků. Původně byla zvažována možnost jejich načítání z Google Calendar, ovšem jednak kvůli počtu českých svátků (jen třináct [13]), a především kvůli požadavku, že svátky mají být upravovatelné (ve svátek se může pracovat), bylo rozhodnuto ve prospěch vlastní databáze. Mimo jiné toto řešení odstranilo potencionální riziko zpomalení systému kvůli komunikaci s cizím serverem.

9 Závěr

Během této diplomové práce jsem si prošel všemi kroky vývoje aplikace a zastupoval jsem veškeré role vývojového týmu. Získal jsem mnoho zkušeností v oblasti vývoje aplikací, především v jednání se zákazníkem. Bohužel například poznatky uváděné v kapitole 2, o struktuře společnosti, jsem shromažďoval během celého procesu, což zpětně hodnotím jako základní nedostatek. Neznalost cílového prostředí zkomplikovala sběr požadavků na výsledný systém.

Některé části funkcionality by šly ještě zdokonalit, uživatelské akce lépe ošetřit a podobně. Cílem diplomové práce bylo nasadit systém do pilotního provozu, který mimo jiné umožní uživatelské testování. Systém splňuje požadavky sepsané ve specifikaci požadavků, ovšem vzhledem k rozsahu práce jsou známy některé jeho nedostatky. Ty nebrání používání systému a budou doplněny na základě výstupů z pilotního provozu.

V současné době je systém nasazen do pilotního provozu a probíhá uživatelské testování. Účastní se jej všechna tři oddělení s omezeným počtem zaměstnanců, jelikož každé oddělení má mírně odlišné nároky na používání systému.

PŘEHLED ZKRATEK

AJAX	Asynchronous JavaScript and XML, skupina technologií využívaných pro komunikaci webových aplikací se serverem, bez nutnosti znovunačtení stránky.
Brief	Textové zadání projektu.
CSS	Cascading Style Sheets, jazyk, kterým je definován způsob zobrazení dokumentů, které jsou psány ve značkovacím jazyce (např. HTML, XML).
HTML	HyperText Markup Language, značkovací jazyk pro definování významu částí dokumentů publikovaných na Internetu.
HTTP	HyperText Transfer Protocol, internetový protokol využívaný k výměně informací mezi webovým serverem a prohlížečem.
IDE	Integrated Development Environment, vývojové prostředí.
JavaScript	Skriptovací jazyk vykonávaný v prohlížeči klienta. Využívaný například k ovládání uživatelského rozhraní.
JSON	JavaScript Object Notation, formát pro výměnu dat. Jeho výhodou je, že lze snadno zpracovávat stojem i člověkem.
PHP	PHP: Hypertext Preprocessor, skriptovací programovací jazyk, používaný převážně pro psaní aplikační logiky na straně serveru.
Projekt	Odpovídá zakázce v interním zakázkovém systému. Dělí se na jednotlivé úkoly.
ShiftPlanning	Online software pro správu pracovních sil. V současné době používán pro sledování vytíženosti zaměstnanců. http://www.shiftplanning.com/

SVN	Subversion, systém pro správu a verzování zdrojových kódů.
Things	Nástroj pro správu úkolů. http://culturedcode.com
Timesheet	Rozvrh vytíženosti zaměstnance.
Tooltip	Prvek grafického uživatelského rozhraní, zobrazující se u kurzoru myši a popisující význam ovládacího prvku, nad kterým se kurzor nachází.
Traffic System	Webová aplikace pro správu a sledování realizace zakázek, aplikace popisovaná v tomto DSP.
Úkol	Část projektu. Každý úkol má přiřazeného jednoho zaměstnance.
XML	Extensible Markup Language, značkovací jazyk využívaný především pro výměnu dat mezi aplikacemi a publikování dokumentů.
Zakázkový systém	Interní zakázkový systém. Sleduje zakázku z pohledu fakturace, objednávek, evidence hodin, interních a externích subdodávek atp.

Literatura

[1] P. Daněk, „Velký test PHP frameworků: Zend, Nette, PHP a RoR,“ 11 září 2008. [Online]. Available: http://www.root.cz/clanky/velky-test-php-frameworku-zend-nette-php-a-ror/. [Přístup získán 11 duben 2012].

[2] „JSON,“ 2012. [Online]. Available: http://www.json.org/json-cz.html. [Přístup získán 2012].

[3] T. P. Group, „PHP Manual,“ 2012. [Online]. Available: http://php.net/manual. [Přístup získán 18 duben 2012].

[4] J. Vrána, „Adminer - Správa databáze v jednom PHP souboru,“ 2012. [Online]. Available: http://www.adminer.org/cs/. [Přístup získán 18 duben 2012].

[5] J. Hewitt, „Doplňky aplikace Firefox,“ 2012. [Online]. Available: https://addons.mozilla.org/cs/firefox/addon/firebug/. [Přístup získán květen 2012].

[6] BinaryAge, „FireLogger :: Doplňky aplikace Firefox,“ 2012. [Online]. Available: https://addons.mozilla.org/cs/firefox/addon/firelogger/. [Přístup získán květen 2012].

[7] Š. Svoboda, „Jak otevřít soubor v editoru z Debuggeru?,“ 2012. [Online]. Available: http://pla.nette.org/cs/jak-otevrit-soubor-z-debuggeru-v-editoru. [Přístup získán 18 duben 2012].

[8] D. Grudl, „Dokumentace | Nette Framework,“ 2012. [Online]. Available: http://doc.nette.org/cs/. [Přístup získán 16 duben 2012].

[9] Knyttl, „Tabella | Nette Framework,“ 2011. [Online]. Available: http://addons.nette.org/cs/tabella. [Přístup získán květen 2012].

[10] L. Perera, „Really Simple Color Picker in jQuery,“ 17 únor 2012. [Online]. Available: http://laktek.com/2008/10/27/really-simple-color-picker-in-jquery/. [Přístup získán 22 duben 2012].

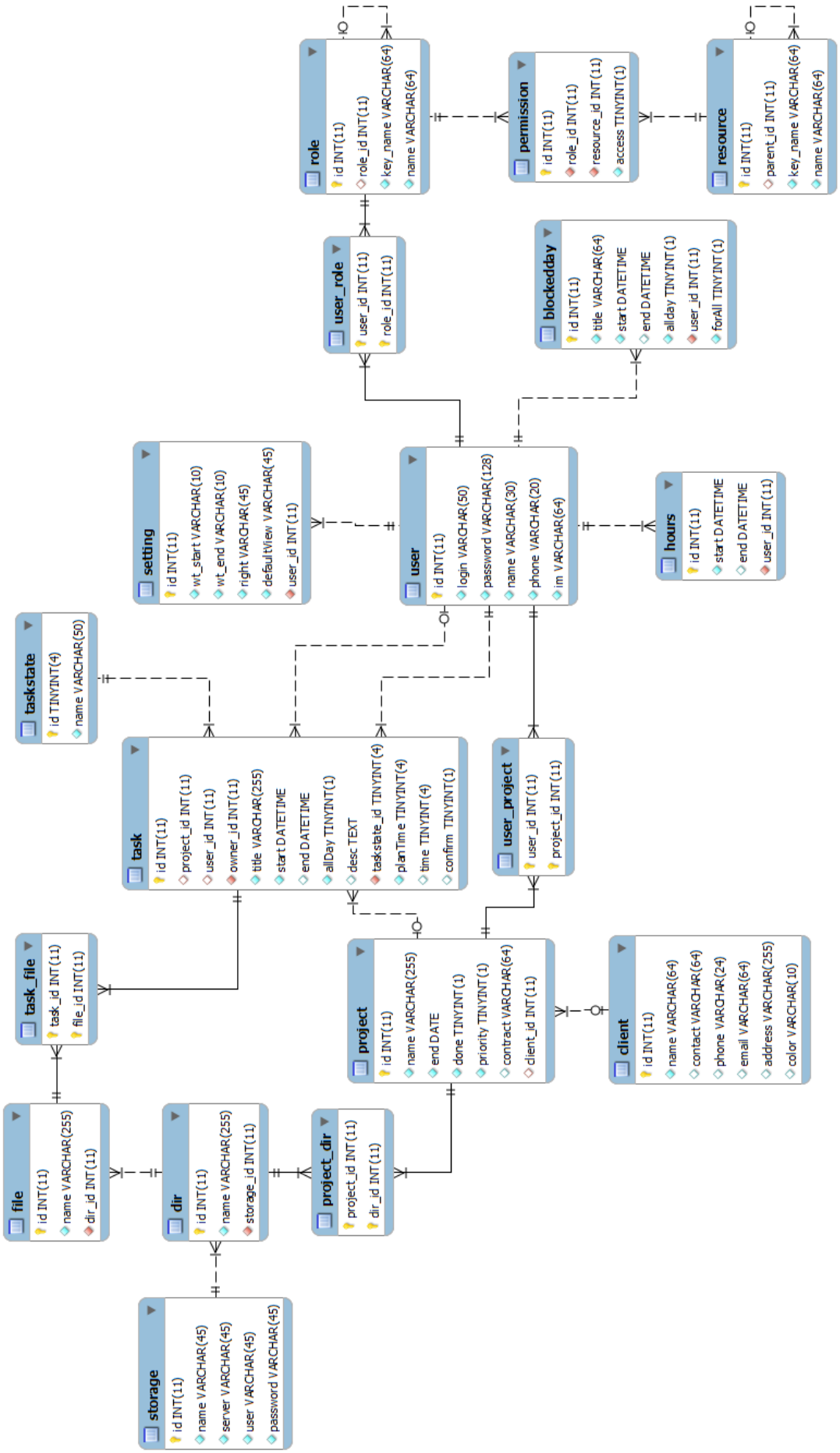
[11] A. Shaw, „FullCalendar - Full-sized Calendar jQuery Plugin,“ 2012. [Online]. Available: http://arshaw.com/fullcalendar/. [Přístup získán 30 duben 2012].

[12] F. komunita, „Issue 143 - Current time indicator in agendaweek and agendaday view,“ 20 březen 2012. [Online]. Available: http://code.google.com/p/fullcalendar/issues/detail?id=143. [Přístup získán 2 květen 2012].

[13] O. 52, „Svátky v České republice,“ 31 říjen 2007. [Online]. Available: http://www.mpsv.cz/cs/74. [Přístup získán 2 květen 2012].

[14] D. Odell, JavaScript, Průvodce programováním ajaxových aplikací, Brno: Computer Press, a.s., 2010.

SEZNAM PŘÍLOH


role
id INT(11)
role_id INT(11)
key_name VARCHAR(64)
name VARCHAR(64)
permission
id INT(11)
role_id INT(11)
resource_id INT(11)
access TINYINT(1)
resource
id INT(11)
parent_id INT(11)
key_name VARCHAR(64)
name VARCHAR(64)
user_role
user_id INT(11)
role_id INT(11)
blockedday
id INT(11)
title VARCHAR(64)
start DATETIME
end DATETIME
allday TINYINT(1)
user_id INT(11)
forAll TINYINT(1)
setting
id INT(11)
wt_start VARCHAR(10)
wt_end VARCHAR(10)
right VARCHAR(45)
defaultView VARCHAR(45)
user_id INT(11)
user
id INT(11)
login VARCHAR(50)
password VARCHAR(128)
name VARCHAR(30)
phone VARCHAR(20)
im VARCHAR(64)
hours
id INT(11)
start DATETIME
end DATETIME
user_id INT(11)
taskstate
id TINYINT(4)
name VARCHAR(50)
task
id INT(11)
project_id INT(11)
user_id INT(11)
owner_id INT(11)
title VARCHAR(255)
start DATETIME
end DATETIME
allDay TINYINT(1)
desc TEXT
taskstate_id TINYINT(4)
planTime TINYINT(4)
time TINYINT(4)
confirm TINYINT(1)
user_project
user_id INT(11)
project_id INT(11)
task_file
task_id INT(11)
file_id INT(11)
project
id INT(11)
name VARCHAR(255)
end DATE
done TINYINT(1)
priority TINYINT(1)
contract VARCHAR(64)
client_id INT(11)
client
id INT(11)
name VARCHAR(64)
contact VARCHAR(64)
phone VARCHAR(24)
email VARCHAR(64)
address VARCHAR(255)
color VARCHAR(10)
file
id INT(11)
name VARCHAR(255)
dir_id INT(11)
dir
id INT(11)
name VARCHAR(255)
storage_id INT(11)
project_dir
project_id INT(11)
dir_id INT(11)
storage
id INT(11)
name VARCHAR(45)
server VARCHAR(45)
user VARCHAR(45)
password VARCHAR(45)

UŽIVATELSKÝ MANUÁL

Obsah

Zaměstnanec

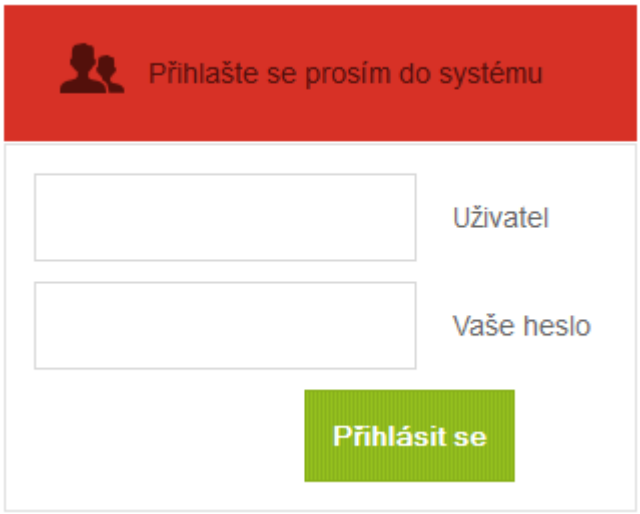
Ředitel oddělení

Administrátor

Zaměstnanec

Přihlášení do systému

Po zadání adresy systému do webového prohlížeče se zobrazí přihlašovací formulář (viz Obrázek B-1).



OBRÁZEK B-1: PŘIHLAŠOVACÍ FORMULÁŘ

Pokud se do systému přihlašujete poprvé, přihlašovací údaje vám byly zaslány e-mailem společně s adresou systému. V takovém případě si, prosím, po přihlášení změňte heslo (viz následující kapitola). Nemáte-li přihlašovací údaje, obraťte se na ředitele svého oddělení.

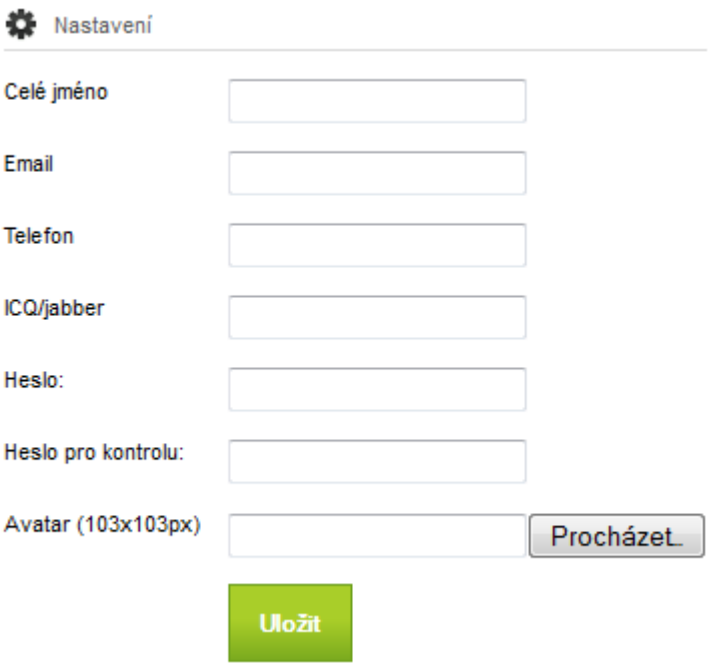
Profilová stránka

Profilová stránka slouží k získání informací o uživateli. Na vlastní profilovou stránku se dostaneme odkudkoli ze systému, klikneme-li v pravém horním rohu na obrázek uživatele a vybereme „Profil“. Zobrazují se zde kontaktní údaje, odpracované hodiny za aktuální den a měsíc a nejbližší úkoly.

Nastavení

Pro nastavení kontaktních údajů a změny hesla pro přihlášení slouží stránka **Nastavení**. Na ni se můžeme dostat z profilové stránky, odkazem v levém menu „Moje nastavení“, nebo odkudkoli v systému, klikneme-li v pravém horním rohu na obrázek uživatele a vybereme „Nastavení“.

Na stránce je formulář (viz Obrázek B-2), který nám umožní zadat (změnit) veškeré informace. Pro zachování stávajícího hesla necháme při ukládání změn obě pole pro heslo prázdná. Pokud chceme **změnit heslo**, zadáme nové do pole „Heslo“ a pro zamezení překlepu jej zopakujeme do pole „Heslo znovu“. Po uložení bude heslo okamžitě změněno.



OBRÁZEK B-2: FORMULÁŘ NASTAVENÍ

Docházka

Pro hlášení odpracované doby má systém jednoduché řešení. V pravém horním rohu je vedle obrázku umístěna ikona (viz Obrázek B-3) signalizující, zda právě pracujete (zelená) či nikoli (červená). Při začátku práce klikneme na červenou ikonu, ta změní barvu na zelenou a čas se začne měřit. Na konci práce ukončíme měření kliknutím na červenou ikonu.

OBRÁZEK B-3: INDIKÁTOR MĚŘENÍ ODPRACOVANÉ DOBY

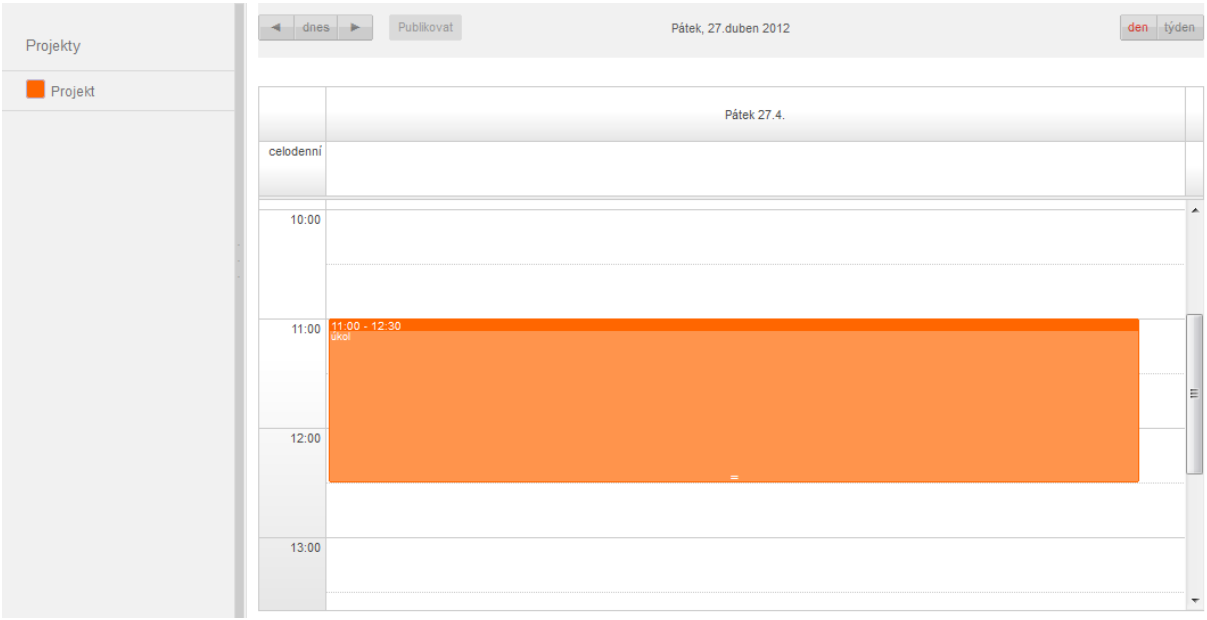
Odpracovanou dobu vidí především ředitel oddělení. Čas strávený v práci za aktuální den můžeme vidět na profilové stránce. Současně je zde zobrazován i odpracovaný čas za aktuální měsíc.

Kalendář

V kalendáři vidíme rozplánované úkoly k projektům na každý den. Ovládacími prvky nad kalendářem můžeme listovat jednotlivými dny (týdny). Tlačítko „Dnes“ nás vrátí na aktuální datum. Tlačítka „den“ a „týden“ slouží k přepínání denního (viz Obrázek B-4) a týdenního zobrazení kalendáře.

Na úkolech v kalendáři je zobrazen plánovaný začátek a konec práce na úkolu a cíl úkolu. Pro detailnější informace můžeme úkol rozkliknout a v zobrazeném dialogu zjistíme veškerá data potřebná k plnění úkolu.

Po dokončení úkolu klikneme na úkol a v dialogovém okně dokončíme úkol v systému. Nejdříve vyplníme čas strávený nad úkolem (v minutách), následně v dolní části přenastavíme výběrové pole ze „zadaný“ na „dokončený“ a změny uložíme tlačítkem „Uložit a zavřít“. Veškeré změny v kalendáři se ostatním projeví až po stisknutí tlačítka „Publikovat“ v horní liště kalendáře.



OBRÁZEK B-4: DENNÍ ZOBRAZENÍ KALENDÁŘE

Vlevo od kalendáře jsou vypsány projekty, na kterých pracujete. Každý projekt je vypsán jako barva a název projektu. Barva slouží k identifikaci, které úkoly v kalendáři patří pod který projekt. Po kliknutí na projekt, se zobrazí jeho detail, abychom, v případě potřeby, o něm mohli zjistit potřebné informace.

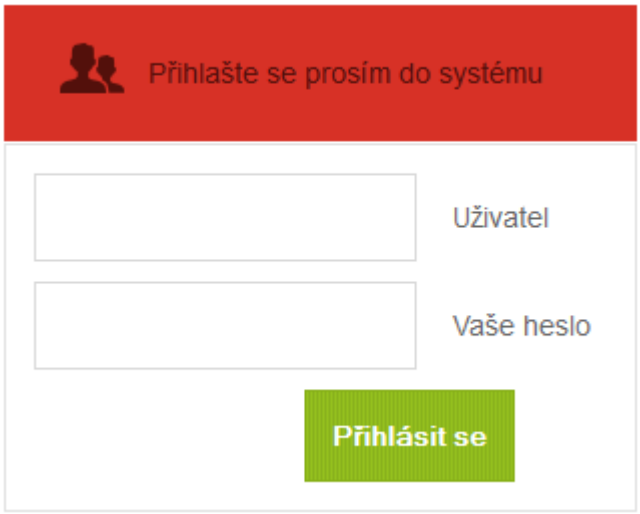
Nový úkol

V případě potřeby si můžeme sami přidělit úkol. To provedeme výběrem časového intervalu na denním zobrazení kalendáře nebo kliknutím na den v týdnu. Zobrazí se dialogové okno pro přidání nového úkolu s předvyplněným zvoleným časovým intervalem. Do titulku okna vyplníme krátký výstižný popis úkolu (v ideálním případě by měl stačit k pochopení cíle úkolu), pokud potřebujeme sdělit více informací, použijeme pole „Popis“. Po výběru projektu se nám pod polem zobrazí deadline projektu, abychom zbytečně nevolili konec úkolu za konec projektu. V případě, že projekt má připojeny nějaké podklady, vypíší se v pravé části okna a zaškrtáním zvolíme soubory potřebné pro práci na úkolu. Čas můžeme vyplnit ve formátu hodiny a minuty oddělené dvojtečkou nebo samostatnou číslicí, která bude převedena na celé hodiny. Plánovaný čas uvedeme v minutách. Na závěr vytvoříme úkol kliknutím na tlačítko „Uložit a zavřít“. Případně můžeme celou akci zrušit tlačítkem „Zrušit“ nebo křížkem v pravém horním rohu. Ostatní vytvořený úkol uvidí až po kliknutí na tlačítko „Publikovat“.

Ředitel oddělení

Přihlášení do systému

Po zadání adresy systému do webového prohlížeče se zobrazí přihlašovací formulář (viz Obrázek B-5).



OBRÁZEK B-5: PŘIHLAŠOVACÍ FORMULÁŘ

Pokud se do systému přihlašujete poprvé, přihlašovací údaje byly zaslány e-mailem společně s adresou systému. V takovém případě si, prosím, po přihlášení změňte heslo (viz následující kapitola). Nemáte-li přihlašovací údaje, obraťte se na administrátora systému.

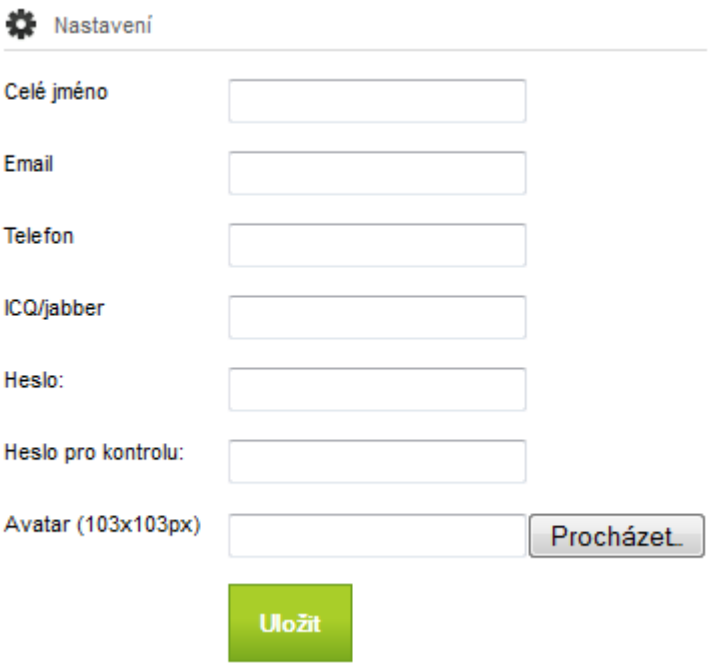
Profilová stránka

Profilová stránka slouží k získání informací o uživateli. Na vlastní profilovou stránku se dostaneme odkudkoli ze systému, klikneme-li v pravém horním rohu na obrázek uživatele a vybereme „Profil“. Zobrazují se zde kontaktní údaje, odpracované hodiny za aktuální den a měsíc a nejbližší úkoly.

Nastavení

Pro nastavení kontaktních údajů a změny hesla pro přihlášení slouží stránka **Nastavení**. Na ni se můžeme dostat z profilové stránky, odkazem v levém menu „Moje nastavení“, nebo odkudkoli v systému, klikneme-li v pravém horním rohu na obrázek uživatele a vybereme „Nastavení“.

Na stránce je formulář (viz Obrázek B-6), který nám umožní zadat (změnit) veškeré informace. Pro zachování stávajícího hesla necháme při ukládání změn obě pole pro heslo prázdná. Pokud chceme **změnit heslo**, zadáme nové do pole „Heslo“ a pro zamezení překlepu jej zopakujeme do pole „Heslo znovu“. Po uložení bude heslo okamžitě změněno.



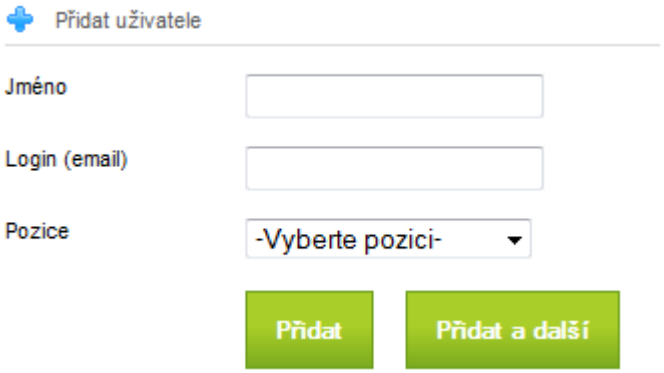
OBRÁZEK B-6: FORMULÁŘ NASTAVENÍ

Správa uživatelů

Na úvodní obrazovce vidíme výpis všech uživatelů našeho oddělení. Kliknutím na uživatele zobrazíme informace o uživateli (jeho profilovou stránku). V levém menu jsou odkazy na správu uživatelů.

Přidání uživatele

Odkaz „Přidat uživatele“ zobrazí formulář pro přidání uživatele do systému (viz Obrázek B-7).



OBRÁZEK B-7: FORMULÁŘ PRO PŘIDÁNÍ UŽIVATELE

Do pole „**Jméno**“ vyplníme skutečné jméno uživatele. „**Login**“ bude použito jako přihlašovací jméno do systému a současně jako e-mail pro komunikaci systému s uživatelem. Je potřeba vyplnit existující e-mailovou adresu. Posledním polem je „**Pozice“**, ve kterém vybereme z předdefinovaných rolí v systému (pozic ve společnosti).

Přidat můžeme pouze jednoho uživatele tlačítkem „**Přidat**“. Po přidání budeme přesměrováni na výpis uživatelů, nebo ihned po vložení můžeme přidat dalšího uživatele tlačítkem „**Přidat a další**“, kdy se po uložení zobrazí opět prázdný formulář.

Novému uživateli je vygenerováno náhodné heslo. Přihlašovací údaje jsou společně s odkazem pro přihlášení odeslány na e-mail zadaný do pole „**Login**“.

Správa klientů

Správu klientů nalezneme v hlavním menu pod odkazem **Kontakty**. Přidání nového klienta provedeme tlačítkem s ikonou plus v pravém horním rohu v hlavičce výpisové tabulky. Pod hlavičkou se zobrazí nový řádek se vstupními prvky, jak je vidět na obrázku (viz Obrázek B-8). Barva klienta ovlivní barvu projektu pro klienta a posléze i barvu úkolu v kalendáři. Po vyplnění požadovaných informací můžeme data uložit stisknutím klávesy *Enter* nebo zeleným tlačítkem na konci řádku. Zrušit změny můžeme klávesou *Esc* nebo červeným tlačítkem.

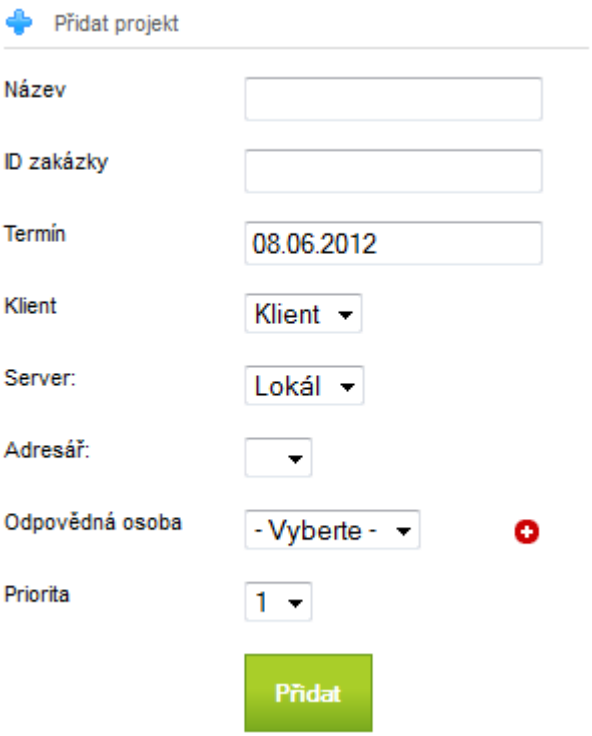
OBRÁZEK B-8: KONTAKTY, PŘIDÁNÍ KLIENTA

Jednotlivé řádky výpisu jsou po kliknutí na řádek editovatelné. Práce s editovatelným řádkem je stejná jako při přidávání záznamu. V hlavičce tabulky je možno filtrovat záznamy. To provedeme vložením hledaného řetězce do pole ve sloupci, ve kterém chceme řetězec hledat. Záznamy je možno řadit dle kteréhokoli sloupce s červenými šipkami v hlavičce. Sloupec, podle kterého se řadí, je zvýrazněn.

Správa projektů

Okno pro správu projektů je rozděleno na dvě části. V pravé je tabulka s výpisem projektů a tlačítkem pro vytvoření nového projektu. Defaultně se zobrazují pouze nedokončené projekty. Pokud chceme vidět dokončené projekty, je k tomuto účelu v levé části okna připraven filtr pro přepínání zobrazení. Výpis můžeme dále filtrovat podle hledaného řetězce v určitém sloupci výpisu. Pomocí červených šipek v hlavičce tabulky měníme řazení záznamů.

Nad tabulkou je umístěno zelené tlačítko „Přidat projekt“, které slouží k vytvoření nového projektu. Kliknutím na něj se zobrazí formulář pro nastavení informací o projektu (viz Obrázek B-9).

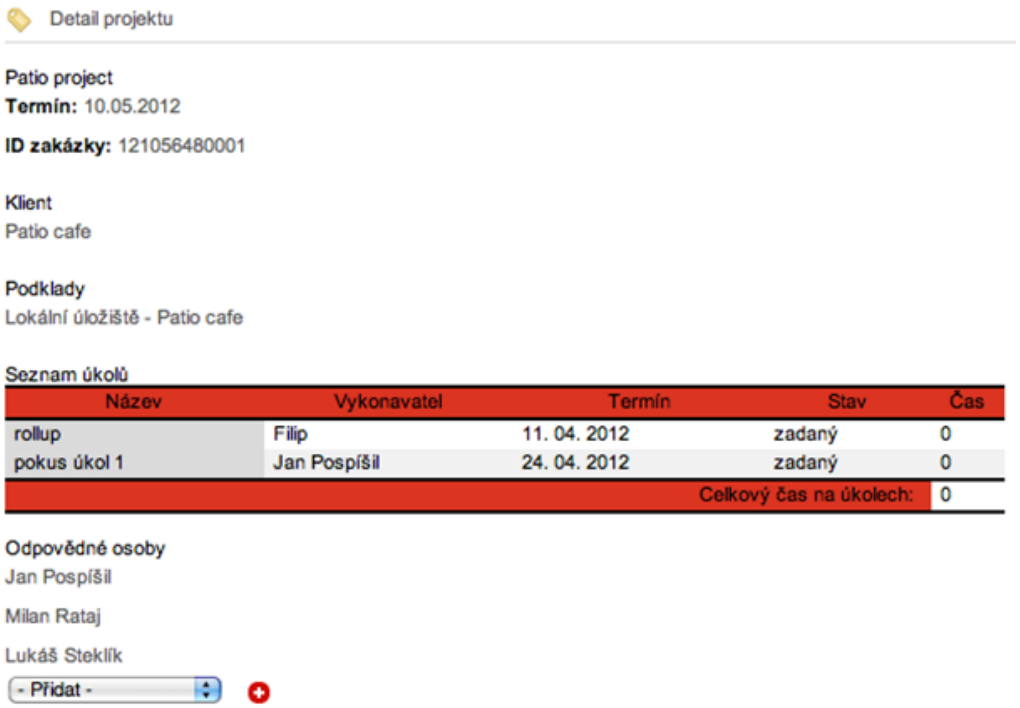


OBRÁZEK B-9: FORMULÁŘ PRO PŘIDÁNÍ PROJEKTU

ID zakázky je číslo zakázky v zakázkovém systému. Termín je datum, do kterého má být projekt dokončen (tzv. deadline). Odpovědných osob může být více a určují uživatele, kteří vidí projekt. Další odpovědnou osobu přidáme ikonou plus za posledním výběrovým polem.

Detail projektu

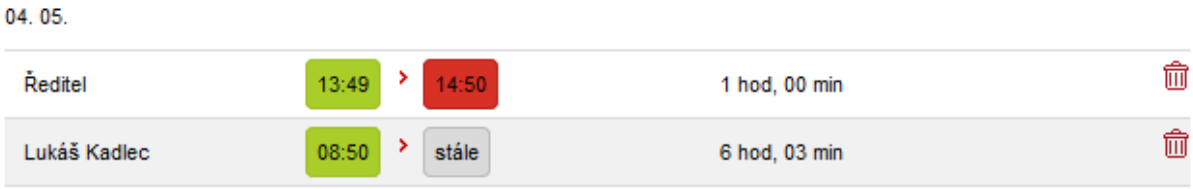
Klikneme-li ve výpisu projektů na název projektu, zobrazí se detail projektu. Zde jsou pohromadě sloučena veškerá důležitá data k projektu. Informace je možno pouze prohlížet. Výjimku tvoří odpovědné osoby, které je možné přidávat.



OBRÁZEK B-10: UKÁZKA DETAILU PROJEKTU

Sledování docházky

Pro sledování odpracované doby zaměstnanců slouží modul **Docházka**, přístupný přes shodně nazvaný odkaz v hlavním menu. Odpracovaná doba zaměstnanců je zobrazována ve výpise (viz Obrázek B-11), který slučuje časy do jednotlivých dní. Zobrazuje se jméno zaměstnance, od kdy do kdy pracoval (případně, že stále pracuje) a celkově odpracovaná doba. No konci řádku je ikona, kterou lze v případě potřeby záznam odstranit.



OBRÁZEK B-11: VÝPIS ODPRACOVANÉ DOBY

Vlevo od výpisu jsou připraveny dva filtry, kterými lze ovlivnit výpis odpracované doby. Defaultně se zobrazují záznamy za poslední týden (*Předchozích 7 dní*) a lze vyfiltrovat jen práce za aktuální den (*Dnes*).

Plánovací kalendář

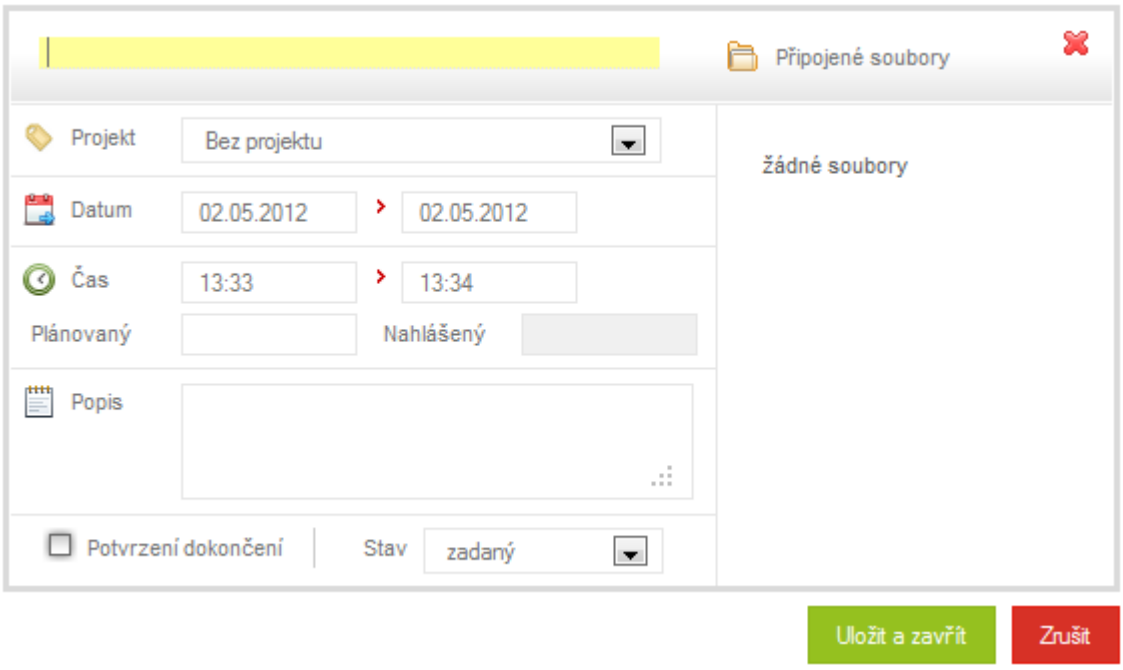
Pod odkazem **Kalendář** v hlavním menu jsou přehledně zobrazeny kalendáře všech zaměstnanců oddělení (viz Obrázek B-12). Nad kalendáři je lišta s ovládacími prvky. Šipky umožňují pohyb v kalendáři, tlačítko „dnes“ nás vrátí na aktuální datum. Tlačítko „Publikovat“ se zaktivní při kterékoli změně na kalendáři. Změny v kalendáři neuvidí nikdo jiný, dokud nejsou tzv. publikovány. Do té doby veškeré změny probíhají pouze v našem kalendáři. Tlačítka „Den“ a „Týden“ umožňují přepínat denní a týdenní (viz Obrázek B-12) zobrazení kalendářů.

V levém panelu se nachází seznam projektů, na kterých pracujeme. Barevná ikona před projektem nám umožní určit, které úkoly v kalendáři spadají do projektu. Kliknutím na projekt zobrazíme detail projektu. V neposlední řadě se v levém panelu zobrazují úkoly, které byly vykonavatelem odmítnuty. Takový úkol lze tažením myši nad požadované místo v kalendáři opět naplánovat.

OBRÁZEK B-12: KALENDÁŘ

Nový úkol

Kliknutím na den v týdnu nebo výběrem časového intervalu se zobrazí dialogové okno pro přidání nového úkolu s předvyplněným zvoleným časovým intervalem (viz Obrázek B-13). Do titulku okna vyplníme krátký výstižný popis úkolu (v ideálním případě by měl stačit k pochopení cíle úkolu), pokud potřebujeme sdělit více informací, použijeme pole „Popis“. Po výběru projektu se nám pod polem zobrazí deadline projektu, abychom zbytečně nevolili konec úkolu za konec projektu. V případě, že projekt má připojeny nějaké podklady, vypíší se v pravé části okna a zaškrtáním zvolíme soubory potřebné pro práci na úkolu. Čas můžeme vyplnit ve formátu hodiny a minuty oddělené dvojtečkou nebo samostatnou číslicí, která bude převedena na celé hodiny. Plánovaný čas uvedeme v minutách. Na závěr vytvoříme úkol kliknutím na tlačítko „Uložit a zavřít“. Případně můžeme celou akci zrušit tlačítkem „Zrušit“ nebo křížkem v pravém horním rohu.



OBRÁZEK B-13: DIALOG PRO PŘIDÁNÍ/EDITACI ÚKOLU

Editace úkolu

Kliknutím na úkol v kalendáři se zobrazí podobný dialog jako při přidávání úkolu (viz Obrázek B-13). Rozdíl je, že můžeme upravovat nahlášený čas (v minutách) a místo tlačítka „Zrušit“ máme možnost úkol odstranit tlačítkem „Smazat“.

Nastavení systému

Úvodní obrazovkou po přihlášení je nastavení systému. Na tuto stránku se můžeme kdykoli vrátit odkazem „Nastavení“ v hlavním menu.

Svátky

Jednou z možností nastavení jsou svátky a dny pracovního klidu. Nastavené události platí pro všechny uživatele systému. Defaultně jsou v systému nastaveny veškeré svátky platné v České republice. Editační formulář i formulář pro přidání nového svátku obsahují pouze dvě pole. Text z pole *„Název“* bude zobrazen v kalendáři na místě určeném polem *„Den“*. Kliknutím na záznam ve výpisu svátku se zobrazí **editační formulář**. Změněné údaje můžeme tlačítkem „Uložit“ zapsat do databáze, nebo tlačítkem „Zrušit“ ignorovat změny. Pod výpisem svátků je odkaz na **přidání svátku**. Ze zobrazeného formuláře můžeme opět bez změny odejít (tlačítko „Zrušit“) nebo přidat nový svátek (tlačítko „Přidat“).

Úložiště podkladů

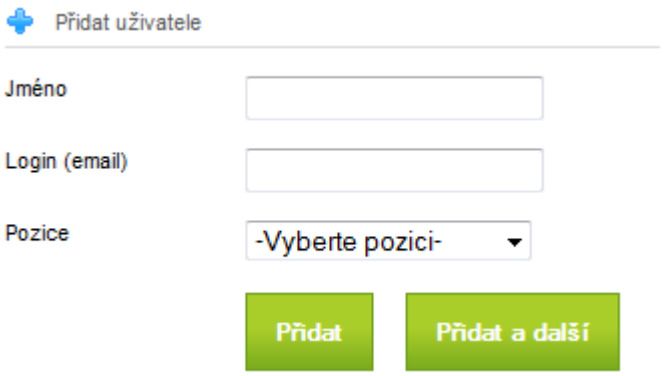
Systém umožňuje, že podklady k projektům mohou být uloženy i na jiných serverech, než je lokální úložiště společnosti. Jedinou podmínkou je, že server musí být pro čtení přístupný přes FTP. Defaultně je v systému nastaven pouze lokální souborový server. Kliknutím na název ve výpisu je možno jej **editovat** a změny uložit nebo zrušit. Pod výpisem je odkaz na **přidání** nového serveru. Pole *Server, Login* a *Heslo* budou používány k připojení se k serveru. Poslední pole *Název*, slouží k identifikaci serveru ve výpisech.

Správa uživatelů

Spravovat uživatele systému můžeme pod odkazem **Uživatelé** v hlavním menu. Na úvodní obrazovce vidíme výpis všech uživatelů systému. Kliknutím na uživatele zobrazíme informace o uživateli (jeho profilovou stránku). V levém menu jsou odkazy na správu uživatelů.

Přidání uživatele

Odkaz „Přidat uživatele“ zobrazí formulář pro přidání uživatele do systému (viz Obrázek B-14).



OBRÁZEK B-14: FORMULÁŘ PRO PŘIDÁNÍ UŽIVATELE

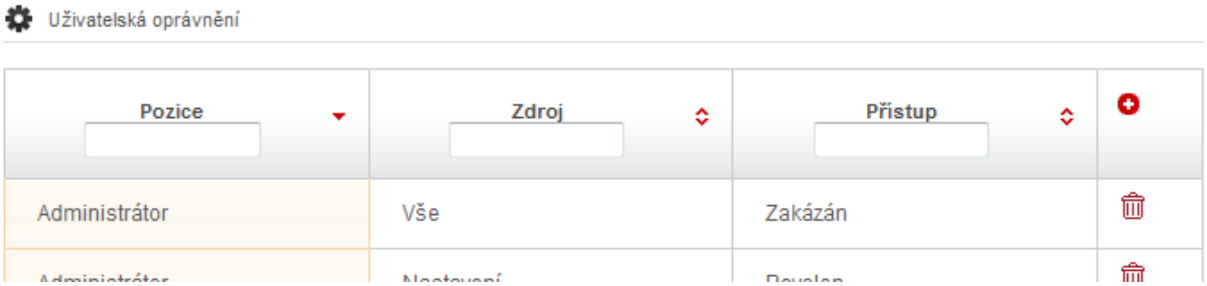
Do pole „**Jméno**“ vyplníme skutečné jméno uživatele. „**Login**“ bude použito jako přihlašovací jméno do systému a současně jako e-mail pro komunikaci systému s uživatelem. Posledním polem je „**Pozice“**, ve kterém vybereme z předdefinovaných rolí v systému (pozic ve společnosti).

Přidat můžeme pouze jednoho uživatele tlačítkem „**Přidat**“, po přidání budeme přesměrováni na výpis uživatelů, nebo ihned po vložení můžeme přidat dalšího uživatele tlačítkem „**Přidat a další**“, kdy se po uložení zobrazí opět prázdný formulář.

Novému uživateli je vygenerováno náhodné heslo. Přihlašovací údaje jsou společně s odkazem pro přihlášení odeslány na e-mail.

Uživatelská práva

Odkazem „Editace uživatelských práv“ se zobrazí tabulka s nastavenými právy jednotlivých pozic v systému (viz Obrázek B-15).



OBRÁZEK B-15: SPRÁVA UŽIVATELSKÝCH PRÁV

V tabulce vytváříme trojice „kdo, kam, co“ s významem jaká pozice má k jakému zdroji (části systému) povolen nebo zakázán přístup. V hlavičce tabulky je pole pro filtraci záznamů a odkaz (šipky) pro nastavení řazení. V pravé části hlavičky je tlačítko „plus“ pro přidání nového oprávnění. Každý záznam v tabulce vypisuje jednotlivé nastavení, které je pomocí tlačítka na konci každé řádky (koš), možno odstranit.

Kliknutím na záznam v tabulce se vybraná řádka změní na editovatelnou (viz Obrázek B-16). Změněné hodnoty můžeme na konci řádky uložit (zelené tlačítko) nebo změny zrušit (červené tlačítko). Nebo můžeme využít kláves *Enter* (uložení) a *Esc* (zrušení).

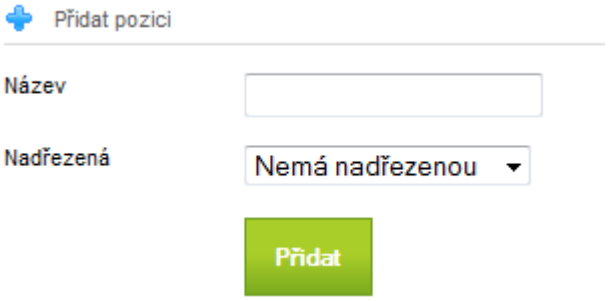
OBRÁZEK B-16: ZÁZNAM TABULKY PŘI PŘIDÁVÁNÍ/EDITACI

Správa pozic

Pozice umožňují hromadně omezit uživatelům přístup do určitých částí systému. Pozice mohou vytvářet hierarchickou strukturu, ve které od sebe jednotlivá oprávnění dědí. Odkazem „Správa pozic“ se dostaneme na stránku se stromovým výpisem existujících pozic (viz Obrázek B-17). Současně na stránce najdeme i formulář pro přidání nové pozice (viz Obrázek B-18).

OBRÁZEK B-17: VÝPIS POZIC

Jednotlivé pozice ve výpisu jsou editovatelné. Klikneme-li na název pozice, změní se formulář pod výpisem na editační (viz Obrázek B-19) a jsou v něm vyplněny informace o pozici.



OBRÁZEK B-18: FORMULÁŘ PRO PŘIDÁNÍ POZICE

Při přidávání pozice stačí vyplnit název pozice, a jaká pozice má být její nadřazenou.

OBRÁZEK B-19: EDITAČNÍ FORMULÁŘ