Grafické rozhraní pro práci s formuláři přes internet
Graphic interface for working with forms placed on Internet

Bc. Marek Kojecký

Diplomová práce
2007

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav aplikované informatiky
akademický rok: 2006/2007

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Marek KOJECKÝ**
Studijní program:	**N 3902 Inženýrská informatika**
Studijní obor:	**Informační technologie**
Téma práce:	**Grafické rozhranní pro práci s formuláři přes internet.**

Zásady pro vypracování:

1. Vytvořte univerzální webovou aplikaci v PHP5, která bude umožňovat generování a správu formulářů.
2. Při tvorbě aplikace využijte objektový návrh a technologii AJAX. Generované formuláře ukládejte do databáze MySQL. Pro úpravu grafického rozhraní použijte CSS.
3. Webové rozhraní pro vyplňování formulářů bude umožňovat tisk a export. Vyplněné údaje se budou ukládat do databáze MySQL.
Aplikace by měla plně nahradit papírové verze formulářů.

Rozsah práce:

Rozsah příloh:

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

Seznam odborné literatury:

GILMORE, W. Jason. Velká kniha PHP5 a MySQL : kompendium znalostí pro začátečníky i profesionály. RNDr. Jan Pokorný. 1. vyd. Brno : Zoner Press, 2005. 711 s. ISBN 80-86815-20-X. SCHLOSSNAGLE, G. Pokročilé programování v PHP 5. ZonerPress, 2005. ISBN 80-86815-14-5 Prokop, M.: CSS – kaskádové styly pro webdesignery. Mobil Media, 2003. Maslakowski, M.: Naučte se MySQL za 21 dní. ComputerPress, 2001.

Vedoucí diplomové práce: **Ing. Martin Sysel, Ph.D.**
Ústav aplikované informatiky

Datum zadání diplomové práce: **13. února 2007**

Termín odevzdání diplomové práce: **28. května 2007**

Ve Zlíně dne 13. února 2007

prof. Ing. Vladimír Vašek, CSc.
děkan

doc. Ing. Ivan Zelinka, Ph.D.
ředitel ústavu

ABSTRAKT

Práce pojednává o webových aplikacích pro tvorbu a následnou správu formulářů. V teoretické části popisuje obecně webové aplikace, použité webové technologie a nejpoužívanější formulářové prvky v jazyce HTML. Praktická část je ještě rozdělena na dvě části pohledu. První pohled je z hlediska programátora a druhý z hlediska běžného uživatele. Praktická část popisuje funkčnost celé aplikace. Diplomovou práci uzavírají náhledy na aplikaci z pozic administrátora a běžného uživatele.

Klíčová slova: elektronické formuláře, webová aplikace, tvorba formuláře, formulář

ABSTRACT

This work deals with web applications for creation and succesive administration of forms. The theoretical part describes web applications, used web technologies and most used forms elements of language HTML in general. The practical part of this work is separated into two parts of view. First is programmer view, second casual users. Practical part describes utility of whole application. Diploma work is closed with point of view on the application of the administrators and casual user position.

Keywords: electronic forms, web application, form creation, form

Rád by jsem touto cestou poděkoval vedoucímu diplomové práce Ing. Martinu Syslovi, Ph.D. za odborné vedení, rady a pomoc při řešení problémů souvisejících s danou prací.

Prohlašuji, že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků, je-li to uvolněno na základě licenční smlouvy, budu uveden jako spoluautor.

Ve Zlíně …….

Podpis diplomanta

OBSAH

ÚVOD

Zatímco devatenácté století bylo nazýváno *„Stoletím páry"*, dobu, v níž právě žijeme, lze směle nazvat *„Stoletím formulářů"*.[7] Kromě mateřského prsu je tím prvním, s čím se v životě setkáte, právě formulář. Rodný list. A pak se s různými formuláři „roztrhne pytel" - zápis do základní školy, vysvědčení, žádost o vydání občanského průkazu, přihláška na střední školu, přihláška na vysokou školu, daňové přiznání a tak dále a tak podobně. Stovky a tisíce formulářů se vinou naším životem jako pověstná červená nit. Nakonec jediným formulářem, se kterým málokdo z nás přijde osobně do styku, je úmrtní list. S trochou škodolibosti lze konstatovat, že hlavním smyslem lidského bytí je vyplnit co nejvíce formulářů.

Je pochopitelné, že v civilizaci, která je posedlá sbíráním informací, nemohl zůstat stranou ani jedinečný vynález lidského ducha - web.

Stejně jako v běžném životě tak i ve virtuálním elektronickém světě existují dva druhy formulářů. Jednoduché a pochopitelné, jejichž klasickým představitelem je složenka typu A a pak ty náročnější.

Hlavním úkolem formulářů na webu je především umožnit uživateli hledanou informaci v záplavě webových stránek vůbec najít. K tomu slouží vyhledávací formuláře. Ty jsou, kromě vzájemné provázanosti dokumentů (čili hypertextových odkazů), hlavní hybnou silou internetu. Ale slouží také ke vzájemné komunikaci mezi provozovatelem webu a jeho návštěvníkem. Například vlastníkům elektronických obchodů umožňuje vydělávat peníze.

Na jedné straně stojí provozovatelé webů a jejich marketingoví odborníci, kteří by nejraději věděli všechno o všech. Na druhé straně se pohybuje většinou bezejmenný dav uživatelů, kteří by nejraději neřekli nikomu nic. Mezi těmito mlýnskými kameny je uvězněn nešťastný webdesigner, jehož téměř nadlidským úkolem je vymyslet formulář tak, aby uspokojil požadavky a přání těch prvních a přitom nevzal radost ze života těm druhým.

I. TEORETICKÁ ČÁST

1 WEBOVÉ APLIKACE

Webové aplikace jsou takové, které jsou zpracovány na webovém serveru, a jež se k uživateli dostanou prostřednictvím internetu nebo intranetu. Tito uživatelé používají ke spuštění webových aplikací nenáročného klienta (webový prohlížeč), který umí data přijatá ze serveru zobrazit a zpracovat. Desktopové aplikace jsou oproti tomu založené na robustních klientech, které provádějí většinu zpracování.

Webové aplikace vzbuzují dojem, že jednoho dne budou mít stejný vzhled a chování, jako jejich vyspělí (a výkonní) příbuzní ze světa desktopu. Způsob, jakým se chová počítačový software, který komunikuje s lidmi, je v dnešní době mnohem více významnější než dříve – uživatelé počítačů jsou totiž různorodí lidé, zatímco v minulosti se hlavně jednalo o technicky zdatné jedince. Dnes je potřeba připravit dobře vypadající zprávu pro vedoucí prodeje a obchodníkovi zase snadno použitelné formuláře pro zadávání dat.

Protože je spokojenost koncových uživatelů jediné, na čem záleží, musí vámi vyvíjené softwarové aplikace vyhovovat všem, kteří s nimi budou pracovat. Proces dospívání aplikací bude u konce až tehdy, když z jejich rozhraní a chování nebude zřejmé, je-li funkčnost dána lokálně nebo vzdáleně. Přenos uživatelských rozhraní prostřednictvím webu bývá problematický jednoduše proto, že prvky, které uživatelé používají ve svých desktopových aplikacích – například přetahování pomocí myši nebo současné řešení více úkolů v jednom okně – nejsou možné.

Dalším problémem při tvorbě webových aplikací je standardizace. Aby se zajistila funkčnost stránky pro většinu návštěvníků, musí být prověřena alespoň ve dvou nebo třech různých prohlížečích.

1.1 Přínosy webových aplikací

Přenášet funkcionalitu aplikace prostřednictvím webu samozřejmě přináší mnoho starostí. Proč se tím tedy trápit a nevytvářet raději jednoduché desktopové aplikace? I se současnými problémy s přívětivostí pro uživatele se totiž webové aplikace staly mimořádně populárními díky mnoha přínosům v technologiích.

- **Webové aplikace je jednoduché a levné dodat uživatelům.** S jejich pomocí mohu společnosti snížit náklady na IT, které spočívají v instalaci software na počítačích jednotlivých uživatelů. Vše, co uživatel pracující s webovou aplikací

potřebuje, je počítač s webovým prohlížečem a spojením do internetu nebo intranetu.

- **Webové aplikace je jednoduché a levné aktualizovat.** Náklady na údržbu software byly vždy významné. Protože se aktualizace stávající části kódu dosti podobá nové instalaci, jsou zde přínosy webových aplikací, zmíněné výše, také významné. Každý uživatel může mít k dispozici novou verzi aplikace chvilku poté, co byla aktualizována na serveru.
- **Webové aplikace mají flexibilní požadavky na uživatele.** Máte-li svou aplikaci instalovanou na serveru – na jakémkoli moderním operačním systému – můžete ji prostřednictvím internetu nebo intranetu používat na jakémkoli stroji, na kterém běží Mac, Windows, nebo Linux. Je-li tato aplikace správně vytvořena, bude dobře fungovat ve kterémkoliv moderním webovém prohlížeči, ať už se jedná o Internet Explorer, Mozilla Firefox, Opera nebo Safari.
- **Webové aplikace umožňují snazší centralizovanou správu dat.** Je-li potřeba přistupovat ke stejným datům z několika různých míst, je jednodušší mít všechna data uložena na jednom místě, než v několika oddělených databázích. Takto lze předejít možným rizikům a problémům se synchronizací a bezpečností. [4]

2 WEBOVÉ TECHNOLOGIE

2.1 HTML

Název **HTML** je zkratkou od *HyperText Markup Language* – textový značkovací jazyk. Slovo *HyperText* zde vyjadřuje možnost vzájemně propojovat texty na základě odkazů, *Markup* označuje schopnost jazyka HTML dávat významy jednotlivým blokům textu s pomocí speciálních značek nazývaných tagy a elementy (např. vypsat část textu tučně nebo ji třeba určit jako nadpis). [12]

HTML je původní jazyk, který se ještě dnes v některých případech používá k vytváření základní obsahové kostry webových stránek. Dříve jazyk HTML sloužil i k formátování vzhledu (v současnosti se k tomu kvůli zachování přístupnosti webu používají kaskádové styly, které umožňují vytvářet vzhled jako druhou, na obsahu nezávislou vrstvu).

Jazyk HTML patří do široké rodiny značkovacích jazyků SGML. Vznikl v roce 1990 ve Švýcarsku a postupně se vyvíjel v závislosti na nejpoužívanějších prohlížečích až k současné verzi HTML 4.01, u které byl vývoj ukončen, neboť na ni navazuje modernější jazyk XHTML.

2.1.1 Syntaxe

Jazyk HTML je charakterizován množinou značek a jejich atributů pro danou verzi definovaných. Mezi značky se uzavírají části textu dokumentu a tím se určuje význam (*sémantika*) obsaženého textu. Názvy jednotlivých značek se uzavírají mezi úhlové závorky („<“ a „>“). Část dokumentu uzavřená mezi značkami tvoří tzv. **element** (prvek) dokumentu. Součástí obsahu elementu mohou být další vnořené elementy. Atributy jsou doplňující informace, které upřesňují vlastnosti elementu. [13]

Ukázka syntaxe

```
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
2 <html>
3 <!-- toto je komentář -->
4    <head>
5       <title>Titulek stránky</title>
6    </head>
7 <!-- tělo dokumentu -->
```

```
 8    <body>
 9       <h1>Nadpis stránky</h1>
10       <p>Toto je tělo dokumentu</p>
11    </body>
12 </html>
```

2.2 CSS

CSS je zkratka pro anglický název *Cascading Style Sheets*, česky **tabulky kaskádových stylů**. Je to jazyk pro popis způsobu zobrazení stránek napsaných v jazycích HTML, XHTML nebo XML.

Jazyk byl navržen standardizační organizací W3C.

Hlavním smyslem je umožnit návrhářům oddělit vzhled dokumentu od jeho struktury a obsahu. Původně to měl umožnit už jazyk HTML, ale v důsledku nedostatečných standardů a konkurenčního boje výrobců prohlížečů se vyvinul jinak. Starší verze HTML obsahují celou řadu elementů, které nepopisují obsah a strukturu dokumentu, ale i způsob jeho zobrazení. Z hlediska zpracování dokumentů a vyhledávání informací není takový vývoj žádoucí. [9]

2.2.1 Syntaxe

Stylový předpis se skládá z posloupnosti pravidel. Každé pravidlo určuje vzhled některého elementu dokumentu, nebo skupiny elementů. Pravidlo začíná tzv. selektorem, který specifikuje („adresuje“) skupinu elementů. Selektor je následován seznamem deklarací, které určují vzhled vybrané skupiny elementů. Celý seznam je uzavřen ve složených závorkách a jednotlivé deklarace jsou odděleny středníkem (tj. za poslední deklarací středník už být nemusí). [10]

Ukázka syntaxe

```
1 h1 {                  /* vzhled nadpisu první úrovně */
2     margin: 5px;      /* okraj šířky 5 pixelů        */
3     font-size: 12pt   /* velikost fontu 12 bodů      */
4    }
5 p  {                    /* styl odstavce       */
6     text-align: center;  /* text centrovat      */
7     line-height: 10pt;   /* výška řádku 10 bodů */
8    }
```

2.2.2 Výhody

Používání kaskádových stylů oproti samotnému HTML přináší v praxi řadu výhod:

- **rozsáhlejší možnosti** – CSS nabízí rozsáhlejší formátovací možnosti než samotné HTML. Např. pro formátování bloku textu – tj. určení vzdálenosti od jejich elementu či okraje stránky nenabízí HTML nic. CSS má vlastnosti padding a margin. V HTML by bylo potřeba vytvořit složitou konstrukci vnořených tabulek.
- **konzistentní styl** – na všech stránkách webové prezentace by měly být všechny nadpisy stejné úrovně, seznamy, zdůrazněné části textu apod. stejného stylu. S použitím formátovacích možností HTML je to obtížné – u každého objektu v každém dokumentu se vzhled objektu stále znovu nastavuje. S použitím CSS je to velmi jednoduché. Vytvoří se soubor stylu, který se připojuje k HTML dokumentu. Ve všech dokumentech jsou pak objekty stejného vzhledu.
- **oddělení struktury a stylu**
- **dynamická práce se styly** – provést změnu stylu webu, který pro formátování vzhledu využívá jen možnosti HTML, znamená najít a nahradit všechny značky a změnit atributy mnoha dalších značek. V případě používání CSS znamená změna stylu webu přepsání jediného souboru – souboru stylů. Internetový prohlížeč Mozilla Firefox dokonce nabízí rozšíření, které umožňuje editovat CSS a přitom sledovat změny ve vzhledu stránky v reálném čase a to bez nutnosti obnovovat obsah stránky (např. pomocí tlačítka F5).
- **formátování XML dokumentů**
- **větší kompatibilita alternativních webových prohlížečů**
- **kratší doba načítání stránky**

2.2.3 Nevýhody

Hlavní nevýhodou CSS je zatím stále špatná podpora v majoritních prohlížečích. Různé prohlížeče interpretují stejný CSS kód jinak a je někdy velmi obtížné jej napsat tak, aby se na všech (resp. na několika vybraných) prohlížečích výsledek zobrazil stejně.

2.3 PHP

PHP (rekurzivní zkratka *PHP: Hypertext Preprocessor*, „PHP: Hypertextový preprocesor“, původně *Personal Home Page*) je skriptovací programovací jazyk, určený především pro programování dynamických internetových stránek. Nejčastěji se začleňuje přímo do struktury jazyka HTML, XHTML či WML, což je velmi výhodné pro tvorbu webových aplikací. PHP lze ovšem také použít i k tvorbě konzolových a desktopových aplikací.

PHP skripty jsou prováděny na straně serveru, k uživateli je přenášen až výsledek jejich činnosti. Syntaxe jazyka kombinuje hned několik programovacích jazyků (Perl, C, Pascal a Java). PHP je nezávislý na platformě, skripty fungují bez úprav na mnoha různých operačních systémech. Obsahuje rozsáhlé knihovny funkcí pro zpracování textu, grafiky, práci se soubory, přístup k většině databázových serverů (mj. MySQL, ODBC, Oracle, PostgreSQL, MSSQL), podporu celé řady internetových protokolů (HTTP, SMTP, SNMP, FTP, IMAP, POP3, LDAP, …)

PHP se stalo velmi oblíbeným nástrojem především díky jednoduchosti použití a tomu, že kombinuje vlastnosti více programovacích jazyků a nechává tak vývojáři částečnou svobodu v syntaxi. S verzí PHP 5 se výrazně zlepšil přístup k objektově orientovanému programování podobný Javě. [5]

2.4 MySQL

MySQL je databázový systém, vytvořený švédskou firmou MySQL AB. Jeho hlavními autory jsou Michael „Monty“ Widenius a David Axmark. Je považován za úspěšného průkopníka dvojího licencování – je k dispozici jak pod bezplatnou licencí GPL, tak pod komerční placenou licencí.

MySQL je multiplatformní databáze. Komunikace s ní probíhá – jak už název napovídá – pomocí jazyka SQL. Podobně jako u ostatních SQL databází se jedná o dialekt tohoto jazyka s některými rozšířeními. [14]

Pro svou snadnou implementovatelnost (lze jej instalovat na Linux, MS Windows, ale i další operační systémy), výkon a především díky tomu, že se jedná o volně šiřitelný software, má vysoký podíl na trhu v současné době používaných databázích. Velmi oblíbená a často nasazovaná je kombinace MySQL, PHP a Apache jako základní software webového serveru.

MySQL bylo od počátku optimalizováno především na rychlost, a to i za cenu některých zjednodušení: má jen jednoduché způsoby zálohování, a až donedávna nepodporovalo pohledy, triggery, a uložené procedury. Tyto vlastnosti jsou doplňovány teprve v posledních letech, kdy začaly nejčastějším uživatelům produktu – programátorům webových stránek – již poněkud scházet. [15]

2.4.1 Syntaxe

Syntaxe SQL není nijak složitá, je zvolena tak, aby připomínala anglický jazyk.

Ukázka syntaxe

```
1 create table dluznici
2   ( prijmeni varchar(25),
3     jmeno varchar(25),
4     dluzi char(1)
5   );
```

2.5 JavaScript

JavaScript je programovací jazyk, jehož kód se vkládá do internetových stránek. Příkazy tohoto jazyka jsou pak provedeny na straně klienta a ne na straně serveru, jak je tomu například u PHP. Tento systém má výhody i nevýhody. Výhodou je mnohem menší zatěžování serveru, na kterém jsou stránky umístěny, a prostředí klienta, které je odlišné od prostředí serveru a lze na něm provádět i věci, které na straně serveru nelze.

JavaScript se většinou používá pro úpravu stránek. Programy vytvořené v JavaScriptu mohou na stránce dělat různé věci: zobrazit datum, změnit polohu okna prohlížeče, otevřít okno nové, provést přesměrování, rozpoznat prohlížeč, provést kontrolu zadaných údajů, atd...

JavaScript však není jediný skriptovací jazyk, který se používá. Největším jeho konkurentem je asi určitě **VBscript**, ten však rozhodně není tak rozšířen. [11]

2.5.1 Syntaxe

Skript (program) JavaScriptu lze vložit do HTML dokumentu dvěma způsoby. Prvním a nejobvyklejším způsobem zapsání programu přímo do souboru obsahující HTML značky mezi značky <SCRIPT> a </SCRIPT>. Jeden dokument HTML může obsahovat i několik

párů těchto značek. Příkazy těchto skriptů se pak provedou podle toho, jak jdou za sebou v pořadí. Přestože je skriptů několik, ve skutečnosti jsou součástmi jednoho programu, takže je možné se v jednom skriptu odkazovat na proměnné a metody definované ve skriptu jiném. Skripty mohou být zapsány v části <HEAD> nebo <BODY> HTML dokumentu. Na umístění skriptu je třeba dát pozor, neboť není například možné manipulovat skriptem s formulářem, který je definován v dokumentu až po skriptu. Naopak například některé vlastnosti objektu Document je třeba nastavit v oddílu <HEAD>, ještě před tím, než prohlížeč začne dekódovat obsah dokumentu v oddíle <BODY>.

Ukázka syntaxe

```
1 <SCRIPT LANGUAGE="JavaScript">
2 document.write("Pouzivate prohlizec: "+navigator.appName);
3 </SCRIPT>
```

2.6 AJAX

AJAX je zkratkou termínu Asynchronous JavaScript and XML (asynchronní JavaScript a XML). Jednoduše řečeno – Ajax může být chápán jako „vylepšený JavaScript“, protože jeho podstata spočívá v tom, že JavaScriptu na straně klienta je umožněno volat v pozadí server a podle potřeby tak získat potřebná data. Tímto způsobem je možné aktualizovat některé části stránky bez nutnosti opětovně načítat stránku.

Ajax slouží pro dosažení lepší rovnováhy mezi aktivitami serveru a klienta při provádění akcí, požadovaných uživatelem. Dosud byly tyto dvě aktivity pokládány za naprosto oddělené, protože v reakci na akce uživatele pracovala vždy jedna z nich. Ajax přichází z řešením pro vyrovnání zátěže mezi klientem a serverem tak, že jim umožní komunikovat na pozadí, zatímco uživatel pracuje se stránkou.

Ajax je tedy o vytváření všestranných a interaktivních webových aplikací takovým způsobem, že během práce uživatele je stránce umožněno asynchronně volat server. Ajax je nástroj, který mohou vývojáři použít k vytváření inteligentnějších webových aplikací, jež se při komunikaci s lidmi chovají lépe.

Technologie AJAXu je implementována ve všech moderních webových prohlížečích, jako jsou Mozilla Firefox, Internet Explorer a Opera, takže k prohlížení webových stránek založených na AJAXu není potřeba instalovat žádné další moduly.

AJAX se skládá z následujících prvků:

- JavaScript je základní složkou AJAXu, která umožňuje budovat funkcionalitu na straně klienta. Ve svých funkcích JavaScriptu budete k manipulaci s částmi HTML stránky používat Dokument Object Model (DOM).
- Objekt *XMLHttpRequest* umožňuje JavaScriptu asynchronně komunikovat se serverem, tak, že zatímco komunikace probíhá na pozadí, může uživatel pokračovat v práci. Tato komunikace jednoduše znamená vytvoření http požadavku na soubor nebo skript, umístěný na serveru. Tyto požadavky se tvoří jednoduše a nezpůsobují žádné problémy s firewallem.
- Na straně serveru je potřebná technologie ke zpracování požadavků, které přichází od klienta.

Pro komunikaci mezi klientem a serverem potřebují obě strany způsob, jak vkládat data a jak jim porozumět. Vkládání dat je ta jednodušší část. Klientský skript, který přistupuje k serveru (pomocí objektu XMLHttpRequest) může prostřednictvím metod GET nebo POST poslat dvojice jméno-hodnota. Tyto hodnoty lze jednoduše přečíst libovolným skriptem na straně serveru.

Tento serverový skript pak zašle odpověď zpět pomocí HTTP, ale na rozdíl od normální webové stránky bude odpověď ve formátu, který bude parsován JavaScriptem v klientovi. Doporučeným formátem je XML, jehož výhodou je velké rozšíření a mnoho knihoven, pomocí kterých je možné s dokumenty XML manipulovat. [4]

AJAX při vývoji nového webu přináší tyto možné výhody:

- Umožňuje vytvářet lepší a přístupnější webové aplikace.
- Díky své popularitě podporuje tzv. vzory, které vývojářům pomáhají, aby se při běžných úkolech vyhnuli vynalézání již dávno známých věcí.
- Využívá stávající technologie.
- Využívá stávající znalosti vývojářů.
- Prvky AJAXu se dokonale integrují s funkcionalitou, danou webovými prohlížeči (změna velikosti stránky, navigace na stránce a podobně).

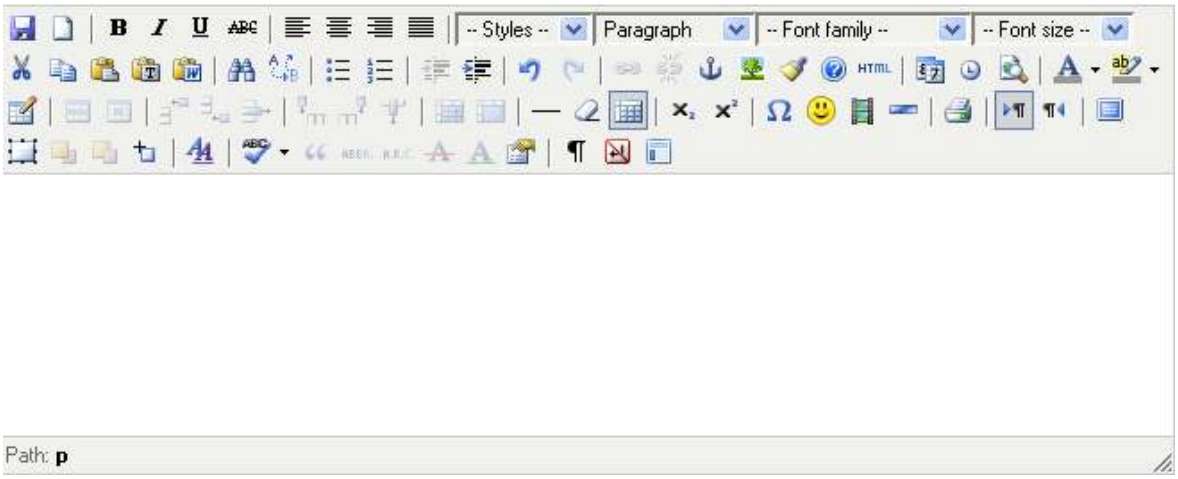
2.7 Ostatní

Mezi ostatní technologie jsou zařazeny technologie, které nám usnadňují práci s formuláři.

2.7.1 TinyMCE

TinyMCE je Wysiwyg editor. Wysiwyg je zkratka, která znamená "what you see is what you get", tedy "co vidíš, to dostaneš". TinyMCE slouží k tvorbě html textu, přičemž uživatel nemusí vůbec znát html kód.

Tento editor patří podle nezávislých testů mezi tři nejlepší on-line editory na světě. Je jednoduché ho integrovat, disponuje rozsáhlou škálou nastavení, jeho výstup XHTML 1.0 je validní, existuje česká lokalizace a je zdarma. [19]



Obr. 1. TinyMCE

3 FORMULÁŘOVÉ PRVKY V HTML

3.1 FORM

Tag FORM označuje formulář. Jedná se o párový tag. Uzavírá skupinu ovládacích polí do jednoho formuláře, který bude najednou odeslán.

Tab. 1. Atributy tagu FORM

Atribut	Význam	Hodnoty
action	skript, který bude zpracovávat data	URL
method	způsob předávání dat	get | post
enctype	způsob zakódování dat	application/x-www-form-urlencoded | multipart/form-data | libovolná mime deklarace
target	cílové okno nebo rám	_self, _blank, _top, _parent, nebo jméno rámu nebo okna

Action obvykle míří na nějaký CGI skript, PHP nebo ASP. Není-li uvedeno, odešlou se data téže stránce.

Výchozí metoda atributu **method** je get. Říká, že se data budou předávat jako součást URL, tedy v řádku adresy. Metod = post zabalí odesílaná data a odesílá je nezávisle, takže nejsou vidět. Post je dobré nastavit u delších formulářů.

Atributem **enctype** se nemusíte zabývat, pokud vám nepůjde o posílání souborů nebo o přesný výstup českých znaků (pak nastavte multipart/form-data). Pro posílání jednoduché pošty doporučuji enctype="text/plain".

Následuje popis vstupních polí formuláře. Dělají se pomocí tagů <input>, <select> a <textarea>. Tyto musejí být umístěny v elementu <form>, ale navíc mohou být třeba i v tabulce. Popisky vstupních polí se dělají normálním textem kolem nich.

3.2 INPUT

INPUT je vstupní pole. Jedná se o nepárový tag.

Tab. 2. Atributy tagu INPUT

Atribut	**Význam**	**Hodnoty**
type	druh vstupního pole	text | password | hidden | radio | checkbox | submit | reset | image | file | buton
name	jméno pole, které se odesílá s daty	libovolné jméno
value	hodnota pole (původní hodnota pole nebo text zobrazovaný na tlačítku)	libovolná hodnota
disabled	políčko bude šedé a nepůjde měnit (jen v některých prohlížečích) v IE se neodesílá	bez hodnoty
readonly	obsah pole nepůjde měnit (fachá jen v některých prohlížečích)	bez hodnoty
align	zarovnání (jako u obrázku)	right, left + těch dalších x možností

3.2.1 Atribut type

Atribut type určuje typ políčka. Input v sobě zahrnuje celou škálu různých kolonek, tlačítek a přepínačů - to všechno závisí na atributu type.

Tab. 3. Atribut type tagu INPUT

Type	**Druh vstupního pole**	**Další atributy**
text	obyčejné textové pole	size= šířka ve znacích maxlength = nejvyšší možný počet zadaných znaků, autocomplete = doplňování známých hodnot (popis níže)
password	textové pole s hvězdičkami	size= šířka ve znacích maxlength- maximum znaků
hidden	skryté pole s předem nastavenou hodnotou	
radio	přepínač puntíků (několik tagů <input type=radio> stejného jména (name) s různými hodnotami tvoří skupinu možností)	checked - atribut bez hodnoty způsobí stisknutí puntíku

checkbox	zatrhávací tlačítko	checked - atribut bez hodnoty způsobí zatržení
submit	potvrzující tlačítko způsobující odeslání formuláře	
reset	tlačítko na smazání všech polí (na přednastavenou hodnotu)	
image	potvrzující tlačítko odesílající navíc souřadnice kliknutí (*name*.x a *name*.y (php je dostává jako *name_x* a *name_y*))	src= URL obrázku (navíc nepoužitelné dynsrc, lowsrc jako u <img>)
file	umožní zadat soubor	accept = MIME typ nabízených souborů
button	tlačítko ovládané skriptovými atributy	

3.3 SELECT

Výběr. Zobrazí obdélníček s možnostmi nebo roletkové menu. Párový tag, jeho obsahem jsou jednotlivé volby „<option>“.

Tab. 4. Atributy tagu SELECT

Atribut	**Význam**	**Hodnoty**
name	jméno pole odesílané s daty	libovolné jméno
multiple	umožnění hromadného výběru (s Ctrl)	bez hodnoty
size	počet zobrazených řádků	Číslo
disabled	políčko bude šedé a nepůjde měnit (jen v některých prohlížečích)	bez hodnoty

Pokud je size rovno 1, bude to roletkové menu. Při size rovno 2 a vyšší se <select> zobrazí jako obdélníček s rolovací lištou. Bude-li hodnot stejně nebo méně než size, nebudou tam ani rolovací lišty.

Šířka selectu, jak se vykreslí na stránce, se odvozuje od nejširší option.

Ze všech HTML tagů jsou s vykreslováním tagu select asi největší problémy, select například nejde dost dobře stylovat přes CSS (pouze tučnost, kurziva a barva pozadí přes <option>). Je to tím, že tento ovládací prvek přebírají prohlížeče (zejm. Internet Explorer) z grafického prostředí operačního systému.

3.4 OPTION

Položka výběru. Jedná se o nepárový tag (třebaže se může zadávat párově), obsahem elementu je text za tagem až do dalšího tagu.

Tab. 5. Atributy tagu OPTION

Atribut	Význam	Hodnoty
value	řetězec odesílaný jako hodnota pole	řetězec
selected	Položka je předem vybrána	bez hodnoty

Text za tagem <option> se zobrazí ve výběru. Pomocí kaskádových stylů (CSS) se v IE 6 dají položky option formátovat pouze omezeně. Dá se jim nastavit jen barva písma (color) a barva pozadí (background-color).

3.5 TEXTAREA

Rozsáhlé vstupní pole. Zobrazuje rámeček s lištou. Je to párový tag. Nemá atribut value, za implicitní hodnotu se považuje obsah elementu. Jinak řečeno, tag <textarea> obklopuje text, který se zpočátku zobrazí uvnitř </textarea>.

Tab. 6. Atributy tagu TEXTAREA

Atribut	Význam	Hodnoty
name	jméno odesílané s daty	jméno
cols	šířka pole ve znacích	číslo
rows	výška pole v řádcích	číslo
disabled	políčko bude šedé a nepůjde měnit (jen v některých prohlížečích, tehdy se neodesílá)	bez hodnoty
readonly	obsah pole nepůjde měnit	bez

	(funguje jen v některých prohlížečích)	hodnoty
wrap	zalamování slov a řádků	hard, soft, off

Tag textarea nemá žádnou obdobu atributu maxlength pro maximální počet znaků, jako je tomu u tagu <input type="text">.

Tab. 7. Atribut wrap tagu TEXTAREA

Wrap	**Význam**
soft	řádky se smějí zalomit jenom v místě mezery. V praxi se zalamují i uprostřed slova, přesahuje-li slovo celý řádek. Odesílá se tak, jak je zapsáno.
hard	řádky se zalamují v místě mezery nebo kdekoliv v příliš dlouhém slově. Pokud se ale zalomí, je tento řádkový zlom také odeslán na server jako konec řádku.
off	řádek se nezalamuje vůbec nikde (Interet Explorer), popř. jen na konci slov (Mozilla). Objevuje se vodorovný scrollbar. Odesílá se tak, jak je zapsáno.

Prohlížeče se liší v tom, jakou hodnotu mají nastavenu jako výchozí. Zatímco v IE a v Mozille je to soft, v Netscape je to pravděpodobně off. [8]

3.6 LABEL

Štítek, popisek pole. Vyskytuje se nejlépe před políčkem, ke kterému se vztahuje, aby to i v prohlížečích, které label nepodporují, dávalo smysl. Výhodou je aktivace pole formuláře, pokud se klikne na štítek, a lepší automatizace (accesskey, css). Vlastní text štítku se zadává jako obsah elementu. Jde o tag párový.

Tab. 8. Atribut tagu LABEL

Atribut	**Význam**	**Hodnoty**
for	svázání s polem stejného identifikátoru	hodnota atributu id u svázaného pole

3.7 FIELDSET

Skupina polí. Párový tag, který opticky sdruží několik prvků formuláře a vykreslí kolem nich slabý rámeček. Čáru rámečku může přerušit text tagu <legend>. <fieldset> nemá žádné atributy kromě obecných. Vzhled se dá upravovat pomocí CSS. Starší prohlížeče tento tag ignorují.

3.8 LEGEND

Popisek skupiny polí (tagu <fieldset>). Zobrazuje se nad skupinou polí přes horní čáru fieldsetu. Musí být zapsána hned za značkou <fieldset>.

Tab. 9. Atribut tagu LEGEND

Atribut	Význam	Hodnoty
align	zarovnání popisku	left | right | center

3.9 OPTGROUP

Skupina voleb ve výběrovém prvku <select>. Párový tag, kterým se obklopí skupinky tagů <option>. V šestkových verzích prohlížečů se potom takové skupinky zobrazí odsazené a budou mít nadpis tučnou kurzívou. Ten nadpis se zadává jako atribut label tagu <optgroup>.

3.10 BUTTON

Tlačítko. Jedinou a hlavní výhodou buttonu oproti tagu <input> je, že se do něj dá vložit libovolný HTML kód, který se na tlačítku zobrazí. Takže se mezi <button> a </button> dají vkládat obrázky, nadpisy, prostě cokoliv. Je to tag párový.

Tab. 10. Atributy tagu BUTTON

Atribut	Význam	Hodnoty
type	druh vstupního pole	submit | reset | button
name	jméno pole, které se odesílá s daty	libovolné jméno
value	hodnota, která by se měla odesílat	libovolná hodnota

V praxi se button chová hodně podobně jako <input> se stejným type.

Při odesílání formuláře by se normálně měly spárovat hodnoty name a value. Implementace tagu button v Internet Exploreru ale obsahuje chybu, kdy se namísto value odesílá text z obsahu buttonu (tedy to, co je vidět na tlačítku, přesněji řečeno vlastnost innerText). Kvůli této chybě se ale button v praxi moc nepoužívá. [8]

II. PRAKTICKÁ ČÁST

4 APLIKACE Z POHLEDU PROGRAMÁTORA

Webová aplikace E-Forms je naprogramována pomocí PHP5, HTML, CSS a dalších webových technologií. Základní myšlenkou při tvorbě této webové aplikace bylo stanovení vzájemných vztahů mezi objekty ve formuláři. Znamená to tedy, že při vytvoření jakéhokoli formulářového prvku je určeno do jakého objektu formuláře bude prvek vložen. Tímto byla vytvořena strategie celé aplikace založena na „rodičích“ a „dětech“ jednotlivých objektů. Při jakékoli operaci jsou data ihned ukládána do databáze MySQL a odtud následně vybírána.

4.1 Tabulky databáze MySQL

Pro tuto webovou aplikaci a její správnou funkčnost byly vytvořeny následující tabulky v databázi MySQL.

4.1.1 Dom

Tabulka DOM byla vytvořena pro uchování vztahů mezi jednotlivými objekty formuláře.

Tabulka DOM je složena z následujících sloupců:

- **idDom –** tento sloupec byl nastaven jako PRIMARY KEY a je zde uloženo identifikační číslo jednotlivých záznamů
- **typeDom** – ukládá se zde typ vloženého objektu (div, obrázek atd.)
- **idObject** – identifikační číslo vloženého objektu z příslušné tabulky (např. tabulka pro ukládání objektu TEXT)
- **idForms** – identifikační číslo formuláře, ve kterém se objekt nachází

4.1.2 Checkbox

Tato tabulka byla vytvořena pro ukládání jednotlivých parametrů objektu Checkbox.

Tabulka CHECKBOX je složena z následujících sloupců:

- **id** – tento sloupec je nastaven jako PRIMARY KEY a je zde uloženo identifikační číslo jednotlivých záznamů

- **name** – jméno objektu Checkbox
- **parent** - „rodiče“ objektu (id objektu, do kterého je CHECKBOX vložen)
- **hodnota** – hodnota, kterou bude objekt nabývat při zaškrtnutí objektu Checkbox
- **fontSize** – velikost písma při výpisu do formuláře
- **fontWeight** – šířka písma při výpisu do formuláře (normální, tučné)
- **color** – barva písma při výpisu do formuláře
- **italic** – styl písma při výpisu do formuláře (kurzíva)
- **font** – font jakým je text vypsán (Arial, Times New Roman)
- **podtrzeni** – styl písma při výpisu do formuláře (podtržení písma)
- **idForms** – identifikační číslo formuláře, ve kterém se objekt CHECKBOX nachází

4.1.3 Divs

Tabulka DIVS byla vytvořena pro ukládání jednotlivých parametrů objektu Div.

Tabulka DIVS je složena z následujících sloupců:

- **id** – tento sloupec je nastaven jako PRIMARY KEY a je zde uloženo identifikační číslo jednotlivých záznamů
- **name** – jméno objektu Div
- **parent** - „rodiče“ objektu (id objektu, do kterého je DIV vložen)
- **children** – „děti“ objektu (id objektů vnořených do objektu DIV)
- **height** – výška daného objektu v pixelech
- **width** – šířka daného objektu v pixelech
- **top** – y-ová souřadnice umístění objektu v pixelech
- **zleva** - x-ová souřadnice umístění objektu v pixelech
- **zindex** – z-idnex objektu DIV
- **textAlign** – zarovnání textu uvnitř objektu DIV

- **bgColor** – barva pozadí objektu
- **border** – ohraničení objektu
- **idForms** – identifikační číslo formuláře, ve kterém se objekt DIV nachází

4.1.4 Form

Tato tabulka byla vytvořena pro ukládání jednotlivých parametrů objektu FORM.

Tabulka FORM je složena z následujících sloupců:

- **id** – tento sloupec je nastaven jako PRIMARY KEY a je zde uloženo identifikační číslo jednotlivých záznamů
- **children** - „děti“ objektu (id objektů vnořených do objektu FORM)
- **idForms** - identifikační číslo formuláře, ve kterém se objekt nachází

4.1.5 Formulare

Tato tabulka byla vytvořena pro ukládání jednotlivých formulářů.

Tabulka FORMULARE je složena z následujících sloupců:

- **idForms** – tento sloupec je nastaven jako PRIMARY KEY a je zde ukládáno identifikační číslo formuláře
- **nameForms** – jméno formuláře
- **povolen** – stav formuláře ke zveřejnění

4.1.6 Inputtext

Tato tabulka byla vytvořena pro ukládání jednotlivých parametrů objektu INPUT.

Tabulka INPUTTEXT je složena z následujících sloupců:

- **id** – tento sloupec je nastaven jako PRIMARY KEY a je zde uloženo identifikační číslo jednotlivých záznamů

- **name** – jméno objektu INPUT
- **parent** - „rodiče“ objektu (id objektu, do kterého je INPUT vložen)
- **width** – šířka objektu v pixelech
- **fontSize** – velikost písma při výpisu do formuláře
- **fontWeight** – šířka písma při výpisu do formuláře (normální, tučné)
- **color** – barva písma při výpisu do formuláře
- **italic** – styl písma při výpisu do formuláře (kurzíva)
- **font** – font jakým je text vypsán (Arial, Times New Roman)
- **idForms** – identifikační číslo formuláře, ve kterém se objekt INPUT nachází

4.1.7 Javascript

Tato tabulka byla vytvořena pro ukládání jednotlivých parametrů objektu JAVASCRIPT.

Tabulka JAVASCRIPT je složena z následujících sloupců:

- **id** – tento sloupec je nastaven jako PRIMARY KEY a je zde uloženo identifikační číslo jednotlivých záznamů
- **name** – jméno objektu JAVASCRIPT
- **parent** - „rodiče“ objektu (id objektu, do kterého je JAVASCRIPT vložen)
- **script** – JavaScriptový kód objektu
- **idForms** – identifikační číslo formuláře, ve kterém se objekt JAVASCRIPT nachází

4.1.8 Vyber

Tato tabulka byla vytvořena pro ukládání jednotlivých parametrů objektu SELECT.

Tabulka VYBER je složena z následujících sloupců:

- **id** – tento sloupec je nastaven jako PRIMARY KEY a je zde uloženo identifikační číslo jednotlivých záznamů
- **name** – jméno objektu SELECT
- **parent** - „rodiče“ objektu (id objektu, do kterého je SELECT vložen)
- **children** - „děti“ objektu (id objektů vnořených do objektu SELECT)
- **multiple** – možnost vícepoložkového výběru
- **fontSize** – velikost písma při výpisu do formuláře
- **fontWeight** – šířka písma při výpisu do formuláře (normální, tučné)
- **color** – barva písma při výpisu do formuláře
- **italic** – styl písma při výpisu do formuláře (kurzíva)
- **font** – font jakým je text vypsán (Arial, Times New Roman)
- **podtrzeni** – styl písma při výpisu do formuláře (podtržení písma)
- **idForms** – identifikační číslo formuláře, ve kterém se objekt SELECT nachází

4.1.9 Optionvyber

Tato tabulka byla vytvořena pro ukládání jednotlivých parametrů objektu OPTION. Tento objekt je vnořen do objektu SELECT (možnosti při výběru)

Tabulka OPTION je složena z následujících sloupců:

- **id** – tento sloupec je nastaven jako PRIMARY KEY a je zde uloženo identifikační číslo jednotlivých záznamů
- **name** – jméno objektu OPTION
- **parent** - „rodiče“ objektu (id objektu, do kterého je OPTION vložen)
- **value** – hodnota, která je nastavena při výběru dané položky
- **idForms** – identifikační číslo formuláře, ve kterém se objekt OPTION nachází

4.1.10 Picture

Tato tabulka byla vytvořena pro ukládání jednotlivých parametrů objektu PICTURE.

Tabulka PICTURE je složena z následujících sloupců:

- **id** – tento sloupec je nastaven jako PRIMARY KEY a je zde uloženo identifikační číslo jednotlivých záznamů
- **name** – jméno objektu OBRÁZEK
- **parent** - „rodiče“ objektu (id objektu, do kterého je OBRÁZEK vložen)
- **value** – hodnota, která je nastavena při výběru dané položky
- **idForms** – identifikační číslo formuláře, ve kterém se objekt OBRÁZEK nachází

4.1.11 Popisek

Tato tabulka byla vytvořena pro ukládání jednotlivých parametrů objektu POPISEK.

Tabulka POPISEK je složena z následujících sloupců:

- **id** – tento sloupec je nastaven jako PRIMARY KEY a je zde uloženo identifikační číslo jednotlivých záznamů
- **name** – jméno objektu POPISEK
- **parent** - „rodiče“ objektu (id objektu, do kterého je POPISEK vložen)
- **object** – id objektu, pro který je POPISEK vložen
- **text** – samotný text popisku
- **fontSize** – velikost písma při výpisu do formuláře
- **fontWeight** – šířka písma při výpisu do formuláře (normální, tučné)
- **color** – barva písma při výpisu do formuláře
- **italic** – styl písma při výpisu do formuláře (kurzíva)
- **font** – font jakým je text vypsán (Arial, Times New Roman)
- **podtrzeni** – styl písma při výpisu do formuláře (podtržení písma)

- **idForms** – identifikační číslo formuláře, ve kterém se objekt POPISEK nachází

4.1.12 Tabulka

Tato tabulka byla vytvořena pro ukládání jednotlivých parametrů objektu TABULKA.

Tabulka TABULKA je složena z následujících sloupců:

- **id** – tento sloupec je nastaven jako PRIMARY KEY a je zde uloženo identifikační číslo jednotlivých záznamů
- **name** – jméno objektu TABULKA
- **parent** - „rodiče“ objektu (id objektu, do kterého je TABULKA vložen)
- **children** - „děti“ objektu (id objektů vnořených do objektu TABULKA)
- **width** – šířka tabulky v pixelech nebo procentech
- **border** – šířka ohraničení tabulky
- **idForms** – identifikační číslo formuláře, ve kterém se objekt TABULKA nachází

4.1.13 TabulkaTr

Tato tabulka byla vytvořena pro ukládání jednotlivých parametrů objektu TR.

Tabulka TABULKATR je složena z následujících sloupců:

- **id** – tento sloupec je nastaven jako PRIMARY KEY a je zde uloženo identifikační číslo jednotlivých záznamů
- **name** – jméno objektu TABULKATR
- **parent** - „rodiče“ objektu (id objektu, do kterého je TABULKATR vložen)
- **children** - „děti“ objektu (id objektů vnořených do objektu TABULKATR)
- **idForms** – identifikační číslo formuláře, ve kterém se objekt TABULKATR nachází

4.1.14 TabulkaTd

Tato tabulka byla vytvořena pro ukládání jednotlivých parametrů objektu TD.

Tabulka TABULKATD je složena z následujících sloupců:

- **id** – tento sloupec je nastaven jako PRIMARY KEY a je zde uloženo identifikační číslo jednotlivých záznamů
- **name** – jméno objektu TABULKATD
- **parent** - „rodiče“ objektu (id objektu, do kterého je TABULKATD vložen)
- **children** - „děti“ objektu (id objektů vnořených do objektu TABULKATD)
- **idForms** – identifikační číslo formuláře, ve kterém se objekt TABULKATD nachází

4.1.15 Text

Tato tabulka byla vytvořena pro ukládání jednotlivých parametrů objektu TEXT.

Tabulka TEXT je složena z následujících sloupců:

- **id** – tento sloupec je nastaven jako PRIMARY KEY a je zde uloženo identifikační číslo jednotlivých záznamů
- **name** – jméno objektu TEXT
- **parent** - „rodiče“ objektu (id objektu, do kterého je TEXT vložen)
- **text** – text objektu
- **idForms** – identifikační číslo formuláře, ve kterém se objekt TEXT nachází

4.1.16 Textarea

Tato tabulka byla vytvořena pro ukládání jednotlivých parametrů objektu TEXTAREA.

Tabulka TEXTAREA je složena z následujících sloupců:

- **id** – tento sloupec je nastaven jako PRIMARY KEY a je zde uloženo identifikační číslo jednotlivých záznamů
- **name** – jméno objektu TEXTAREA
- **parent** - „rodiče“ objektu (id objektu, do kterého je TEXTAREA vložen)
- **cols** – šířka objektu v pixelech
- **rows** – výška objektu v pixelech
- **fontSize** – velikost písma při výpisu do formuláře
- **fontWeight** – šířka písma při výpisu do formuláře (normální, tučné)
- **color** – barva písma při výpisu do formuláře
- **italic** – styl písma při výpisu do formuláře (kurzíva)
- **font** – font jakým je text vypsán (Arial, Times New Roman)
- **podtrzeni** – styl písma při výpisu do formuláře (podtržení písma)
- **idForms** – identifikační číslo formuláře, ve kterém se objekt TEXTAREA nachází

4.1.17 Users

Tato tabulka byla vytvořena pro ukládání parametrů jednotlivých uživatelů.

Tabulka USERS je složena z následujících sloupců:

- **id** – tento sloupec je nastaven jako PRIMARY KEY a je zde uloženo identifikační číslo jednotlivých záznamů
- **jmeno** – jméno uživatele
- **prijmeni** – příjmení uživatele
- **nick** – přihlašovací jméno uživatele
- **email** – e-mail uživatele
- **heslo** – heslo uživatele
- **formuláře** – id vyplněných formulářů uživatele

4.2 Administrátorská část

4.2.1 Vložení formulářového prvku do formuláře

Při vytvoření nového formuláře dochází k vytvoření hlavního objektu formuláře, kterým je objekt „form“. Tento je uložen do databázové tabulky „form“. Do tohoto objektu jsou následně vkládány jednotlivé formulářové prvky („děti“ objektu „form“). Tímto krokem začíná tvorba každého formuláře.

Při vytvoření formulářového prvku je objekt vložen do objektu „div“, který má předem stanovené parametry. Je to z toho důvodu, aby se s objektem dalo ihned po vložení pohybovat a následně editovat jeho parametry.

Nejdříve je tedy vytvořena nová třída cMysql, která zajišťuje připojení k databázi s předem nadefinovanými parametry.

```
1 //--- vytvoření objektu DB a následné připojení k DB
2 $userDB = new cMysql($MYSQL_SERVER, $MYSQL_USER, $MYSQL_PASSWORD,
  $MYSQL_DATABASE);
```

Poté je do tabulky „divs“ vložen objekt „div“, který zapouzdřuje formulářový prvek z výše uvedeného důvodu. Nadřazeným objektem („rodičem“) takového divu je nastaven hlavní objekt formuláře čili objekt „form“. Následně je zjištěno id posledního záznamu v této tabulce, které je zapotřebí pro záznam do tabulky „dom“. Objekt „div“ je tedy zároveň uložen do tabulky „dom“, která uchovává vztahy mezi objekty formuláře. Opět je zjištěno id posledního záznamu, tentokrát v tabulce „dom“. Takové id slouží pro určení nadřazeného prvku („rodiče“) vkládaného formulářového prvku.

```
3 //vytvoření divu
4  $div = $userDB->query("INSERT INTO divs
  (name,parent,width,height,top,zindex,textAlign,bgColor,border,zleva,
  idForms) VALUES
5 (\"$name\",\"$parent\",'300','100','10','0','left','#FFFFFF','0','10
  ',\"$idForms\");");


6  $idObjectDiv = mysql_insert_id();
7  $divDom = $userDB->query("INSERT INTO dom(typeDom,idObject,idForms)
  VALUES ('divs','$idObjectDiv','$idForms');");
8  $idDomDiv=mysql_insert_id();
```

V případě, kdy je zjištěno id posledního záznamu v tabulce „dom“, je do příslušné databázové tabulky vkládaného objektu vložen záznam s jeho parametry (např. vytvoření objektu „input“ je uloženo do tabulky „inputtext“). Nadřazeným objektem („rodičem“) je div, do kterého je prvek vložen. Opět dochází ke zjištění id posledního záznamu v tabulce daného objektu. Tento se poté uloží i do databázové tabulky „dom“, ve které je id objektu nastaveno právě na poslední záznam z předcházející tabulky.

```
 9 //--- uložení parametrů do DB
10  $result = $userDB->query("INSERT INTO inputtext
   (name,parent,width,fontSize,fontWeight,color,font,italic,idForms)
   VALUES
11 (\"$name\",\"$idDomDiv\",\"$width\",\"$fontSize\",\"$fontWeight\",\"
   $color\",\"$font\",\"$italic\",\"$idForms\");");


12  //--- zjištění id posledního vloženého prvku
13  $idObjectInput = mysql_insert_id();
14
15  //--- uložení objektu do DOM
16  $result2 = $userDB->query("INSERT INTO dom
   (typeDom,idObject,idForms) VALUES
   ('inputtext','$idObjectInput','$idForms');");
```

Poté je ještě jednou zjištěno id posledního záznamu v tabulce „dom“, které je použito pro nastavení podřazených objektů („dětí“) u na začátku vytvořeného divu.

```
17  $idDom = mysql_insert_id();
18  $result3 = $userDB->query("UPDATE divs SET children=".$idDom."
   WHERE id=".$idObjectDiv.");
```

Nakonec je do hlavního objektu, v tomto případě „form“, uložen podřazený objekt („dítě“).

4.2.2 Editace formulářových prvků

Pomocí volby úpravy, je možné vložené formulářové prvky editovat. Prvek, který má být editován může být vybrán ze všech formulářových prvků.

Nejdříve jsou z databáze a tabulky vybrány všechny prvky daného formuláře. Podle záznamu z tabulky „dom“ je vybrán konkrétní prvek z tabulky daného záznamu. Všechny tyto záznamy jsou následně vypisovány do tagu „select“.

```
1 $userDB = new cMysql($MYSQL_SERVER, $MYSQL_USER, $MYSQL_PASSWORD,
  $MYSQL_DATABASE);
```

```
2 $result = $userDB->query('SELECT * FROM dom where
  idForms='.$_SESSION["idForms"].' AND typeDom<>"form";');
3 while($row = mysql_fetch_object($result))
4 {
5  $result1 = $userDB->query('SELECT * FROM '.$row->typeDom.' Where
    id='.$row->idObject.' AND idForms='.$_SESSION["idForms"].';');
6        while($row1 = mysql_fetch_object($result1))
7        {
8          echo '<option value="'.$row->typeDom.'-'.$row1-> name.'">'.
           $row->typeDom;
9          echo '-'.$row1->name;
10         echo "\n\t";
11       }
12 }
13 echo '</select>';
```

Vybraný prvek je následně editován ve stejném formuláři, ve kterém se dané prvky vkládají.

Další možností výběru je označení daného formulářového prvku. Toto je realizováno pomocí JavaScriptu.

Vymazání jednotlivých objektů probíhá stejným způsobem jako jejich editace.

4.2.3 Výpis formuláře

Výpis formulářových prvků do formuláře používá stejný skript jak pro administrátorskou část tak pro část uživatelskou.

Celý výpis je realizován tak, že každý vypisovaný prvek formuláře je uložen do pole formulářových prvků.

Nejdříve jsou vybrány všechny záznamy z tabulky „dom“ pro daný formulář, který se má vypsat.

```
1 //--- vybere všechny prvky z tabulky DOM pro aktuální formulář
2 $result = $userDB->query("SELECT * FROM dom where
  idForms='$idForms';");
```

Jestliže je v této databázové tabulce nalezen nějaký záznam pro daný formulář, skript pokračuje. Je vybrán konkrétní objekt z příslušné tabulky, který je dán výsledkem výběru dat z tabulky „dom“.

```
3 //--- jestliže jsou v tabulce DOM záznamy pokračuje dál
4 while($dataDom = mysql_fetch_object($result))
5 {
6  //--- vybere konkrétní prvek, který má id rovno id z tabulky DOM
7  $result1 = $userDB->query("SELECT * FROM ".$dataDom->typeDom."
  WHERE id='".$dataDom->idObject."' AND idForms='$idForms';");
```

Poté dochází ke stanovení typu objektu, který byl vybrán předcházejícím dotazem na databázi MySQL. Jestliže záznam splňuje danou podmínku, dojde k nastavení nového id, které je vybráno ze záznamu databázové tabulky „dom“ daného objektu. V případě, že jsou u objektu nastaveny podřazené objekty („děti“), jsou odděleny podle znaku „ ; “ a následně přiřazeny prvku nově vytvořeného pole. Jestliže objekt nemá žádné podřazené objekty, je tato část vynechána. Mezi tím je vytvořena nová třída daného objektu (např. pro objekt „div“ je to třída „div()“).

```
 8 if($dataDom->typeDom == "divs")
 9 {
10  $newid = $dataDom->idDom;
11  $parent = $dataObject->parent;
12  $children = explode(";",$dataObject->children);
13  $objekty[$newid] = new div();
14  if(is_array($children) && $children[0]!="") $objekty[$newid]->
    children = $children;
```

Následuje nastavení nadřazeného objektu („rodiče“) a přiřazení parametrů objektu nově vytvořené položce pole.

```
15 //--- nacteni proměnných z tabulky objektu TABLE
16 if($dataDom->typeDom == "tabulka")
17 {
18  $newid = $dataDom->idDom;
19  $parent = $dataObject->parent;
20  $children = explode(";",$dataObject->children);
21  $objekty[$newid] = new tabulka();
22  if(is_array($children) && $children[0]!="") $objekty[$newid]->
```

```
    children = $children;
23   $objekty[$newid]->setParent($parent);
24   $objekty[$newid]->name = $dataObject->name;
25   $objekty[$newid]->width = $dataObject->width;
26   $objekty[$newid]->border = $dataObject->border;
27 }
```

Tento cyklus tedy trvá do té doby, než jsou vypsány všechny objekty daného formuláře. Poté je zavolána funkce „makeXHTML()“, která má za úkol samotný výpis zdrojového kódu formuláře. Ještě před zavoláním této funkce musí být určen počáteční objekt, od kterého má být formulář vypsán. Jedná se o objekt „form“ daného formuláře. Protože je požadován výpis několika formulářů, nebylo možné nastavit tuto počáteční hodnotu na první záznam v tabulce „dom“.

```
28 //--- nalezení počátečního prvku v tabulce DOM pro výpis objektů
29 $result2 = $userDB->query("SELECT * FROM dom where typeDom='form'
   AND idForms='$idForms';");
30 while($row = mysql_fetch_object($result2))
31 {
32   $start = $row->idDom;
33 }
34
35 $styles="";
36 $xhtml="";
37
38 makeXHTML($objekty,$start,0);
```

Samotná funkce „makeXHTML()“ má tři vstupní parametry. První parametr určuje co má být vypsáno, v tomto případě pole objekty. Druhý parametr určuje počáteční záznam v tabulce „dom“, od kterého má být zahájen výpis a třetí parametr určuje odsazení pro lepší vzhled zdrojového kódu.

Nejdříve byly v této funkci nadefinovány globální proměnné „styles“ a „xhtml“, do kterých je ukládán výsledek výpisu. Jestliže je nastavena proměnná objekty, je zahájen výpis zdrojového kódu do výše uvedených globálních proměnných. Hned na začátku funkce byl zařazen cyklus pro odsazení jednotlivých tagů zdrojového kódu formuláře. Ten tyto tagy odsazuje podle úrovně, ve které se formulářový prvek nachází.

```
1 function makeXHTML($objekty,$id,$level){
2  global $styles,$xhtml;
3  if(isset($objekty[$id])){
4  //--- Odsazení tagů
5  $level++;
6  for($i=0,$odsazeni="";$i<$level;$i++) $odsazeni.="\t";
```

Následně je určeno o jaký typ objektu se jedná a výpis samotného zdrojového kódu je dán posloupností primitivních příkazů. Jestliže má objekt uloženy nějaké podřazené objekty, jsou vypsány mezi tagy daného objektu. Výpis kódu pro CSS je naprogramován zavoláním funkce „vypisCss()“ daného objektu.

```
7 //--- kod pro výpis objektu INPUT TEXT
8 elseif($objekty[$id]->subtype=="text")
9 {
10  $styles.=$objekty[$id]->vypisCss();
11  $xhtml.= $odsazeni."<".($objekty[$id]->type)."
    type='".$objekty[$id]->subtype."' name='".$objekty[$id]->name."'
    id='inputtext-".$objekty[$id]->name."' />\n";
12 }
```

Cyklus této funkce je opakován až do doby, kdy je zdrojový kód celého formuláře uložen do globálních proměnných. Výpis samotného formuláře je realizován pomocí následujících řádků kódu.

```
13 <!-- zacatek vypis -->
14 <style type='text/css'>
15 <?php echo $styles; ?>
16 </style>
17 <?php echo $xhtml; ?>
18 <!-- konec vypis -->
```

4.2.4 Zpřístupnění vytvořeného formuláře uživateli

Při dokončení vytváření formuláře je následně vygenerována databázová tabulka pojmenována jako „formulář_“ + id daného formuláře.

Tato tabulka je generována tak, že jsou vybrány všechny prvky formuláře z databázové tabulky „dom“, které jsou určeny k zadávání hodnot (např. INPUT, CHECKBOX atd.)

Z každého takového záznamu je vybráno jméno prvku, které se odesílá metodou POST při vyplnění příslušného formuláře. K těmto jménům jsou následně přidány znaky pro vytvoření sloupce nově vygenerované databázové tabulky. Jestliže se nejedná o první sloupec generované tabulky, je k tomuto jménu přidán znak „ , “, který odděluje jednotlivé sloupce tabulky.

```
1  $userDB = new cMysql($MYSQL_SERVER, $MYSQL_USER, $MYSQL_PASSWORD,
   $MYSQL_DATABASE);
2  $result = $userDB->query("SELECT * FROM dom WHERE
   idForms='".$idForms."'AND (typeDom='checkbox' OR typeDom='inputtext'
   OR typeDom='vyber' OR typeDom='textarea')");
3  while($row=mysql_fetch_object($result))
4  {
5   $result1 = $userDB->query("SELECT * FROM ".$row->typeDom." WHERE
   id=".$row->idObject.";");
6   while($row1=mysql_fetch_object($result1))
7   {
8         if($i=="0")
9         {
10              $objekty.='`'.$row1->name.'` TEXT NOT NULL';
11        }
12        else
13        {
14              $objekty.=',`'.$row1->name.'` TEXT NOT NULL';
15        }
16        $i++;
17 }
```

Vygenerování tabulky je poté vytvořeno posledním řádkem skriptu.

```
18 $vytvor = $userDB->query("CREATE TABLE formular_".$idForms." (id INT
   NOT NULL PRIMARY KEY AUTO_INCREMENT,".$objekty.",user varchar(255)
   NOT NULL,datum varchar(255) NOT NULL);");
```

K vygenerovaným sloupcům tabulky jsou přidány ještě dva sloupce. Prvním je sloupec „user“, kde bude při vyplnění zaznamenáno přihlašovací jméno uživatele, který formulář vyplnil. Druhým sloupcem je datum vyplnění formuláře.

Po vygenerování formulářové tabulky, je nastaven daný formulář pro zpřístupnění jednotlivým uživatelům a to tak, že je do tabulky „formulare“ k danému formuláři nastavena hodnota povolen na 1.

```
19 $result2 = $userDB->query("UPDATE formulare SET povolen=1 WHERE
   idForms=".$idForms."");
```

4.3 Uživatelská část

4.3.1 Výpis formuláře

Výpis formuláře, který má být vyplněn je realizován stejným způsobem jako výpis v administrátorské části.

4.3.2 Uložení vyplněného formuláře

Vyplněný formulář je ukládán do databázové tabulky, která tomuto formuláři náleží. Název tabulky je složen z formulář_ + identifikační číslo formuláře (např. formulář_1).

Metodou POST jsou tedy odeslány veškeré vstupní hodnoty na skript „print.php“. V tomto skriptu jsou data uložena do databázové tabulky.

Hlavním úkolem je vygenerovat sloupce tabulky pro uložení.Tyto sloupce jsou generovány tak, že jsou vybrány všechny prvky formuláře z databázové tabulky „dom“, které jsou určeny k zadávání hodnot (např. INPUT, CHECKBOX atd.) Z každého takového záznamu je vybráno jméno prvku, které se odesílá metodou POST při vyplnění příslušného formuláře.

```
 1 if(isset($_POST["save"]))
 2 {
 3  $i=0;
 4  $idForms=$_GET["idForms"];
 5  $userDB = new cMysql($MYSQL_SERVER, $MYSQL_USER, $MYSQL_PASSWORD,
   $MYSQL_DATABASE);
 6  $result = $userDB->query("SELECT * FROM dom WHERE
    idForms='".$idForms."'AND (typeDom='checkbox' OR typeDom=
   'inputtext' OR typeDom='vyber'
 7  OR typeDom='textarea')");
 8  while($row=mysql_fetch_object($result))
 9  {
10        //--- generování proměnných pro uložení do DB
11        $result1 = $userDB->query("SELECT * FROM ".$row->typeDom."
          WHERE id=".$row->idObject.";");
```

```
12          while($row1=mysql_fetch_object($result1))
13          {
```

Následně jsou vygenerovány sloupce formulářové tabulky a proměnné, které budou do těchto sloupců uloženy. Do této tabulky je následně uložen také datum, který je zjištěn php funkcí „time()“ a následně upraven do zřetelnějšího tvaru. Všechny tyto záznamy jsou poté uloženy do databázové tabulky příslušného formuláře.

```
14 if($i=="0")
15 {
16  $objekty.='`'.$row1->name.'`';
17  $promenne.= "'".$_POST[$row1->name]."'";
18 }
19 else
20 {
21  $objekty.=',`'.$row1->name.'`';
22  $promenne.=",'".$_POST[$row1->name]."'";
23 }
24  $i++;
25 }
26 }
27 $datum = StrFTime("%d/%m/%Y %H:%M:%S", Time());
28 $_SESSION["datum"]=$datum;
29 //--- uložení hodnot do tabulky příslušného formuláře
30 $result2 = $userDB->query("INSERT INTO formular_".$idForms." ("
   .$objekty.",user,datum) VALUES (".$promenne.",
   '".$_SESSION['nick']."', '".$datum."');");
```

Poslední operace, která je vykonána pro uložení vyplněného formuláře, je nastavení tohoto formuláře u příslušného uživatele. Nejprve je načten záznam daného uživatele z tabulky „users“, ze které je vybrán sloupec „formulare“. Jestliže není tento sloupec prázdný, je záznam doplněn o nově uložený formulář tak, že k aktuální hodnotě je přidána nová hodnota. Záznam nového formuláře do sloupce je vytvořen pomocí id formuláře + znak „ _ “ + datum vyplnění formuláře. Tento slouží pro možnost vícenásobného vyplnění formuláře.

```
31 //--- update pole formulare u tabulky users pro nový vyplněný
   formulář
```

```
32 $result3 = $userDB->query("SELECT * FROM users WHERE
   nick='".$_SESSION['nick']."';");
33 while($row2=mysql_fetch_object($result3))
34 {
35  $formulare = $row2->formulare;
36  $updateFormulare=$formulare.$idForms.'_'.$datum.';';
37  $result4 = $userDB->query("UPDATE users SET formulare='
   ".$updateFormulare."' WHERE nick='".$_SESSION['nick']."';");
38  }
```

Tímto je celý proces uložení vyplněného formuláře hotov.

4.3.3 Odeslání formuláře e-mailem

Odeslání formuláře e-mailem je realizováno pomocí třídy „PHPMailer()“. Tato třída je volně dostupná kdekoli na internetu.

Celý proces odesílání začíná vytvořením nové třídy „PHPMailer()“. Následně je určeno kódování pro odesílaný e-mail a nastavení parametrů pro odesílání e-mailů. Důležitou součástí je také předmět e-mailu.

```
1 $mail = new PHPMailer();
2
3 $mail->CharSet = "cp1250";
4
5 //--- nastavení parametrů e-mailu
6 include_once "../inc/configMail.php";
7
8 $body = "Váš formulář ze systému EForms";
```

Pro lepší přehlednost je také z databázové tabulky formuláře vybráno jméno odesílaného formuláře, po kterém je pojmenována vygenerovaná příloha neboli vlastní formulář.

Dalším důležitým krokem je získání e-mailové adresy uživatele. Tato je vybrána ze záznamu uživatele v tabulce „users“.

```
9 $userDB = new cMysql($MYSQL_SERVER, $MYSQL_USER, $MYSQL_PASSWORD,
  $MYSQL_DATABASE);
10 $result = $userDB->query("SELECT * FROM formulare WHERE idForms= ¨
   '".$idForms."';");
11 while($row=mysql_fetch_object($result))
```

```
12 {
13  $form=$row->nameForms;
14 }
15
16 //--- získání e-mailové adresy
17 $result = $userDB->query("SELECT * FROM users WHERE nick= '".
   $_SESSION['nick']."';");
18 while($row=mysql_fetch_object($result))
19 {
20  $user=$row->email;
21 }
```

Následně dochází k samotnému odeslání e-mailu na danou adresu a vytvoření přílohy, která má být e-mailem odeslána.

```
22 //--- vytvoření formuláře jako příloha e-mailu
23 $priloha="<style>".$_SESSION['styles']."</style>".$_SESSION['xhtml']
   ;
24 $mail->Body    = $body;
25 $mail->AddAddress($user);
26 $jmenoSouboru=$form.".html";
27 $mail->AddStringAttachment($priloha, $jmenoSouboru);
```

Poslední operací, která se vykoná při odeslání e-mailu, je výpis upozornění, která mají být vypsána v případě úspěšného či neúspěšného odeslání e-mailu. Zároveň také dochází k výmazu adresy příjemce a přílohy, která byla s e-mailem odeslána.

```
28 if(!$mail->Send())
29 echo "<div class='ErrMessage'>Nastala chyba! E-mail nebyl odeslán!
   </div><br>";
30 else echo "<div class='ErrMessage'>E-mail byl odeslán!</div>";
31
32 // Smazání všech adres a příloh
33 $mail->ClearAddresses();
34 $mail->ClearAttachments();
```

4.3.4 Tisk vyplněného formuláře

Při tisku vyplněného formuláře dochází k vypsání všech formulářových prvků stejným způsobem jakým se vypisují v celé webové aplikaci. Jediným rozdílem je ten, že nejsou ve skriptu „print.php“ zobrazeny formulářové prvky (INPUT, TEXTAREA atd.), které jsou nahrazeny výslednou podobou formuláře. Podobu konečného formuláře stanovuje administrátor, který tento formulář vytváří.

Současně je také při tisku schováno menu s ovládacími tlačítky a nakonec je zavolána JavaScriptová funkce „print()“, která celý tento formulář vytiskne.

5 APLIKACE Z POHLEDU UŽIVATELE

Do aplikace se uživatel dostane zaregistrováním a následným přihlášením. Existují dvě možnosti přihlášení. První možností je přihlášení jako administrátor a druhou možností je přihlášení jako běžný uživatel.

Obr. 2. Náhled webové aplikace E-Forms

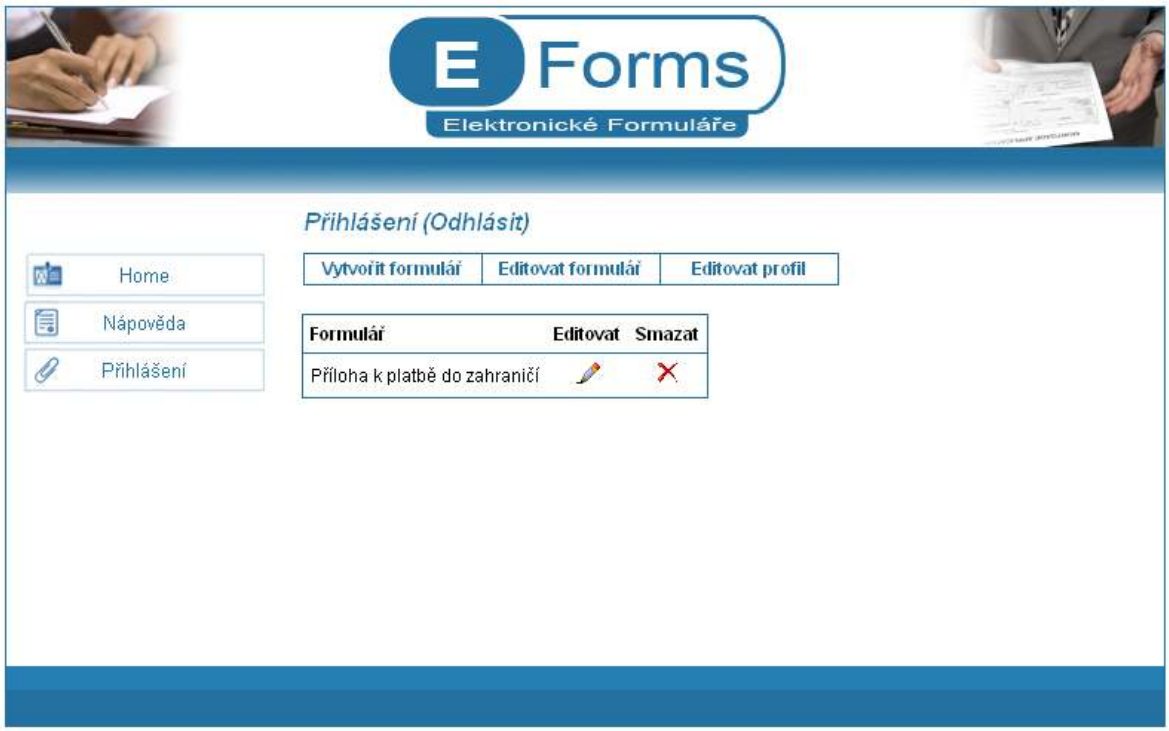
5.1 Administrátorské prostředí

5.1.1 Správa formulářů

Po přihlášení uživatele jako administrátor, má uživatel v nabídce malé menu. Toto menu je složeno ze tří položek. První položkou je „Vytvořit formulář“. Zvolením této položky může administrátor vytvořit nový formulář k editaci. Zadáním jména formuláře a kliknutím na tlačítko vytvořit, bude do databáze formulář přidán.

Obr. 3. Položka „Vytvořit formulář“ v hlavním menu aplikace

V případě, že budete chtít administrátor takhle vytvořený formulář editovat musí kliknout na druhou položku menu, kterou je „Editovat formulář“. Zde je umístěn výpis všech existujících formulářů. Administrátor může formuláře editovat nebo vymazat.



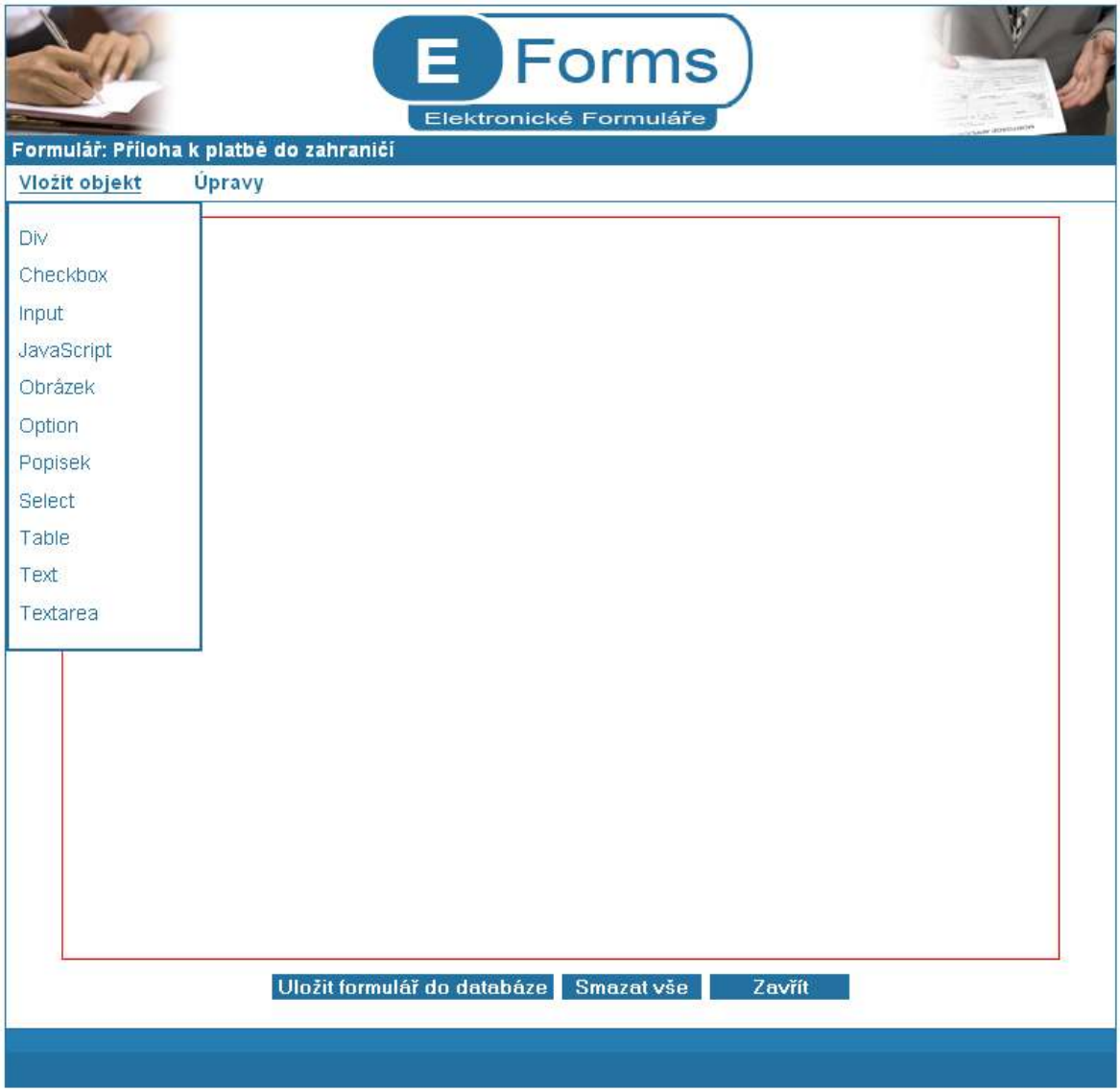
Obr. 4. Položka „Editovat formulář“ v hlavním menu aplikace

Poslední položkou v hlavním menu administrátora je položka „Upravit profil“. Zde si může administrátor změnit své údaje a e-mailovou adresu.

5.1.2 Vytváření formulářů

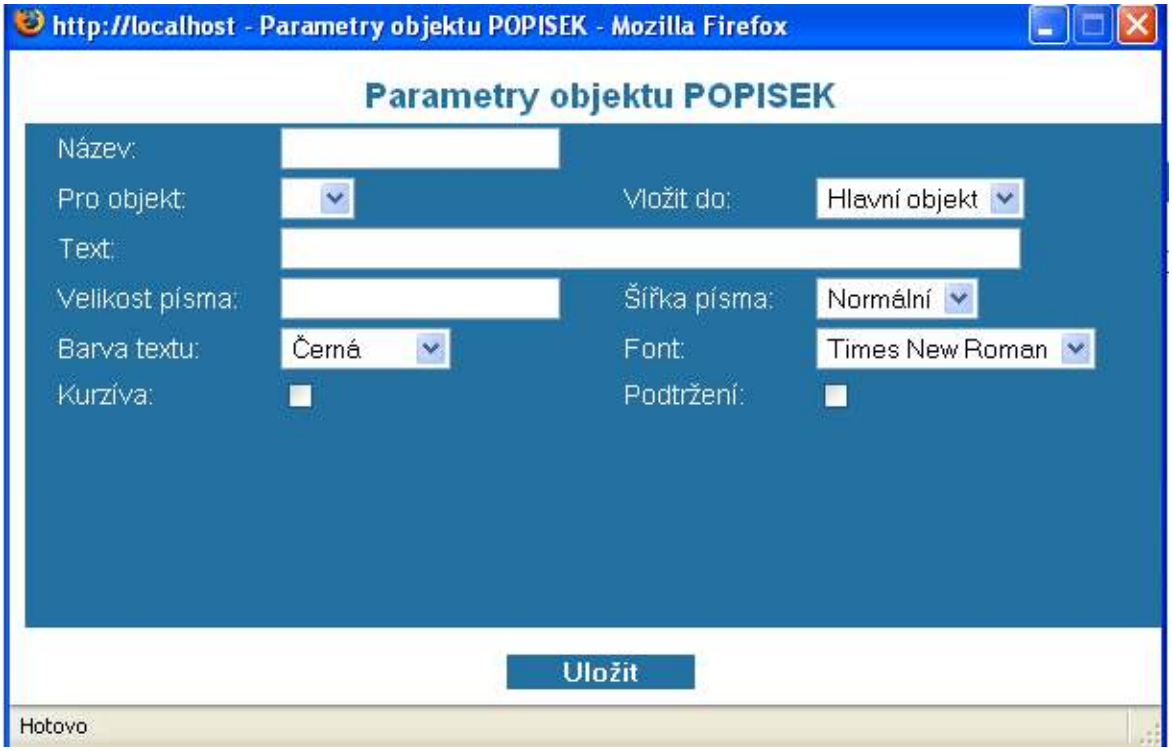
Při kliknutí administrátora v menu „Editovat formulář“ na ikonku daného formuláře pro editaci bude otevřeno nové okno s návrhovým prostředím pro generování formulářů.

Administrátor má v takovém případě v nabídce menu položky pro vytváření a editaci nových formulářových prvků.



Obr. 5. Prostředí pro vytváření formulářů

Kliknutím na některý z uvedených formulářových prvků se otevře nové okno, ve kterém administrátor zadá příslušné hodnoty do formuláře a prvek se následně vloží do formuláře.



Obr. 6. Okno pro vkládání nových popisků

Okna všech prvků, které se dají vložit do formuláře jsou si téměř totožná. Liší se pouze zadávanými parametry.

5.2 Uživatelské prostředí

5.2.1 Přehled formulářů

Po přihlášení uživatele jako běžný uživatel, se objeví v nabídce malé menu. Stejně jako u administrátorské části má tři položky. První položkou je položka „Formuláře“. V této části jsou vypsány všechny zpřístupněné formuláře. Zvolením jednoho z nich se daný formulář otevře v novém okně, ve kterém je možno formulář vyplnit.

Obr. 7. Položka „Formuláře“ v uživatelském rozhraní

Druhou položkou menu v uživatelském rozhraní je „Vyplněné formuláře“. Do této položky se formuláře dostávají v okamžiku, kdy uživatel některý z formulářů vyplní a uloží do databáze. U těchto formulářů je také zobrazen datum pro lepší orientaci, kdy byl formulář vyplněn.

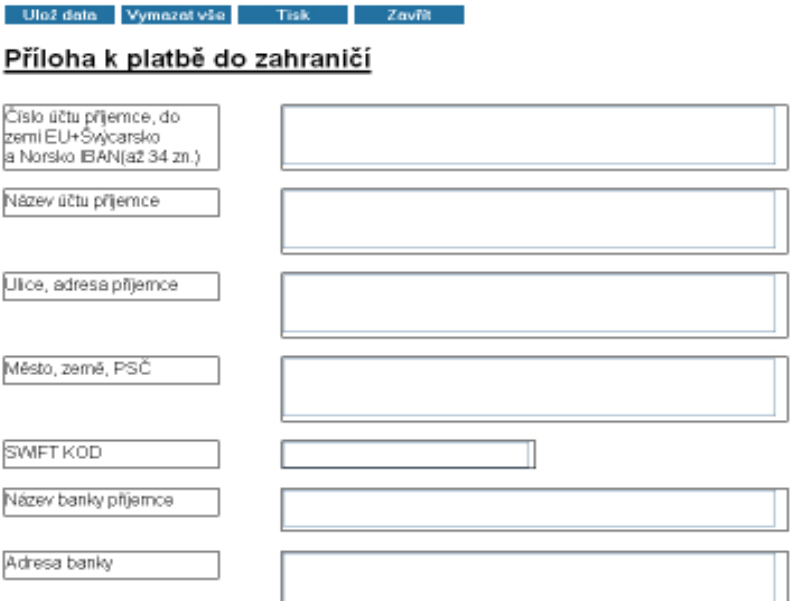
Obr. 8. Položka „Vyplněné formuláře“ v uživatelském rozhraní

Poslední položkou menu je „Editovat údaje“. V této části může uživatel editovat své údaje včetně e-mailové adresy.

Obr. 9. Položka „Editovat údaje“ v uživatelském rozhraní

5.2.2 Rozhraní pro vyplňování formulářů

Kliknutím na daný formulář se uživateli otevře nové okno, ve kterém se formulář vyplňuje.



Obr. 10. Okno pro vyplnění formuláře

V tomto okně je možné formulář uložit, vytisknout nebo odeslat e-mailem. Tyto funkce není zapotřebí více rozebírat.

ZÁVĚR

Cílem této práce bylo vytvořit grafické rozhraní pro práci s formuláři na internetu, které by umožňovalo generování a správu formulářů. Tato aplikace by měla plně nahradit papírové formuláře. Grafické rozhraní bylo naprogramováno pomocí webových technologií HTML, CSS, PHP5 a JavaScript, ke kterým byly přidány ještě neméně podstatné technologie jako jsou AJAX a v neposlední řadě i TinyMCE, které zjednodušuje administrátorovi vkládání textů do formuláře. Nakonec byla vytvořena kompletní webová aplikace zabývající se elektronickými formuláři.

Hlavní myšlenkou při tvorbě webové aplikace, nazvané „EForms“, bylo přiblížení uživatelského prostředí k běžným desktopovým aplikacím a také zjednodušení jednotlivých kroků při práci, což ocení zejména uživatelé aplikace.

Výhodou je zpřístupnění formulářů z kteréhokoli počítače připojeného k síti internet. Také není zapotřebí mít nainstalován žádný složitý software na každé pracovní stanici, ale stačí pouze webový prohlížeč. Tím se ušetří spousta nákladů na tento software.

Strategií vytváření formulářů bylo uchování vztahů mezi jednotlivými objekty. To znamená, že každý objekt vložený do formuláře má svůj nadřazený objekt („rodič“) a objekty, které jsou tvořeny párovými tagy, můžou mít i objekt podřazený („dítě“). Tato strategie je dodržována při generování formuláře, ale také při jeho výpisu do zdrojového kódu.

Vytvoření daných tabulek MySQL bylo dobrou volbou, jen by se mohlo vygenerovat více tabulek „dom“ pro lepší přehlednost a zrychlení práce s databází MySQL. Tyto tabulky by mohly být generovány jednotlivě pro každý formulář. Tím by se zmenšil počet záznamů v každé tabulce a data by se stala přehlednější. Ukládání jednotlivých objektů ihned při vložení do databáze MySQL bylo zřejmě také dobré řešení.

Jako každá webová aplikace se musí vyvíjet, tak i tato bude vyvíjena nadále takovým způsobem, aby se její uživatelské rozhraní co nejvíce přiblížilo desktopovým aplikacím. Také bude aplikace rozšířena o řadu dalších funkcí, které by se mohly uplatnit třeba na univerzitě.

V opačném případě by se tato aplikace stala zanedlouho digitálním odpadem, jelikož vývoj webových technologií jde rychlým krokem kupředu.

ZÁVĚR V ANGLIČTINĚ

The subject matter of my thesis was focused on creating of the graphic interface for working with forms on the Internet which would make generating and controlling of forms possible. This application should fully replace paper forms. Graphical user interface was programmed with the help of web technologies HTML, CSS, PHP5 and JavaScript. No less important technologies, such as Ajax and TinyMCE were added. This tool simplifies inserting texts into the form. In the end was created the complete e-form web application.

The main idea when creating this web application, called EForms, was bring the user interface closer to the common desktop applications and also simplify each step at work, which will appreciate mostly the users of the application.

The advantage is retrieval of forms from whichever computer connected to the Internet. Also there is no need for anyone to have some complex software installed on each workstation. A simple web browser is enough. It saves money through the mediation of cutting down the expenses for such a software.

The strategy of creating the forms was to keep each object related with other objects. That means each object inserted into the form has it's own superior object ("parent") and objects which are created by pair tags, can have also object submissed ("child"). This strategy is kept when generating the form, but also during it's printout into the source code.

The creation of given MySQL tables was a good choice, but for a better lucidity and making the work with MySQL database faster, more "dom" tables could be created. These tables could be generated individually for every form. This would make the number of records lower in each table and data would be more transparent. Saving of individual objects immediately after inserting into the database MySQL came right as a good solution.

Every web application has to be developed and this one is not an exception. It will be developed from now so that its user interface would bring itself as much close to the desktop applications as possible.

The application will be also extended by many functions which could be applied for example at the University. Otherwise this application would become a digital rubbish soon because evolution of web technologies is making a fast progress.

SEZNAM POUŽITÉ LITERATURY

[1] SCHLOSSANAGLE, GEORGE. *Pokročilé programování v PHP5*. Zoner Press: Brno, 2004.

[2] PROKOP, MAREK. *CSS kaskádové styly pro webdesignéry*. CP Books a.s.: Brno, 2005.

[3] DELLWING, E., DELLWING, I. *JavaScript příručka programátora.* Grada Publishing a.s.: Praha, 2003.

[4] DARIE, C., BRINZAREA, B., CHERECHES-TOSA, F., BUCICA, M. *AJAX a PHP – tvoříme interaktivní webové aplikace PROFESIONÁLNĚ*, Zoner Press: Praha, 2006.

[5] GILMORE, W., J. *Velká kniha PHP5 a MySQL*. Zoner Press: Praha, 2005.

[6] MASLAKOWSKI, M. *Naučte se MySQL za 21 dní*. Computer Press: Praha, 2001.

[7] PLAVÁČEK: *Použitelné a přístupné webové formuláře.* [online]. [cit. 2007-04-28]. Dostupný z WWW: < http://www.plavacek.net/formulare/index.html>.

[8] JAK PSÁT WEB: *Formuláře.* [online]. [cit. 2007-04-26]. Dostupný z WWW: <http://www.jakpsatweb.cz/html/formulare.html>.

[9] WIKIPEDIE: *Cascading Style Sheets.* [online]. [cit. 2007-04-10]. Dostupný z WWW: < http://cs.wikipedia.org/wiki/Cascading_Style_Sheets>.

[10] PECAN.CZ: *Co je to CSS?* [online]. [cit. 2007-04-12]. Dostupný z WWW: <http://pecan.cz/index.php?id=16>.

[11] BUILDER: *Co je to JavaSript?* [online]. [cit. 2007-04-12]. Dostupný z WWW: <http://www.builder.cz/art/javascript/js_seznameni.html>.

[12] ADAPTIC: *Co je to HTML?* [online]. [cit. 2007-04-18]. Dostupný z WWW: <http://www.adaptic.cz/znalosti/slovnicek/html.htm>.

[13] WIKIPEDIE: *HTML.* [online]. [cit. 2007-04-12]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/HTML>.

[14] WIKIPEDIE: *MySQL*. [online]. [cit. 2007-04-12]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/Mysql>.

[15] ADAPTIC: *MySQL*. [online]. [cit. 2007-04-12]. Dostupný z WWW: <http://www.adaptic.cz/znalosti/slovnicek/mysql.htm>.

[16] DATABÁZOVÝ SVĚT: *MySQL*. [online]. [cit. 2007-04-14]. Dostupný z WWW: <http://www.dbsvet.cz/view.php?cisloclanku=2004030102>.

[17] OWEBU.CZ: *TINYMCE*. [online]. [cit. 2007-04-19]. Dostupný z WWW: <http://www.owebu.cz/obecne/vypis.php?clanek=897>.

[18] DATABÁZOVÝ SVĚT: *MySQL*. [online]. [cit. 2007-04-14]. Dostupný z WWW: <http://www.dbsvet.cz/view.php?cisloclanku=2004030102>.

[19] MOXIECODE: *What is TinyMCE?*. [online]. [cit. 2007-04-23]. Dostupný z WWW: <http://tinymce.moxiecode.com/>.

[20] LUPA: *Hashovaní funkce*. [online]. [cit. 2007-04-29]. Dostupný z WWW: <http://www.lupa.cz/clanky/hasovaci-funkce-jak-se-odolava-hackerum/>.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

id Identifikační číslo.

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

PŘÍLOHA P I: NÁVOD K VYTVOŘENÍ FORMULÁŘE

Přihlášení do aplikace

Nejprve je nutné se do aplikace přihlásit. Toto provede tak, že vyplníte políčka pro přihlašovací jméno a heslo. Do políčka jméno zadejte „admin“ a do políčka heslo zadejte heslo k tomuto účtu.

Pokud přihlášení proběhlo úspěšně, zobrazí se administrační rozhraní (viz. Obrázek 1). Pokud přihlášení selhalo, nic se nestane.

Obrázek 1 – Administrátorské rozhraní pro správu formulářů

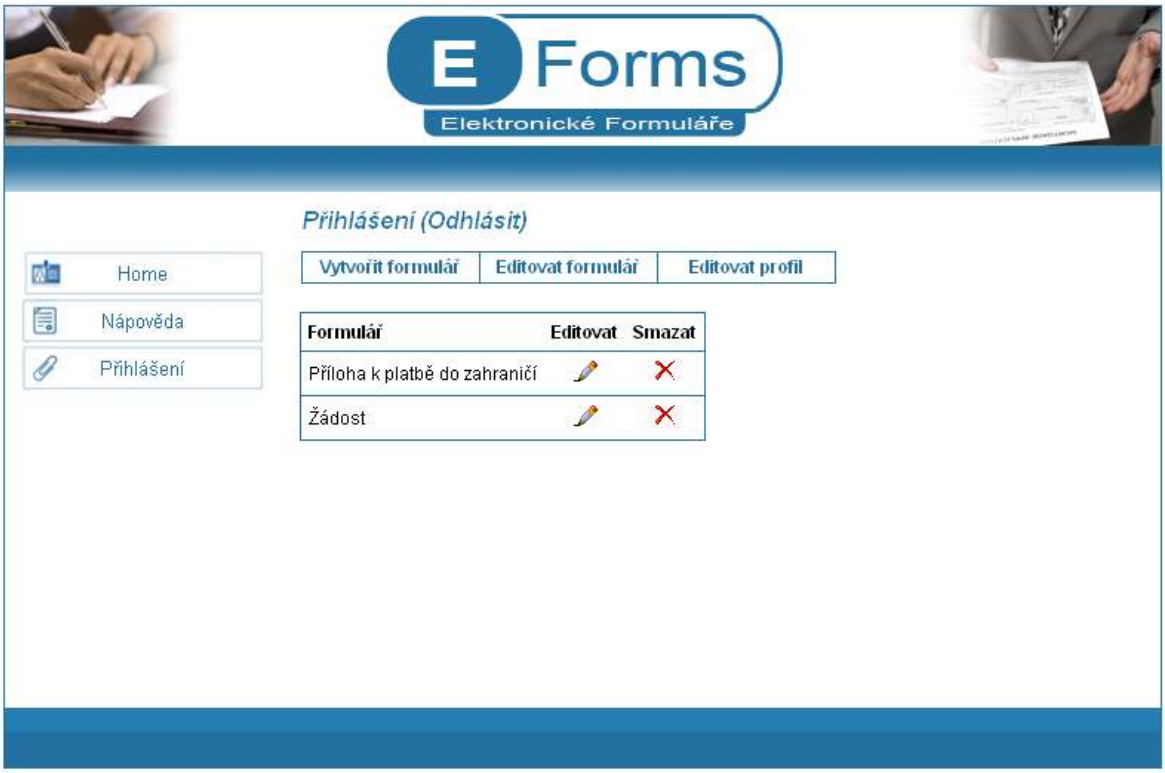
Odhlášení

Odhlášení ze systému provedete kliknutím na odkaz „Odhlásit“ vedle nadpisu stránky.

Vytvoření formuláře

Kliknutím, na položku „Vytvořit formulář“, se zobrazí pole pro zadání názvu nového formuláře. Zde zadejte „Žádost“. Po kliknutí na tlačítko „OK“ se vypíše hláška „Formulář byl úspěšně vytvořen!“. Jeho návrh zrealizujte v menu „Editovat formulář“. V takovém případě klikněte na položku menu „Editovat formulář“.

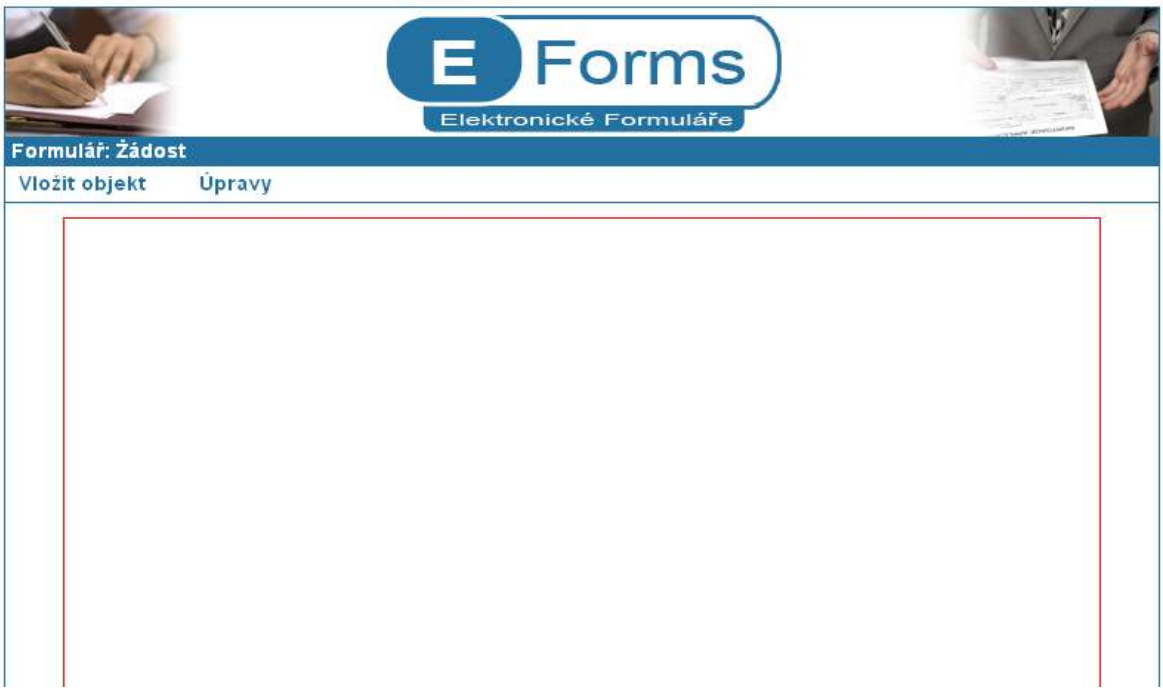
Zobrazí se zde všechny dosud vytvořené formuláře (viz. Obrázek 2).



Obrázek 2 – Vytvořené formuláře

Editace formuláře

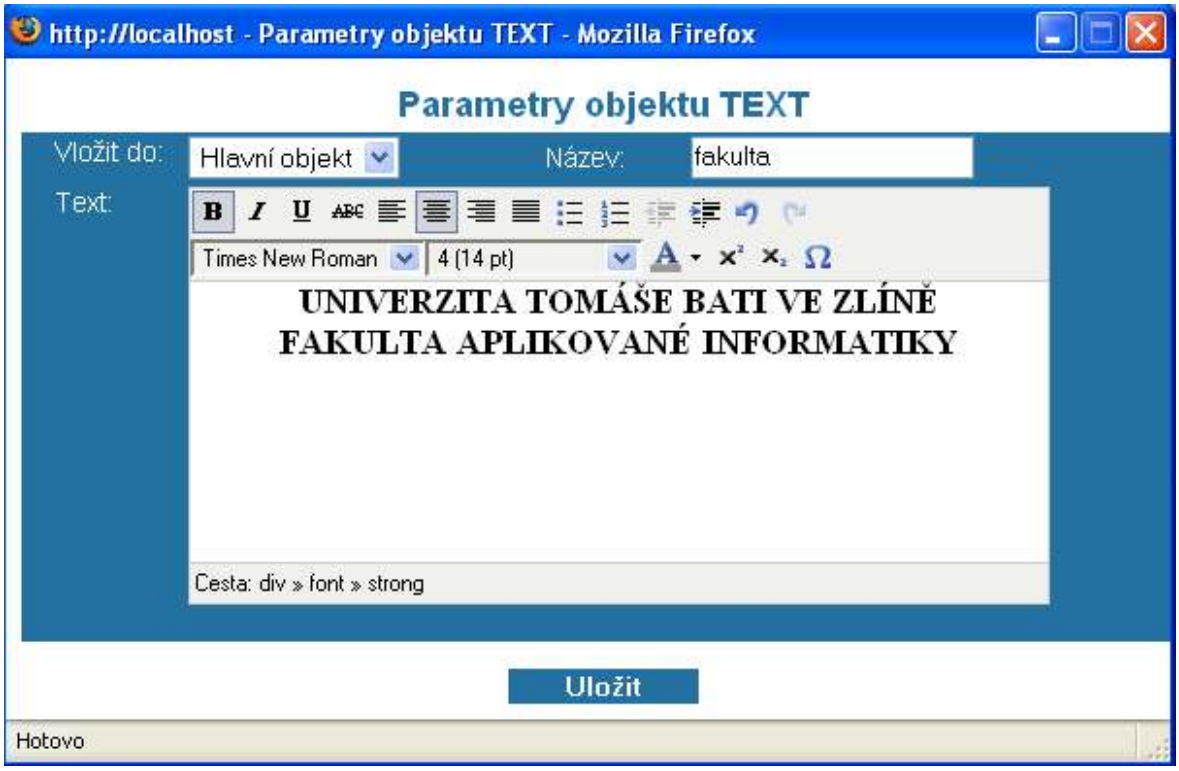
Kliknutím na ikonku „editovat“ u formuláře „Žádost“ se zobrazí nové okno pro návrh formuláře (viz. Obrázek 3).



Obrázek 3 – Okno pro vytváření a editaci formulářů

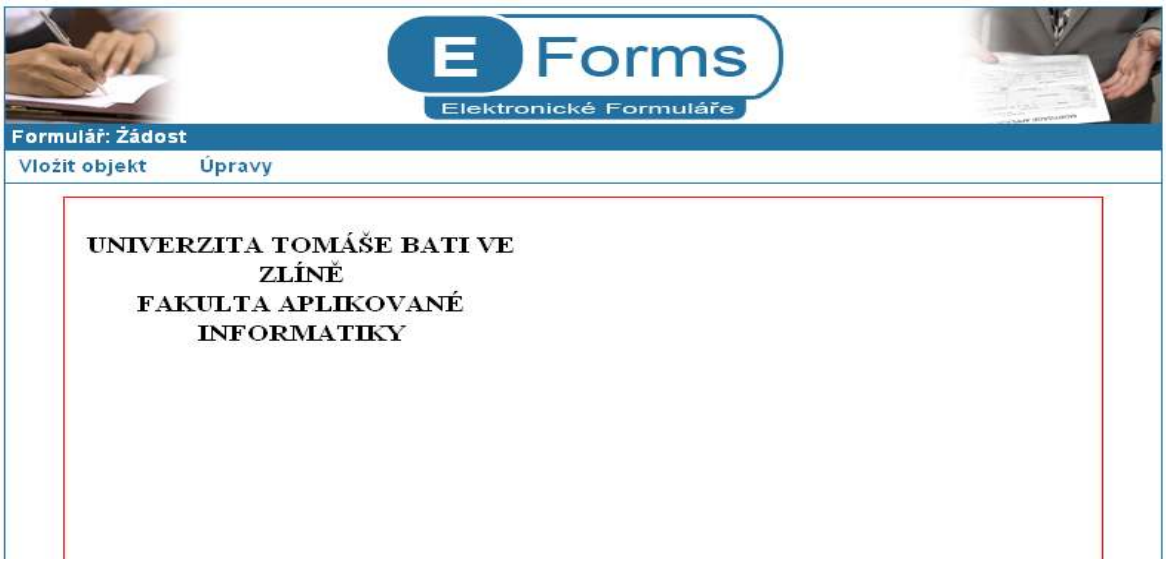
Vytváření samotného formuláře

Klikněte na položku menu „Vložit objekt“. V této nabídce zvolte „Text“. Zobrazí se nové okno, do kterého zadáte parametry tohoto objektu (viz. Obrázek 4).



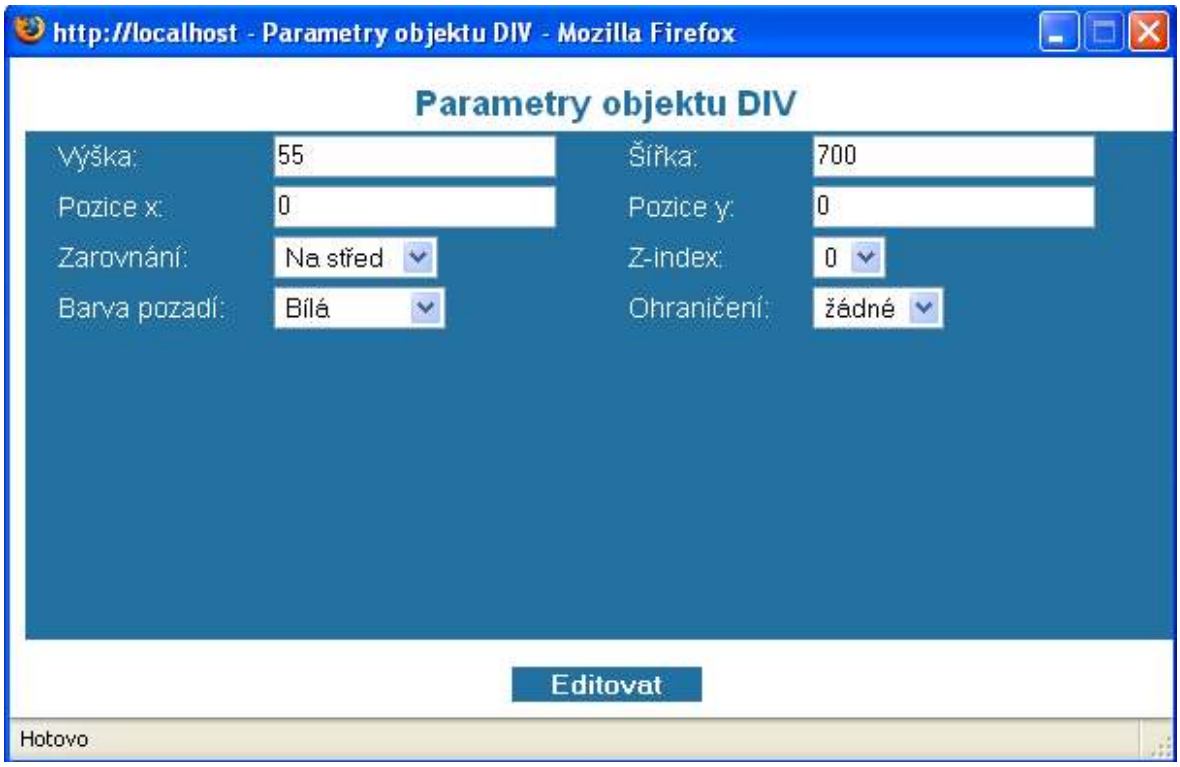
Obrázek 4 – Parametry objektu „Text“

Po kliknutí na tlačítko „Uložit“ se aktuální okno zavře a do formuláře je vložen text s zadanými parametry. Návrhové okno bude vypadat následovně (viz. Obrázek 5).



Obrázek 5 – Návrhové okno po vložení textu

Teď je na řadě umístění textu na pozici, kterou požadujeme. Přejeďte myší na položku menu „Úpravy“. Vyberte z možností položku „Editovat“. Objeví se seznam možností na výběr. Z tohoto seznamu vyberte „divs-fakulta“ a klikněte na tlačítko „Editovat“. Do nově otevřeného okna zadejte hodnoty políček (viz. Obrázek 6).

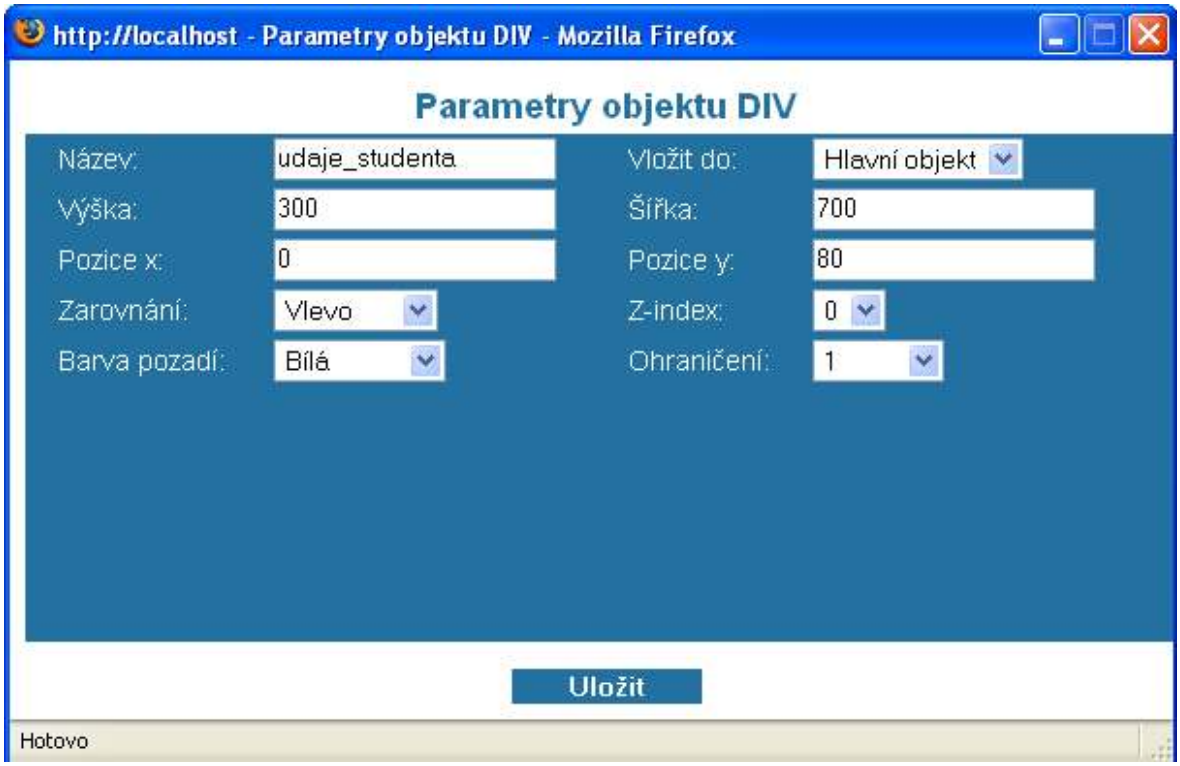


Obrázek 6 – Editované parametry objektu text

Potom klikněte na tlačítko „Editovat“. Tento text bude zobrazen uprostřed stránky formuláře (viz. Obrázek 7).

Obrázek 7 – Editovaný text

Pokračujte vložením objektu div, který nám poslouží jako rámeček, do kterého budou přidány informace o studentovi. Toto vložení provedete stejným způsobem jako vložení textu. Vyberete položku „Div“ a zadáte parametry objektu div (viz. Obrázek 8).



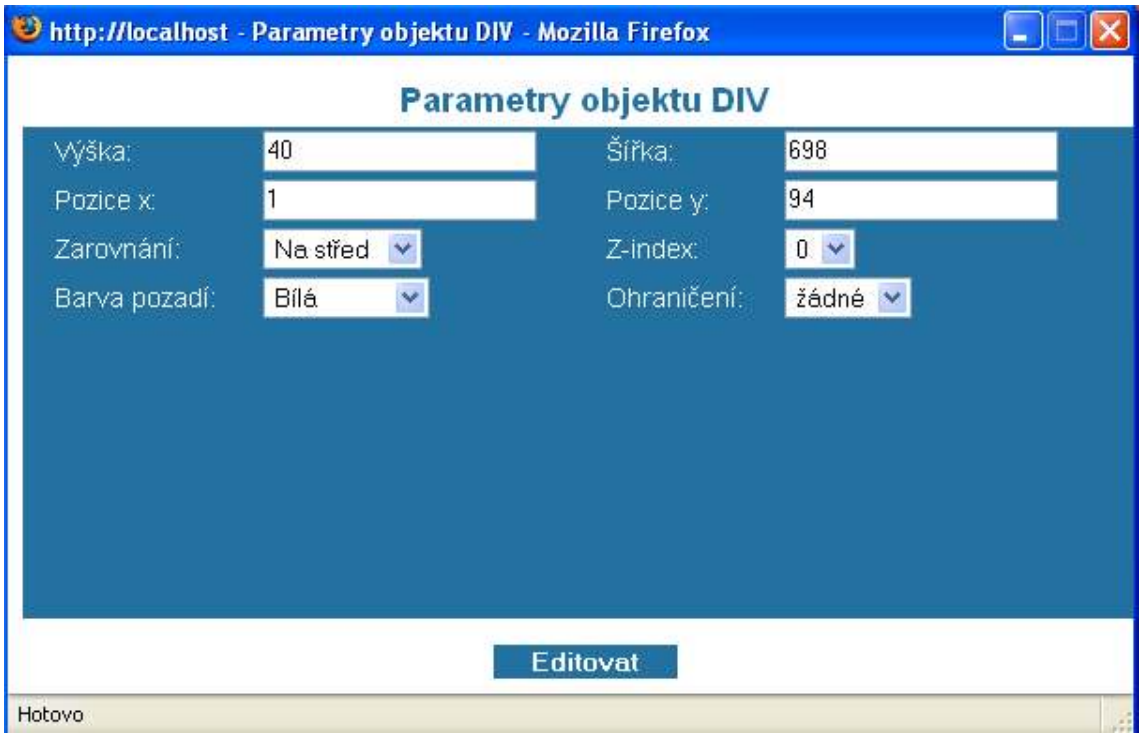
Obrázek 8 – Parametry rámečku

Kliknutím na tlačítko se div vloží do formuláře a vytvoří požadovaný rámeček.

Dalším krokem je vložení textu „ŽÁDOST“ do vytvořeného rámečku. Toto provedete stejným způsobem jakým jste vložili první text z jinými parametry (viz. Obrázek 9).

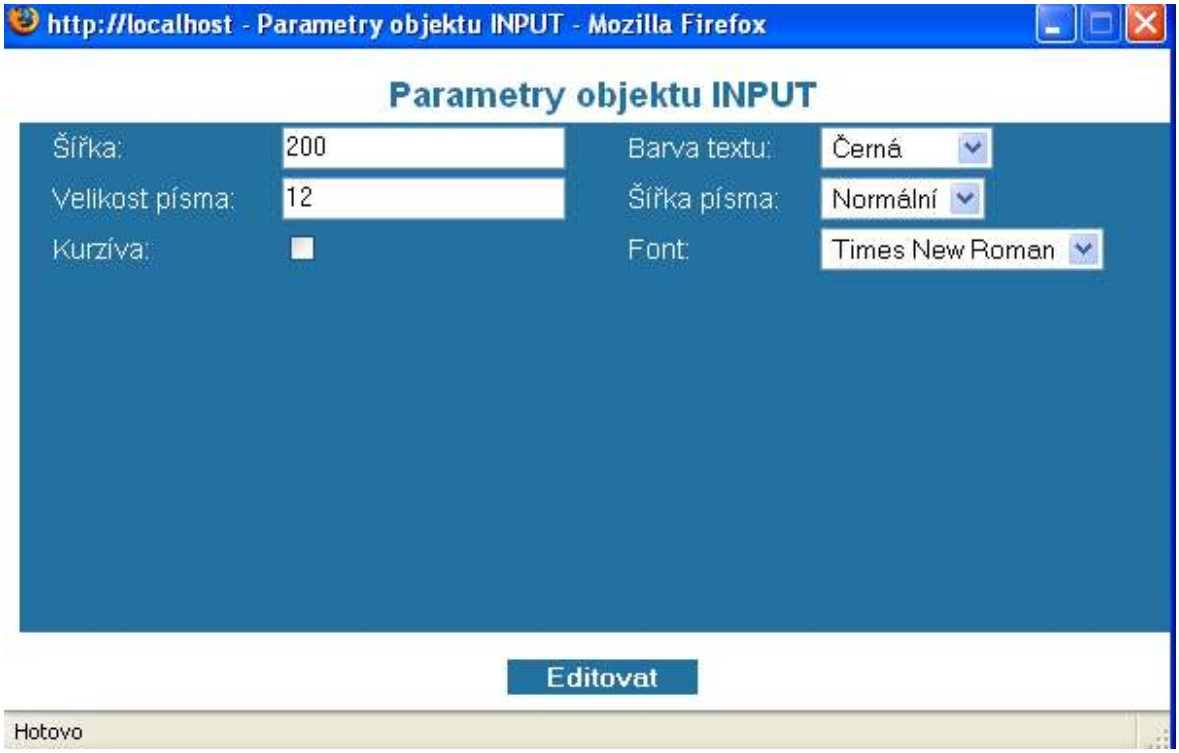
Obrázek 9 – Parametry textu „ŽÁDOST“

Následovně upravíme parametry divu tak, abychom dostali požadované umístění textu „ŽÁDOST“ (VIZ. Obrázek 10).



Obrázek 10 – Parametry editovaného objektu

Kliknutím na položku „Input“ v menu „Vložit objekt“ se zobrazí nové okno pro zadání parametrů objektu input. Parametry jsou následující (viz. Obrázek 11).



Obrázek 11 – Parametry vkládaného objektu input

Nyní upravíme parametry divu pro přesné umístění vstupního pole input. To můžete udělat výše uvedeným způsobem nebo označit daný div kombinací klávesy Ctrl a levým tlačítkem myši. Objekt se označí zeleným rámečkem a teď už stačí jen kliknout na položku „Editovat“ v menu „Úpravy“. Parametry budou následující (viz. Obrázek 12).

Obrázek 12 – Parametry editovaného divu pro vstupní pole input

Kliknutím na „Popisek“ v položce menu „Vložit objekt“ vytvoříte popisek pro vstupní pole se zadanými parametry (viz. Obrázek 13).

Obrázek 13 – Parametry vkládaného popisku

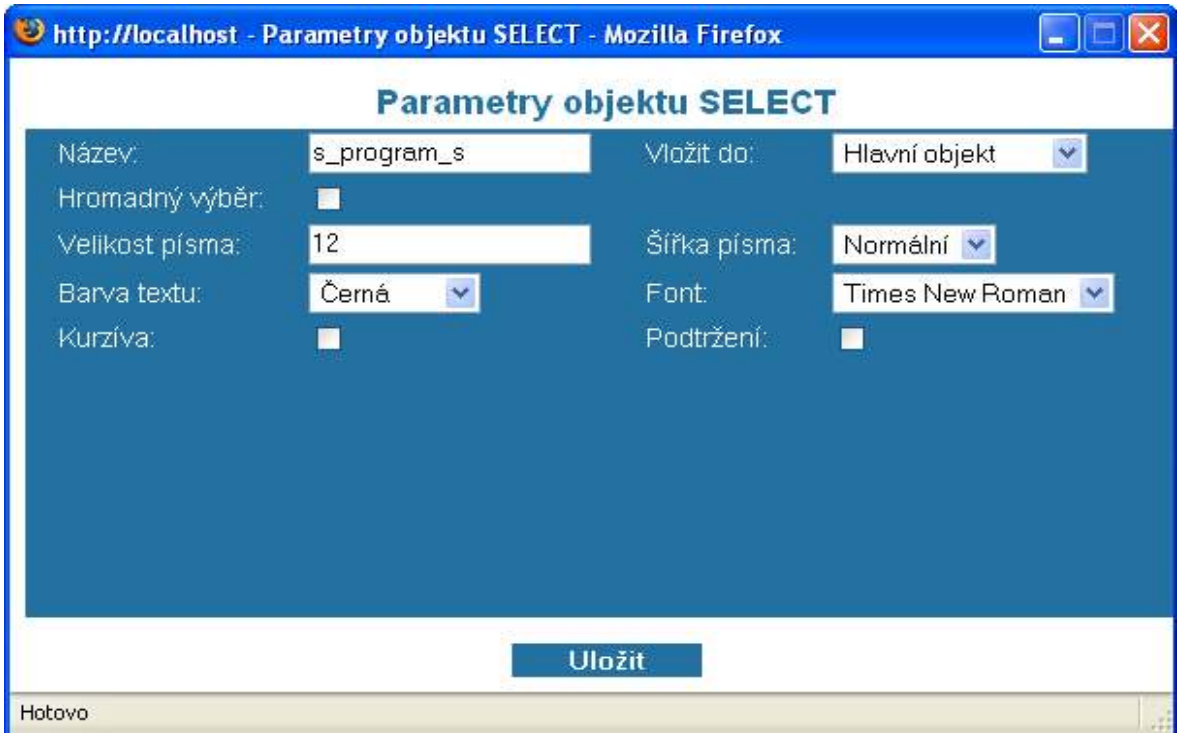
Nyní je zapotřebí opět upravit parametry divu, ve kterém je popisek vložen. Tyto parametry jsou následující (viz. Obrázek 14).

Obrázek 14 – Úprava parametrů editovaného divu

Postup při vkládání dalších popisků a vstupních polí je totožný. Liší se opět jen v zadaných parametrech. Postupujte stejným způsobem jako v předcházejících krocích. Předloha pro další kroky viz. Obrázek 15.

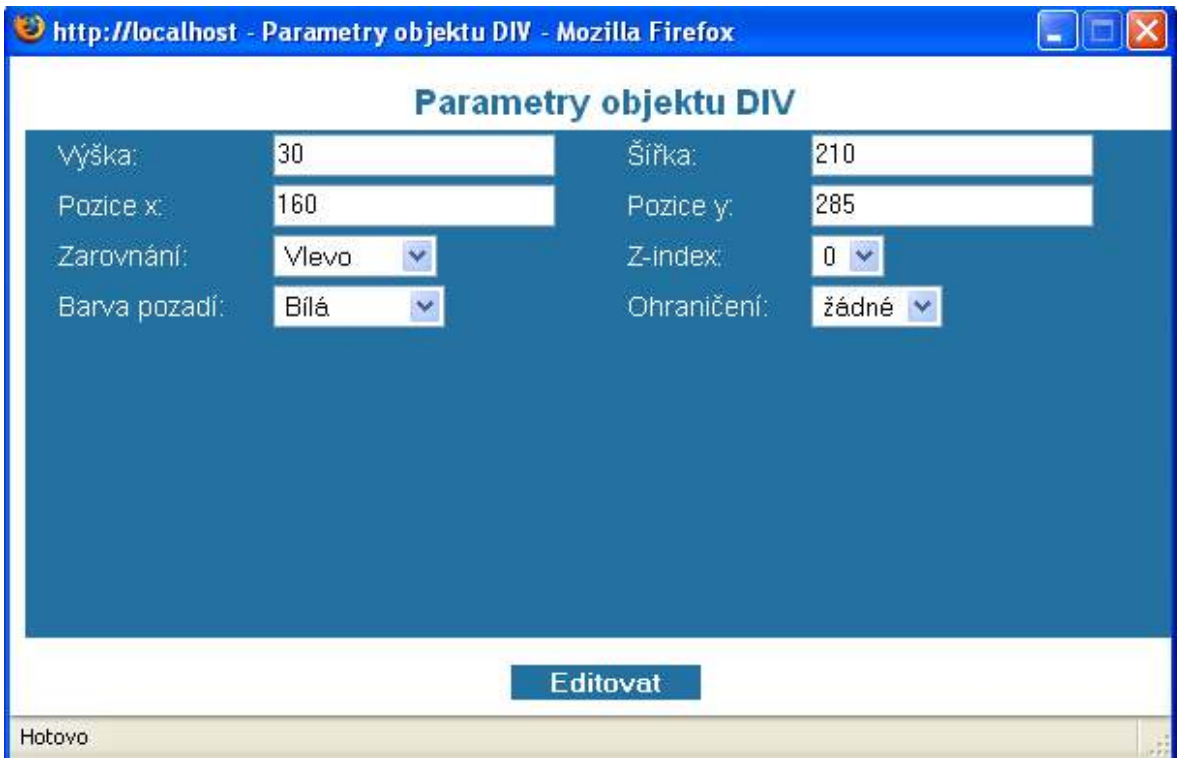
Obrázek 15 – Výsledek po dalších krocích

Vložením objektu select nahradíme formulářový slogan „nehodící se škrtněte“. To provedete tak, že kliknete na položku „Select“ v menu „Vložit objekt“. Opět se otevře nové okno, kde zadáte následující parametry (viz. Obrázek 16).



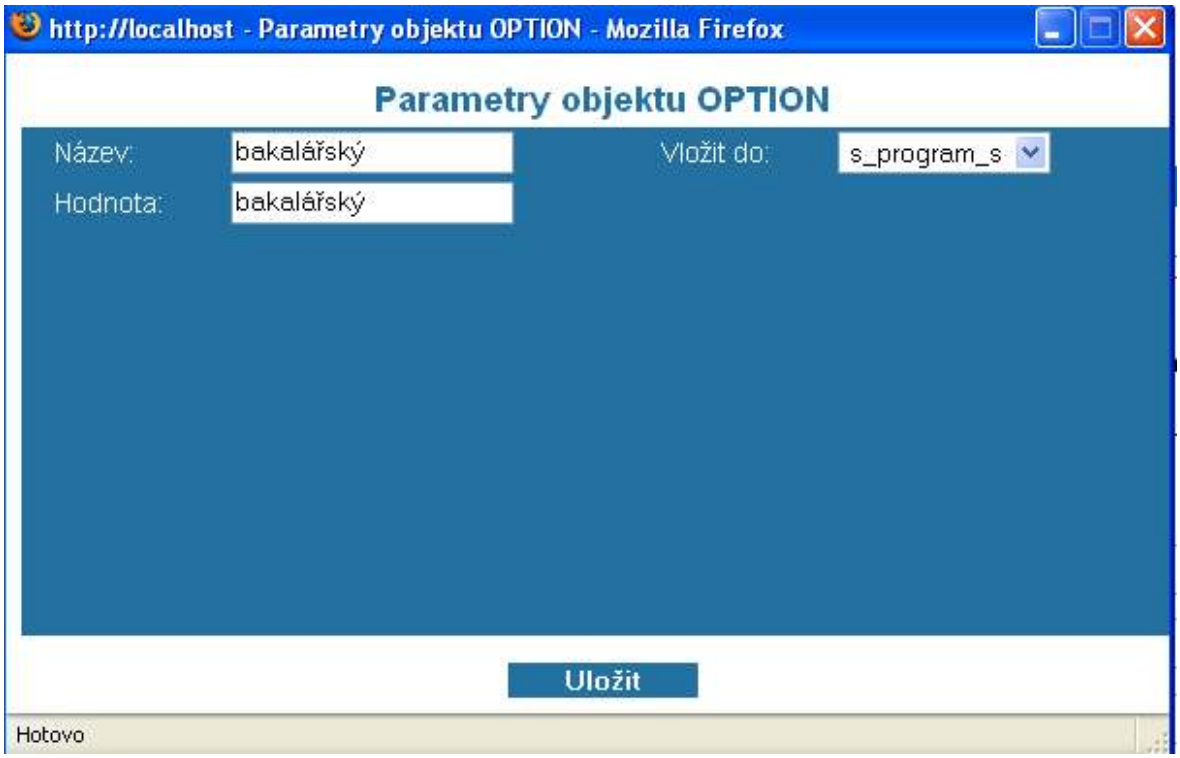
Obrázek 16 – Parametry objektu select

V tomto okamžiku opět upravíme objekt div, ve kterém je objekt select vložen. Parametry jsou následující (viz. Obrázek 17).



Obrázek 17 – Parametry vloženého objektu select

Novou věcí je přidání možností do samotného výběrového pole neboli objektu select. To provedete kliknutím na položku „option“ v menu „Vložit objekt“. Opět se otevře nové okno, kde zadáte následující parametry (viz. Obrázek 18).



Obrázek 18 – Přidání položky do výběrového pole

Další položka se vloží naprosto stejným způsobem jako položka předcházející. Změní se pouze políčka „Název“ a „Hodnota“. Do obou těchto polí zadáte „magisterský“.

K dokončení vytvoření části pro údaje studenta je zapotřebí ještě vložit popisek pro výběrové pole. Toto uděláte stejně jako v předcházejících případech. Pouze do políčka „Text“ vepíšete slovní spojení „Studijní program:“ a objekt div obklopený tímto objektem nastavíte na stejné souřadnice.

Poslední operaci, kterou musíte udělat je zmenšení výšky objektu div na 250 pixelů, který nám vykresluje rámeček.

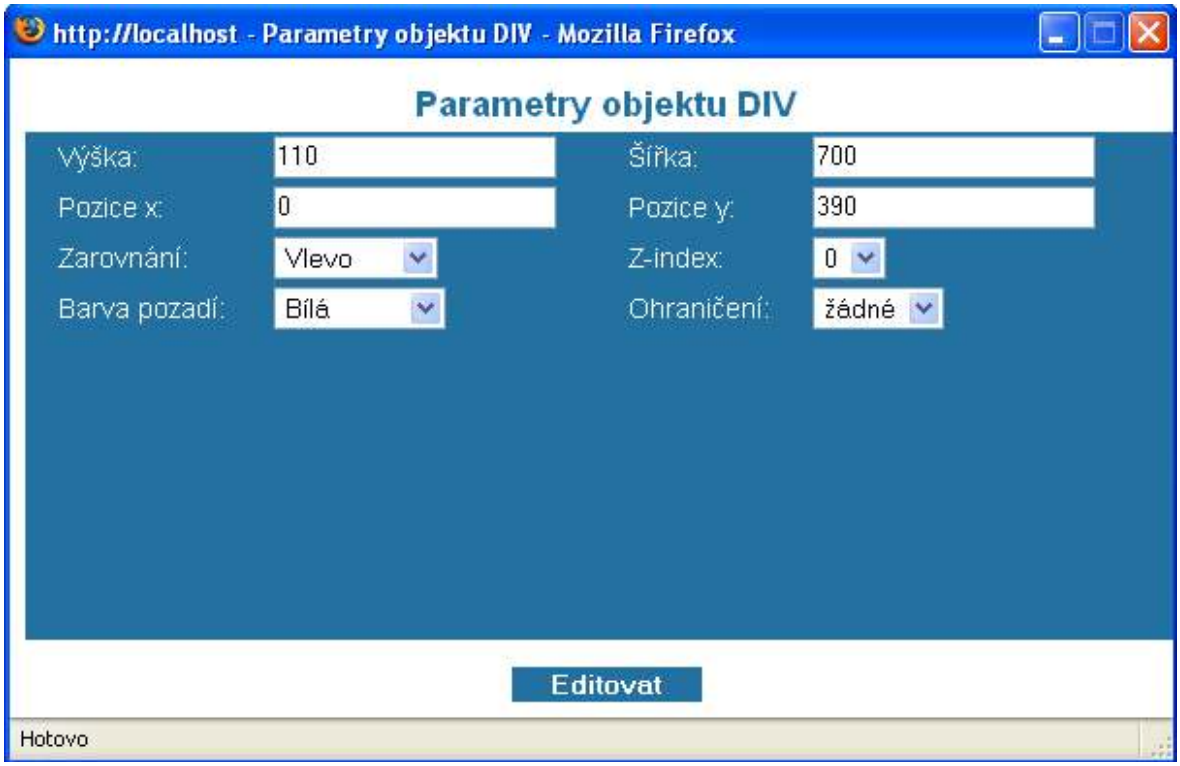
Tímto jste dokončili vytvoření části formuláře pro zadávání údajů studenta. Náhled jak by měl formulář vypadat naleznete na následujícím obrázku (viz. Obrázek 19).

Obrázek 19 – Náhled navrhnuté části formuláře

Nyní klikněte na položku „Textarea“ v menu „Vložit objekt“. Objeví se nové okno s parametry objektu textarea. Jejich parametry zadejte následovně (viz Obrázek 20).

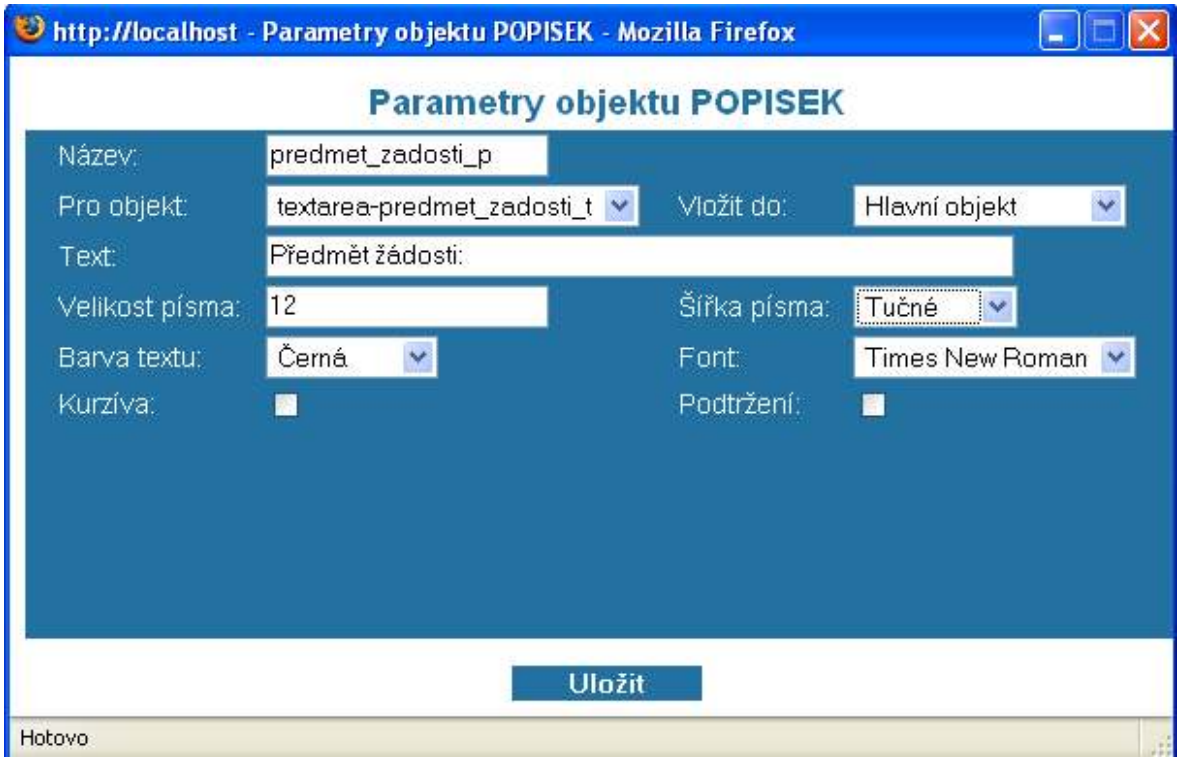
Obrázek 20 – parametry objektu textarea

Stisknutím levého tlačítka a pohybem myši přesuňte objekt textarea pod část osobních údajů studenta. Div obklopený tímto objektem editujte kliknutím na položku „Editovat“ v menu „Úpravy“. Následně vyberte jméno objektu „divs-predmet_ zadosti_t“ pro editaci. Parametry editujte následovně (viz. Obrázek 21).



Obrázek 21 – Parametry objektu div

Klikněte na menu „Vložit objekt“ a zvolte položku „Popisek“. Parametry, které zadáte jsou uvedeny na následujícím obrázku (viz. Obrázek 22).

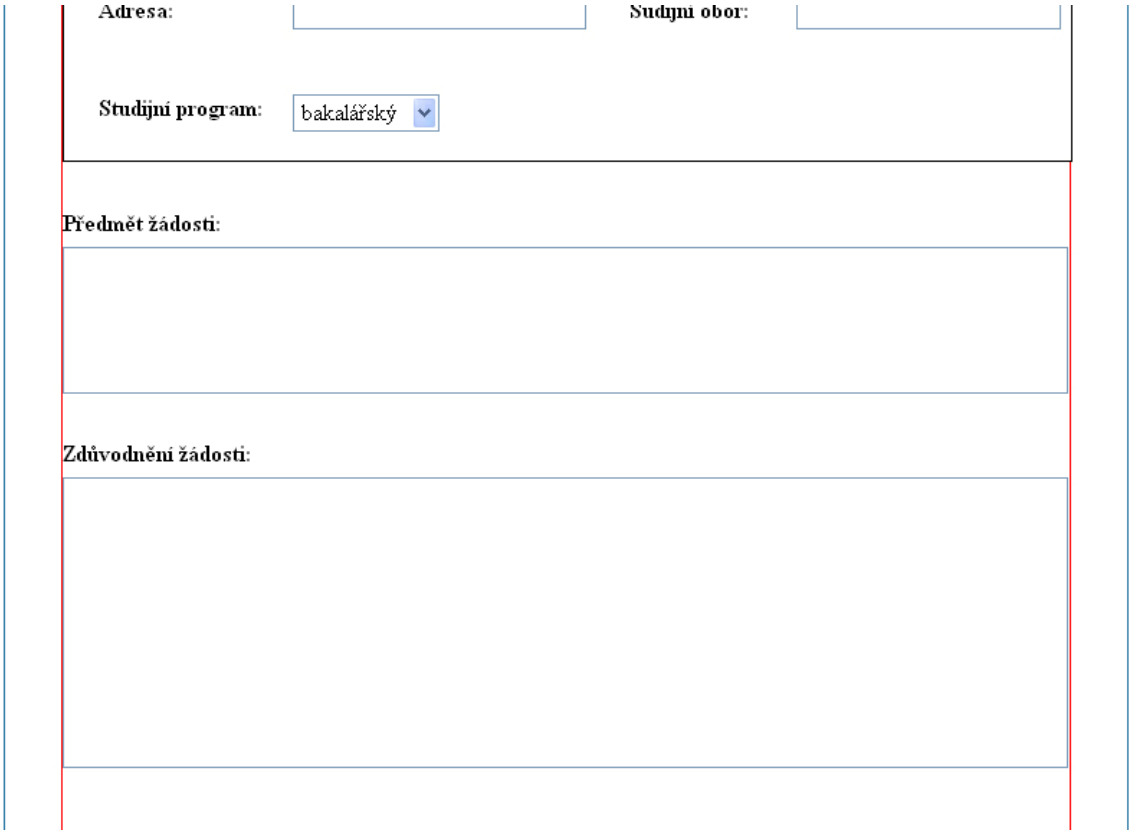


Obrázek 22 – Parametry objektu popisek

Nyní editujte parametry divu, který tento objekt obklopuje. Parametry budou vypadat následovně (viz. Obrázek 23).

Obrázek 23 – Parametry objektu div

Vložení objektu textarea proveďte ještě jednou včetně jeho popisku. Objekty budou umístěny pod naposled vytvořeným objektem textarea ve stejném stylu. Formulář po těchto úpravách můžete vidět na následujícím obrázku (viz. Obrázek 24).



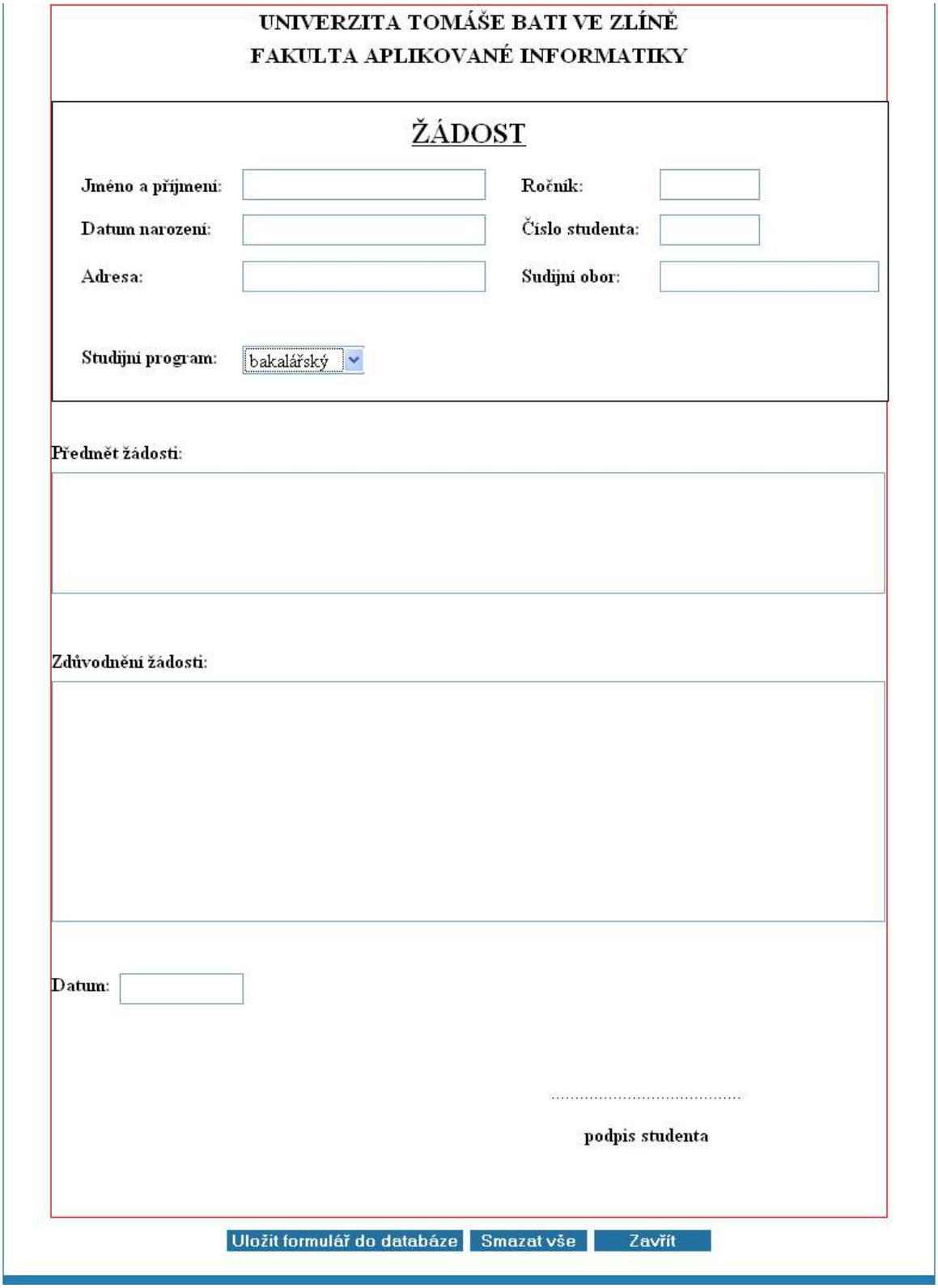
Obrázek 24 – Formulář po předcházejících úpravách

Posledními úkoly pro dokončení tohoto formuláře bude vytvořit políčko pro zadávání datumu a políčko pro podpis studenta.

Vstupní pole datumu bude klasický objekt input, Pro určení místa podpisu studenta použijeme objekt text, který bude mít hodnotu několika teček.

Tyto objekty umístěte na konec formuláře.

Jestliže jste postupovali správně, došli jste ke stejnému výsledku jaký je na následujícím obrázku (viz. Obrázek 25). Teď již zbývá jen formulář uložit a zveřejnit běžným uživatelům. To provedete kliknutím na tlačítko „Uložit formulář do databáze“ v dolní části návrhového okna.



Obrázek 25 – Náhled konečné verze formuláře