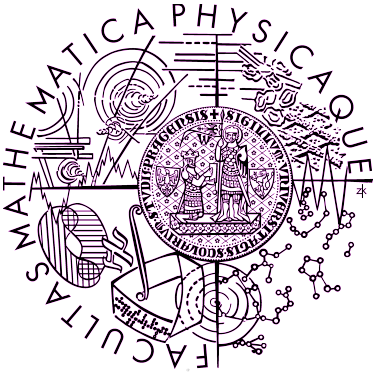
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Vojtěch Mencl

Internetová aukce

Ústav formální a aplikované lingvistiky

Vedoucí bakalářské práce: Mgr. Jiří Kocanda

Studijní program: Programování

2006

Poděkování

Na tomto místě bych chtěl poděkovat svému vedoucímu bakalářské práce panu Mgr. Jiřímu Kocandovi za podnětné připomínky a vedení, a to zejména při analýze a návrhu aplikace.

Prohlášení

Prohlašuji, že jsem svou bakalářskou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce a jejím zveřejňováním.

V Praze dne 9/8/2006 Vojtěch Mencl

Obsah

Název práce: Internetová aukce
Autor: Vojtěch Mencl
Katedra (ústav): Ústav formální a aplikované lingvistiky
Vedoucí diplomové práce: Mgr. Jiří Kocanda
E-mail vedoucího: kocanda@atlas.cz

Abstrakt: Obchodování pomocí internetu je moderní způsob, jak nabízet své výrobky nebo služby. Mezi zajímavé způsoby obchodování na internetu patří elektronická aukce, jejíž analýza, návrh a implementace je cílem této práce. Inaukce je webová aplikace zprostředkovávající aukční nebo bazarový prodej předmětů prostřednictvím internetu. Uživatelé přistupují k aplikaci pomocí webového prohlížeče v různých rolích, které se liší uživatelskými právy. Po přihlášení do systému mají uživatelé s příslušnými právy možnost předměty nabízet či kupovat. K dispozici je také role administrátora s možností správy účtů a předmětů, administrace a monitoringu systému.

Klíčová slova: aukce, Internet, bazar, prodej, online

Title: Internet auction
Author: Vojtěch Mencl
Department: Institute of Formal and Applied Linguistics
Supervisor: Mgr. Jiří Kocanda
Supervisor's e-mail address: kocanda@atlas.cz

Abstract: Trading on the Internet is a modern way of offering products and services. Electronic auction belongs among interesting ways of trading on the Internet. The aim of this work is to analyze, design and implement e-commerce. Inaukce is a web application which convey auction or bazaar sale of various items on the Internet. Users access the application in different roles using web browser. These roles differ in users rights. After being logged on, users with appropriate rights can offer or buy items. In the administrative role, there is a option to manage user accounts and traded items. System monitoring is also available.

Keywords: auction, Internet, bazaar, sale, online

Související dokumenty

1) Funkční specifikace – soubor FunkcniSpecifikace.pdf
2) Uživatelská příručka – v souboru Prilohy.pdf jako Priloha A
3) Administrátorská příručka – v souboru Prilohy.pdf jako Priloha B
4) Instalační příručka – v souboru Prilohy.pdf jako Priloha C
5) Programátorská dokumentace – v souboru Prilohy.pdf jako Priloha D
6) Technická dokumentace – v HTML podobě v adresáři /Doc/TechDoc/ spustitelná souborem index.html

Kapitola 1
Úvod a cíl práce

1.1 Úvod do problematiky

Zatímco aukce jako forma prodeje je stará jako lidstvo samo (vychází prapůvodně ze smlouvání při obchodování), elektronická aukce je nový fenomén, který má před sebou velmi dynamický vývoj a velkou budoucnost.

Internetová aukce je trhem, kde se proti sobě střetává nabídka s poptávkou. Jako na každém trhu se kupující snaží získat věc koupě za co možná nejnižší cenu, naopak nabízející prodávající se snaží prodat za nejvyšší cenu. Zdá se, že oba mají jiný záměr a nelze tedy dojít k uspokojivému řešení. Jenže oba mají i jeden společný cíl, a to uskutečnit obchod prodej. Dochází tedy ke smlouvání, speciálně v aukci, k přihazování. Přihazování značí zvyšování ceny, za kterou je nakupující danou věc ochoten koupit. Může docházet i k tomu že proti sobě o danou položku soutěží více poptávajících, čímž dochází brzy k rychlému růstu ceny z ceny tzv. vyvolávací (původní cena, stanovena buď samotným majitelem nabízené věci, zástupcem aukčního domu či soudním znalcem) na cenu prodeje. Je to agresivní druh prodeje, kdy se jeden kupující snaží odstranit z obchodu pomocí vyšší nabízené ceny svého soka. Musí však dojít k tomu, že cena nabízená kupujícím je akceptována prodávajícím a naopak, jinak k obchodu nedojde.

Z hlediska právního však zákon č. 26/2000 Sb. na internetovou aukci nepamatuje. V současné době je snahou České asociace dražebníků tento zákon novelizovat a funkci internetových aukcí uzákonit. Internetová dražba by měla být jako celek chápána v rámci již zákonem stanovených pojmů podle [11] takto:

„*Jednání probíhající prostřednictvím veřejně přístupného informačního systému (Internet), jehož účelem je přechod vlastnického nebo jiného práva k předmětu dražby konané na návrh navrhovatele, při němž se licitátor obrací na předem neurčený okruh osob s výzvou k podávání nabídek a při němž na osobu, která za stanovených podmínek učiní nejvyšší nabídku, přejde příklepem licitátora vlastnictví nebo jiné právo k předmětu dražby, nebo totéž jednání, které bylo licitátorem ukončeno z důvodu, že nebylo učiněno ani nejnižší podání*“.

1.2 Cíl práce

Cílem této bakalářské práce je vytvořit internetovou aplikaci zprostředkovávající pohodlný nákup a prodej předmětů prostřednictvím aukce. Maximální důraz je kladen na uživatelskou přívětivost. Vytvořená aplikace by měla umožňovat co nejjednodušší zadání předmětů do prodeje, správu a především nákup předmětů. Internetová aukce by měla mít standardní vzhled a ovládání a měla by uživatele co nejméně vyrušovat od samotného nákupu či prodeje.

Internetová aukce by měla simulovat aukce jejichž organizátory jsou aukční domy. Stejně tak jako aukční domy, tak i Inaukce by měla poskytovat bezpečný prodej a nákup předmětů. Je nutné, aby účastnící aukce byli registrováni. Tím se zajistí možnost prověření identity a bezpečnost obchodů. Další formou zabezpečení by měla být šifrovaná komunikace mezi serverem a klienty účastníků aukce. Aplikace by měla umožňovat snadné a přehledné procházení jednotlivých aukcí a vyhledávání konkrétních nabízených položek podle různých atributů. Aplikace umožní kromě aukčního prodeje také standardní bazarový prodej.

Při vývoji aplikace by měl být kladen důraz na možnost případných budoucích změn v aplikaci. Jakékoliv změny by měly být možné provést velmi jednoduše a s co nejmenšími vedlejšími efekty.

Pro úspěšné vytvoření aplikace je nutné dodržet několik zásad pro její vývoj. Konečnému vývoji aplikace musí předcházet funkční analýza a vytvoření Funkční specifikace. Na základě Funkční specifikace je nutno provést návrh softwarového řešení. Tento návrh by měl být pak sepsán ve formě Technické specifikace. Podle Technické specifikace se sestaví harmonogram práce a implementace s odhady časových náročností jednotlivých úkolů. Vývoj by měl probíhat podle tohoto harmonogramu. Na závěr je nutno provést uživatelské testy vytvořené aplikace a vytvořit dokumentace aplikace důležité pro další úpravy a rozvoj aplikace.

1.3 Motivace

V současnosti se objevuje na českém trhu velké množství nových internetových obchodů, z nichž pouze malá část využívá myšlenky využít pro nacházení ideální ceny aukční prodej. V zahraničí je obchodování na internetových aukcích jeden z nejoblíbenějších a nejefektivnějších způsobů nákupu a prodeje veškerého druhu zboží, produktů a služeb. Nesporná výhoda tohoto obchodování spočívá v ušetřeném čase,

který by jinak byl nutný při aukcích v aukčních domech. Další výhodou je značná cenová výhodnost nákupu. Bude-li chtít uživatel internetu aukci vyzkoušet, stačí vyplnit registrační formulář na internetových stránkách dražebníka a pak už jen přihazovat k ceně jím zvoleného předmětu.

Kromě samotného nákupu či prodeje aplikace většinou umožňují snadnou změnu jednotlivých stránek aplikace, která může být využita pro reklamní účely apod.

1.4 Jak číst tuto práci

Dokument Bakalářská práce popisuje a zdůvodňuje nejdůležitější rozhodnutí a kroky vykonané při analýze a vývoji aplikace a částečně popisuje zvolené postupy a jejich zástupné možnosti. Výklad v Bakalářské práci neseznamuje čtenáře učebnicovým stylem s výsledným softwarovým dílem. Tuto funkci plní Uživatelská a Administrátorská příručka společně s Instalační příručkou. Pro porozumění dalším částem Bakalářské práce je vhodné se nejdřív seznámit s vytvořenou aplikací prostřednictvím těchto dokumentů a pak pokračovat ve čtení následujících kapitol. Z programátorského hlediska je aplikace popsána do detailů v Programátorské dokumentaci.

1.5 Přehled kapitol bakalářské práce

Druhá kapitola se zabývá analýzou a návrhem aplikace. V kapitole je popsaná tvorba Funkční specifikace a všechny kroky a rozhodnutí, která byla vykonaná před započetím implementace.

Třetí kapitola popisuje implementační fázi vývoje aplikace. Některé části kapitoly volně navazují na příslušné části druhé kapitoly a doplňují je o implementační detaily, které nebyly v době analýzy a návrhu aplikace ještě známé.

Ve čtvrté kapitole jsou stručně popsané jednotlivé dokumenty, které jsou nedílnou součástí softwarového díla. Důraz je kladen na formální úpravu dokumentů a jejich funkci.

Závěrečná kapitola hodnotí splnění cíle bakalářské práce. Jsou zde shrnuty praktické zkušenosti z testovacího používání vytvořené aplikace.

Kapitola 2
Analýza a návrh

2.1 Funkční specifikace

Na základě komunikace se zadavatelem práce byl vypracován návrh aplikace, podle kterého byla vytvořena Funkční specifikace. Tento dokument zachycuje požadavky zadavatele specifikované v průběhu analýzy a návrhu aplikace. Na základě Funkční specifikace by měla vzniknout Technická specifikace[1] a následně samotné softwarové dílo. V závěrečných fázích vývoje aplikace sloužila Funkční specifikace k ověření splnění požadavků zadavatele.

2.2 Dostupný software

V současnosti je na českém Internetovém trhu mnoho internetových aukcí, které mají více či méně shodné funkce. Mezi největší a nejoblíbenější Internetové aukce patří servery aukce.cz, aukro.cz, aukce.atlas.cz, i-aukce.cz či odklepnuto.cz. Všechny tyto aukce používají software, který je vytvořen na zakázku a není dále šířen. Software se liší prakticky jen uživatelským rozhraním, funkce všech těchto aplikací jsou stejné. Při analýze se z tohoto faktu vycházelo. Samotný software internetové aukce je však na českém trhu nabízen jen sporadicky. Mezi dostupné aplikace patří např. reverzní aukce od spol. Bluepixel (cena od 30 000Kč) či PROe.biz.

2.3 Definování atributů kvality

Nejdůležitějším bodem při počátečním návrhu aplikace bylo definování atributů kvality, z nichž se posléze vytvářela Funkční specifikace a měly nemalý vliv i na samotnou implementaci aplikace. Některé atributy byly vymezeny zadáním bakalářské práce, zbylé se musely vytvořit s ohledem na požadavky zadavatele. Výsledné atributy kvality:

- Správa uživatelských účtů a přístupových práv.
- Zabezpečení přístupu do systému pomocí uživatelských jmen a hesel.
- Možnost nastavení registrace uživatelů.
- Možnost nastavení způsobu prodeje prodávajícími uživateli.
- Zprostředkování prodeje předmětů formou bazaru nebo aukce.

[1]Funkční specifikace obsahuje velkou část z Technické specifikace, proto nebyl tento dokument yytvořen

- Snadný a bezpečný přístup administrátorů k datům uživatelů a jejich snadná změna.
- Zabezpečená komunikace pomocí protokolu SSL.
- Vzdálený přístup uživatelů do systému pomocí webového prohlížeče.
- Možnost nahrávat obrázky a dokumenty k jednotlivým předmětům
- Snadné prohlížení a vyhledávání prodávaných předmětů.
- Začlenění předmětů do kategorií
- Jednotný vzhled webových stránek aplikace

2.4 Architektura systému

Ze zadání práce nepřímo vyplývá použitá architektura systému. Bylo zvoleno moderní pojetí internetových aplikací a tedy byl vybrán model třívrstvé architektury jakožto pokračovatele, z dnešního pohledu již zastaralé, dvouvrstvé architektury (čerpáno z [16]).

Model třívrstvé architektury rozlišuje tyto vrstvy:

1. Prezentační vrstva – obsahuje funkce uživatelského rozhraní. Obvykle existuje několik prezentačních vrstev pro různé druhy zařízení, platformy a prostředí
2. Aplikační vrstva – tvoří prostředníka mezi vrstvou prezentační a vrstvou datovou. Obsahuje tzv. business logiku aplikace. V této vrstvě dochází k transformací dat mezi vstupně / výstupními požadavky a datovou vrstvou.
3. Datová vrstva – obsahuje funkce pro přístup k informacím v datovém úložišti

Třívrstvá architektura nabízí řadu nesporných výhod, jako například snadnou změnu poskytovatele datového úložiště či zjednodušení přístupu k datům. Na rozdíl od dvouvrstvé architektury, klientovi není dovoleno přímo komunikovat s datovou vrstvou. Použití třívrstvé architektury ale neznamená, že pro každou vrstvu musí být vyhrazen samostatný počítač. Aplikace Inaukce může být provozována na jednom počítači, což znamenalo obrovskou výhodu při vývoji a testování aplikace.

Prezentační vrstva není jen dominantou klientů, ale může být umístěna také na serveru, jako v případě aplikace Inaukce, kdy se server stará o generování HTML stránek.

Ve složitějších aplikacích je možné definovat i více než tři vrstvy. Většinou je ale od sebe odstíněna prezenční vrstva od vrstvy datové. Příkladem může být např. vrstva pro kontrolu přístupových práv a zabezpečení v aplikaci Inaukce. Na tuto funkčnost lze pohlížet jako na podvrstvu aplikační vrstvy z třívrstvé architektury, ale také jako na samostatnou vrstvy. Záleží pouze na úhlu pohledu.

2.5 Vzdálený přístup uživatelů do systému pomocí webového prohlížeče

Webová aplikace je aplikace poskytovaná uživatelům z webového serveru (aplikační logika) přes počítačovou síť Internet, nebo její vnitropodnikovou obdobu (intranet). Pro přístup k aplikační logice se v Inaukci používá webový prohlížeč. Webový prohlížeč funguje jako tenký klient, což je klient, který neobsahuje žádnou aplikační logiku. Aplikační logika je umístěna na aplikačním serveru, ke kterému tenký klient přistupuje. Samotný klient pak slouží pouze k zobrazování informací a k přenosu uživatelských vstupů zpět na aplikační server. Uživatelé tak k aplikačnímu serveru přistupují z různých míst pomocí unifikovaného rozhraní. Výhody takového řešení:

- Především, uživatelé mohou být zcela mobilní. Nezávisí na stanici, ze které se přihlásí. Vždy dostanou své nastavení, aplikace, data, vše v podobě, v jaké je opustili na jiném zařízení.
- Velice se zjednodušuje údržba aplikací, řízení jejich funkčnosti a v konečném důsledku také dochází ke zvyšování bezpečnosti. Uživatelé do klientů nemohou instalovat vlastní programy, stejně tak nemají přístup ke zdrojovému kódu aplikační logiky.

Nevýhody tohoto řešení:

- Především je to závislost na připojení k síti a komunikace se serverem. Tenký klient je bez přístupu k serveru redukován na minimum základních funkcí. Dále jeho odezva je vinou přenosu dat po síti pomalejší.

Jak bylo zmíněno v předchozí kapitole, aplikace Inaukce je strukturována jako třívrstvá[2]. V této nejběžnější formě je webový prohlížeč první vrstvou (prezentační), aplikační server[3], jehož součástí jsou nástroje pro dynamické generování stránek (v našem případě, PHP), je vrstvou střední (logickou) a databáze je vrstvou třetí (datovou).

[2]Pojednání o počtu vrstev je v kap. 2.4
[3]V tomto případě je webový server zároveň i aplikačním serverem.

Webový prohlížeč posílá požadavky střední vrstvě, která je obsluhuje prostřednictvím dotazů do databáze (resp. její aktualizací) a generováním uživatelského rozhraní. Schopnost aktualizovat a spravovat webové aplikace bez nutnosti instalovat software na potenciálně tisíce uživatelských počítačů je hlavním důvodem zvolení webových prohlížečů pro presentační vrstvu v aplikaci Inaukce.

Webové rozhraní v některých směrech omezuje funkčnost a možnosti klienta. Metody známé z desktopových aplikací, jako je například vykreslování na obrazovku, nejsou standardními technologiemi prohlížečů podporovány. Proto pro přidání funkčnosti je využíváno skriptování na straně klienta (JavaScript), zvláště pro vytvoření dojmu interaktivity bez nutnosti opětovného načtení stránky, které zatěžuje tok v síti a samotný server a zpomaluje práci uživatelů.

2.6 Uživatelské účty, role a přístupová práva

Uživatelským účtům lze přiřadit jednotlivé uživatelské role, které se rozlišují pomocí uživatelských práv. Vlastnictví některého z práv pak znamená možnost vykonávat akce pro danou roli.

Kontrola práv uživatelů přihlašujících se pod svými uživatelskými účty je jedním z atributů bezpečnosti aplikace Inaukce. Před provedením každé akce se nejprve zkontroluje, zda má daný uživatel právo tuto akci provést. Pokud toto právo má akce se provede, v opačném případě je uživatel informován o neúspěchu.

Uživatelé mohou vystupovat celkem v pěti rolích:

- *uživatel anonymní* – vstupuje do systému bez přihlášení. Anonymní uživatel nemusí být tedy registrován v databázi aukčního serveru. Může pouze prohlížet a vyhledávat nabízené předměty. Byla zvažována varianta, kdy anonymní uživatel nebude moci procházet předměty. Potřeba registrace by však odradila potenciální zákazníky, kteří by jako anonymní uživatelé mohli nalézt vhodný předmět a až posléze by se registrovali. Po dohodě se zadavatelem je preferován maximální užitek prodávajících uživatelů.
- *uživatel kupující* – vstupuje do systému po zadání uživatelského jména a hesla. K získání tohoto uživatelského účtu je nutná předchozí registrace. Uživatel kupující je dostatečně identifikován svým uživatelským jménem a e-mailovou adresou. Pro zakoupení předmětu tedy stačí být registrován s platnou e-mailovou adresou a přihlášen do systému pod svým uživatelským jménem.

Tento návrh byl oproti Funkční specifikaci rozšířen o možnost nastavení zobrazení jednotlivých registračních polí a nutnosti pole při registraci vyplnit. Dále pak vzhledem k možnosti nastavení výběrové registrace[4], je na dražebníkovi, jaké registrační údaje bude považovat za jednoznačně identifikující a jak bude docházet k jejich ověření. Uživatel kupující má práva anonymního uživatele rozšířená o možnost nákupu předmětů.

- *uživatel prodávající* – vstupuje do systému po zadání uživatelského jména a hesla. K získání tohoto uživatelského účtu je nutná předchozí registrace. Oproti funkční specifikaci byla v implementaci přidána možnost administrátorského nastavení způsobu udílení prodejních práv:
    a) Uživatel získá prodejní právo ihned po požádání. Tedy nezíská toto právo ihned po registraci, protože bylo marketingovým záměrem identifikovat uživatele, kteří mají zájem jen předměty kupovat a uživatele, kteří chtějí prodávat či kupovat.
    b) Uživatel požádá o prodejní právo a je zařazen do fronty čekatelů na prodejní právo. Dražebník má pak možnost dalšího prověření osobních údajů uživatele, popř. zpoplatnění zisku prodejního práva, ap.

  Prodávající uživatel má stejná práva jako kupující. Navíc pak může zadávat položky do databáze položek, tj. nabízet předměty k prodeji. Ke každé prodejní položce, kterou sám zadává do databáze může přiřadit mnoho podrobností, například foto předmětu či možné vložení případných textových souborů. Prodávající má k disposici přehlednou správu vlastních předmětů, které prodává s možností manipulace s předměty a podrobnostmi.

- *administrátor* - vstupuje do systému po zadání uživatelského jména a hesla. Práva administrátora jsou oproti běžným uživatelským účtům (kupující, prodávající) rozšířena o 3 důležité oblasti práv, jsou to:
    a) *Správa uživatelských účtů* - možnost vytvářet, rušit a měnit u. účty kupujících a prodávajících. Při automatické registraci kupujících je důležitá hlavně schopnost vytváření u. účtů prodávajících..
    b) *Správa předmětů* - možnost přidávat předměty do prodeje, rušit a měnit prodeje předmětů, měnit atributy předmětů.

[4]Lze nastavit nepřímou registraci, kdy se žádosti uživatelů o registraci posílají do fronty čekatelů.

c) *Administrace systému* - možnost rušit, vytvářet, měnit nabídky předmětů. Možnost zobrazovat nastavení systému není však možné toto nastavení měnit.
d) *Monitoring systému* - možnost sledovat prodeje předmětů, statistiky přístupů do systému, statistiky zobrazení jednotlivých prodejů, možnost sledovat pohyb jednotlivých uživatelů v systému.

Zpočátku byla zvažována pouze jedna administrátorská role, která by spojovala současné role administrátora a superusera. Po důkladnější analýze vyvstala potřeba důsledně odlišit dražebníka, jakožto provozovatele internetové aukce s maximální odpovědností a licitátorů, jakožto pověřených osob řídících jednotlivé aukce a řešících problémy uživatelů. Licitátoři nemají právo měnit pravidla aukce nebo výrazněji zasahovat do chodu aukce. Pro ně je určena administrátorská role. Pro dražebníka je určena role superusera.

- *superuser* - vstupuje do systému po zadání uživatelského jména a hesla. Superuser má stejná práva jako administrátor rozšířená o možnost vytvářet administrátorské účty a měnit nastavení systému, například způsob registrace. Tato práva přísluší dražebníkovi, tedy provozovateli internetové aukce.

V implementaci Inaukce byla ponechána možnost volby jednotlivých přístupových práv, které nekorespondují s definovanými uživatelskými rolemi. Například administrátoři mají možnost uživateli udělit právo prodeje, ale ne nákupu, atd. Jsou však upozorněni na nestandardnost takovéhoto zacházení.

2.7 Uživatelská jména, hesla a autentizace

Pro Inaukci je nutné, aby uživatelé byly při některých akcích jednoznačně rozlišováni. Mezi tyto akce patří mimo jiné nákup a prodej předmětů. Bez rozlišení uživatelů by nebylo možné určit, jaký uživatel kupuje či prodává daný předmět a tedy by nebylo možné tyto akce uskutečnit. Zároveň je nutné, aby nedocházelo k záměně identifikace uživatelů, jak náhodné, tak záměrné.

K identifikaci uživatelů slouží systém uživatelských účtů. Každý uživatelský účet je jednoznačně identifikován unikátním uživatelským jménem. Uživatelské jméno si uživatel zvolí při registraci společně se zadáním osobních a dalších údajů. Uživatelské jméno se využívá k určení uživatele při přihlášení do systému. Při

přihlášení do systému musí dojít k autentizaci uživatele, aby nebylo možné se vydávat za někoho jiného.

2.7.1 Uložení přihlašovacích údajů

Aplikace by měla příslušným způsobem uchovávat, evidovat a spravovat přihlašovací, osobní i jiné údaje registrovaných uživatelů. Zejména registrační údaje, jako jsou uživatelská jména a hesla, by měla být na serveru uložena na bezpečném místě, typicky v databázi s omezeným přístupem, což bylo zvoleno jako finální řešení, vzhledem k výhodnosti centrálního uložení dat a poskytované bezpečnosti databáze MySQL.

V průběhu vytváření fyzického datového modelu bylo nutné zvážit uložení hesla v databázi. Hesla (podle [14]) není vhodné ukládat v čisté podobě, ale upravená některou jednocestnou hashovací funkcí, například SHA nebo MD5. Nakonec byla zvolena funkce SHA1. Nebudou tak čitelná pro administrátora, ale ani pro případného narušitele, který by se nějakým způsobem k databázi hesel dostal, což značně omezuje možnosti útočníka, jak původní podobu hesla odhalit. Jedna z možností je útok hrubou silou, tzn. zkoušet stejnou jednocestnou funkcí šifrovat jedno potenciální heslo za druhým a porovnávat je s uloženými záznamy. Proto by měla být zvolená hashovací funkce dostatečně silná, aby byl takový postup v přiměřeném čase výpočetně nedosažitelný. To je důvod, proč byla nakonec zamítnuta funkce MD5[5].

Dále vyvstalo další nebezpečí a to kolize hesel – když dva uživatelé zadají stejné heslo, tak má i stejný hash a když to jeden z uživatelů jakkoliv zjistí, může se přihlásit i na druhého uživatele. Tento problém byl vyřešen pomocí vygenerování náhodného řetězce[6], který se s heslem vhodně smíchá. Metoda není kritická a zřetězení náhodného řetězce s heslem nesnižuje bezpečnost. Přestože toto řešení lehce zvyšuje paměťovou i časovou složitost, jeho výhody jsou nesporné.

2.7.2 Autentizace

Autentizace je proces, při kterém se ověřuje, zda je uživatel nebo entita opravdu ten, za koho se vydává. V zásadě existují tři možné způsoby, jak je možné provádět úvodní uživatelskou autentizaci (podle [12]). Při každé autentizaci se vždy jeden nebo více z nich používá:

- Sdělení tajné informace ve formě klíče či hesla.

[5]Více o volbě hashovacích funkcí v kap. 3.7.2
[6]V angl. terminologii se používá výraz *salt*.

- Předložení daného *tokenu*[7].
- Využití *biometriky*[8] přistupujícího uživatele.

Drtivá většina webových aplikací implementuje pro svou jednoduchost a relativně dostačující bezpečnost první přístup, stejně tak je i v Inaukci, tedy autentizaci zadáním uživatelského jména a hesla. Ostatní dvě metody se používají zejména v systémech náročných na bezpečnost, a to ještě v různých vzájemných kombinacích. Pro internetovou aukci formátu určeným v zadání by byly takovéto způsoby autentizace nevhodné, jelikož by tím odradily většinu potenciálních cílových zákazníků Inaukce. V principu se ale stále jedná o totéž. Ať už zadáním hesla, protažením identifikační karty či sejmutím otisku prstu poskytuje uživatel počítači nějakou informaci, která je typická jen pro něj a kterou je pro kohokoliv jiného nesnadné napodobit.

Je důležité, aby hesla volená uživateli byla nesnadno odhadnutelná. Inaukce by měla mít alespoň základní kontrolu volby hesla při registraci uživatele pro zvýšení bezpečnosti.

Pokročilejší vlastností, která nebyla v Inaukci implementována, je *zamykání účtů* při autentizaci. Když se útočník bude pokoušet přihlásit na jeden účet a několikrát po sobě zadá chybné heslo, účet se zamkne. Opětovné odemčení účtu pak může provést jen administrátor, anebo k němu dojde automaticky, ale až po určité časové prodlevě. Mechanizmus uzamykání chrání uživatelské účty před útokem hrubou silou. Povolený počet pokusů se pohybuje zpravidla mezi 5 až 10, doba uzamčení účtu pak v řádech desítek minut až několika hodin.

2.7.3 Zapomenutí hesla

Pro případ, že *uživatel své heslo zapomene*, poskytuje aplikace funkci automatického obnovení přístupu k účtu, aniž by bylo nutné osobně kontaktovat administrátora. Je nutné si uvědomit, že taková možnost představuje bezpečnostní riziko, protože se tak účet zpřístupňuje uživateli, který není schopen se dostatečně autentizovat. Bylo zvažováno několik možností:

[7]Termín *token* pochází z anglického jazyka a označuje známku, znak pravosti. Do kategorie tokenů spadají třeba veškeré paměťové nebo čipové karty, jimiž se zaměstnanci prokazují při vstupu na pracoviště, mezinárodní studentské karty ISIC atd.

[8]Slovo *biometrie* pochází z řečtiny a označuje měření určitých vlastností člověka. Biometrické systémy pak jsou schopny na základě unikátnosti těchto vlastností identifikovat uživatele. Zmiňovaných vlastností existuje celá řada a lze je rozdělit do dvou základních skupin – fyziologické nebo behaviorální. Mezi reprezentanty první uvedené kategorie patří například otisky prstů a oční duhovka, do behaviorálních charakteristik (týkajících se chování) spadá kupříkladu dynamika podpisu nebo ověřování hlasu mluvčího.

1. *zodpovězení kontrolní otázky*, kterou si uživatel nastavil při registraci. Toto je silně nedoporučovaný přístup, neboť většina běžně používaných kontrolních otázek má nějakou přímou souvislost s daným uživatelem (rodné jméno manželky, číslo řidičského průkazu apod.), kterou si snadno odvodí i případný útočník.
2. *zasílání přístupových údajů e-mailem* na adresu, kterou uživatel určil při původní registraci. V tomto případě také existují značná bezpečnostní rizika. Největší riziko se skrývá v samotné e-mailové službě. Přenos e-mailových zpráv po síti je nešifrovaný, samotné zprávy jsou otevřené a čitelné pro kohokoliv. Pokud aplikace tento způsob obnovy přístupu podporuje, měla by zasílat náhodně vygenerované jednorázové heslo. Po jeho prvním použití by měl být uživatel ihned vyzván ke změně hesla na jinou hodnotu. Tento způsob byl zvolen pro jeho dostatečnou bezpečnost a relativní jednoduchost.

Zvolený způsob zasílání přístupových údajů e-mailem v sobě skrývá chybu, která byla objevena až v průběhu implementace. Pokud zná libovolný uživatel uživatelské jméno[9] dalšího uživatele, může mu použitím této funkčnosti zneplatnit heslo. Po požádání o zneplatnění hesla je uživateli zaslán e-mail s náhodným kódem pro ověření žádosti o zneplatnění. Pokud uživatel skutečně nezná heslo, následuje odkaz který obdržel v této zprávě. V odkazu je použit zmiňovaný náhodný kód, který je na serveru ověřen. Pro vyřešení tohoto problému musela být přidána další MySQL tabulka *usercodes*[10], která se používá pro uchovávání jednoznačného kódu požadavku na získání nového hesla na straně serveru. Pokud je kód v tabulce obsažen, je uživateli zaslán druhý e-mail s novým heslem k jeho účtu.

Možná další implementace: všem heslům v aplikaci je obecně dobré nastavit určitou *omezenou životnost*. Systém pak například každé dva měsíce vyzve uživatele, aby si své aktuální heslo změnil. Po každé editaci registračních údajů by měla aplikace zasílat uživateli na jeho adresu notifikační e-mail o provedených změnách. Toto vylepšení v aplikaci nebylo implementováno z důvodu nedostatku času.

[9]Uživatelská jména jsou veřejná.
[10]Tabulka usercodes není uvažována v logickém a fyzickém datovém modelu Funkční specifikace.

2.8 Registrace uživatelů

Uživatelé, kteří chtějí získat uživatelský účet pro přístup do systému musí správně vyplnit registrační formulář. V administrátorském nastavení lze určit, která pole registračního formuláře budou uživateli zobrazena a která pole bude muset uživatel nutně vyplnit. Tato funkčnost nebyla ve Funkční specifikaci přímo určena, ale je důležitá, vzhledem k rozdílným požadavkům na identifikaci uživatelů jednotlivých dražebníků.

Po vyplnění registračního formuláře uživatelem se provede akce, která záleží na administrátorském nastavení:

1. Uživatel je přímo registrován do systému.
2. Uživateli je zaslán e-mail pro ověření jím zadané e-mailové adresy. Tento způsob je doporučován a je jediným způsobem, jak lze jednoduše ověřit e-mailovou adresu uživatele.

Byly uvažovány i další varianty, které však byly po konzultaci se zadavatelem zavrženy. Mezi nejdiskutovanější patřila možnost ověřit osobní údaje uživatele prostřednictvím poštovní služby. Uživateli by byl zaslán dopis na jím zadanou kontaktní adresu s náhodně vygenerovaným kódem. Uživatel by musel po obdržení dopisu tento kód zadat do daného formuláře v internetové aukci a tím by byl registrován do systému. Tato možnost lze využít dražebníkem pro ověření údajů uživatelů žádajících prodejní právo.

2.9 Formy a uskutečnění prodeje

Ve Funkční specifikaci Inaukce jsou rozlišovány dva typy prodeje předmětů: bazarový a aukční prodej. V původním zadání byl obsažen pouze prodej aukční, po komunikaci se zadavatelem byl přidán do zadání i prodej bazarový vzhledem k jednoduchosti implementace a zvýšení funkčnosti aplikace.

Průběh aukce i bazaru je detailně popsán ve Funkční specifikaci. Obsahuje však sporný bod, který nelze jednoduše na aplikační úrovni implementovat. Tento bod spočívá v povinnosti kupujícího uživatele v případě zakoupení předmětu na aplikační úrovni, tento předmět zakoupit také fakticky[11]. Vzhledem k tomu, že Inaukce je koncipována jen jako zprostředkovatel prodeje předmětů, nelze ani jednu stranu na aplikační úrovni donutit ke koupi či prodeji daného předmětu, tak jak je popsáno v dokumentu Pravidla aukce dle zákona. Pro uplatnění pravidel by bylo nutné ověřit

[11] Jedná se o oba typy prodeje.

totožnost obou účastníků prodeje a následně předat problém do soudního řízení. K tomu by bylo nutné zavést sofistikovanější způsob autentizace, například pomocí biometriky a podobně. Současný návrh Inaukce však takovýto způsob nepodporuje. Je tedy zcela na dražebníkovi, jak tento problém vyřeší. Při komunikaci se zadavatelem jsme došli k závěru, že žádné řešení tohoto problému nebude do Inaukce zahrnuto.

2.10 Vyhledávání předmětů

Vyhledávání předmětů slouží k nalezení podmnožiny předmětů, které splňují určité požadavky uživatele. Inaukce umožňuje dva typy vyhledávání. Je to rychlé vyhledávání, které je přístupné ze všech stránek aplikace a podrobné vyhledání přístupné ve vlastní sekci.

V původním návrhu aplikace Inaukce se uvažovalo pouze o vyhledávání předmětů v sekci určené výhradně pro vyhledávání předmětů. Při testování aplikace se došlo k závěru, že navržený vyhledávací formulář je pro mnohé případy příliš složitý. Dále ze statistické analýzy vyhledávání předmětů cílovými uživateli vyšlo najevo, že maximálně prohledávané atributy předmětů jsou jejich jména a popisy. Zbylé atributy jsou prohledávány pouze zřídka. Proto bylo vzhledem k maximalizaci uživatelského komfortu navrženo a implementováno tzv. rychlé vyhledávání předmětů. To je tvořeno boxem pro vyhledání předmětů, který se umísťuje na každou stránku aplikace. Uživatelé tedy nemusejí vstupovat do sekce pro vyhledávání. Cenou za rychlost a jednoduchost přístupu k rychlému vyhledávání je jeho funkčnost. Rychlé vyhledávání prochází pouze atributy jméno a popis předmětu.

2.11 Prohlížení předmětů

Kromě vyhledávání předmětů by mělo by mu být uživateli k dispozici přehledné procházení předmětů podle kategorií do kterých tyto předměty přísluší.

Pro tento účel by měla sloužit stránka obsahující přehledný výpis kategorií s vyznačením aktuální kategorie, pro níž se předměty vypisují a dále samotný výpis předmětů patřící do aktuální kategorie. Pro výpis předmětů se zvažovaly dvě možnosti:

- Vypsat jen předměty patřící do aktuální kategorie
- Vypsat předměty patřící do aktuální kategorie a všech jejích podkategorií

Po dohodě se zadavatelem byla zvolena druhá možnost. Při tomto výpisu se nejprve vypíší všechny předměty, jelikož všechny předměty patří do kořenové kategorie nebo jejích podkategorií. Při výběru podkategorií dochází k filtraci předmětů a postupnému

zužování výběru, což spíše odpovídá procházení předmětů. Pro uživatele je tento způsob také mnohem přehlednější.

Z výpisu předmětů by se mělo dát při vybrání konkrétního předmětu přejít na zobrazení detailů předmětu, kde by měly být vypsány všechny dostupné atributy tohoto předmětu a také by zde měla být možnost daný předmět zakoupit či provézt příhoz.

2.12 Obrázky a dokumenty předmětů

Ke každému předmětu by se měl dát nahrát obrázek či dokument popisující daný předmět. Ostatní uživatelé by pak měly mít možnost tyto obrázky a dokumenty prohlížet. Vzhledem k nutnosti vytvořit fyzický datový model během funkční analýzy, bylo nutné se rozhodnout v jaké formě a kde se budou soubory ukládat[12]. Po pečlivém zvážení byla vybrána pro uchovávání předmětů databáze MySQL, vzhledem k centralizaci uchovávání dat a poskytovanému rychlému adresování, takto uložených souborů.

Obrázky by měly být zobrazovány u předmětů ve výpisu předmětů ve zmenšené podobě a ve výpisu detailů předmětu ve skutečné velikosti.

Funkce nahrávání dokumentů k jednotlivým předmětů byl jeden z požadavků zadavatele, přestože tvoří bezpečnostní riziko. Obsah souborů, které chtějí uživatelé uložit nelze nijak jednoduše kontrolovat. a tedy lze uchovávat libovolné soubory, které nemají s předměty nic společného. Takto lze tuto funkci využívat pro uchovávání souborů na Internetu. Proto by měl být maximální počet dokumentů nahraných k předmětu omezen. Dále by měla být omezena velikost těchto dokumentů. Tím by se mělo alespoň částečně zabránit zneužití této funkce.

2.13 Správa uživatelských účtů a předmětů

Jedním z hlavních požadavků zadavatele byla snadná a přehledná správa uživatelských účtů a předmětů. Systém musel být navržen s ohledem na tento požadavek, tak aby administrátoři (a nikdo jiný) mohli provádět úpravy a manipulace uživatelských účtů a předmětů.

V případě zvoleného centrálního uložení dat v MySQL lze správu provádět pomocí administračních programů pro řízení databáze, jako jsou například PHPMyAdmin či mysqld, mysqladmin a podobně. Samotné použití těchto aplikací je

[12]Více o tomto tématu v kap. 3.8.1

pro běžného uživatele Inaukce velmi obtížné a vzhledem k závislostem jednotlivých databázových tabulek zcela nevhodné, jelikož odporuje požadavku zadavatele na snadnost a přehlednost. Navíc se jakýkoliv zásah do databáze prostřednictví těchto programů nedoporučuje vzhledem k výše zmíněným závislostem tabulek. Správa předmětů a uživatelů musela být tedy navržena jako součást aplikace.

Aplikace Inaukce by měla nabízet plnou kontrolu nad předměty ukládanými uživateli do databáze. Administrátoři by měli mít možnost měnit atributy či vlastníky předmětů, předměty mazat nebo naopak vkládat. K tomu by jim měly slouží přehledné výpisy předmětů a výpisy detailů předmětů společně s výběry možností správy předmětů. Vzhledem ke shodnému požadavku pro vyhledávání a výpis předmětů i pro běžné uživatele by měly být tyto funkce implementovány univerzálně pro použití v obou případech. V tomto případě by byl zaručen i jednotný vzhled. Funkčnost by závisela na právech přistupujícího uživatele, ale byla by do jisté míry omezena univerzálností řešení. Přístup k administračním funkcím by byl zpřístupněn pro administrátory ihned po přihlášení do systému. Druhé zvažované řešení by zpřístupnilo administrační funkce pro správu předmětů až po vstupu do administrační sekce (zde se sdružují všechny administrátorské funkce). Pro rychlejší administraci bylo zvoleno první řešení, které však potlačuje možnost pohybu administrátorů v Inaukci jakožto běžných uživatelů.

Aplikace Inaukce by měla mít plnou kontrolu nad uživatelskými účty. Administrátoři by měly mít možnost měnit atributy uživatelských účtů, přidávat nebo mazat uživatelské účty. Systém by měl dále nabízet přehledné výpisy a vyhledávání uživatelů dle konkrétních požadavků uživatelů. Na rozdíl od správy předmětů je vhodné, aby správu uživatelů byla, vzhledem k citlivosti dat s ní souvisejících, umístěna v administrační sekci. Pokud by byl takový požadavek zadavatele, mohl by být přístup ke správě uživatelů zabezpečen opětovným zadáním hesla, popřípadě další autorizací. Nebo by mohl být správa zpřístupněna pouze účtu superuser. Vzhledově by měla být správa uživatelů totožná se správou předmětů, aby byl splněn požadavek přehlednosti a tím i jednoduchosti.

2.14 Jednotný vzhled

Jednotný vzhled a systém navigace umožňuje uživateli rychlou orientaci a trvalou informaci o tom, kde se momentálně nachází. Z úvodní stránky lze vstupovat do

jednotlivých sekcí u nichž je informativní část soustředěna vždy do jedné části a zbylé části stránky zůstávají neměnné. Stejně tak se dodržuje i barevné ladění celých stránek.

2.15 Kategorie předmětů

Požadavkem zadavatele bylo, aby byl každý předmět v databázi zařazen do některé z nabízených kategorií. Bylo tedy nutné navrhnout strukturu kategorií a při navrhování fyzického datového modelu bylo nutné zvážit budoucí implementaci[13] této struktury ve zvolené datové struktuře.

Z hierarchické struktury kategorií vyplývá, že pro uložení kategorií bude nejlepší stromová struktura. Dále bylo nutné si ujasnit jaké akce může provádět aplikace typu internetová aukce s kategoriemi:

- základní operace se stromem – CRUD operace s uzly (kategoriemi) = CREATE, RETRIEVE, UPDATE,DELETE
- rozbalení stromu ve vybraném uzlu
- nalezení všech podkategorií dané kategorie
- zobrazení cesty k vybranému uzlu od kořene

Vzhledem k požadované jednotnosti a centralizaci uložení dat bylo vybráno uložení kategorií v tabulce databáze. Dále bylo nutné uvažovat prostředky zvolené databáze pro zacházení se stromy. MySQL bohužel zatím nepodporuje uložení stromů v tabulkách databáze[14], proto bylo zvoleno uložení stromu v datové struktuře známé pod názvem *Nested set* nebo také *DFS strom* – tato struktura se ukládá do tabulky databáze MySQL. Toto řešení je podle [6] nejvýkonnějším řešením reprezentace stromu pro zmíněné akce, přičemž se kladl důraz na akce nalezení všech podkategorií dané kategorie a zobrazení cesty k vybranému uzlu od kořene[15].

Kategorie jsou reprezentovány jedním stromem, který má vždy alespoň jeden uzel. Proto může být splněna podmínka, že každý předmět je zařazen do jedné kategorie.

[13]Více o zvolené implementaci uložení kategorií předmětů a samotném návrhu v kap. 3.8.2
[14]MySQL verze 4.x a nižší nemají žádnou podporu stromových struktur, tak jako např. ORACLE, kde se využívá v PL/SQL klauzulí START WITH a CONNECT BY.
[15]Více o dalších možných řešeních je popsáno v kap. 3.8.2

2.16 Zasílání e-mailových zpráv

V mnoha případech je nutné zaslat e-mail pro ověření či notifikaci. Mezi tyto případy patří například zasílání e-mailových zpráv pro ověření e-mailové adresy[16]. Bylo zvažováno, nevyužití e-mailové služby vzhledem k velmi malé podpoře e-mailových služeb poskytovateli free hostingu, který měl být cílovým pro umístění Inaukce. Po konzultaci se zadavatelem se však došlo k závěru, že nevyužití e-mailových služeb by rapidně snížilo funkčnost internetové aukce. Jelikož bylo zasílání e-mailů zahrnuto do návrhu Inaukce, zúžilo se tím možné nasazení Inaukce na free serverech a posunulo se spíše na komerční hosting, který většinou e-mailové služby podporuje. Možností, jak se tomuto vyhnout by bylo podporovat jak zasílání e-mailů, tak implementovat i způsob, který zasílání e-mailů nevyužívá. Tento způsob byl nakonec implementován, přestože původně nebyl uvažován ve Funkční specifikaci. Veškeré případy, při kterých se původně zasílali e-mailové zprávy, jsou nastavitelné v administrátorském nastavení tak, že se lze zasílání e-mailů vyhnout.

2.17 Datový model

V průběhu funkční analýzy byl vytvořen logický a fyzický datový model. Oba modely jsou obsaženy v Funkční specifikaci aplikace.

Návrh logického datového modelu probíhal bez problémů. K vytvoření modelu byl použit modelovací program Dia [8], který je přímo uzpůsoben pro tvorbu ER diagramů. Při návrhu fyzického datového modelu jsem však narazil na problém uložení stromů kategorií. Při analýze tohoto problému jsem musel brát v potaz použití databáze MySQL jako databázového serveru, ačkoliv takovéto úvahy patří spíše do implementační fáze vývoje aplikace. Databáze MySQL zatím neobsahuje podporu pro uchovávání a dotazování stromových dat. Proto bylo nutné nejprve nalézt vhodný způsob, jak tuto podporu nahradit[17]. Výsledkem bylo použití tzv. DFS stromů uložených v tabulce MySQL databáze, které je popsáno v kap. 3.8.2. Toto řešení je implementačně jednoduché a dostatečně rychlé.

Pro vytvoření fyzického datového modelu byl použit program DBDesigner4 [7], který byl použit i ve fázi implementační a testovací, jelikož umožňuje přímou úpravu a testování tabulek v databázi podle vytvořeného datového modelu.

[16]Pokud je nastaveno ověřování e-mailové adresy v administrátorském nastavení.
[17]Analýza návrhu datové struktury pro uložení stromů kategorií je popsána v kap. 2.15

Datový model je podrobně popsán v Programátorské příručce. Logický a fyzický datový model je znázorněn ve Funkční specifikaci.

2.18 Zabezpečená komunikace pomocí protokolu SSL

Přestože tento atribut nepatří přímo do funkční specifikace, ale spíše do implementační fáze, byl jedním z požadavků, které vznesl zadavatel. Muselo se s ním tedy počítat již v samotném návrhu aplikace. Bylo důležité, aby hledaný server, na kterém bude Inaukce umístěna podporoval protokol SSL. Bohužel tím došlo také k omezení na straně klienta, jehož webový prohlížeč musí také podporovat protokol SSL. Na druhou stranu se rapidně zvýšila bezpečnost komunikace mezi serverem a klientem, která je velmi důležitá pro zachování diskrétnosti a korektnosti jednotlivých prodejů či zabezpečení osobních údajů. Protokol SSL se využívá až po přihlášení uživatele do systému, kdy mohou probíhat důvěrné činnosti spojené s nákupem či prodejem předmětů.

Kapitola 3

Implementace

3.1 Výběr vývojových nástrojů

Aplikace Inaukce je standardní internetová aplikace skládající se z PHP, JavaScript a SQL skriptů. Vhodné vývojové prostředí by mělo podporovat vývoj všech těchto skriptů, mělo by mít pokud možno syntaktický zvýrazňovač pro přehlednou tvorbu těchto skriptů. Vzhledem k freewarovému rozšíření nástroje a stálému vývoji byl vybrán nástroj PsPad [9].

Pro vytvoření logického datového modelu byl použit program Dia [8]. Pro vytvoření fyzického datového modelu byl použit program DBDesigner4 [7], který byl využit i ve fázi implementační a testovací, jelikož umožňuje přímou úpravu a testování tabulek v databázi podle vytvořeného datového modelu.

Pro vytvoření obrázků byl použit nástroj Adobe Photoshop CS2.

3.2 Použitý software a systémové požadavky

3.2.1 Software na straně serveru

Vzhledem k požadavku zadavatele na nulovou cenu software presentační a datové vrstvy, byla možnost výběru velmi zúžena.

- Jako webový server byl použit server Apache. Apache je jednoduchý, ale přitom velmi výkonný web server, který je dostupný jak pro 32bitová Windows, ta i pro platformu Unix/Linux. Dostupnost na platformách Windows a Unix byl jeden z vedlejších požadavků zadavatele. Pro Apache mluví i podpora různých programovacích jazyků, které jsou součástí serveru ve formě modulů. Tyto moduly jsou mnohem menší než samotné běžné distribuce při zachování stejné funkčnosti, dosahuje se také výrazného zrychlení – což se však projevuje na zhoršení bezpečnosti[18].

  PHP může být zkompilováno jako samostatný interpret CGI nebo jako modul Apache. V PHP nastaveném jako interpret CGI je skript PHP pokaždé

[18]Otázka volby PHP jako modulu či CGI je probrána také v Instalační příručce.

interpretován, webový server pro něj vytvoří novou instanci interpretu, která tento skript zpracuje. To má za následek snížení výkonu a zjištění odolnosti webového serveru proti chybným PHP skriptům. Pokud je PHP zkompilováno jako modul Apache, běží ve stejném adresovém prostoru jako sám proces webového serveru a to poskytuje vyšší výkon (jak popisuje [3]). Pro Inaukci bylo použito PHP jako modulu Apache, tj. preference výkonu aplikace před robustností a odolností proti chybám PHP skriptů, které jsou v maximální možné míře potlačeny.

- Pro ukládání dat byla použita volně šiřitelná robustní relační databáze MySQL. Jedním z mnoha důvodů byla maximální podpora PHP a MySQL. Dalším důvodem výsledky testů volně šířitelných databází [17]. V těchto testech vyšla databáze MySQL s tabulkami InnoDB [2] jako vítěz i vzhledem k poskytovaným transakcím. Neposledním důvodem byl fakt, že většina freehostigů podporuje MySQL, zatímco ostatní databáze nikoliv.
- Pro zabezpečení přenosu dat mezi serverem a klientem je použit protokol SSL. SSL je na serveru Apache implementován pomocí modulu mod_ssl a knihovny OpenSSL. Toto řešení abstrahuje aktuální funkce SSL do modulu, čímž umožňuje jeho dynamické načítání. Lze použít i implementaci Apache-SSL, která je také volně šířitelná. (existují i komerční implementace např. StrongHold a RavenSSL – funkčnost všech těchto řešení by měla být shodná).

Aplikace Inaukce je přenositelná, vzhledem k operačnímu systému. Avšak, server Apache je lépe podporován operačními systémy Linux. Použití této platformy přináší malé zlepšení výkonu. Kromě slabého rozdílu ve výkonu v zásadě není rozdíl mezi jednotlivými operačními systémy. Kromě podpory výše zmíněných aplikací, Inaukce nemá konkrétní požadavky na operační systém.

3.2.2 Software na straně klienta

Klient zprostředkovává styk uživatele se serverem. Technické požadavky na používání aplikace Inaukce nejsou náročné. Stačí mít pouze přístup k internetu a mít k dispozici libovolný doporučený internetový prohlížeč. Bližší informace jsou popsány v Instalační příručce.

Komunikace mezi počítačem klienta a centrálním serverem aplikace Inaukce probíhá nešifrovaně, pokud není uživatel přihlášen – tedy je v roli anonymního

uživatele. V této roli smí uživatel pouze prohlížet předměty, tedy nešifrovaná komunikace by neměla být na závadu. Pokud je uživatel přihlášen k aplikaci Inaukce, jsou všechna zasílaná i přijímaná data v šifrované podobě. Pro šifrování se používá buď 56 bit nebo 128 bit šifra, podle toho, jakou úroveň má prohlížeč uživatele. Další ochranou je heslo, kterým se přihlašuje uživatel do aplikace. Heslo lze kdykoliv změnit (popsáno v Uživatelské příručce). Tato změna se ukládá na server v zašifrované podobě a tím je zajištěno, že jej neznají ani administrátoři aplikace Inaukce.

Pro komunikaci a chod aplikace nejsou používána *Cookies*. Prohlížeč proto nemusí *Cookies* podporovat nebo povolovat.

3.3 Použití XHTML vs. HTML

XHTML (podle [4]) je odvozenina syntakticky striktního jazyka XML, na nějž je naroubován starší, naopak velmi volný jazyk HTML. Zásadní rozdíl mezi oběma platformami (XML a HTML) je tedy v tom, jak se s jejich dokumenty nakládá. Dokumenty XML zpracovává XML-parser, který velmi přísně kontroluje syntaxi a na jakékoli chybě skončí, zobrazí chybové hlášení a dokument nezobrazí. Zatímco dokumenty HTML zpracovává HTML-parser, jemuž je už historicky dána nadmíra tolerance, jakoukoli chybu se snaží opravit, chybějící značky domyslet, přebývající přehlédnout.

Je těžké polemizovat, zda používání XHTML přináší nějaké objektivní výhody. Osobně jsem přesvědčen, že všechna tvrzení o tom, že dokumenty XHTML jsou menší nebo rychlejší nebo že se snáze zpracovávají, jsou jen akademická a v praxi mají téměř nulový efekt. Výběr XHTML pro Inaukci se řídil těmito důvody:

- XHTML nutí dělat čisté a striktní stránky, které jsou nejefektivnější – proto všechny stránky aplikace Inaukce splňují normu XHTML 1.0 Strict DTD.
- XHTML je oproti HTML živým jazykem, který je stále vyvíjen.

3.3.1 Chyba MSIE ve zpracování XHTML stránek

To, jak se dokument zpracuje, říká jeho MIME-typ. MIME-typ je přiřazen každému bloku dat, který se posílá po webu. Posílá se v hlavičce HTTP a podle něj se prohlížeč rozhoduje, binárním kódem, který dostává na vstupu, má naložit. Hlavičky HTTP obvykle odesílá webový server, který nejčastěji vychází z přípony souboru. Pro "hybrid" XHTML pak byl definován další typ application/xhtml+xml. Pokud je stránka v jazyce

XHTML poslána s korektním MIME typem, prohlížeč kód stránky zpracovává pomocí parseru XML. Zpracuje jej striktně, bez tolerance chyb. Jenže nejpoužívanější prohlížeč světa, MSIE, obsahuje několik závažných chyb[19]. Především typy začínající "application" (tedy např. application/xhtml+xml) nezpracovává vůbec a nanejvýš nabídne, že si stahovaný soubor lze uložit na disk. Takže jako XML se dokumenty XHTML (zatím) zobrazovat v MSIE nedají. Musí se tedy použít typ text/html, a tedy parser HTML. Problém je v tom, že každý XHTML dokument (stejně jako každé jiné XML) by měl začínat XML prologem, neboli značkou ve tvaru:

```
<?xml version="verze" encoding="kódování"?>
```

Jenže závažná chyba v HTML parseru MSIE 6 způsobí, že pokud na tento prolog narazí, je chybně zpracován. Bohužel ale specifikace jazyka XHTML říká, že tento prolog se může vynechat jenom tehdy, když je použito kódování UTF-8. V opačném případě[20] prolog být uveden musí. V tomto případě jsem se tedy vědomě dopustil drobné chyby, oproti specifikaci XHTML 1.0 Strict DTD.

3.4 Tvorba XHTML stránek

Tvorba XHTML stránek je v aplikaci Inaukce dynamická a provádí se pomocí PHP skriptů na straně serveru. Další možností, která byla zvažována je zpracování na straně klienta. Zpracování na straně serveru a generování webových stránek nabízí oproti technologiím na straně klienta více výhod:

- minimalizuje provoz na síti omezením potřeby vzájemné komunikace prohlížeče a serveru
- zrychluje načítání stránek, protože v konečné fázi stahujeme stránku v XHTML
- obchází problémy s kompatibilitou prohlížečů
- může prohlížeči poskytnout data, která nejsou na straně klienta k dispozici
- umožňuje lepší zabezpečení, protože můžeme zakódovat věci, které prohlížeč nikdy neuvidí

U větších projektů, jako je aplikace Inaukce, je velmi důležité mít důsledně oddělenou aplikační vrstvu aplikace od prezentační. Je dobré uvažovat o skutečnosti, že dobrý programátor nemusí být dobrý grafik, či HTML kodér a tedy je vhodné tyto související části oddělit.

[19]Jedná se o MSIE do verze 6.x, verze 7.x je v době psaní této práce stále vyvíjena.
[20]Tedy u nás častých kódování windows-1250 a iso-8859-2.

Jedním z řešením tohoto úkolu by mohlo být XSLT - programátor vygeneruje v podstatě libovolné XML a grafik si ho převede jak chce pomocí XSLT. Problém je v tom, že většina HTML kodérů XSLT neumí a učit se ho nechtějí, protože jim to příliš připomíná programování.

Proto se spíše používá řešení založené na rozličných šablonovacích systémech. To funguje tak, že programátor v kódu jenom naplní PHP proměnné, grafik vytvoří šablonu, která tyto proměnné využívá, a šablonovací systém to celé poskládá dohromady. Další velkou výhodou šablonovacích systémů je to, že pokud neřeknete jinak, tak jsou všechna vypisovaná data automaticky escapovaná, takže nehrozí nebezpečí *XSS*[21].

Jeden z nejoblíbenějších šablonovacích systémů je Smarty, který však posouvá aplikační logiku blíže k prezentační a tedy dává grafikovi velmi silnou zbraň. Pro tvorbu aplikace Inaukce je použit jednoduchý a velmi rychlý šablonovací systém FastTemplate, který je však starší[22], a musel být pro potřeby Inaukce značně upraven.

3.4.1 FastTemplate

FastTemplate[23] je velmi používaný balíček PHP (čerpáno z [15]), který obsahuje jednoduchý šablonovací systém. Tato technika zcela odděluje kód od HTML, což je jeden z důvodů, proč byla knihovna zvolena jako součást projektu, dalším důvodem je velmi snadná změna stránek grafického ztvárnění stránek bez změny funkčních PHP skriptů. Spolu s využitím kaskádových stylů je FastTemplate velmi mocným nástrojem.

Základní myšlenkou FastTemplate je jediná stránka skládající se z více logických částí (šablon) s pojmenovanými jednotlivými částmi stránky. To jsou názvy šablon. Každá šablona může obsahovat více proměnných. Proměnné jsou nahrazovány buď čistým textem (tj. HTML) nebo výstupem jiných šablon. Základním principem tvorby stránek pomocí FastTemplate je vytváření zdola nahoru, začíná se „malými“ prvky, které se vkládají do „větších“. Tento proces pokračuje až do sestavení stránky a jejího zobrazení.

Pro umožnění představy, zde příklad:

[21] *Cross site Scripting (také XSS nebo CSS)* nastává, když dynamické webové stránky zpracují data podstrčená uživatelem a zobrazí výstup, aniž by byl řádně ověřen. Tato data mají většinou formu odkazu, jehož součástí je také škodlivý obsah, a jsou šířena všemi prostředky Internetu.
[22] Od verze 4.0 je nefunkční.
[23] Více je popsáno v Programátorské dokumentaci.

```
<!- šablona nejvyšší úrovně - toplevel.tpl ->
<HTML>
<HEAD>
<TITLE>{TITLE}<TITLE>
<BODY>
{CONTENT}
</BODY>
</HTML>
```

V projektu Inaukce jsou šablony uloženy v adresáři Templates. Šablonu nejvyšší úrovně tvoří soubor Hlavni.tpl. Takovýto základ šablony by měl šablonou nejvyšší úrovně pro téměř všechna sídla. Obsahuje dvě proměnné ({TITLE} a {CONTENT}), které jsou vyplněny, když se řekne FastTemplate, aby analyzovalo šablonu nejvyšší úrovně. Zbytek je jen čisté HTML.

Pro vytvoření výstupu a vyplnění hodnot je potřeba nějaká akce PHP:

```
<?php include "class.FastTemplate.php";
//vytvoreni nove instance tridy FastTemplate
//prvni parametr urcuje cestu, kde jsou sablony ulozeny
$tpl = new FastTemplate(".");
//definice mapovani z nazvu sablony ma nazev souboru
$tpl->define(array("toplevel" => "toplevel.tpl"));
//prirazeni hodnot promennym
$tpl->assign("TITLE","Nadpis sablony");
$tpl->assign("CONTENT","Obsah stranky");
//analyza sablony nejvyssi urovne (FastTemplate doplnuje
hodnoty)
$tpl->parse("MAIN","toplevel");
//vypis
$tpl->FastPrint();?>
```

Proměnná MAIN se zde používá jako proměnná nejvyšší úrovně. V tomto případě nemá žádný zvláštní účel. Proměnné jsou zapsány v šablonách ve tvaru {NAZEV}, poněvadž se v kódu PHP na ně odkazuje pouze jejich názvem. Názvy proměnných musí splňovat stejné podmínky jako normální proměnné PHP.

3.5 Grafické uživatelské rozhraní

Dnešním světovým trendem na poli elektronické prezentace je ústup od přehnaně vyumělkovaných, hardwarově náročných a nejrůznějšími flashovými prvky překombinovaných prezentací (byť graficky velmi zdařilých, bohužel finančně poněkud náročnějších) směrem k přehledným, jednoduše řazeným, kde je kladen důraz především na účelovost, snadnou orientaci a informativnost.

Webdesign – take-away value – pod tímto pojmem je představován (podle [10]) první dojem,

vznikající při prvním shlédnutí webových stránek. Velmi vysoký důraz při tvorbě stránek je právě kladen na grafiku a barvu. Technická proveditelnost a kvalitní kódové zpracování je věc v pořadí druhá. Nejdříve musí web návštěvníka zaujmout a upoutat jeho pozornost. Pokud jsou stránky přehledně uspořádané a bez rušivých elementů, se správně zvolenou kombinací barev, návštěvník – potencionální klient je upoután a nic ho neodrazuje od dalšího prohlížení. Pokud se návštěvník na stránkách neorientuje, má tendenci prohlížení ukončit.

3.5.1 Barvy

Jako základní barva aplikace byla vybrána modrá. Modrá je barva klidu, míru, nebe a vody a obecně je to asi nejoblíbenější barva, a to i navzdory tomu, že je velmi chladná a odměřená. Je spojována se stabilitou a jistotou a lidé mají tendenci jí důvěřovat. Je to také barva harmonie, čistoty a trpělivosti. Lidé, kteří ji mají rádi, bývají uzavření s malým okruhem přátel a více přemýšlí, než mluví. U modré je dobré zmínit, že na ženy působí zcela odlišně než na muže. Mají před ní vetší respekt, mají pocit určité nadřazenosti této barvy a spojují ji tak často s vůdčí rolí a tvrdým obchodem.

3.5.2 Kaskádové styly

Vzhledem k použití striktního XHTML kódu bylo nutné grafickou strukturu stránek vyjádřit pomocí kaskádových stylů. Strategie, která byla použita, je nejpokročilejším využitím kaskádových stylů - spočívá v tom, že se CSS svěří veškeré formátování dokumentu, včetně rozvržení stránky a polohování jednotlivých prvků. Protože se rozvržení stránky v této koncepci obejde bez tabulek, vžil se pro ni název beztabulkový (tableless) design.

Základem kvalitního návrhu stránky je v tomto případě perfektně strukturovaný zdrojový kód XHTML. Kvalitní struktura kódu je zásadní jednak proto, že umožňuje nejefektivnější formy využití kaskádových stylů (kontextové a další pokročilé typy selektorů), a jednak proto, že starší prohlížeče a některá další menšinová zařízení zobrazují ve skutečnosti jen prosté HTML, bez explicitního formátování. Uživatel je pak odkázán jen na implicitní formátování dle možností svého prohlížeče a pouze kvalitně strukturovaný obsah si v tom případě podrží dostatečnou přehlednost a použitelnost.

Stylové předpisy ovlivnily podporu starších prohlížečů. Podporované prohlížeče a jejich verze jsou popsány v Instalační příručce.

3.5.3 Grafická struktura stránek

Aplikace Inaukce má v dnešní době nejpoužívanější rozvržení stránky (podle [13]), které je složeno z jednotného záhlaví a zápatí, více méně jednotného levého menu a proměnnou pravou část. Takovéto rozvržení se dnes považuje za ideální z hlediska orientace návštěvníka a přináší mnoho výhod mezi něž například patří přehlednost či stejný motiv příbuzných stránek. Odkazy v menu jsou odlišeny od okolního textu a obsahují odkaz na hlavní stránku, což je základní atribut „dobré“ orientace.

Pro vytvoření tohoto rozvržení existuje několik postupů. Většina používaných postupů se dá dobře ilustrovat při omezení výkladu na tvorbu levého sloupečku. S tím jsou spojeny dva problémy, lépe řečeno rozhodnutí:

1. Jak umístit sloupeček s odkazy vlevo (konstrukční problém)
2. Jak naplnit a udržovat obsah toho sloupečku aktuální při případných změnách (organizační problém)

Naplnění a údržba je popsána v kapitole 3.4, popř. detailněji v Programátorské příručce - použité řešení využívá knihovny FastTemplate pro oddělení XHTML a PHP kódu. Umístění sloupečku odkazů vlevo je problém užší - omezuje se na XHTML, popřípadě CSS.

Možnosti umístění sloupečku vlevo:

- *Rámy* - Z metodiky rámů se vlastně obliba levého sloupečku vyvinula. Rámy představují velmi snadnou cestu tvorby a aktualizace levého sloupečku (a historicky nejstarší). Mají však řadu chyb a problémů mezi něž patří špatné ukládání, tisk či odkazování.
- *Tabulkou přes celou stránku* - Dnes nejpoužívanější metoda, protože se ve všech prohlížečích zobrazuje téměř stejně. Nevýhodou však je pomalé zobrazování stránky, protože celá stránka se zobrazí až ve chvíli, kdy se načte celá ta tabulka.
- *Pozicováním* - Pomocí CSS pozicování lze umístit libovolný objekt kamkoli do stránky a tedy lze text stránky posunout vedle levého menu. Velkou výhodou oproti použití tabulky je postupné načítání a zobrazování obsahu stránky. Nevýhodou pak je slabá podpora starších prohlížečů.
- *Obtékaným divem* - Na menu se použije plovoucí oddíl = tag . Do divu se vloží obsah menu. Stylem se divu nastaví obtékání (float) a šířka (width).

```
<div style="float:left; width: 150px">plovoucí
     oddíl</div>
<div style="float:left; width: 600px">text
     stránky</div>
```

Toto řešení bylo použito v implementaci aplikace Inaukce.

3.6 Databáze

Aplikace Inaukce využívá pro ukládání dat databázi MySQL. Pro bezproblémový chod aplikace musí být některé akce zpracovávány jako transakce, stejně tak je vhodné použití cizích klíčů. Standardní typ tabulek MyISAM databáze MySQL však ani jedno nepodporuje. Proto bylo nutné použít tabulky databáze Inaukce jiný typ. Mezi možná řešení patří typ InnoDB či BDB[24], které je však stále ve fázi vývoje a tedy lehce nestabilní, proto bylo použit stabilní typ InnoDB pro všechny tabulky databáze.

3.6.1 InnoDB

Nejpodstatnější funkcí pro Inaukci, kterou InnoDB nabízí, je plná podpora transakcí a cizích klíčů. Mezi další zajímavé funkce patří zamykání na úrovni záznamů (MyISAM umí zamykat pouze tabulky), konzistentní čtení dat (tzv. multiverzování -- každý výběr vidí celou databázi ve stejném tvaru, jak na začátku výběru dat, tak na jeho konci), nelimitovaná velikost dat (využívám pro vkládání souborových dat), automatické opravování po havárii a také vysoká rychlost srovnatelná s MyISAM. Při vkládání dat by měl být dle vývojářů InnoDB dokonce rychlejší než MyISAM, což dokazují všemi možnými testy na svých stránkách. Opět ale platí pravidlo "něco za něco". InnoDB nepodporuje některá rozšíření MySQL, jako například FULLTEXT indexy, indexy na TEXT a BLOB sloupcích a podobně. Tyto indexy by v aplikaci Inaukce našly použití hlavně pro indexaci sloupce *description* tabulky *subjects*.

3.6.2 Komunikace s databází

PHP podporuje rozhraní API pro přístup k velkému počtu databází mimo jiné MySQL. Pro přístup k databázi MySQL lze použít dvě rozhraní API. Obecné rozhraní *ODBC* se používá pro psaní všeobecných databázových aplikací, které však podporuje jen všeobecné funkce a je pomalejší než rozhraní API pro konkrétní databázi. Na druhou stranu aplikace, která používá toto rozhraní je přenositelná vůči databázovému serveru.

[24]Další typy tabulek, jsou popsány v Programátorské příručce – použití jen v odůvodněných případech.

Už od návrhu aplikace se počítalo s využitím databáze MySQL jakožto centrálním úložištěm dat, a proto bylo použito *API MySQL* v PHP, bez možnosti přenositelnosti vzhledem k typu databázového úložiště, což přináší slabé zrychlení.

Pro přístup do databáze nepotřebuje běžný uživatel znát přístupové údaje. Ty jsou zadávány administrátorem při instalaci Inaukce a uloženy do konfiguračního souboru. Konfigurační údaje jsou uloženy v souboru Variables.php, ke kterému by měl mít přístup pouze superuser[25].

3.6.3 Tabulky databáze

Tabulky jsou vytvářeny při instalaci Inaukce. K vytváření tabulek se používá instalační SQL skript Inaukce.sql, který obsahuje příkazy pro vytvoření jednotlivých tabulek. Při instalaci jsou dále tabulky naplněny implicitními daty, které jsou uloženy v tabulce Insert_def_data.sql.

3.7 Uložení přihlašovacích údajů, autentizace a autorizace

Podle analýzy popsané v kapitole 2.7 bylo implementováno uložení přihlašovacích údajů, autentizace[26] a autorizace. Ty jsou nutné v případě, že uživatelé chtějí prokázat svou totožnost, aby mohli vykonávat nákupy, prodeje či správu.

3.7.1 Uložení přihlašovacích údajů

Podle fyzického modelu byly přihlašovací údaje uloženy v tabulce users. První návrh počítal s tím, že bude vytvořena samostatná tabulka pro uložení přístupových údajů. Z vypracovaného ER-schématu vyplynulo, že by výsledné schéma nebylo v 3. normální formě. Proto se přistoupilo k sjednocení těchto tabulek, které nikterak nenaruší bezpečnost uložení citlivých dat.

Pro maximalizaci zabezpečení přístupových údajů jsou hesla ukládána jako hash řetězec. Ke každému původnímu heslu se ukládá náhodný tzv. salt řetězec o 13 Bytech. Tento řetězec se připojuje k heslu a z výsledného řetězce se vytváří hash řetězec o 40 Bytech. Tím se zabraňuje útoku na stejná hesla.

[25]Zabezpečení konfiguračního souboru je podrobněji popsáno v Instalační příručce.
[26]Místo *autentizace* (z něm.) se také používá pojem *autentifikace* (z fran.) či *autentikace* (z angl.)

3.7.2 Hashovací funkce

Důležitou otázkou z hlediska bezpečnosti aplikace byla volba vhodné hashovací funkce. Na přelomu let 2005/2006 byl publikován článek [1] Vlastimila Klímy o nalézání kolizí MD5 na běžném počítači. Před tímto článkem byla hashovací funkce MD5 stále ještě považována za bezpečnou funkci a proto jsem ji v prvních fázích implementace používal, avšak v březnu 2005 byla definitivně poražena. Vlastimil Klíma objevil novou myšlenku tunelování hašovacích funkcí. Tunely umožňují nahradit současné metody mnohonásobné modifikace zpráv a exponenciálně zkrátit čas hledání kolizí. S jejich využitím lze zkrátit čas hledání kolize hašovací funkce MD5 z původních osmi hodin na minutu na běžném notebooku (Intel Pentium, 1.6 GHz). Metoda je použitelná pro libovolnou inicializační hodnotu. Tunely mohou být využity k urychlení hledání kolizí i jiných hašovacích funkcí. Metoda byla experimentálně ověřena. Naštěstí nalezené tunely nejdou aplikovat přímo na funkci SHA1,256,512,... a tedy tyto metody zatím ještě obstály. Z hlediska výkonosti je nejrychlejší SHA1, avšak je také nejméně bezpečná. Pro aplikaci Inaukce je však takováto míra bezpečnosti zcela dostatečná, a proto je aplikací využívána (vytváření hash hesla, náhodných řetězců, apod.).

3.7.3 Uživatelská práva

Každý uživatel systému má přiřazena přístupová práva. Tyto práva jsou uložena v tabulce *users* společně s dalšími informacemi o uživateli.

3.7.4 Uživatelská autentizace

Principiálně se autentizace rozděluje na dva typy: *uživatelskou* a *entitní* (podle [12]). K uživatelské autentizaci dochází v aplikaci Inaukce při přihlašování zadáním uživatelského jména a hesla. Poté server zašle klientovi *session token*, což je náhodně vygenerovaný identifikátor aktuální session. Ten si prohlížeč „uchovává“ v podobě PHP proměnné, která je posílána metodou GET. Společně s každým dalším požadavkem pak prohlížeč zasílá tento session token, čímž je prováděna autentizace entitní. Ta úzce souvisí se správou aktuální session a bude o ní řeč v kapitole 3.7.10.

Aplikace Inaukce implementuje relativně bezpečnou formu uživatelské autentizace při níž uživatel zadává své unikátní uživatelské jméno a heslo do XHTML formuláře. Pro vyšší bezpečnost se heslo zadává do vstupního formulářového pole typu *password*, které

vstupní znaky maskuje. Navíc není toto pole nikdy automaticky předvyplňováno na základě předchozích vstupů, jak činí novější prohlížeče u běžných textových polí.

Po zadání přihlašovacích údajů je přihlašovací formulář výhradně odesílán HTTP metodou POST. Podoba GET požadavku se totiž včetně všech odesílaných parametrů ukládá do historie prohlížeče i do případných logů po cestě, a tudíž je snadno odcizitelná. Komunikace mezi klientem a serverem je zabezpečená během uživatelské autentizace (pokud proběhne úspěšně, tak i po ní), kdy sítí putuje uživatelské jméno a heslo. K tomuto účelu se používá protokol *SSL*[27].

Následně dochází k ověření uživatelského jména a hesla. K heslu se přidá *salt* řetězec a z výsledného řetězce se vytvoří pomocí funkce *SHA1* hash řetězec. Tento hash se porovnává s řetězce uloženým ve sloupci *passw* tabulky users pro dané uživatelské jméno.

```
If (sha1(concat(passwadd,formheslo)) == passw)...
```

Pokud heslo souhlasí uživatel může být přihlášen do systému v opačném případě je mu nabídnut opět formulář pro vyplnění přihlašovacích údajů.

3.7.5 Možné řešení – Autentizace pomocí HTTP

PHP podporuje standardní HTTP autentizaci prostřednictvím proměnných $_SERVER['PHP_AUTH_USER'] a $_SERVER['PHP_AUTH_PW']. Protokol HTTP umožňuje serveru zaslat hlavičku:

```
WWW-Authenticate: Basic realm="JMENO"
```

a kód ze série 4xx, konkrétně `401 Unauthorized` klientovi. Po přijetí této hlavičky klientem bude uživatel dotázán na autentizaci. Většinou si klient zadané údaje dočasně pamatuje a další spojení probíhají už bez účasti uživatele.

[27] O využití a samotném protokolu SSL je popsáno více v kap. 3.11

```
//dotaz do databáze, kterým se ověřuje přihlášení
//uživatele, popř. přihlašovací údaje
$row = mysql_fetch_assoc(
  mysql_query("SELECT * FROM uzivatele WHERE login = '".
   addslashes($_SERVER['PHP_AUTH_USER'])."' AND heslo = '".
   md5($_SERVER['PHP_AUTH_PW']) . "'"));
//pokud uživatel není přihlášen($row == false), zobrazí se
//mu přihlašovací formulář v samostatném okně
if (!$row) {
  header('HTTP/1.1 401 Unauthorized');
  header('WWW-Authenticate: Basic realm="Název aplikace"');
  page_header("Přihlášení");
  echo "<p>Pro přístup na tuto stránku se musíte
    přihlásit.</p>\n";
  page_footer();
  exit;
}
```

Výhoda tohoto přístupu spočívá v relativně nižší pracnosti a také v tom, že uživatel se může přihlásit standardním způsobem přes http://login:heslo@www.example.com, byť se to asi málokdy použije - v IE 6 v XP SP2 byl tento způsob zápisu adresy z bezpečnostních důvodu navíc zakázán. Někdo může za výhodu považovat také to, že se pro přihlášení zobrazí standardní dialog prohlížeče, pro jiného to ale bude spíše nevýhoda. Jasná nevýhoda je ta, že přihlašovací data se přenášejí při každém stažení stránky nešifrovaně a navíc ve standardizované podobě, takže je pro různé sniffery jednodušší je získat. Problém je také s odhlášením - např. v Mozille stačí poslat hlavičku 401 (uživateli se místo informace o odhlášení zobrazí přihlašovací dialog), v Internet Exploreru bylo nutné otevřít adresu s neplatným uživatelem (např. http://neexistuje@www.example.com), kvůli výše uvedenému omezení ale není možné už ani to. Poslední zádrhel se skrývá v tom, že pokud nejsou pod kontrolou všechny skripty na celé doméně, může záškodník použít stejný realm[28] a tím potenciálně získat přihlašovací údaje uživatele (pokud ho přiměje po přihlášení k aplikaci navštívit své stránky). To je také důvod, proč se při zapnutém *safe_mode*[29] do realmu doplňuje UID vlastníka skriptu. To pochopitelně vzhled standardního přihlašovacího dialogu poněkud nabourává.

[28]Údaj o přihlášení obsahující uživatelské jméno a heslo.
[29]Nastavení PHP většinou v souboru php.ini.

3.7.6 Možné řešení – Autentizace pomocí session cookies

Ve většině případů je proto asi lepší si přihlašování zařídit sám a využít přitom možností cookies. Informace o přihlášenosti uživatele se dá ukládat do session.

```
//dotaz do databáze, kterým se ověřují přihlašovací údaje
if (isset($_POST["auth_login"])) {
  if (mysql_result(
    mysql_query("SELECT COUNT(*) FROM uzivatele WHERE
      login = '$_POST[auth_login]' AND heslo = '".
      md5($_POST["auth_heslo"]) . "'"), 0)) {
    session_regenerate_id();//ochrana před Session Fixation
    $_SESSION["logged"] = true; //informace o přihlášení
  }
}
//pokud není uživatel přihlášen, zobrazí se mu přihlašovací
//formulář
if (!isset($_SESSION["logged"])) {
  page_header("Příhlášení");
  if (isset($_POST["auth_login"])) {
    echo "<p>Neplatné přihlašovací údaje.</p>\n";
  }
  echo "<form action='' method='post'>\n";
  echo "Login: <input name='auth_login' maxlength='30'
    />\n";
  echo "Heslo: <input type='password' name='auth_heslo'
    />\n";
  echo "<input type='submit' value='Přihlásit' />\n";
  echo "</form>\n";
  page_footer();
  exit;
}
```

Tento způsob je o něco pracnější, výhoda je ta, že máme vše pod kontrolou. Pozor je ale potřeba dát na zabezpečení session proměnných. Odhlášení je triviální, stačí provést PHP příkaz `unset($_SESSION["logged"])`, kterým se odstraní informace o přihlášení a uživatel je odhlášen.

Se správou session je spojeno několik různých útoků. Když někdo získá cizí Session ID a použije ho pro sebe, jedná se o *Session Hijacking*. Dalším typem útoku je *Session Fixation*. Jedná se o útok, kdy útočník klientovi nastaví nějakou hodnotu Session ID (podstrčením odkazu na webu, v e-mailu nebo např. přímou editací uživatelových cookies) a jakmile se uživatel přihlásí, tak tuto Session ID použije pro sebe.

Pokud uživatel nemá povolené cookies a je zapnutá direktiva *session.use_trans_sid*, přenáší se Session ID v URL. To sice umožňuje použít sessions i

uživatelům s vypnutými cookies, ale přináší to další bezpečnostní rizika. URL se totiž ukládá na řadě míst (v historii prohlížeče, v logu serveru, jako Referer se dostane i na odkazované servery, ...) a tím zjednodušuje útoky Session Hijacking. Kvůli útoku Session Fixation je vhodné mechanismus předávání Session ID v URL zcela vypnout direktivou *session.use_only_cookies*. Pokud session proměnné nejsou používány k uchovávání citlivých informací, může se tento mechanismus nechat zapnutý. Většina vyhledávačů již umí ignorovat session identifikátory používané PHP, takže stránky s různými Session ID se ve vyhledávačích neukládají vícekrát, ve vyrovnávací paměti ale pochopitelně pořád ano.

Kód zajišťující přihlášení se zpravidla umísťuje do souboru *auth.inc.php*. Do všech souborů vyžadujících přihlášení se pak tento soubor vkládá. Výhoda je velká – uživatelům lze nabídnout jakýkoliv odkaz (např. v e-mailu) a pokud se na stránku podívají nepřihlášení, rovnou se jim zobrazí přihlašovací dialog, po jehož vyplnění se stránka zobrazí.[30] Pokud se přihlašovacímu dialogu předá adresa stránky, kterou uživatel chce zobrazit, dá se dosáhnout stejné funkčnosti.

Nevýhodou tohoto přístupu je nutná podpora cookies v klientově prohlížeči. V dnešní době cookies podporuje naprostá většina prohlížečů, ale jejich podporu lze jednoduše vypnout.

3.7.7 Možné řešení – Autentizace pomocí výzva-odpověď a SSL

Pokud je používán protokol HTTPS, může být také využit pro přihlášení pomocí klientských certifikátů, ale obzvláště s jejich generováním je o něco více práce. Bezpečnost je každopádně s přihlašováním pomocí hesla nesrovnatelná.

Pomocí protokolu HTTPS lze zajistit šifrovaný přenos všech informací a ideálně se tak hodí mimo jiné pro přihlašovací formuláře. Pokud tento protokol nelze použít (u malých projektů proto, že vám nevyjde vstříc hosting, u velkých z výkonnostních důvodů), přenáší se všechna data nešifrovaně a zdatný uživatel je může po cestě odposlouchávat. Bezpečné přihlašování se ale dá zajistit i na nezabezpečeném protokolu.

Technika výzva-odpověď funguje tak, že server pošle klientovi výzvu, klient k této výzvě připojí své heslo a serveru pošle otisk tohoto spojení. Server na své straně provede totéž a pokud výsledky odpovídají, tak uživatele přihlásí, jinak ho odmítne.

[30]V některých případech je preferováno použití přesměrování na další skript pomocí zaslání hlavičky Location

Bezpečnost tohoto řešení je založena na tom, že server každou výzvu posílá jen jednou a pokud se útočníkovi podaří odpověď klienta zachytit, k ničemu mu to neposlouží, protože stejnou výzvu už server nikdy nepošle.

K technické realizaci tohoto řešení je zapotřebí na straně serveru i klienta funkce na výpočet otisku hesla spojeného s výzvou. V PHP i dalších serverových jazycích jsou hashovací funkce k dispozici již v základu, takže máme situaci poměrně jednoduchou, na straně klienta budeme muset sáhnout po externí knihovně - např. JavaScript pro MD5 i SHA1 nabízí v BSD licenci Paul Johnston.

Pro lepší zabezpečení aplikace Inaukce doporučuji přejít na tento způsob přihlášení a uchovávání informace o přihlášení. Z důvodů technické náročnosti však toto řešení nebylo implementováno.

3.7.8 Session tokens

Po ověření přihlašovacích údajů je nutné, aby informace o přihlášení byla uchována. HTTP je sám o sobě bezstavový protokol, za normálních okolností tak server odpovídá na jednotlivé dotazy klienta, aniž by si je navzájem dával do souvislostí. Proto byl zaveden na aplikační úrovni mechanizmus sessions. Pro každou session se vygeneruje při prvním dotazu jedinečný identifikátor – session token. Ten se odešle klientovi, který jej pak odesílá na serveru s každým dalším dotazem metodou GET. Server si tak podle něj může navzájem pospojovat všechny klientovy dotazy.

V dnešní době se však mnohem více prosazuje varianta popsaná v kapitole, kdy se využívá cookies pro uchovávání session token na straně klienta. Tato varianta je mnohem bezpečnější. Předávání session tokenu pomocí přepisování všech odkazů ve stránce, kdy se session token přidává jako další parametr k danému odkazu zvyšuje riziko ukradení session případným narušitelem. Kvůli upravené podobě všech otevíraných URL se totiž aktuální session token ukládá do historie prohlížeče i do všech možných logů během cesty po síti. Vzhledem k požadavku zadavatele na maximální přístupnost musela být implementována tato verze, jelikož cookies nemusí být podporovány všemi uživateli Inaukce.

Session token musí být unikátní, nesnadno odhadnutelný, odolný proti dekompozici. Nebezpečí ukradení tokenu útokem hrubou silou, kdy útočník postupně zkouší jeden potenciální token za druhým, eliminuje jeho dostatečná délka, díky které se tento způsob útoku stane v daném časovém limitu výpočetně neuskutečnitelný.

Zvolená hashovací funkce SHA1 vytvářející z náhodného řetězce 40 Bytový hash řetězec je dostatečně bezpečné řešení.

Pro vytváření a spravování session tokenů slouží třída *Session*. Při vytvoření session tokenu pro danou session je hash řetězec uložen do nového řádku tabulky *actsessions* společně s dalšími informacemi o tomto session. Při každém dotazu je pak zaslaný session token kontrolován s příslušným řádkem v tabulce.

Možnost dalšího vývoje: session token je vhodné nějakým způsobem navázat na aktuálního klienta, aby se zabránilo jeho ukradení či replay–attacku, například omezením na konkrétní IP adresu, daný typ prohlížeče apod. Pro ukládání tokenu by se měly používat neperzistentní cookies (také často nazývané „session cookies“), které prohlížeč neukládá na disk a uchovává je jen v RAM paměti. Při zavření okna prohlížeče je tak session token ztracen a případný další uživatel počítače nemůže ve stejné session pokračovat.

3.7.9 Autorizace

Autorizací se rozumí proces ověření přístupových oprávnění uživatele vstupující do informačního systému. Inaukce implementuje *přístupový kontrolní mechanizmus*, který řídí, kteří uživatelé mohou přistupovat k jednotlivým zdrojům, které funkce mohou využívat apod. Ochraňuje aplikaci před neautorizovaným prohlížením, změnami nebo kopírováním dat.

Při autorizaci se ověřuje, zda má uživatel dostatečná oprávnění pro přístup k určitému souboru či pro provedené určité akce. Tato kontrola se provádí na základě jeho přístupových práv. Neboli se předpokládá předchozí úspěšná autentizace, na jejíž spolehlivosti je autorizace plně závislá.

Jsou definovány tři nejpoužívanější *modely řízení přístupu*, které se ale v praxi navzájem často kombinují: model volný, direktivní a založený na rolích:

1. *model volný* - O přístupu ke zdroji či informaci je rozhodováno na základě identity uživatele a/nebo jeho členství v některé uživatelské skupině, tedy na základě autentizačních údajů, které uživatel předtím poskytl (uživatelské jméno, heslo, hardwarový klíč apod.). Důležitým prvkem je, že každý vlastník zdroje či informace sám rozhoduje o přidělení přístupových práv ostatním uživatelům zcela podle vlastního uvážení. Nevýhodou je, že administrátoři systému nemohou centrálně a jednotně řídit přístup ke všem informacím či souborem

uloženým na serveru. Tento mechanizmus je běžný například v unixových filesystémech.

2. *model direktivní* -Rozhodování o přidělení práv nezávisí na libovůli jednotlivých uživatelů. Každý uživatel je zařazen na určitou úroveň důvěryhodnosti, která mu byla přidělena administrátorem. Zdrojům a informacím se také přidělují různé úrovně citlivosti. Při přístupu uživatele k některému objektu se úroveň jeho důvěryhodnosti porovnává s úrovní citlivosti daného objektu. O povolení či zamítnutí přístupu je poté rozhodnuto na základě tohoto porovnání. Popsaný mechanizmus se používá v aplikacích extrémně náročných na zabezpečení. Tento model je částečně implementován v aplikaci Inaukce, přičemž zdroje mají přidělené úrovně citlivosti, které nelze měnit.
3. *model založený na rolích* - Přístup ke zdrojům a informacím je založen na rolích a odpovědnostech každého uživatele v rámci organizace nebo uživatelské základny. O přidělení rolí rozhoduje administrátor aplikace. Ten musí být schopen správně rozlišit potřeby jednotlivých rolí a jejich náročnost na zdroje. Zároveň by měl mít na paměti zásadu „Least Privilege“ neboli co nejmenších oprávnění. Každý uživatel by měl mít v aplikaci přístup jen a pouze k těm částem, které skutečně potřebuje využívat, nikoliv k těm, které by někdy potenciálně mohl využít. Tento model je také implementován. Administrátoři mají k dispozici role, které mohou přidělovat uživatelům.

Možnost další vývoje: přístupový kontrolní mechanismus by mohl uvažovat i členství ve skupinách, či mohl být založen například na aktuálním čase, IP adrese či doméně klienta, typu šifrování datového přenosu, počtu přihlášení či dotazů uživatele za poslední den apod.

3.7.10 Entitní autentizace

Session je vytvářena hned při prvním dotazu klienta, bez ohledu na to, zda je uživatel přihlášen či nikoliv. Po úspěšné autentizaci uživatele, kdy je ověřeno zaslané uživatelské jméno a heslo, je po přihlášení, vzhledem k větší bezpečnosti, vytvořen nový session token a jsou zaznamenány informace o přihlášení a uživateli do tabulky *actsessions*.

Prohlížeč pak s každým dalším dotazem zasílá opět jen session token. Aplikace však už ví, který uživatel je s ním asociován – dochází k entitní autentizaci na základě

zaslaného session tokenu. Tak je tomu až do doby, kdy se uživatel odhlásí z aplikace. Po odhlášení je, vzhledem k bezpečnosti, zrušena i celá session.

Hlavním důvodem evidence sessions je možnost navázat na ně některé proměnné a přenášet si jejich hodnoty napříč jednotlivými požadavky uživatele. Tyto proměnné jsou uchovávány v tabulce actsessions, kde klíčem k jejich hodnotám je právě session token. V minulosti se pro tento účel používaly přímo cookies. To však není příliš šťastné řešení, neboť kterýkoliv uživatel může libovolným způsobem ovlivnit obsah zasílaných cookies. Například pokud bychom chtěli mezi různými požadavky přihlášeného uživatele přenášet jeho identifikační číslo nebo rozsah přiřazených oprávnění, může si je případný útočník změnit na jinou hodnotu, zaslat takto modifikovaný požadavek a získat tím jednoduše práva úplně jiného uživatele.

3.7.11 Odhlášení

Mechanizmus sessions a na ně navazovaných přihlášených uživatelů, tak jak byl zatím popsán, má jednu zásadní vadu. Jestliže se totiž uživatel po skončení své práce zapomene explicitně odhlásit, zůstává na serveru otevřená session, navíc ještě s přiřazeným přihlášením uživatele. Případný útočník má tak neomezeně dlouhou dobu, po kterou má čas hrubou silou vyzkoušet postupně všechny možné podoby příslušného session tokenu.

Z tohoto důvodu se zavedl timeout nepoužívaných sessions resp. timeout přihlášení. Aplikace sleduje u každého přihlášeného uživatele čas uplynulý od jeho posledního dotazu na server. Překročí-li určitou hodnotu, je uživatel automaticky odhlášen resp. je jeho session zrušena. Dostatečně krátký timeout v kombinaci s dostatečně dlouhým session tokenem[31] zajistí, že dojde ke zrušení příslušné session dříve, než útočník stihne zaslat i jen zlomek možných tokenů.

V některých případech ale může být timeout přihlášení nepříjemný. Jedná se například o situace, kdy uživatel vybírá předměty z výpisu předmětů. Při vybírání uplyne timeout a systém provede automatické odhlášení. Po odeslání dalšího požadavku, např. na koupi předmětu pak server pošle uživateli zpět login formulář. Pak je pracně nalezený předmět ztracen.

Možnost další vývoje: takové nežádoucí chování aplikace by proto měl framework ošetřit. V principu jde o to nějakým způsobem dočasně uchovat všechna

[31]SHA1 generuje hash řetězec dlouhý znaků.

zaslaná formulářová data a zpracovat je bezprostředně po úspěšném přihlášení do aplikace.

3.8 Předměty a kategorie

Všechny předměty aplikace Inaukce jsou uchovávány v tabulce *subjects*. Tato tabulka obsahuje v každém řádku jeden předmět, jehož veškeré atributy jsou v tomto řádku uloženy.

3.8.1 Obrázky a dokumenty předmětů

Každý předmět smí mít omezený počet obrázků a jeden dokument[32]. Nahrávání souborů od uživatele představuje vysoké bezpečnostní riziko, a proto bylo nutná implementace s maximální opatrností. První dobrou zásadou, která byla praktikována, je používání pro práci s nahranými soubory pole $_FILES a být si vědom různých velikostních limitů.

Další věc, která musela být rozhodnuta, je umístění nahrávaných souborů. V zásadě existují dvě možnosti:

1. *Ukládání souborů do adresáře na disku* – musí se pamatovat na více věcí. Především musí mít do tohoto adresáře právo zápisu webový server. Pokud aplikace běží na sdíleném webhostingu bez správného zabezpečení, musí se počítat s tím, že ostatní uživatelé budou moci soubory z tohoto adresáře mazat a přepisovat. Kroky nutné provádět před uploadem souboru:
   - Zamezení spouštění skriptů v tomto adresáři direktivou engine.
   - Kontrola typu nahrávaného souboru.
   - Rozhodout se, zda soubor ukládat pod původním názvem či novým vygenerovaným.
2. *Ukládání souborů do databáze* – soubory se ukládají do sloupce typu BLOB. Tím se mírně zhoršuje výkon a pro stahování těchto souborů se musí vytvořit vlastní skript. Uložení souborů v databázi je však mnohem bezpečnější.

Ukládání souborů do adresáře je rychlejší, nezhoršuje výkon databáze. Problém však nastává, pokud se rapidně zvětší počet souborů uložených v adresáři (k čemuž může u aplikace Inaukce dojít). Jednak bude při původním uložení názvů souborů docházet k častým kolizí, co je horší, zvýší se zátěž na disk serveru a zaslání souboru klientovi

[32]Důvody byly vysvětleny v kap. 2.12

bude trvat neúnosně dlouhou dobu. Uložení souborů v databázi tento problém řeší díky indexům, proto byla implementována tato varianta.

3.8.2 Kategorie

Z Funkční specifikace vyplývá, že každý předmět musí být zařazen do některé z nabízených kategorií. Kategorie tvoří strom[33]. Tento strom je potřeba nějak uchovávat v MySQL tabulce. MySQL bohužel nepodporuje práci se stromy, tak jak je tomu např. v databázích ORACLE, a proto muselo být implementováno složitější řešení uchovávání jednotlivých uzlů. Možná řešení:

1. *Sebereferenční tabulky* – Nejjednodušším způsobem uložení hierarchické struktury v SQL tabulce jsou tzv. sebereferenční tabulky, definující strom seznamem následníků[34]. Sebereferenční tabulky využívají vazby rodič-syn/dcera v hierarchické struktuře. SQL tabulka by mohla být vytvořena příkazem:

   ```
   CREATE TABLE categories(
     id INT NOT NULL PRIMARY KEY,
     name VARCHAR(32),
     parent INT NOT NULL);
   ```

   Jak je vidět, každý uzel má unikátní identifikátor (číslo id) a má svého rodiče (kořen bude mít NULL). Problém je však s implementací operací se stromovou strukturou. Většina operací musí být implementována na úrovni relační vrstvy při neúnosném zatížení vrstvy datové. Velkého ulehčení zle docílit načtením celého stromu a jeho zpracováním aplikací. To sice ušetří práci databázi, ale nelze ji dost dobře použít při získávání všech předmětů patřících do kategorií podstromu, kterých může být velmi mnoho, apod. Předností sebereferenčních tabulky je jejich jednoduchost - při práci s ní si bohatě se vystačí se znalostí rekurze. Cenou ovšem je neúměrné zatížení databázového serveru zvláště v případě, že se stromem bude pracovat často, což se u internetového obchodu děje obvykle s každou zobrazenou stránkou, či když strom bude hezky košatý a vysoký. Odlehčit lze jistou úrovní cachování stránek, nicméně toto řešení je dočasné a nepříliš výkonné.
2. *Genealogické stromy* - Genealogický strom také využívá vazbu rodič-syn mezi uzly, ale navíc pro každý uzel definuje i tzv. genealogický identifikátor. Tento identifikátor je unikátní pro každý uzel a dají se z něj vyčíst informace o jeho

[33]Strom je neorientovaný graf, jehož každé dva vrcholy jsou spojeny právě jednou cestou.
[34]Toto řešení se používá, pokud existuje podpora databáze pro práci se stromy.

předcích (rodičích, prarodičích, prapra...) i potomcích. Identifikátor potomka totiž získáme tak, že za identifikátor předka připojíme identifikátor potomka. Zvolíme-li za identifikátor písmeno abecedy, pak bude naše rozšířená tabulka obsahovat tyto záznamy:

id	name	parent	path
1	Operační systémy	0	A
2	Unix	1	AA
3	Linux	1	AB
4	Windows	1	AC
5	Red Hat	3	ABA
6	Mandriva	3	ABB

Délka atributu *Path* tak omezuje výšku stromu a zvolená abeceda pak maximální počet potomků každého uzlu. Chceme-li si ušetřit nepříjemnosti s nedostatkem písmen a přepočítáváním po mazání, lze použít jiný identifikátor. V praxi se často používá např. speciální oddělovač následovaný číselnou sekvencí. Identifikátor uzlu Red Hat by byl "/1/3/5&qout;. Operace se stromem jsou pak mnohem rychlejší, přestože se v některých případech musí provádět režie na opravu geneze uzlů (atribut Path).

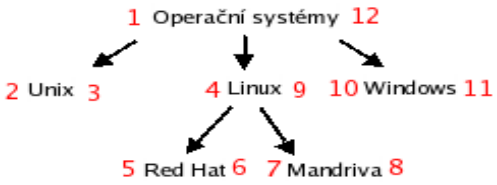
3. *Nested set aneb DFS strom* - Posledním a dle mého názoru nejvýkonnějším a tedy implementovaným řešením je tzv. nested set reprezentace stromu. (Pozn.: Tento název používá Joe Celko [6] a z různých článku se zdá, že není sám.).

   SQL tabulka by mohla být vytvořena příkazem:

```
CREATE TABLE categories(
  id INT NOT NULL PRIMARY KEY,
  name VARCHAR(32),
  parent INT NOT NULL,
  left INT NOT NULL,
  right INT NOT NULL);
```

   Vychází ze sebereferenční tabulky, kterou rozšiřuje atributy left a right. Jejich hodnoty jsou získány průchodem stromu DFS (depth first search) algoritmem. Strom se prochází doleva do hloubky. Časy navštívení a opuštění uzlu se použijí jako hodnoty atributů left resp. right v SQL tabulce.

Interval <left;right> libovolného uzlu je podintervalem intervalu vlastního rodiče (odtud nested set = vnořené množiny). Tato vlastnost plyne z toho, jak DFS prochází strom a ukládá časy a právě ona nám ulehčí práci s hierarchickou strukturou v SQL. DFS strom se zdá být velmi vhodný pro statické, či málo



upravované struktury. Uplatnění si však zajisté najde i v případě potřeby vyhledávání ve velmi rozsáhlých hierarchických strukturách. Všechny operace kromě CRUD jsou extrémně rychlé. CRUD operace provádí pouze administrátor a to jen těsně po instalaci Inaukce nebo velmi zřídka, tedy určité zpomalení nevadí.

3.8.3 Prodej předmětů

Prodej předmětů je implementován pomocí tabulky *subjectsbuyers*. Do této tabulky jsou zaznamenávány veškeré koupě předmětů a provedené příhozy. Tabulka tedy slouží zároveň i pro logování prodeje předmětů. Libovolný předmět může zakoupit uživatel s právem kupujícího uživatele. Předmět je zakoupen v případě bazarového prodeje ihned potom, co první uživatel kupující zakoupí předmět. V případě aukčního prodeje, je předmět zakoupen po vypršení doby ukončení prodej, pokud některý z uživatelů kupujících provedl příhoz na tento předmět. Uživatel, který je držitelem nejvyššího podání se stává kupcem předmětu. Při zakoupení předmětu se zaznamená do tabulky *subjectsbuyers* kupec, předmět, za jakou cenu předmět zakoupil a kdy, stejně tak se děje i při provedení příhozu.

3.8.4 Vyhledávání předmětů

Aplikace Inaukce poskytuje dva typy vyhledávání. Je to rychlé vyhledávání, které je přístupné ze všech stránek aplikace a podrobné vyhledání. Pro provedení požadavku na vyhledání předmětů s uživatelem zadanými vlastnostmi se nejprve vytváří SQL dotaz. Tento SQL dotaz bere v potaz všechny požadavky mezi které vkládá AND. Aplikace Inaukce umožňuje také vyhledávání předmětů obsahujících určitý řetězec znaků ve

jméně předmětu či popisu. Vzhledem k typu MySQL tabulek InnoDB, které nepodporují fulltextové indexy, je velmi náročné vyhledávání vzorku v atributu *description* tabulky subjects, jelikož může být délka řetězce velmi dlouhá. Pro snížení nároků na databázi by mohlo být zakázáno hledání shody se vzorkem v tomto atributu.

3.9 Monitoring a statistika

Monitoring systému slouží pro uživatele jako zdroj statistik, pocházejících ze sledování systému z dlouhodobějšího hlediska nebo zprostředkovává informace o aktuálním stavu systému. Nabízí přehledný popis statistik o předmětech v databázi a uživatelích. Pro monitoring a statistiku se používají následující tabulky:

- Z tabulky *subjects* se dají vyčíst informace o počtu předmětů celkem, v jednotlivých stavech, zakoupených za určitou dobu. Společně s tabulkou *subjectsbuyers* tvoří log kupování předmětů.
- Z tabulky *users* se dají vyčíst informace o počtu uživatelů, uživatelů s danými právy.
- Z tabulky *actsessions* se dají vyčíst informace o počtu uživatelů, kteří jsou online a jací to jsou a jaká mají práva (tabulka *users*).
- Z tabulky *accessapplicationstatistics* se dají vyčíst informace o přístupů uživatelů do systému. Tato tabulka má pouze statistickou a logovací funkci.
- Z tabulky *accesspagestatistics* se dají vyčíst informace o přístupů uživatelů k jednotlivým stránkám aplikace. Tato tabulka má pouze statistickou a logovací funkci.

3.10 Zasílání e-mailových zpráv

Aplikace Inaukce má jako svůj atribut kvality zasílání e-mailů v následujících případech:

- Ověření e-mailové adresy při registraci.
- Zasílání informativních e-mailů kupujícím uživatelům, jejichž podání bylo přehozeno.
- Zasílání informativních e-mailů prodávajícím uživatelům, jejichž předmět byl zakoupen.

Implementace zasílání e-mailů používá PHP funkci *mail()*, která používá pro svou činnost nastavený smtp server[35]. Při odeslání e-mailů je kontrolován úspěch zaslání zprávy.

[35]Přímo pro platformy Windows, či nepřímo pro UNIXové platformy a další.

Čím může být způsobeno neúspěšné odesílání?

1. funkce *mail()* je zakázaná - Bohužel je funkce *mail*() zakázána zejména na free serverech pro které je aplikace Inaukce primárně určena. Administrátor však může v tomto případě vypnout zasílání e-mailů.
2. je špatně nastavený smtp server (v php.ini na win-serverech) – Tento případ je řešen v Instalační příručce.
3. Chybný *mail démon* nebo je přetížená síť – v tomto případě neexistuje jednoduché řešení.

3.11 Zabezpečené připojení pomocí SSL

SSL je vrstva/protokol zabezpečující data na přechodu mezi aplikační a transportní vrstvou (protokolem TCP/IP). Lze zajistit šifrování přenášených dat a autentizaci serveru pomocí digitálních certifikátů. Pro použití SSL je třeba mít na straně serveru nainstalovanou podporu SSL, což bohužel většina free serverů nemá. Také jej musí podporovat klientův prohlížeč, což v současnosti podporují téměř všechny. To, že se klient připojil na webové stránky zabezpečené pomocí SSL, pozná podle adresy (obsahuje navíc písmeno s, např. https://server.cz) nebo podle indikace prohlížeče. Obyčejně je zabezpečení přenosu indikováno ikonou zamčeného zámku ve stavovém řádku okna prohlížeče. V závislosti na nastavení prohlížeč informuje o přechodu mezi zabezpečeným a nezabezpečeným režimem. Pokud má klient upozorňování zapnuto, objeví se dialogové okno, ve kterém může případně odmítnout přechod do nezabezpečeného režimu. V zásadě je lhostejné, kdo s aplikací Inaukce pracuje, jen chceme zabezpečit, aby informace, které si klient s aplikací vymění (přihlášený uživatel) nemohly uniknout, certifikát pro klienta proto není potřebný. Při navázání spojení se serverem přes https nabídne prohlížeč k použití certifikát serveru (ten musí být k dispozici, zajistí to správce serveru) a žádný další není potřebný.

Kapitola 4

Dokumentace

4.1 Funkční specifikace

Funkční specifikace zachytává požadavky zadavatele softwarového díla specifikované v průběhu analýzy a návrhu aplikace. Na základě Funkční specifikace by měla vzniknout Technická specifikace a následně samotné softwarové dílo. V závěrečných fázích vývoje aplikace slouží Funkční specifikace k ověření splnění požadavků zadavatele a k tvorbě funkčních testovacích scénářů.

Funkční specifikace je psaná v trpném rodu formálním jazykem. Funkční specifikace slouží uživateli, který nemusí být vzdělaný v oblasti informačních technologií, proto nezachází do technických detailů a pokud možno vyhýbá se používání příliš odborných pojmů.

Požadavkem zadavatele na funkční specifikaci, bylo vypracování některých UML diagramů. Jsou to Diagramy hierarchie procesů, Diagramy procesních vláken a Diagramy užití společně se jejich některými scénáři. Pro autora to bylo první seznámení s modelovacím jazykem UML (hlavní zdroj [5]).

4.2 Uživatelská příručka

Uživatelská příručka popisuje softwarové dílo z uživatelského hlediska. Cílem uživatelské příručky je co možná nejsrozumitelnějším způsobem seznámit uživatele s aplikací. Obsahuje popis všech vlastností aplikace z hlediska normálního uživatele[36].

Pro dodržení formálního stylu a zároveň pro snadnější pochopení je Uživatelská příručka napsána v er-formě. Pro názornost a možnost čtení bez přístupu k pracovní stanici s nainstalovanou aplikací Inaukce je text příručky doplněn o ukázkové obrázky většiny stavů aplikace.

4.3 Instalační příručka

Instalační příručka popisuje postup instalace softwarového díla a doplňujících součástí a popis potřebných nastavení systémových prostředí a dalších aplikací. Instalační příručka dále obsahuje popis kroků, které je nutno provést před prvním spuštěním aplikace.

[36]Anonymního, kupujícího či prodávajícího uživatele.

Instalační příručka aplikace Inaukce je chápaná jako doplnění Uživatelské příručky, avšak pro snadnější pochopení je psána ve formě učebnice.

4.4 Administrátorská příručka

Administrátorská příručka je chápána jako doplnění Uživatelské příručky. Obsahuje popis všech vlastností aplikace z hlediska administrátora[37].

4.5 Programátorská příručka

Programátorská příručka vysvětluje principy, na základě kterých aplikace vnitřně funguje. Popisuje použité technologie, algoritmy, logiku rozdělení zdrojových kódů do skriptů a dále do tříd, logiku názvosloví identifikátorů apod.

Programátorská příručka je psaná formálním jazykem a v trpném rodu. Časté je používaní anglických výrazů a někdy i informatického žargonu, což je způsobeno příliš rychlým vývojem informačních technologií, na který spisovná čeština nedokáže dostatečně rychle reagovat.

[37]Administrátora či superusera.

Kapitola 5

Závěr

5.1 Praktické zkušenosti s aplikací

Při vyvíjení Inaukce bylo nutné tuto aplikaci testovat. To lze v zásadě dvěma způsoby:

1. Nahrát aplikaci na ostrý Internetový server a testovat ji přímo na tomto serveru. Tento způsob vyžaduje stálé připojení k Internetu, nutnost zálohování dat a přináší problémy s samotným testováním aplikace.
2. Vytvořit si domácí testovací server, na kterém lze simulovat provoz jako na ostrém serveru pomocí shodného softwaru a nastavení. Tato varianta byla upřednostněna.

Pro instalaci softwaru byl vybrán balíček Xampp[38] pro platformu Windows, který obsahuje server Apache, modul PHP, databázi MySQL a další aplikace. Výhodou tohoto balíčku je vzájemná nakonfigurovanost těchto tří programů. Programy je nutné nakonfigurovat tak, aby simulovaly prostředí na předpokládaném ostrém serveru. K serveru se přistupuje pomocí adresy http://localhost či http://127.0.0.1.

Jelikož je Inaukce multiuživatelský systém, bylo potřeba aplikaci otestovat také v ostrém provozu s více uživateli. Zde však nastal problém s výběrem hostingového serveru. Inaukce je určena primárně pro free servery, a proto bylo nutné nalézt vhodný server, který by pokryl všechny požadavky aplikace Inaukce. Mezi hlavní požadavky Inaukce patří databáze MySQL a možnost nastavení InnoDB typu MySQL tabulek, skriptování PHP a zpřístupněné používání PHP funkce *mail*(), podpora SSL na straně serveru.

Průzkum free hostinogových serverů:

server	***InnoDB tabulka***	***Mail()***	***SSL***
Webzdarma.cz	NE	NE	NE
Host.sk	ANO	NE	NE
Ic.cz	NE	ANO	ANO
Fbi.cz	ANO	NE	NE
Php5.cz	NE	ANO	NE
Xnet.cz[39]	ANO	ANO	NE

[38]Existují i další jako PHPTriad, PHP Home, EasyPHP.
[39]Pouze domény 2.řádu – platí se roční poplatek.

Jak vyplývá z tabulky, na našem trhu neexistuje free hostingový server, který by podporoval všechny tři požadavky. Nejzásadnější je z nich podpora InnoDB tabulek, jelikož mimo jiné bez ní neplatí referenční vztahy mezi cizími klíči, které jsou důležité při DELETE příkazech. Nakonec byl pro ostrý provoz zvolen server Host.sk pro testování CRUD příkazů a server Ic.cz pro testovnání zabezpečení a zasílání e-mailů.
Jak se bude chovat aplikace Inaukce ve skutečnosti by ukázalo až její praktické nasazení a používání na serveru podporující všechny tři hlavní požadavky.

Pro otestování aplikace byl vytvořen SQL soubor Insert_test_data.sql, který obsahuje testovací data do databáze MySQL pro Inaukci.

5.2 Splnění cíle

V rámci této bakalářské práce byla vytvořena aplikace Inaukce sloužící jako internetová aukce či bazar. Uživatelé přistupují k aplikaci prostřednictvím webového prohlížeče v různých rolích. Jednotlivé role se liší uživatelskými právy. Po přihlášení do systému mají uživatelé s příslušnými právy možnost nabízet své předměty. Prodávané předměty lze snadno prohlížet podle různých kritérií. Uživatelé s příslušnými právy pak mají možnost kupovat předměty za pevnou cenu (bazarový prodej) nebo se pokusit koupit daný předmět v dražbě, tedy nabídnout za daný předmět nejvyšší sumu peněz (aukční prodej). K dispozici je také role administrátora s možností správy účtů a předmětů, administrace a monitoringu systému.

Uživatelské rozhraní aplikace bylo navržené tak, aby co nejméně odpoutávalo klienta od nákupu a aby se dosáhlo maximální uživatelské přívětivosti.

Inaukce je původně koncipována pro nasazení na free serverech, může však být nasazena i pro komerční využití.

Vývoj aplikace probíhal dle standardních postupů. Nejdříve byla vytvořena ve spolupráci se zadavatelem podrobná Funkční specifikace, na základě které byla naprogramována výsledná aplikace Inaukce, napsána dokumentace, konkrétně Uživatelská příručka, Administrátorská příručka, Instalační příručka, Programátorská dokumentace, Referenční příručka, která je součástí Programátorské dokumentace a Technická dokumentace. Projektová dokumentace je nahrazena rozsáhlejší Bakalářskou prací, která zastřešuje celé dílo.

Cílem této bakalářské práce bylo rozšířit si znalosti o Internetových obchodech a aukcích a dokázat takovou aukci i vytvořit. Cíl byl splněn, finální stav aplikace Inaukce odpovídá Funkční specifikaci. Drobné odlišnosti jsou výsledkem nepřesné

specifikace procesů, které vyplývají z autorovy neznalosti UML, jehož některé diagramy byly ve specifikaci zadavatelem požadovány. Některé kapitoly Bakalářské práce obsahují možnosti dalšího vývoje aplikace, které se v průběhu implementace objevily a z důvodů časové tísně nebyly implementovány.

Literatura

[1] Vlastimil Klíma:
Tunnels in Hash Functions: MD5 Collisions Within a Minute, 2005

[2] MySQL AB: MySQL Reference Manual, WWW
http://dev.mysql.com/doc/

[3] Jesus Castagneto, Harish Rawat …:
PHP Programujeme profesionálně, 2004

[4] Frank Boumphrey, Cassandra Greer …:
XHTML Průvodce vývojáře, 2002

[5] Hana Kanisová, Miroslav Müller:
UML srozumitelně, 2004

[6] Joe Coelko:
Joe Celko's Trees and Hierarchies in SQL for Smarties, 2004

[7] DBDesigner. WWW,
http://www.fabforce.net/

[8] Dia. WWW,
http://www.gnome.org/projects/dia

[9] PsPad, WWW,
http://www.pspad.com

[10] Akon CZ. WWW,
http://www.akon.cz

[11] Zákon č. 26/2000 Sb. o veřejných dražbách, WWW,
http://www.drazebnici.cz/cad-scripts/cadweb.exe/include?file=clanek19.inc

[12] Průvodce zajímavým světem autentizace, WWW,
http://www.zive.cz/h/Uzivatel/AR.asp?ARI=121526

[13] Levé menu, WWW,
http://www.jakpsatweb.cz/leve-menu.html

[14] Jakub Vrána: ukládání hesel, WWW, 2005,
http://php.vrana.cz/ukladani-hesel

[15] Bill Cunningham: Web Development with PHP 4.0 and FastTemplate 1.1.0, WWW, 2001,
http://www.linuxjournal.com/article/4573

[16] Třívrstvá architektura, WWW,
http://www.sweb.cz/pichlik/archive/2004_11_07_archive.html
[17] Srovnání databází MySQL vs PostgreSQL vs Firebird, WWW,
http://www.root.cz/clanky/mysql-vs-postgresql-vs-firebird/

Příloha

Obsah CD-ROM

K této práci je přiložen CD-ROM, na kterém jsou uloženy zdrojové soubory aplikace, textové a html soubory dokumentací a bakalářské práce a repository obsahující vývoj celé práce.

Adresář/soubor	*Obsah*
BakalarskaPrace.pdf	Soubor obsahuje Bakalářskou práci
FunkcniSpecifikace.pdf	Funkční specifikace
Prilohy.pdf	Soubor obsahuje přílohy k Bakalářské práci: Uživatelskou dokumentaci, Administrátorskou dokumentaci, Instalační příručku a Programátorskou dokumentaci.
TechDoc/index.html	Technická dokumentace
/Doc	Adresář obsahuje Dokumentaci a Bakalářskou práci v samostatných souborech
/Inaukce	Adresář obsahuje zdrojový kód aplikace
/Repository	SVN Repository celé práce.
/Xampp	Distribuce Apache, PHP, MySQL
/PHPDocumentor	Distribuce dokumentačního nástroje PHPDocumentor