VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

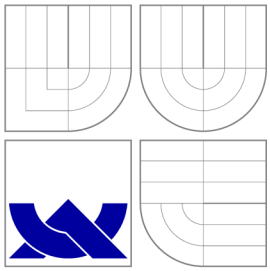
INFORMAČNÍ SYSTÉM LETIŠTĚ

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

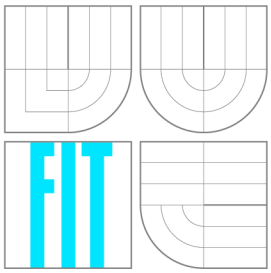
AUTOR PRÁCE
AUTHOR

JANA VELEBOVÁ

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

INFORMAČNÍ SYSTÉM LETIŠTĚ

INFORMATION SYSTEM OF AIRPORT

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JANA VELEBOVÁ
AUTHOR

VEDOUCÍ PRÁCE — Ing. ROMAN LUKÁŠ, Ph.D.
SUPERVISOR

BRNO 2008

Abstrakt

Cílem této práce je vytvořit informační systém dle požadavků sportovního letiště v Medlánkách. Tento systém bude sloužit členům aeroklubu jako zdroj informací. Jeho hlavní součástí je rozvrh leteckého provozu určený pilotům ve výcviku. Ten je generován jednou, na začátku letecké sezóny instruktorem a je zpřístupněn pouze členům aeroklubu. Vstup do informačního systému je umožněn pouze registrovaným členům aeroklubu prostřednictvím loginu a hesla. Jendotliví členové mají různá oprávnění, která jim jsou přidělena vedením aeroklubu.

Klíčová slova

informační systém, PHP, CSS, smarty, HTML, umělá inteligence, rozvrh, letiště

Abstract

Scope of this thesis is to create information system for public airport in Brno-Medlanky. This system will be used by members of aeroclub as a main stream of information. The core feature of this system is air traffic scheduling for pilot training. This schedule is generated by instructor one time at the beginning of the flight season and can be accessed inside this system only. Access to the system is permitted for the registered members of the aeroclub only. User was verified by user name and password and each member have different level of permissions. This permissions was assigned by the administrators of this information system.

Keywords

information system, PHP, CSS, smarty, HTML, artificial intelligence, timetable, airport

Citace

Jana Velebová: Informační systém letiště, bakalářská práce, Brno, FIT VUT v Brně, 2008

Informační systém letiště

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Romana Lukáše Ph.D.

.....
Jana Velebová
14. května 2008

Poděkování

Děkuji panu Ing. Romanu Lukášovi Ph.D. za odborné vedení a pomoc při práci na mé bakalářské práci.

Obsah

Kapitola 1

Úvod

S rozvojem internetu se nám otevírají možnosti, které jsme v minulosti neměli. Dostupnost internetových služeb už není neřešitelný problémem a jeho poskytovatelů stále přibývá.

To je i důvod, proč je v dnešní době vhodné využívat informační systémy jako zdroj informací prostřednictvím webových stránek. Výhoda je zejména v tom, že uživatelé mohou z pohodlí svého domova získávat informace bez nutnosti návštěvy dané instituce.

Jak ale víme, internet není jedno z nejbezpečnějších míst, a pokud se rozhodneme ho použít k zveřejnění dat, musíme zajistit jejich bezpečnost. Což sebou nese nemalé komplikace. Proto je dobré předem zhodnotit možná rizika a těm se vhodnými prostředky snažit zabránit.

Fází vývoje celého informačního systému je několik. Od specifikace požadavků, přes návrh, implementaci až po testování a správu již fungující aplikace. Proto bychom neměli ani jednu z nich podcenit. Zvláště pokud je našim úkolem spravovat systém i v budoucnu.

Informační systém letiště je určen členům aeroklubu v Brně - Medlánkách. Bude jim sloužit jako zdroj informací jak osobních, tak informacích o letadlech, leteckém provozu a dění na letišti.

Jeho hlavní součástí je rozvrh leteckého provozu určený především pilotům ve výcviku. Tento rozvrh je řešen metodou z umělé inteligence a je přizpůsoben podmínkám zadaných ve specifikaci.

Kapitola 2

Použité jazyky a prostředky

Pro vytvoření informačního systému jsem použila několik volně dostupných programů, které mi usnadnily vytvořit všechny jeho součásti.

Hlavní součástí informačního systému je databáze, pro uchovávání dat. K jejich zobrazení slouží webová rozhraní, které umožňuje přihlášení do systému a následné vyhledání a zpracování informací.

Pro jednotlivé součásti informačního systému jsem zvolila následující technologie a jazyky.

2.1 Databáze MySQL

Pro práci s databází uloženou na MySQL serveru jsem použila klientskou utilitu `phpMyAdmin`, která umožňuje pracovat s databází přes webové rozhraní. Tento přístup je praktický a zajišťuje okamžité změny v databázi. Díky kterým můžeme ihned testovat spávnost výstupních dat.

Pro vytvoření databáze a tabulek jsem použila jazyk *SQL (Structured Query Language)*. Ten možňuje data definovat, zejména vytvořením struktury a zajišťuje i možnost manipulace s uloženými daty.

2.2 Webové rozhraní

PHP 5 (Hypertext Preprocessor) je dnes velmi populární, na platformě nezávislý jazyk. Zpracovává se na straně serveru a je vhodný pro práci s databází. Mezi nejznámější PHP servery patří Apache.

Práce s PHP je spojena i se znalostí jazyka HTML. Vímě, že se php skripty zpracovávájí na straně serveru, to znamená, že se do prohlížeče odešle pouze výsledek. To je i důvod, proč při zobrazení zdrojového kódu v prohlížeči žádný PHP kód nevidíme.

Jazyk PHP je v dnešní době velmi rozšířený. Proto jsem neměla problém se získáváním potřebných informací jak o něm samotném, tak o velké škále funkcí, které nabízí.

Mnoho informací jsem se také dočetla v knihách Mistrovství v PHP5 [3] a Pokročilé programování v PHP 5 [11].

Šablony a smarty je vhodné použít, chceme-li vytvořit flexibilní web. Použítím smarty zajistíme oddělení prezenční logiky od logiky aplikace. Toto rozdělení nám umožní upravo-

vat design stránek jen v šablonách bez nutnosti zasahovat do zdrojových kódů PHP. Což oceníme především u práce v týmu.

Se šablonami jsem se setkala poprvé, a to byl i důvod, proč práce na systému ze začátku neprobíhala příliš rychle. Postupem času jsem si ale techniku práce s nimi celkem osvojila. K tomu mi napomohla i kniha PHP moduly, rozšíření, akcelerátory [13].

HTML (HyperText Markup Language) je základní prostředek pro tvorbu webových stránek. Každý HTML soubor by měl obsahovat základní prvky (tzv. tagy), jako jsou hlavička a tělo. Ty jsou důležité pro prohlížeče, kteřé podle nadefinovaných informací rozpoznají a správně zobrazí dané informace.

I když znám HTML už nějakou dobu, nepoužívám ho každý den. A občas se hodí mít po ruce alespoň základní syntaxi jazyka. K tomu mi posloužila i kniha Webmaster v kostce [1] kde jsem si našla potřebné informace.

CSS (Cascading Style Sheets) slouží ke grafické úpravě webových stránek. Styly lze vkládat přímo do HTML kódu, nebo je můžeme importovat, z externích souborů. Jsou vhodné pro vytvoření uživatelsky příjemného prostředí nejen vzhledově, ale zpřehledňují i aspekt funkčnosti celých webových stránek.

Pro rozšíření dosavadních znalostí jsem použila knihy CSS filtry, hacky a pokročilé postupy [4] a CSS pokračujeme s kaskádovými styly profesionálně [9].

JavaScript je multimediální, objektově orientovaný, skriptovací jazyk. Je zpracovávaný na straně klienta a programy v něm napsané je možné vkládat do webových stránek, u kterých zajišťuje jejich interaktivitu. Jsou vhodné zejména pokud chceme na svých stránkách na něco upozornit, nebo chceme-li uživateli ulehčit práci. Například při vyplňování různých formulářů ať už použitím kalendářů, či zvýraznění nevyplněných položek.

Kapitola 3

Specifikace požadavků

Požadavky jsou jednou z nejdůležitějších součástí, bez které by informační systém nemohl vzniknout. Proto je důležité jim věnovat velkou pozornost.

Ve většině případů jsou tyto požadavky zadávány zákazníkem, což může být spojeno s nemalými problémy. Musíme si uvědomit, že zákazník nemá přesnou představu o tom, jak probíhá programování a s jakými problémy se můžeme setkat. V tomto případě je dobré se zákazníkem projít celou specifikaci a upozornit ho na možné problémy, nebo dokonce i na různé úpravy vedoucí k optimalizaci a větší efektivitě celého systému.

Můžeme se setkat i s lidmi, kteří mají programátorské znalosti a domluva s nimi bývá obvykle snadnější. Nemusíme jim například do detailů vysvětlovat, proč zrovna tuto variantu návrhu považujeme za nevhodnou. To ale neznamená, že domluva s programátory bývá snadná.

Shodneme-li se se zákazníkem na všech bodech specifikace, je dobré si ji nechat písemně potvrdit. Vyvarujeme se tím nepříjemnostem do budoucna. A má-li zákazník dodatečné požadavky, není problém po zhodnocení všech faktorů daný požadavek do specifikace doplnit s tím, že se termín dodání informačního systému při rozsáhlejších úpravách může změnit.

Specifikace požadavků pro informační sytém sportovního letiště v Medlánkách bohužel nebyla až tak úplná. To vedlo i k určitým nepříjemnostem při samotné implementaci, kdy jsem narazila na problémy, se kterými jsem se při návrhu nesetkala. Z toho alespoň plyne ponaučení do budoucna.

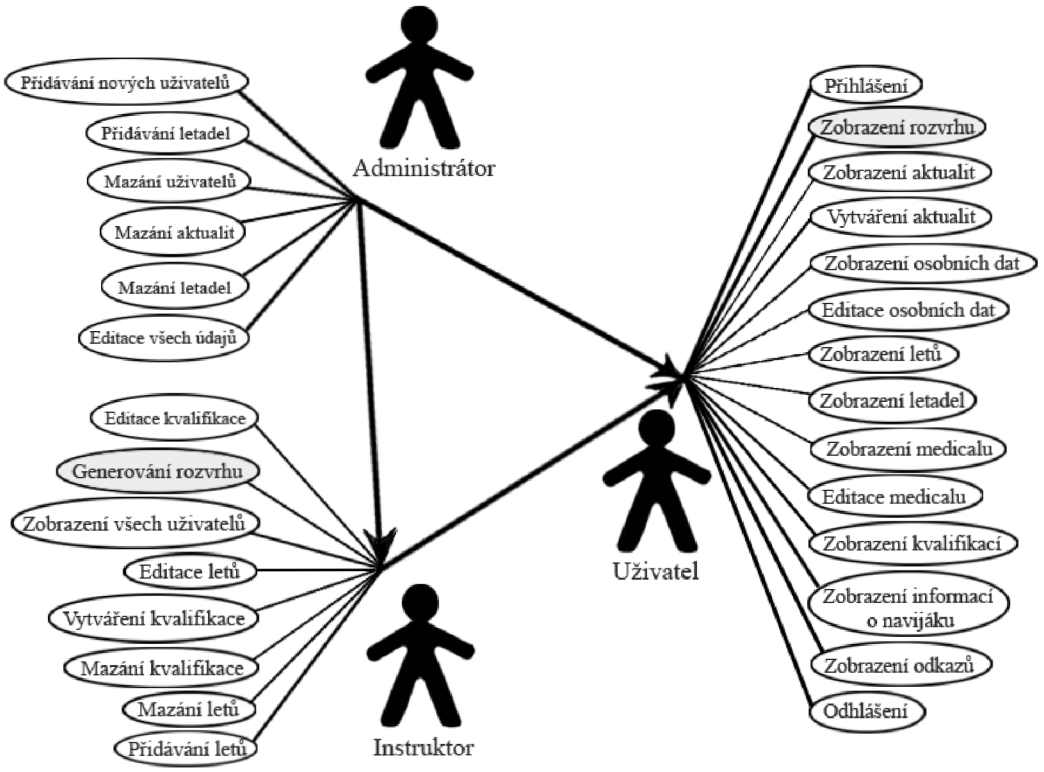
3.1 Detailní specifikace informačního systému letiště

- Systém by měl nahradit dosavadní způsob správy informací a dat na letišti. Má umožňovat přístup všem členům aeroklubu přes webové rozhraní se zajištěním bezpečného uložení a správy dat.
- Jednotliví členové aeroklubu budou rozděleni do skupin dle přístupových práv a funcí vykonávajících na letišti. Tyto skupiny by se měly lišit oprávněním na manipulaci s daty. Zejména bude-li se jednat o vkládání, mazání a editaci.
- Systém bude členům aeroklubu sloužit jako zdroj aktuálních informací, nejen o nich samotných, ale i o letadlech, leteckém provozu a dění na letišti.
- Vstup do systému by měl být umožněn jen registrovaným členům, které má oprávnění registrovat pouze administrátor, nebo osoba k tomu pověřená.

- Měla by být zajištěna přehlednost, jednoduchost a nenáročnost uživatelsky příjemného prostředí. S ohledem na vhodně zvolený design bez reklamních a rušivých prvků.

Bližší specifikace je znázorněna v diagramu užití 3.1.

3.2 Diagram užití (Use Case Diagram)



Obrázek 3.1: Diagram užití pro informační systém sportovního letiště v Medlánkách

Pokud při implementaci narazíme na něco, co ve specifikaci není, ale přesto by to mohlo vést k větší efektivitě, je dobré tuto variantu projednat se zákazníkem. Nejsme-li vázáni na zákazníka, či jiné aspekty, můžeme danou věc implementovat bez jakýchkoli komplikací.

Kapitola 4

Návrh systému

Před samotnou implementací je dobré si vše nejdříve navrhnout. Omezíme tak množství problémů, se kterými bychom jsme se mohli setkat při implementaci.

Návrh musíme také přizpůsobit dostupným prostředkům a našim znalostem. Zvláště, nemáme-li dostatek času na osvojení nových technologií.

Způsobů, jak můžeme návrh realizovat, je mnoho. Proto je na nás, jaký si zvolíme. Někdy nám postačí pouze návrh nakreslený na papíře, můžeme však použít i softwarové aplikace, které nám mohou, v určitém aspektu, práci usnadnit.

Pokud v této fázi narazíme na nějaký problém, nemusíme předělávat celou aplikaci a stačí jednoduše zvolit vhodnější variantu postupu. Pokud narazíme na složitější problém, je dobré na něj upozornit zákazníka a domluvit se s ním na možných úpravách. Řešení obvykle nebývají až tak náročná a jsou realizována v mnohem kratší době, než v případě, kdy by se problém objevil až při samotné implementaci. U vývoje jakékoli aplikace platí, že čím více času musíme mít na její dokončení, tím více nám rostou finanční náklady. Proto se snažíme vše dokončit v co nejkratším čase a s co nejlepšími výsledky.

Během vlastní implementace nám návrh slouží jako vzor, což ale neznamená, že jej během vývoje nemůžeme měnit. A to především proto, že při implementaci se můžeme setkat s problémy, se kterými jsme při vlastním návrhu nepočítali. Tyto problémy mohou být různě závažné. A hodně záleží na tom, jak dobře byl samotný návrh vytvořen.

Tato oblast je velmi náročná zejména proto, že až časem, stráveným nad vytvářením mnoha návrhů, jsme schopni správně odhadnout možná rizika a konflikty. Jednoduše, chybami se člověk učí a čím více návrhů vytvoříme, tím lepší pak máme představu o možných konfliktech a způsobu jejich řešení.

4.1 Návrh databáze

Databáze je tvořena tabulkami, kde každá tabulka obsahuje informace o různých objektech. K těmto datům je umožněno přistupovat přes webové rozhraní informačního systému, který umožnuje jejich modifikaci. A to pouze osobám, k tomu určeným.

Databáze obsahuje dva druhy tabulek, které jsou přizpůsobeny svému účelu.

4.1.1 Tabulky

Slouží pro uchovávání informací, které jsou následně zpracovány. Každá tabulka má nadefinovaný primární klíč a může obsahovat i klíče cizí. Ty slouží k rozšíření samotné tabulky.

K dispozici máme tabulky:

Člen_ak uchovává všechny potřebné informace o všech členech aeroklubu, jako jsou například jméno, přijmení, adresa a mnohé další. Obsahuje také jeden cizí klíč. Tento cizí klíč je identifikátor práv, která jsou v systému nadefinovaná v samostatné tabulce. U uživatele proto uchováváme pouze identifikátor.

Medical je průkaz dokládající absolvování povinné lékařské prohlídky, bez které by nemohl pilot létat. Důležitými údaji, které musíme uchovávat, je datum vydání a datum platnosti, třídu pro kterou byl průkaz vydán a lékaře, který prohlídku provedl. Jelikož jde o samostatnou tabulku, je zde nutné uchovávat identifikační číslo člena aeroklubu, podle kterého je prohlídka přidělena. Jelikož jsou prohlídky platné dva roky a není potřeba uchovávat historii již neplatných prohlídek, je v databázi uložena pouze ta aktuální a platná prohlídka. To znamená, že jeden uživatel může mít pouze jeden záznam o lékařské prohlídce, kterou má oprávnění editovat.

Naviják je zvláštní druh kvalifikace umožňující manipulaci s navijákem. K jeho získání je nutné absolvovat kurz, který je ukončen závěrečnou zkouškou. Po úspěšném vykonání zkoušky je v současné době platnost této kvalifikace neomezená. Proto je v tabulce uchováváno pouze identifikační číslo člena aeroklubu a datum vydání. Je zde i položka datum platnosti, ale to především z důvodu možných změn v budoucnu.

Aktuality umožňují informovat piloty o aktuálním dění na letišti. Vytvářet aktuality mohou všichni členové, ale mazání aktualit je umožněno pouze administrátorovi. Tabulka obsahuje krom svého identifikačního čísla i čas vytvoření aktuality, identifikátor člena, který ji vytvořil, nadpis a text. Aktuality jsou při výpisu seřazeny podle data.

Kurz je určena pro uchovávání informací o vygenerovaném rozvrhu. Uchovávány jsou informace pouze o jednom kurzu. Při opětovném vygenerování se současná data smažou a jsou vložena data nová. Tabulka obsahuje den, skupinu žáků, instruktora a letadlo. Každý řádek odpovídá jednomu dni kurzu. Pomocí této tabulky je vygenerovaný rozvrh pro daný kurz možné zobrazit.

Kvalifikace je určena nejen členům aeroklubu, ale také jednotlivým letadlům. U letadel je kvalifikace převážně informativní, ale u jednotlivých členů nám říká, s jakými typy letadel mohou létat. Kvalifikaci může zadat instruktor, třeba po úspěšném dokončení leteckého výcviku, nebo administrátor v případě, že člen při vstupu do aeroklubu již některé kvalifikace má. Tabulka kvalifikací obsahuje pouze identifikátor kvalifikace, její název a zkratku.

Práva jsou určeny k oddělení jednotlivých skupin uživatelů informačního systému. Podle těchto práv je ošetřen přístup do jednotlivých částí informačního systému. To, že je tato tabulka samostatně, umožňuje přidávání nových práv bez zásahu do struktury databáze. Tabulka obsahuje krom identifikátoru také název a popisek jednotlivých práv.

Letadlo je tabulka uchovávající informace o letadlech. Například majitele, provozovatele, typ, kategorii a mnoho dalších potřebných informací týkajících se letadla. Tabulka obsahuje

několik cizích klíčů, podle kterých můžeme vyhledat přesnou kategorii letadla či kvalifikaci nutnou pro jeho pilotáž.

Kategorie_letadla přesně definuje danou kategorii letadla. Kategorií letadel může být několik. Například motorové, bezmotorové nebo ultra-lehké letadlo. Tabulka obsahuje identifikátor a danou kategorii.

Let uchovává informace o jednotlivých letech a všech jeho účastnících, času startu, přistání, a také informaci o přistání do terénu, ke kterému když dojde, je letadlo posláno na technickou prohlídku. V tabulce najdeme i cizí klíče, pomocí kterých se můžeme dozvědět, o jaký typ startu, letu či úlohy se jednalo.

Typ_start je jedna z důležitých informací. Každý typ startu sebou nese odlišné informace. Na rozdíl od motorových letadel rozlišujeme u bezmotorových letadel dva typy startů. Start s vlečnou, v případě vytažení letadla do určité výšky pomocí jiného letadla (tzv. vlečné), nebo navijákový start, který vytáhne bezmotorové letadlo navíjením ocelového lana připevněného k letadlu do výšky od 150 do 300 metrů. U typu startů ukládáme identifikační číslo a typ startu.

Let_typ udává typ letu, který může být například orientační, vyhlídkový či jinak specifikovaný. V tabulce ukládáme pouze identifikátor jednotlivých typů a jejich název.

Úloha specifikuje daný typ úlohy při samotném letu. Úlohy rozlišujeme podle jejich identifikátoru a názvů.

4.1.2 Propojovací tabulky

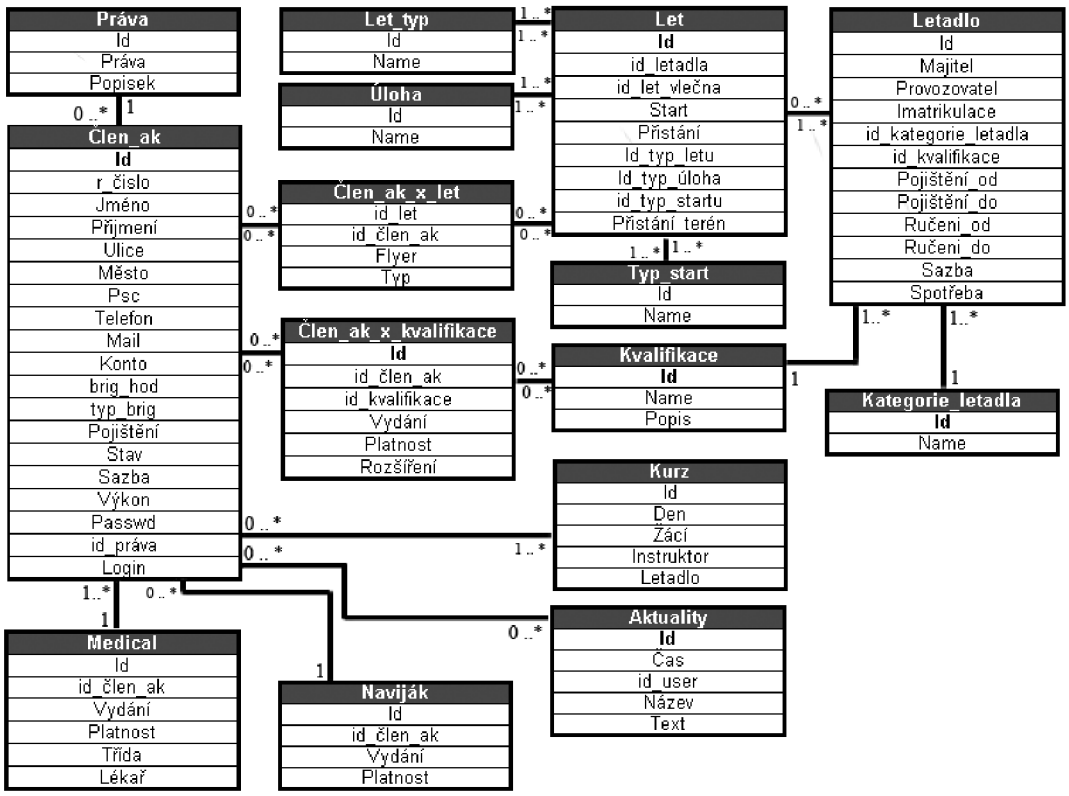
Jsou používány v případě, potřebujeme-li propojit více tabulek. Umožňují flexibilitu a možnou rozšiřitelnost. Obsahují převážně cizí klíče, kterými dané tabulky můžeme propojit. Při ukládání dat do tabulek spojených s propojovacími tabulkami musíme ošetřit, aby byly data uloženy na všech potřebných místech. Jinak by mohlo dojít k zadání neúplných informací, které by mohly způsobit špatné, nebo neúplné přidělení dat.

K dispozici máme dvě propojovací tabulky:

Člen_ak_x_let propojuje člena aeroklubu s absolvovaným letem. A to zejména proto, že jednomu letu může být přiděleno více účastníku. Zvlášť v případě, kdy jeden je pilot a ostatní tvoří posádku letadla. Tabulka obsahuje cizí klíče, jejichž hodnota udává komu má být let přidělen a ve kterém letadle byl let uskutečněn. Dále je zde položka, která určuje, zda-li byla daná osoba pilotem, nebo nikoli a typ pilota. Typ pilotů rozlišujeme zejména ve výcviku, kdy může být pilotem i žák.

Člen_ak_x_kvalifikace propojuje člena aeroklubu s kvalifikací, kterých může mít člen dle svých možností několik. I zde jsou použity cizí klíče pro správně přidělení kvalifikace danému uživateli. Dalšími položkami jsou datum vydání, datum platnosti a možné rozšíření kvalifikace.

4.2 ER diagram



Obrázek 4.1: ER diagram

4.3 Uživatelské rozhraní

Je velmi důležité zvláště pro uživatele. Proto byl měl být návrh přizpůsoben nejen veškerým požadavkům, ale měl by zajišťovat i uživatelsky příjemné prostředí. Vhodně zvolený design, který bude zajišťovat přehlednost a jednoduchost, by měl být základem.

Je dobré si vytvořit i více odlišných návrhů v různých barvách a s různými prvky. Z návrhů pak vybereme ten, který nám bude nejvíce vyhovovat. Návrhy také můžeme ukázat více lidem, a nechat je, aby nám k nim řekli svůj názor. A podle zjištěných poznatků pak můžeme dané návrhy upravit.

Pokud vytváříme informační systém na zakázku, bývá obvyklé vytvořené návrhy konzultovat přímo se zákazníkem, který si daný systém objednal. A podle jeho připomínek pak návrh upravit.

Návrh by měl být také přizpůsoben tématické oblasti, ve které bude používán. To znamená, že pokud budeme vytvářet návrhy pro letiště, můžeme použít prvky spojené s létáním či letadly. To ale není podmínkou. Přesto by však na první pohled mělo být jasné, pro jakou oblast je daný systém určen.

Bylo by hodně matoucí, kdybychom vytvářeli systém pro letiště a na úvodní stránce by byly zobrazeny lodě nebo ponorky. I když by se mohlo jednat o opravdu uchvacující vzhled, asi by na první pohled nebylo zřejmé, že se nacházíme na stránkách letiště.

V dnešní době máme k dispozici mnoho nástrojů a technologií, které nám naši práci usnadňují. Je pouze na nás, co si pro implementaci systému zvolíme.

Kapitola 5

Implementace informačního systému

Vlastní implementace je asi nejnáročnější část, ve které dochází k realizaci specifikovaných požadavků a vytvoření uživatelského rozhraní. Zabírá převážne nejvíce času v celém vývoji daného systému.

Nejdříve bylo nutné vytvořit databázi. Tu jsem vytvořila pomocí klientské utilitu phpMyAdmin, ve které jsem vytvořila tabulky obsahující testovací data.

Vlastní implementaci jsem realizovala na svém počítači, kde jsem si nainstalovala potřebné prostředky pro vytvoření databáze a správnou funkčnost zpracování PHP kódů. Pro práci jsem použila MySql server a webový server Apache.

Práce tzv. na localhostu je rychlá a nepotřebuje připojení k internetu, což vzhledem ke kvalitám internetových poskytovatelů můžeme hodnotit jako výhodu.

Podle specifikovaných požadavků jsem postupně implementovala požadované funkce. Během této implementace jsem ale musela modifikovat základní strukturu databáze. Tyto změny vedly k větší efektivitě a přehlednosti celého systému. Což bylo spojeno s nutnými úpravami již funkčních části.

5.1 Práce s databází

Pro základní funkčnost systému je nutné zajistit správnou komunikaci aplikace s databází. Kdyby komunikace nefungovala spávně, nebylo by možné se do systému ani přihlásit, natož v něm pracovat.

Podle typu oprávnění umožňujeme uživatelům pracovat s databází na více úrovních. A to zejména pro zajištění bezpečnosti dat. Základní a nejvíce omezené možnosti mají uživatelé, kteří jsou na letišti v pozici pilotů, nebo pilotů ve výcviku. Instruktoři jsou na tom o něco lépe a administrátor může skoro vše. Tyto skupiny oprávnění se mohou měnit, to je však už v kompetenci vedení aeroklubu, které o těchto věcech rozhoduje.

Rozpis oprávnění pro jednotlivé skupiny

- **Uživatel** - zobrazení informací bez možnosti editace a mazání dat, krom editace osobních údajů
- **Instruktor** - zobrazení, omezené editace a mazání dat

- **Administrátor** - skoro neomezené možnosti práce s aplikací

Pro komunikaci s databázi je v systému vytvořená třída *db.class.php*, která usnadňuje používání jednotlivých funkcí souvisejících s databází.

5.2 Možnosti informačního systému

Jsou přizpůsobeny požadavkům definovaných ve specifikaci. Podle přiřazených práv mohou v systému pracovat v různých částech.

Systém uživateli umožňuje

- přihlášení a odhlášení
- zobrazení a editaci osobních údajů
- výpis všech jeho letů s časovými údaji
- zobrazení rozvrhu leteckého provozu pilotům ve výcviku
- zobrazení informací o všech dosažených kvalifikacích
- zobrazení a možnost editace lékařské prohlídky
- zobrazení letadel
- zobrazení a vytváření aktualit

Systém instruktorovi umožňuje mimo všech funkcí které má uživatel

- generování rozvrhu leteckého provozu pilotům ve výcviku
- možnost editace,přidávání a mazání kvalifikace
- zobrazení všech členů aeroklubu
- editaci, přidávání a mazání letů

Administrátorovi systém umožňuje mimo všech funkcí které má instruktor

- editace, přidávání a mazání členů aeroklubu
- přidávání a mazání letadel
- mazání aktualit

Jelikož je informační systém určen pouze registrovaným členům a bude součástí webového rozhraní stránek aeroklubu, není nutné zde zobrazovat všechny informace o letišti. Dokonce implementace fotogalerie nebo diskuzního fóra by byla zbytečná, protože tyto součásti jsou již vytvořeny.

Při implementaci jsem pro každý objekt zvolila vlastní třídu, která obsahuje všechny potřebné funkce. Mezi nejzákladnější třídy patří třída *Kurz, User, Plane, Fly a Page*. Dále

bylo použito několik pomocných tříd pro práci s databázi, třídu pomocných funkcí a mnohé další.

V aplikaci můžeme narazit na dva typy funkcí, veřejné a privátní. Privátní funkce mohou být volány pouze prostřednicvím jiné funkce a většinou zajišťují kontrolu dat, čí získávání potřebných informací z databáze.

Aby bylo možné výsledné informace získané v jednotlivých funkcích zobrazit, musíme je nejdříve poslat šabloně, ve které si nadefinujeme jak bude výsledná stránka vypadat.

```
$tpl = new template();
//přiřadí proměnné $part řetězec view
$tpl->assign('part','view');
//přiřadí proměnné $users obsah proměnné $data
$tpl->assign('users',$data);
//v proměnné $user_right si pošleme práva přihlášeného uživatele
$tpl->assign('user_rights',$u->_data['prava']);
return $tpl->fetch('user.class.tpl');  //zajistí zobrazení šablony
```

Strukturu stránek vytváříme pomocí *HTML* přímo v šablonách, které mají koncovku *tpl*. To, že se jedná o příkazy a funkce *Smarty* indentifikují složené závorky. Pomocí nich můžeme pracovat s *assign*ovanými proměnnými. Pro zobrazení všech částí informačního systému slouží šablona *page.class.tpl*, která zajišťuje vytvoření hlavičky, těla a zápatí stránek. Další použité šablony obsahují pouze jednotlivé části, které se za daných podmínek zobrazí. Správné zobrazení pouze vybraných částí zajištuje proměnná *part*, která obsahuje název dané části .

```
{*víme, že se má zobrazit úvodní přihlašovací stránka*}
 {if $part eq 'login'}
   {*vytvoření hlavicky a menu*}
   {call object='page' method='top' active='login'}
     {*HTML kód*}
   {*zápatí a konce ukončení všech tagů stránky}
   {call object='page' method='bottom'}
```

Bylo také nutné zajistit, aby se uživatelům s různým typem oprávnění zobrazovaly pouze jim určené informace. Proto se do šablon posílají i práva přihlášených uživatelů, podle kterých probíhá výběr zobrazených informací. Zobrazeny byly pouze ty položky a stránky, na které má uživatel právo.

Tímto postupem se zamezilo situaci, kdy se uživatel kliknutím na položku dozví, že na zobrazení dané sekce nemá oprávnění. Je to sice také jedna z možností, ale osobně si myslím, že to uživatele přinejmenším zmate a bude se snažit přijít na to, proč právě jemu byl přístup odmítnut.

Dalším krokem bylo vytvoření uživatelského rozhraní a celkového vzhledu stránek.

Zde se uplatňuje využití šablon, které nám v mnohém práci usnadní. Hlavní výhodou je především to, ze veškeré změny potřebné pro úpravu vzledu byly uskutečněny pouze v šablonách bez nutnosti zásahu do php kódů. Pro rozmístění komponent na stránce a celkového vzhledu jsem použila *CSS*.

Při návrhu byl zohledněn fakt, že se jedná o informační systém a není nutné zde zobrazovat podrobné infromace o činnostech aeroklubu ani jakékoli reklamní prvky či upoutávky. A vzhledem k účelu těchto stránek jsem zvolila spíše jednoduchý a nenáročný design 2.

5.3 Bezpečnost

Výhodou webových informačních systémů je přístupnost na internetu. To však může vést ke vzniku nemalých problémů. A zajištění bezpečnosti není jednoduché.

Než začneme cokoli dělat, je dobré zvážit všechna možná rizika, která nám hrozí. Z nich pak vybrat ta, která jsou pro nás nejnebezpečnější a k nim stanovit podmínky a pravidla pro zabezpečení celého systému. Nesmíme ale také zapomenout na zvídavé uživatele, kteří by se mohli snažit napadnout systém zevnitř.

Útokům a chybám webových aplikací se věnuje i kniha Hacking bez tajemství [7]. Ta upozorňuje na nedokonalost používaných technologií, které jsou snadno zneužitelné. Proto je dobré si tuto knihu přečíst a uvedeným problémům se vyhnout. I když asi nezamezíme všem možným útokům, můžeme je alespoň omezit.

Doporučená opatření:

- dostat základy bezpečnosti do podvědomí všech uživatelů systému
- vhodně zvolit ochranné prostředky
- zhodnotit možná rizika a jejich následky
- navrhnout obecná opatření při napadení systému
- vytvořit bezpečnostní plán kontroly systému

Při implementaci informačního systému jsem se zaměřila především na to, aby se do systému nedostal neoprávněný uživatel. Vstup do systému je zajištěn pomocí loginu a hesla. Ty jsou uživatelům přidělena při registraci, kterou provede administrátor. Dle požadavků není možná registrace samotnými uživateli. Pokud dojde při přihlášení k chybě, to znamená, že login nebo heslo nebylo správné, vypíše se chybové hlášení a je umožněno se uživateli přihlásit znovu. Počet možných přihlášení není omezen.

Heslo je zašifrováno pomocí metody SHA 1, která zajišťuje, že se nikde neposílá v nezašifrované podobě.

Dále také bylo nutné zajistit, aby se přihlášení uživatelé nedostali do sekcí, ke kterým nemají přístup. To je naimplementováno tak, že se v každé funkci posílají mimo jiných dat i práva přihlášeného uživatele, podle kterých je zajištěno správné zobrazení všech prvků na stránce. To je na první pohled zřejmé při přihlášení jednotlivých uživatelů, kterým je nabídnuto odlišné menu. Také nabízené funkce se u zobrazených dat liší a to podle oprávnění daných ve specifikaci.

Může se však také stát, že si uživatelé navzájem posílají odkazy. V tuhle chvíli ale teoreticky nevíme, jestli má uživatel na danou stránku přístup. Proto je mu zobrazena stránka, kde se může přihlásit. Pokud dojde k úspěšnému přihlášení do systému, zkontrolujeme, má-li daný uživatel na zobrazení požadované stránky oprávnění. Pokud ne, vypíšeme chybové hlášení. Pokud ano, přesměrujeme jej přímo z přihlašovacího formuláře na danou stránku. To je zajištěno tzv. proměnnou *back*, která obahuje potřebné informace pro zobrazení správné stránky.

Stejný princip je použit při podvržení linku, kdy si kdokoli může dosadit do odkazu co uzná za vhodné.

Je také důležité kontrolovat správnost dat zadaných do formulářů, tyto možnosti jsou ale omezené. Zvláště v případě, kdy nemáme přesně definované jednotlivé prvky, které se ve výrazu vyskytovat nemohou. To ale neznamená, že je zbytečné kontrolu vynechat.

Chceme-li se vyhnout možným problémům, je dobré omezit i délku vkládaných řetězců.

Při kontrole jsem se také zaměřila i na správný typ dat, a správnost zadaných informací. Například při vyplňování mailové adresy, jsem pomocí regulárního výrazu ověřila, obsahuje-li zadaný text požadované prvky splňující definici mailové adresy.

Funkce pro kontrolu mailové adresy:

```
 function IsEmail($e){
   return eregi("^[a-zA-Z0-9]+[_a-zA-Z0-9-]*(\.[_a-z0-9-]+)*@
                  [a-z??????0-9]+(-[a-z??????0-9]+)
                 *(\.[a-z??????0-9-]+)*(\.[a-z]{2,4})$", $e);
}
```

5.4 Testování

Jediná možnost, jak ověřit spávnou funkčnost aplikace, je neustále něco testovat. Ať už při implementaci, kdy testujeme to, co jsem právě udělali, tak při dokončení daného projektu.

Testovat můžeme skoro cokoli. Zvláště testování bezpečnosti není dobré podceňovat. Také správnost zobrazených dat a funkčnost jednotlivých komponent je nezbytná.

Z vlastních zkušeností vím, že je dobré testovat pokaždé, dojde-li k jakékoli změně v systému. A to například pomocí kontrolních výpisů. Zejména v případě, kdy musíme upravit tabulku v databázi. Někdy nás může překvapit i zdánlivá maličkost, které si na první pohled nevšimneme, protože máme pocit, že to prostě fungovat musí, když už to jednou fungovalo.

Při testování jsem se zaměřila především na:

- Funkčnost aplikace, zejména na dobře vypsané a odeslané hodnoty a bezchybné přesměrování všech odkazů na požadované stránky, které jsou součástí informačního systému
- Správné zobrazení webového rozhraní v prostředí několika webových prohlížečů (Firefox, IE, Opera)
- Bezpečnost

Především v oblasti bezpečnosti bylo nutné zajistit, aby nedošlo k proniknutí do systému neoprávněnou osobou.

Kapitola 6

Implementace rozvrhu

Rozvrh je určen pilotům přihlášeným do plachtařského výcviku pro získání pilotního průkazu pilota kluzáků. Tento výcvik začíná obvykle na podzim teoretickou výukou a po úspěšném složení závěrečných zkoušek pokračuje praktickou výukou probíhající na sportovním letišti v Brně - Medlánkách [2].

Vzhledem k možnostem letiště je kurz otevřen pouze v případě, kdy bude minimální počet přihlášených členů v daném roce 20 a maximální počet nebude více jak 30 členů. To souvisí i s počtem instruktorů a letadel, které má aeroklub k dispozici. S ohledem na zkušenosti z předchozích let je aeroklub ochoten jednomu kurzu přidělit právě čtyři instruktory a čtyři letadla.

Pro každý výcvikový den s otevřeným leteckým provozem je zajištěna pozemní radiová služa AFIS a lidé pro provoz navijáku nebo vlečné.

Pomocí navijáku je větroň vytažen do výšky 150 až 300 metrů. Tato výška je dostačující pro základní výcvik. Při aerovleku se letadla dostávají mnohem výš, což umožňuje nácvik krizových situací a zdokonalení leteckých dovedností.

Rozdíl mezi těmito typy startů není v rozvrhu patrný, protože pro oba typy startů musí být splněny stejné základní podmínky a je pouze na instruktorovi, který rozhodne, může-li pilot od navijákových startů přejít na starty s vlečnou.

6.1 Požadavky

Byly specifikovány celkem přesně. A to zejména proto, že výcvikový kurz musí probíhat podle předem daných směrnic. Vzhledem k náročnosti rozvrhu nebyly při implementaci zohledněny faktory ovlivňující letecký provoz. Jako například meteorologické podmínky, náhlá neúčast jednotlivých členů výcviku a technické poruchy letadel. Tyto faktory při generování rozvrhu na začátku sezóny totiž neznáme. A pokud bychom je při generování vkládali náhodně, stejně by výsledný rozvrh nebyl adekvátní pro danou situaci.

Upřesnění požadavků:

- žáci jsou rozděleni do dvou skupin na každý den
- v každé skupině může být minimálně pět a maximálně osm žáků na intruktora
- každá skupina má přiděleného jednoho instruktora a jedno letadlo
- žáci ani instruktoři nesmějí létat dva dny po sobě

- v jednom dni jsou k dispozici dvě letadla
- výběr letadel je omezen pouze tím, že si nemůžeme vybrat letadlo, které je již přiděleno jiné skupině

Podle daných podmínek je rozvrh pomocí metody z umělé inteligence vygenerován na všechny dny výcviku. Počet dní se pro jednotlivé výcviky může lišit. Záleží na počtu přihlášených žáků. Aby mohl být kurz dokončen, musí mít všichni jeho účastníci mimo instruktorů odlétaný daný počet hodin.

6.2 Postup implementace

Prvním krokem bylo vybrat z databáze pomocí jednoduchého dotazu žáky, kteří se přihlásili do kurzu. Zde bylo nutné ověřit, je-li počet těchto žáků v rozsahu možností pro otevření kurzu. Kdyby bylo žáků málo, nemohl by být rozvrh vygenerován. Při překročení maximální kapacity kurzu je nutné přebývající žáky vyřadit. Ale to je v kompetenci aeroklubu. Obdobným způsobem byly získány informace o instruktorech a letadlech.

V rozvrhu předpokládáme, že všichni žáci musí odlétat minimálně sedm hodin, aby mohli dokončit kurz. Tento počet hodin je úměrný všem úlohám a samostatným letům, které musí žáci splnit. Jeden let má průměrnou délku asi pět minut, a žák může během dne uskutečnit v průměru šest startů. Z čehož vyplývá, že při každém odlétaném dnu se žákovi připočte půl hodiny.

Do rozvrhu nelze přímo implementovat jednotlivé úlohy. Každý žák ve svém výcviku postupuje individuálně a je jen na instruktorovi, kdy umožní žákovi pokračovat novou úlohou.

Nejdříve jsem vytvořila objekt **kurz**, kterému jsem vložila informace o všech žácích, instruktorech a letadlech. K tomuto účelu byly vytvořeny funkce:

- `addLetadlo($p['id'], $p['typ']` - vložení letadlo do kurzu
- `addInstruktor($i['id'], $jmeno)` - vložení instruktora do kurzu
- `addStudent($s['id'], $jmeno, $hodin)` - vložení žáka do kurzu

Kurz se skládá z jednotlivých dnů, ve kterých jsou dvě skupiny žáků. Každé z těchto skupin je přidělen jeden instruktor a letadlo podle předem specifikovaných pravidel.

6.2.1 Rozdělení do skupin

Další fází generování kurzu je rozdělení jednotlivých žáku, instruktorů a letadel do dvou skupin. To zajišťují následující funkce:

- `borrowAircraft()` - zajistí výběr volného letadla
- `findInstructor($instructors)` - vybere vhodného instruktora
- `findStudent($students,$instId)` - vybere nejvýhodnějšího žáka

Není možné předem odhadnout počet lidí ve skupině, proto je v rozvrhu tento počet v rozmezí pěti až do osmi žáků na skupinu vygenerován náhodně.

Jako první dojde k vybrání jednoho ze čtyř instruktorů do jedné skupiny pomocí funkce `findInstructor($instructors)`. Pokud je daný instruktor přiřazen první kupině, je mu přidělena *penalizace*. Následně dojde k výběru druhého instruktora který je přiřazen druhé skupině. A opět je mu přidělena *penalizace*. Tato penalizace zajišťuje, že daný instruktor nemůže být vybrán dvakrát po sobě. Simuluje to pauzu, kdy má instruktor den na odpočinek po náročném dnu. Aby byl výběr instruktorů opravdu náhodný, je jejich pořadí předem proházeno.

V dalším kroku vybereme pro jednotlivé skupiny letadlo. U letadel máme pouze jednu specifikovanou podmínku omezující jeho výběr. Ta spočívá v tom, že letadlo může být vybráno pouze v případě, není-li již přiděleno jiné skupině. To je ošetřeno ve funkci *borrowAircraft()*, která vrací *TRUE* v případě, že si dané letadlo můžeme vypůjčit. Pokud je letadlo přiděleno, je mu nastaven příznak *inUse* a při dalším výběru již toto letadlo nebude k dispozici.

Výběr volného letadla se odvíjí od proměnné *inUse*. Pokud je tato proměnná nastavena, vrací funkce *borrowAircraft() FALSE* a naopak. Aby byl výběr letadel náhodný, jsou předem proházena.

Nyní instruktorům pro daný den vybereme žáky. Zde ale můžeme narazit na více konfliktů:

- žáci nemohou létat dva dny po sobě
- v jednom dni mohou být žáci přiděleni pouze do jedné skupiny
- při výběru mají přednost žáci s menším počtem odlétaných hodin
- pokud mají žáci stejný počet hodin, vybere se náhodně jeden z nich
- žáci si mohou u jednotlivých instruktrů specifikovat, nakolik chtějí být zařazeni do jeho skupiny. Tento výběr se ale zohledňuje pouze, pokud mají nálétáno dostatek hodin. Dojde-li k situaci, že má žák nalétáno málo hodin, je přednostně přidělen do skupiny bez ohledu na to, u kterého instruktora by chtěl být.

Výběr toho nejvýhodnějšího žáka obstarává funkce `findStudent($students,$instId)`. Obdobně jako u instruktorů proházíme pořadí žáků, aby byl výběr opravdu náhodný. Ve funkci i kontrolujeme, mají-li žáci odlétaný požadovaný počet hodin. Pokud dosáhnou maximálního počtu hodin, nejsou již do výběru přiděleni.

Přednostně jsou žáci vybíráni k instruktorovi, kterému mají přidělené největší sympatie. Ale to pouze v připadě, mají-li nalétaný dostatečný počet hodin. Pokud ne, jsou přiděleni k prvnímu volnému instruktorovi, aby svou časovou ztrátu mohli co nejdříve dohnat.

6.2.2 Cost funkce

A nyní se dostáváme k nejdůležitější funkci, která obstarává ohodnocení, tzv. *cenu* žáka, který je přidělen danému instruktorovi.

Celková cena se odvíjí od veškerých hodnot, které si žák během své cesty všemi požadovanými funkcemi nasbíral. Zohledňuje se zde počet odlétaných hodin, penalizace a hodnota preferovanosti k instruktorovi.

Tato funkce je volána při výběru instruktorů a žáků a vždy je vybrán ten, který je pro danou situaci nejvýhodnější.

```
function calcCost($instId){
 //vypocitame "cenu" soucastneho studenta
  $cost = $this->odlHodin + $this->dpenal;
 //pridame bonus za instruktora
  if($instId == $this->pref){
    $cost = $cost - $this->instPen;
  }
 }
 return $cost;
  }
```

6.2.3 Vygenerování dalšího dne kurzu

Vše, co bylo dosud popsáno, obstarává vygenerování jednoho dne pro dvě skupiny. Aby však byl rozvrh kompletní, musíme jej vygenerovat na tak dlouho, dokud všichni žáci nenalétají požadovaný počet hodin.

Proto je nutné vygenerovat další den a to pomocí funkce `dalsiDen()`. V této funkci dochází k počítání dnů kurzu, k uvolnění letadel a zrušení penalizací. Vzhledem k tomu, že `calcCost($instId)` funkce zajišťuje výběr nejvýhodnějšího člena, víme, že žáci letící jako první, bubou mít větší cenu, než ti, kteří poletí jako druzí. Proto budou vybráni přednostně ti s výhodnější cenou.

6.2.4 Testování

Samotná implementace by se neobešla bez průběžného testování správnosti výstupu. A to v každé části od načítání informací, přes správné přiřazení až po vyhodnocení výsledků.

Postupně jsem kontrolovala správné přidělování penalizace, ohodnocení žáků a jejich vhodné přidělení do skupin. Také při rozdělení instruktorů a letadel bylo nutné dbát na dodržení všech pravidel. Z výsledného rozvrhu však některé aspekty patrny nejsou, proto jsem používala kontrolní výpisy, ve kterých byly všechny jednotlivé prvky ovlivňující výběr zobrazeny i s hodnotami a jmény jednotlivých členů a letadel pro lepší přehlednost.

Ukázka kontrolního výstupu:

```
DEN 1 skupina 1:
Letadlo:    L13 Blaník
Instruktor: Tomáš Vondráček(id:25)
Jméno: Antonín Horáček   hodiny: 0.5  pref.instr:25
Jméno: Šáša Šamšula      hodiny: 0.5  pref.instr:25
Jméno: Karel Bohac       hodiny: 0.5  pref.instr:25
Jméno: Dominik Boháček   hodiny: 0.5  pref.instr:25
Jméno: Adam Straka       hodiny: 0.5  pref.instr:25

DEN 1 skupina 2:
Letadlo:    L23 Super Blaník
Instruktor: Tonda Kouldelka(id:22)
Jméno: Daniel Horáček    hodiny: 0.5  pref.instr:22
Jméno: Bedřich Novák     hodiny: 0.5  pref.instr:22
```

```
Jméno: Alexander Třetí   hodiny: 0.5  pref.instr:22
Jméno: Jan Hora          hodiny: 0.5  pref.instr:22
Jméno: Karolína Světlá   hodiny: 0.5  pref.instr:22

DEN 2 skupina 1:
Letadlo:    L23 Super Blaník
Instruktor: Petr Zajíček(id:23)
Jméno: Adam Křepelka     hodiny: 0.5  pref.instr:23
Jméno: Anna Veselá       hodiny: 0.5  pref.instr:23
Jméno: Adam Sýkora       hodiny: 0.5  pref.instr:23
Jméno: Jarmila Boháčová  hodiny: 0.5  pref.instr:24
Jméno: Adam Kos          hodiny: 0.5  pref.instr:22
Jméno: Johanka Orleánská hodiny: 0.5  pref.instr:25
Jméno: Jan Horák         hodiny: 0.5  pref.instr:24
Jméno: Jan Žižla         hodiny: 0.5  pref.instr:25

DEN 2 skupina 2:
Letadlo:    L13AC Blaník
Instruktor: Daniela Jakubková(id:24)
Jméno: Daniel Hora       hodiny: 0.5  pref.instr:24
Jméno: Honza Hus         hodiny: 0.5  pref.instr:24
Jméno: Antonín Horáček   hodiny: 1    pref.instr:25
Jméno: Karel Bohac       hodiny: 1    pref.instr:25
Jméno: Dominik Boháček   hodiny: 1    pref.instr:25
Jméno: Daniel Horáček    hodiny: 1    pref.instr:22
Jméno: Alexander Třetí   hodiny: 1    pref.instr:22
Jméno: Adam Straka       hodiny: 1    pref.instr:25
```

Zde je názorně vidět, že na začátku mají žáci instruktora, kterého si preferovali a postupem času je jejich výběr patrný čím dál méně. To je způsobeno všemi nastavenými podmínkami.

6.2.5 Zhodnocení

Pomocí dané metody z oblasti umělé inteligence jsem dospěla k celkem dobrým výsledkům. Vzhledem k použité náhodnosti při výběru jednotlivých účastníků kurzu je možné generovat rozvrh několikrát, pokaždé s odlišným výsledkem.

Vygenerovaný rozvrh je uložen do databáze v připravené tabulce a je možné si ho zobrazit v sekci náhledu rozvrhu 1. Pokud budeme rozvrh generovat několikrát, nebude se do databáze ukládat vícekrát než jednou. A to tak, že při novém generování se smaže obsah tabulky pomocí `$db->query('','TRUNCATE TABLE rozvrh')` a následně se uloží nově vygenerovaný rozvrh. Toto řešení se mi zdá v celku optimální a ukládání více rozvrhů mi připadá neefektivní.

Generování rozvrhu obstarávají členové aeroklubu ve funkci instruktorů. Rozvrh je vygenerován na začátku letecké sezóny a měl by sloužit po celou dobu trvání kurzu.

Kapitola 7

Možné rozšíření a správa systému

Vzhledem k rychlému rozvoji informačních technologií je nutné počítat i s rozvojem informačních systémů. Proto je důležité už nyní vytvářet flexibilní a dynamicky rozšiřitelné aplikace.

Měli bychom dopředu počítat s tím, že systém se bude používat třeba i několik let, během kterých může dojít ke změnám. A našim úkolem je se na tyto změny připravit už nyní.

Například tím, že systém přizpůsobíme krom požadavků zákazníka i na možná vylepšení, o kterých víme už při implementaci. Zvlášte v případě, poskytneme-li zákazníkovi správu systému i do budoucna. Výhoda dlouhodobé spolupráce je na první pohled zřejmá. Udržíme si zákazníka a především pracujeme na něčem, co jsme vytvořili. A zákazník má jistotu, že pokud nebude něco fungovat, může se na nás kdykoli obrátit.

Což může na druhou stranu znít i jako nevýhoda. Ale pokud svou práci děláme pořádně a zodpovědně, nemáme se čeho bát.

Možných rozšíření informačního systému pro sportovní letiště v Brně-Medlánkách není málo. Zejména proto, že různé směrnice se neustále mění a lidí zajímajících se o létaní přibývá.

Nesmíme také zapomenout, že se pohybujeme v oblasti leteckého provozu, a to sebou nese nemalá rizika. Především v tom, že systém by měl vyhovovat všem existujícím normám a specifikacím. To je ale také jeden z velkých problémů, se kterým se můžeme setkat. Zejména v tom, že aby systém mohl opravdu fungovat, musíme čas od času jistá pravidla obejít. Dokonce jsem se setkala i s tím, že různí lidé si pravidla vykládájí poněkud jinak. A nám nemusí být přesně jasné, kterou z variant zvolit. V takových případech se většinou přikláníme k již zaběhnutým a osvědčeným způsobům.

Také možnosti designu jsou skoro neomezené. A každému se líbí něco jiného. Přesto není na škodu čas od času udělat pár změn, které vedou k inovaci webového rozhraní, či dokonce k usnadnění práce v systému.

Důležitou a mnohdy opomíjenou oblastí je bezpečnost. Někdy se setkáváme se systémy, které se od doby svého vzniku nerozvíjejí. To však vede k jejich postupnému úpadku. Zvláště dnes je důležité se na ochranu dat dostatečně zaměřit. Pravidelné kontroly sytému by měly být naplánovány už při dokončení systému. Abychom byli schopni možné útoky odhalit ještě před tím, než způsobí velké škody.

Kapitola 8

Závěr

S informačními systémy se setkáváme skoro každý den a většina z nás to bere jako normální součást našeho života. Přesto však nemáme potřebu jim porozumět a vystačíme si s pocitem, že vše, funguje jak má.

Ale to ne vždy bývá pravda. K chybám v informačních systémech dochází a můžeme se s nimi potkat kdekoli. Přesto však musíme rozlišit jejich závažnost.

Setkáme-li se při placení nákupu s neznámou položkou na účtence, může se jednat o chybně zpracovaný kód. Většinou si toho ani nevšimneme, pokud se nejedná o velký cenový rozdíl daných položek. Ale pokud nám někdo převede z účtu veškerý zůstatek na neznámý účet někde na Bahamách, všimneme si toho poměrně rychle.

Proto je také důležité při samotném návrhu počítat s možnými riziky. Většinou se problémy ohledně bezpečnosti během implementace rozrostou. Přesto je však dobrým návrhem můžeme omezit.

Zejména pracujeme-li s důvěrnými a osobními údaji, zdravotní dokumentací či finančními zdroji. S rozvojem technologií vývoje informačních systémů roste i riziko prolomení bezpečnostní ochrany.

Ale navrhnout a implementovat bezpečný systém ještě neznamená, že nemůže dojít k jeho zneužití. Jedním z ohrožujích faktorů jsou samotní uživatelé. A to především díky svému nezodpovědnému chování a důvěřivosti. Zvláště přihlašovací údaje by neměly být dostupné neoprávněným osobám. Není vyjimkou, kdy si uživatelé tyto údaje sdělují, nebo je nechávají na veřejně přístupných místech.

Ani volba složitého hesla, či jeho časté změny obvykle nevede k očekávaným výsledkům. Někteří uživatelé si je nedokáží zapamatovat, a proto si je zapisují a ukládají na jim dobře přístupná místa.

Jedním z možných řešení jsou školení, na kterém se uživatelé mohou o bezpečnostních rizikách dozvědět. Přesto však záleží na povaze jednotlivých uživatelů.

Důležitou vlastností systému je i jeho vzhled. Ten bývá obvykle prací grafiků, kteří by měli dodržovat určitá pravidla. Jako je přehledné rozmístění jednotlivých prvků, vhodně zvolené barvy a celkově, uživatelsky příjemné prostředí. I přesto je tato práce velmi kreativní a hodně záleží na vkusu uživatelů.

Téma mé práce jsem si zvolila převážně proto, že oblast létaní je mi blízká a vytváření webových aplikací mě zaujala. Zejména poznávání nových jazyků a technologií. Přesto jsem však až během programování samotného systému narazila na mnohé problémy, se kterými jsem nepočítala.

S jazykem PHP jsem se již setkala, ale neuvědomila jsem si, jaké jsou jeho možnosti a osvojování nových postupů mi zabralo spoustu času. Zvláště práce se smarty pro mě byla

úplně novou záležitostí.

Během implementace jsem musela udělat několik zásahů do databáze. A to především pro větší přehlednost, efektivnost a možnost dalšího rozšíření celého systému.

Přes všechny úskalí mě práce bavila, a do budoucna mi udělila mnohá ponaučení. Proto doufám, že toto není moje poslední práce v této oblasti.

Na závěr bych chtěla upozornit, že data v databázi jsou testovací a případná shoda s existujícími údaji je náhodná. Pouze názvy některých typů letadel a kvalifikací byly získány ze stránek aeroklubu [2].

Literatura

[1] Stehen Spaingour a Robert Eckstein. *Webmaster v kostce.* Computer press, 1999. ISBN 80-7226-450-8.

[2] Stránky aeroklubu Medlánky. Výcvik pilotů. `http://www.akmedlanky.cz/sluzby/vycvik-pilotu/`.

[3] Derick Rethans Andi Gutmans, Stig Saether Bakken. *Mistrovství v PHP5.* Computer press, 2005. ISBN 80-251-0799-X.

[4] Andy Budd. *CSS filtry, hacky a pokročilé postupy.* Zoner press, 2005. ISBN 978-80-86815-54-1.

[5] Clint Eccher. *Webdesign techniky a vzorová řešení.* Computer press, 2005. ISBN 80-251-0547-4.

[6] W. Jason Gilmore. *Velká kniha PHP 5 MySQL.* Zoner press, 2004. ISBN 80-86815-20-X.

[7] Mike Sheman Joel Scambray. *Hacking bez tajemství: webové aplikace.* Computer Press, 2003. ISBN: 80-7226-769-8.

[8] Andrew B. King. *Zrychlete své webové stránky.* Zoner press, 2004. ISBN 80-86815-02-1.

[9] Eric Meyer. *Eric Meyer o CSS profesionálně.* Zoner press, 2005. ISBN 80-86815-17-X.

[10] Lukáš Žitník Miroslav Kučera. *Inspirativní webdesign.* KNIHY.iDNES.cz, 2005. ISBN 80-86593-42-8.

[11] George Schlossnagle. *Pokročilé programování v PHP 5.* Zoner press, 2004. ISBN 80-6815-14-5.

[12] Toby Segaran. *Programming Collective Intelligence.* Computer press, August 2007. ISBN 10: 0-596-52932-5.

[13] David Sklar. *PHP moduly, rozšíření, akcelerátory.* Zoner press, 2005. ISBN 80-86815-19-6.

[14] Petr Tiňo Vladimír Kvasnička, Jiří Pospíchal. *Evolučné algoritmy.* STU v Bratislavě, 2000. ISBN 80-227-1377-5.

Seznam příloh

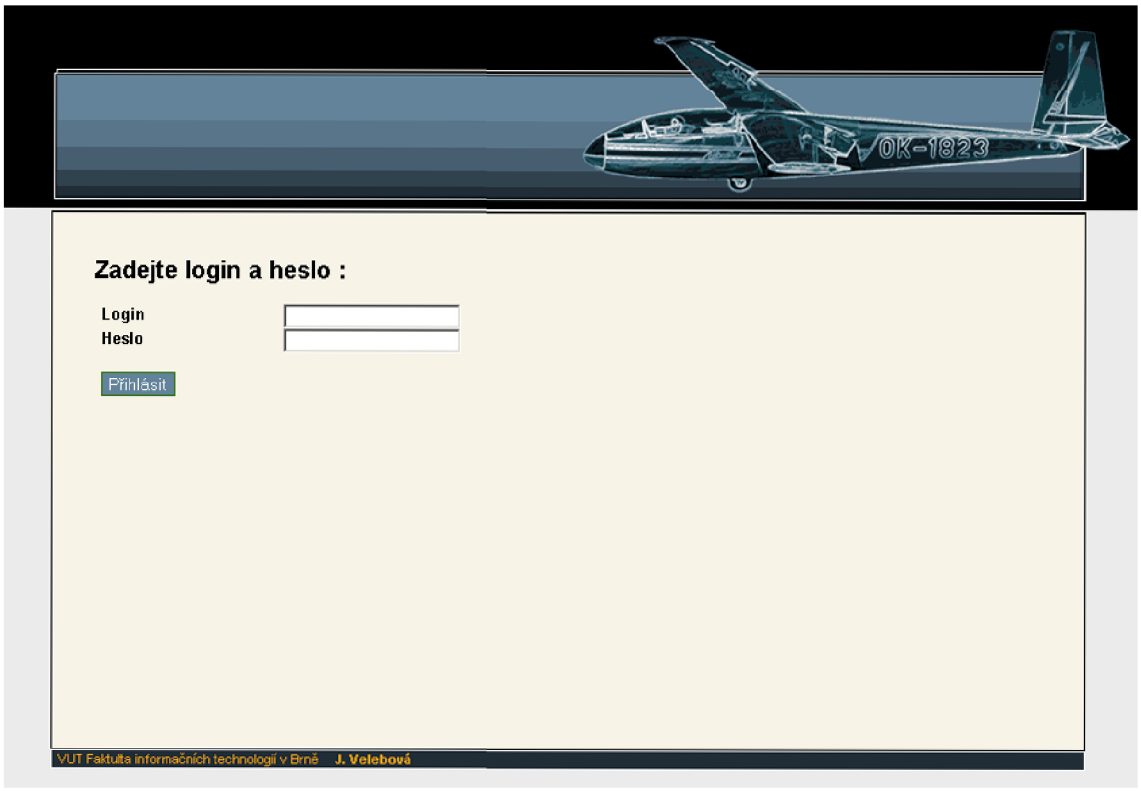
Náhledy informačního systému

Vygenerovaný rozvrh

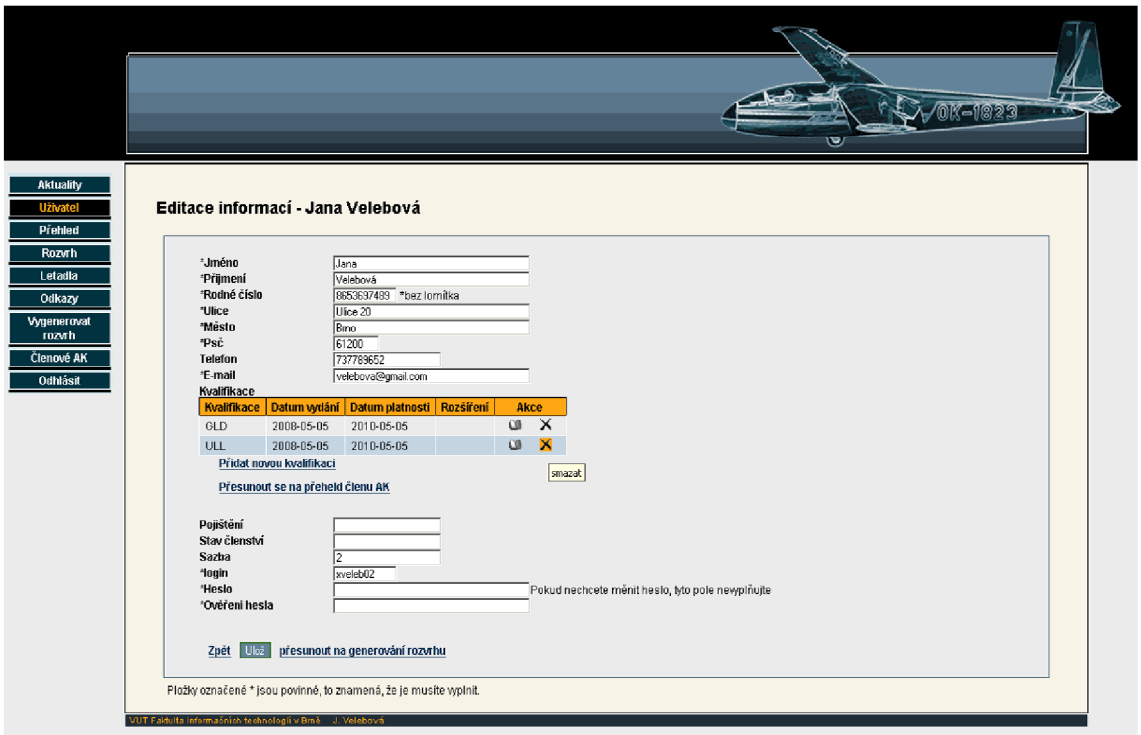
Vygenerovat další rozvrh Generovat další

Den	Skupina	Žáci	Instruktor	Letadlo
1	První	Šáša Šamšula,Adam Kos,Karel Bohac,Jarmila Boháčová,Adam Sýkora,Daniel Hora,Karolína Světlá,Jan Hora	Tonda Kouldelka	L12 Blaník
	Druhá	Antonín Horáček,Daniel Horáček,Bedřich Novák,Jan Žižla,Johanka Orleánská,Adam Straka,Jan Horák	Petr Zajíček	L23 Super Blaník
2	První	Anna Veselá,Honza Hus,Alexander Třetí,Lojza Zavorálek,Adam Křepelka	Tomáš Vondráček	L12Se Blaník
	Druhá	Dominik Boháček,Adam Sýkora,Adam Kos,Jan Žižla,Jan Hora,Johanka Orleánská,Karolína Světlá,Antonín Horáček	Daniela Jakubková	L12 Blaník
3	První	Karel Bohac,Šáša Šamšula,Daniel Hora,Adam Straka,Bedřich Novák,Jan Horák,Jarmila Boháčová	Tonda Kouldelka	L12Se Blaník
	Druhá	Daniel Horáček,Honza Hus,Adam Křepelka,Dominik Boháček,Anna Veselá,Alexander Třetí	Petr Zajíček	L13 Blaník
4	První	Lojza Zavorálek,Jan Hora,Johanka Orleánská,Antonín Horáček,Jan Žižla,Adam Sýkora,Adam Kos,Karolína Světlá	Tomáš Vondráček	L12 Blaník
	Druhá	Bedřich Novák,Jan Horák,Daniel Hora,Karel Bohac,Jarmila Boháčová,Daniel Horáček,Adam Straka,Šáša Šamšula	Daniela Jakubková	L13 Blaník
5	První	Honza Hus,Anna Veselá,Dominik Boháček,Adam Křepelka,Alexander Třetí	Petr Zajíček	L12Se Blaník
	Druhá	Lojza Zavorálek,Adam Kos,Jan Hora,Johanka Orleánská,Karolína Světlá,Antonín Horáček	Tonda Kouldelka	L23 Super Blaník
6	První	Jarmila Boháčová,Jan Žižla,Daniel Horáček,Adam Sýkora,Jan Horák,Adam Straka	Daniela Jakubková	L23 Super Blaník
	Druhá	Šáša Šamšula,Bedřich Novák,Karel Bohac,Daniel Hora,Dominik Boháček,Adam Křepelka,Anna Veselá	Tomáš Vondráček	L12 Blaník
7	První	Honza Hus,Alexander Třetí,Lojza Zavorálek,Johanka Orleánská,Jan Hora,Karolína Světlá,Antonín Horáček,Adam Kos	Tonda Kouldelka	L12Se Blaník
	Druhá	Jan Žižla,Jarmila Boháčová,Adam Straka,Karel Bohac,Daniel Hora,Bedřich Novák	Petr Zajíček	L13 Blaník
8	První	Adam Křepelka,Anna Veselá,Adam Sýkora,Jan Horák,Dominik Boháček,Šáša Šamšula,Daniel Horáček	Tomáš Vondráček	L12Se Blaník
	Druhá	Honza Hus,Lojza Zavorálek,Alexander Třetí,Adam Kos,Antonín Horáček	Daniela Jakubková	L23 Super Blaník
9	První	Daniel Hora,Karolína Světlá,Adam Straka,Bedřich Novák,Karel Bohac,Jan Žižla,Jarmila Boháčová	Tonda Kouldelka	L13 Blaník
	Druhá	Johanka Orleánská,Jan Hora,Adam Sýkora,Anna Veselá,Adam Křepelka,Šáša Šamšula,Jan Horák,Daniel Horáček	Petr Zajíček	L23 Super Blaník
10	První	Honza Hus,Dominik Boháček,Lojza Zavorálek,Alexander Třetí,Antonín Horáček	Daniela Jakubková	L12 Blaník

Obrázek 1: Ukázka vygenerovaného rozvrhu



Obrázek 2: Ukázka vzhledu informačního systému



Obrázek 3: Ukázka vzhledu informačního systému