SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE

Fakulta chemickej a potravinárskej technológie

Ústav informatizácie, automatizácie a matematiky

Web aplikácia na spracovanie a analýzu údajov

Vypracoval: Jana Kmeťová

Školiteľ: Ing. Ľuboš Čirka, PhD.

Humenné 2008

Slovenská technická univerzita v Bratislave **Fakulta chemickej a potravinárskej technológie**
Oddelenie informatizácie a riadenia procesov **Akademický rok: 2007/2008**

ZADANIE BAKALÁRSKEJ PRÁCE

Autor práce:	**Jana Kmeťová (28076)**
Študijný program:	automatizácia, informatizácia a manažment v chémii a potravinárstve
Kombinácia študijných odborov:	5.2.14 automatizácia, 5.2.52 priemyselné inžinierstvo
Vedúci práce:	Ing. Ľuboš Čirka, PhD.
Miesto vypracovania:	Humenné

Názov témy: **Web aplikácia na spracovanie a analýzu údajov**

Rozsah práce: 30 strán

Dátum zadania bakalárskej práce: **18. 02. 2008**
Termín odovzdania bakalárskej práce: **23. 05. 2008**

L. S.

Jana Kmeťová
riešiteľ bakalárskej práce

prof. Ing. Dr. Miroslav Fikar
vedúci pracoviska

prof. Ing. Dr. Miroslav Fikar
garant študijného programu

Ďakujem vedúcemu bakalárskej práce
Ing. Ľubošovi Čirkovi, PhD. za pomoc pri
získavaní vedomostí z oblasti programovania
v jazykoch HTML, PHP a SQL (MySQL)
ako i celkovo za vedenie, rady a pripomienky,
ktoré mi poskytol pri vypracovaní bakalárskej práce

Abstrakt

Cieľom práce je vytvoriť modul na vyúčtovanie zahraničnej pracovnej cesty, ktorý bude súčasťou informačného systému Ústavu informatizácie, automatizácie a matematiky na Fakulte chemickej a potravinárskej technológie v Bratislave. Modul je vytvorený v jazyku HMTL, PHP a JavaScript s podporou databázy MySQL. V podstate sa jedná o elektronický formulár, ktorého údaje sú ukladané do databázy. Okrem formulára je vytvorená aj výstupná šablóna vo formáte PDF. Výsledný PDF dokument, vyplnený údajmi z databázy, reprezentuje elektronickú formu tlačiva na vyúčtovanie zahraničnej pracovnej cesty. Vyplňovanie údajov je veľmi jednoduché, rýchle a v prípade chybného zadania sa údaje dajú ľahko opraviť a meniť.

Abstract

The goal of this project is to develop an electronic travel expense claim form, a module which will be built into the information system currently used by the Institute of informatics, automatisation and mathematics at the Faculty of chemistry and food technology in Bratislava. It si programmed with HTML, PHP and Java Script tools and is supported by the MySQL database, where the data is stored. The submitted form can be exported into a PDF file using the custom-written output template for printing and archivation. The entry of data is very simple and intuitive and the data can be easily edited and corrected.

Obsah

Zoznam skratiek

ARPA	Advanced Research Projects Agency (Agentúra pokrokových výskumných projektov)
ARPANET	Advanced Research Project Agency Network (Sieť Projektovej agentúry pre moderný výskum)
PHP	Hypertext Preprocessor (Skriptovací programovací jazyk)
PERL	Practical Extraction and Reporting Language (Programovací jazyk)
C	Programovací jazyk
HTML	HyperText Markup Language (Hypertextový značkový jazyk)
WWW	World Wide Web (Celosvetová sieť)
MySQL	Databázový systém
SQL	Structured Query Language
DDL	Data Definition Language
SDL	Storage Definition Language
VDL	View Definition Language
DML	Data Manipulation Language
DTD	Document Type Definition (Definícia typu dokumentu)
XHTML	eXtensible Hypertext Markup Language (Hypertextový značkovací jazyk)
PDF	Portable Document Format (Súborový formát)

Zoznam obrázkov

Úvod

V dnešnej modernej dobe, kde už skoro všetky úradné tlačivá a formuláre nahrádzajú formuláre v elektronickej forme, vznikajú elektronické dokumenty, ktoré šetria čas, peniaze a sú efektívnejšie. Dajú sa jednoducho upravovať, meniť alebo tlačiť.

Cieľom mojej bakalárskej práce bolo vytvorenie webovej aplikácie na spracovanie a analýzu údajov. Tento elektronický formulár bude slúžiť pre zamestnancov ústavu, ktorí budú vyslaní na zahraničnú pracovnú cestu. K vyplnenému formuláru sa bude môcť zamestnanec hocikedy vrátiť a doplniť, prípadne zmeniť vyplnené údaje. Takto vyplnený formulár je možné vytlačiť a zaradiť do evidencie. Formulár má uľahčiť zaznamenávanie a spracovanie údajov oproti starému zaužívanému ručnému vypisovaniu tlačív.

1 TEORETICKÁ ČASŤ

1.1 História počítačovej siete [1]

Začiatkom šesťdesiatych rokoch sa objavili prvé myšlienky na vytvorenie siete, ktorá by navzájom prepojovala najdôležitejšie vojenské, vládne a vedecko-výskumné počítače. Malo ísť o sieť, ktorá bude fungovať aj v prípade výpadku niektorého z uzla. Pracovníci RAND Corporation prišli s riešením – vybudovať siete bez centrálneho uzla. Ak bude niektorá linka zničená, informácia bude hneď vedená k príjemcovi inou trasou. Preto bola v USA vládou založená organizácia pod názvom Advanced Research Projects Agency (ARPA), ktorú poverili špeciálnym výskumom. Prvá počítačová sieť bola testovaná začiatkom roku 1968 v Národnom výskumnom laboratóriu vo Veľkej Británii, čo bolo iba malé spojenie v rámci jednej budovy. V krátkom čase prišlo americké ministerstvo obrany s požiadavkou vybudovať podobnú sieť, ktorá dostala označenie ARPANET. Ukázala sa byť veľmi úspešná a preto rástol záujem o jej pripojenie. Hlavným využitím tejto siete bola elektronická pošta a okrem konferencií slúžiacim k výmene vedeckých informácií sa objavili aj konferencie určené pre zábavu. Postupne sa k tejto sieti pripojovali ďalšie inštitúcie, najmä univerzity. Sieť bola nekomerčná záležitosť, na jej vybudovanie prispela americká armáda a rôzne vládne agentúry. Podnikatelia o ňu nemali záujem, pretože nevedeli ako ju využiť. Preto sa uvádza, že v roku 1984 bolo k Internetu pripojených iba 1000 počítačov.

1.2 Jazyk PHP [4]

PHP je skriptovací jazyk pre tvorbu dynamického webu, ktorý sa zrodil v roku 1994. Jeho autor Rasmus Lerderf vytvoril jednoduchý systém pre počítanie prístupu k svojím stránkam, napísaný v jazyku PERL. O nejaký čas bol systém prepísaný do jazyka C, pretože perlovský kód dosť zaťažoval server. Sada skriptov bola vydaná v tom istom roku pod názvom „Personal Home Page Tools“, skrátene PHP.

Základná syntax je prevažne tvorená príkazmi, ktoré vykonávajú určitú funkciu a krok za krokom tak prebieha algoritmus vytvoreného programu.

```
<?php
prikaz1;
prikaz2;
. . .
?>
```

Najčastejšie používaným príkazom v PHP je príkaz echo, ktorý vypisuje zadaný text na štandardný výstup – do HTML stránky.

PHP umožňuje vkladať svoj programový kód priamo do zdrojového kódu stránok HTML. Na odlíšenie značkovania HTML od PHP kódu sa používa reťazec <?php ako začiatočný oddeľovač a ?> ako ukončovací oddeľovač PHP kódu. Kód PHP nemusí byť umiestnený na jednom mieste v stránke - môže sa kedykoľvek ukončiť, pokračovať kódom HTML a neskôr v stránke opäť vložiť príkazy PHP:

```
<html>
<head>
<title>Ukážka</title>
</head>
<body>
<?php
echo "<p>Toto je cez php</p>";
?>
<h1>Nadpis</h1>
<p>Toto je statická časť v html</p>
<?php
echo "<p>Toto je opäť cez php</p>";
?>
</body>
</html>
```

Žiaden program si nevystačí len s používaním príkazov a vyvolávaním funkcií. Obyčajne je potreba previesť určitú časť programu iba za splnenie istých podmienok. Ich použitie je jednoduché. Používajú sa vtedy, ak potrebujeme, aby sa program sám rozhodol ako bude pokračovať, ktorú časť skriptu vykoná alebo koľko krát ju bude opakovať. Najjednoduchší príkaz pre vetvenie predstavuje príkaz if. Jeho syntax znamená – ak je výraz podmienka vyhodnotený ako pravdivý, prevedie sa zoznam príkazov. Často však zároveň chceme nejaké príkazy previesť aj tak, ak podmienka splnená nie je. K tomu slúži časť else, ktorá znamená inak vykonaj zoznam príkazov.

```
if (podmienka) {
   zoznam príkazov
}
else {
   zoznam príkazov2
}
```

1.3 Cyklus for, while, do-while

Cykly umožňujú opakovanie určitých časti programu. V PHP sú k dispozícií tri druhy cyklov: for, while, do-while. Pretože vykonávajú v zásade to isté, sú medzi nimi drobné odlišnosti.

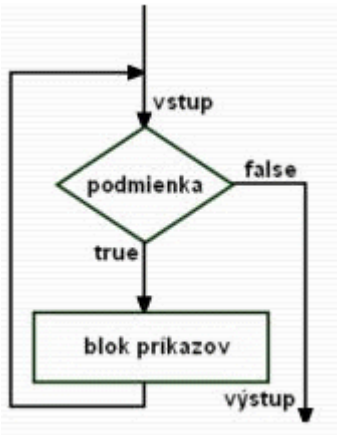
1.3.1 Cyklus for [6]

Je to druh cyklu s podmienkou na začiatku. Používa sa vtedy, ak vopred poznáme počet opakovaní cyklu. Pracuje sa s tromi výrazmi (štartovací výraz, ukončovací výraz, krok cyklu), ktoré sú uzavreté okrúhlymi zátvorkami oddelené bodkočiarkou. Prvý výraz udáva počiatočnú hodnotu pre opakovanie a vyhodnocuje sa iba raz pri vstupe do slučky. Druhým výrazom je podmienka, ktorá sa vyhodnocuje pred každým vstupom do tela slučky. Tretí výraz udáva, o koľko sa bude premenná zvyšovať - klasicky sa napríklad zväčší vždy o jednotku (i++).

```
for (vyraz_start; vyraz_stop; krok_cyklu) {
   // blok príkazov
}
```

1.3.2 Cyklus while [2]

Cyklus while sa používa v prípadoch, kedy nie je známy počet priechodov ako v prípade cyklu for, iba podmienka pre ukončenie. Základom tohto cyklu je, že podmienka sa testuje na začiatku. Znamená to, že cyklus sa v určitých prípadoch nemusí vykonať ani jeden krát.



Obr. 1: Cyklus while

Cyklus while je veľmi podobný podmienke if, čo môžeme vidieť z obrázku. Najskôr sa testuje podmienka a potom sa vykonajú jednotlivé cykly. Rozdiel je v tom, že ak má podmienka hodnotu true, vykoná sa blok príkazov a znova sa testuje, či ju spĺňa. Ak podmienka nadobudne hodnotu false, slučka skončí a vykoná sa prvý príkaz za slučku.

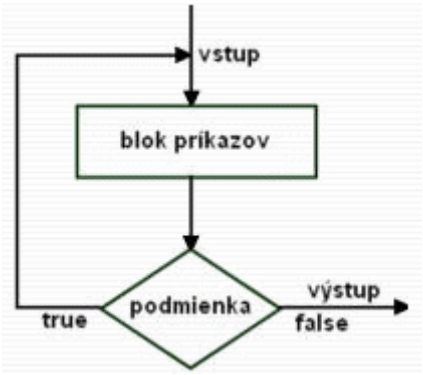
Základná syntax cyklu while je nasledujúca

```
while (podmienka)
{
  // blok príkazov
}
```

1.3.3 Cyklus do-while [3]

Menej často používaný cyklus je cyklus do-while, ale v štruktúrovanom programovaní má svoje nezastupiteľné miesto.

Tento príkaz sa podobá príkazu while. Veľmi dôležitý rozdiel je v tom, že podmienka sa vyhodnocuje až na konci cyklu. Blok príkazov sa vždy aspoň jeden krát vykoná, pretože je posunutý o jeden cyklus. Pri príkaze while tomu tak nemusí byť, ak sa hneď prvá hodnota nerovná požadovanej.



Obr. 2: Cyklus do-while

Syntax cyklu do-while vyzerá takto:

```
do {
   príkazy
} while (podmienka);
```

1.4 Príkaz exit

Príkaz exit slúži k ukončeniu spracovania celého skriptu, bez ohľadu na jeho aktuálny stav. Tento príkaz sa používa v prípade, že dôjde k chyba, po ktorej sa nedá rozumným spôsobom zotaviť. Alebo v prípade, že programátor je lenivý a danú situáciu nechce riešiť. Príkaz exit je možné ako parameter zadať celé číslo reprezentujúci chybný kód, či textový reťazec s chybným hlásením, ktoré sa pred ukončením skriptu vypíše.

```
<?php
…
if (!file_exists($subor))
exit (“Neexistuje požadovaný súbor”);
…
?>
```

1.5 Vývoj databázy [5]

Pojem databáza dnes už nie je určite nikomu cudzí. Ľudia majú potrebu evidovať a zhromažďovať informácie už odpradávna. Celá dnešná moderná spoločnosť je postavená na databázových systémoch, od evidencie občanov cez zdravotníctvo, hospodárstvo, školstvo, až po letectvo, výskum alebo sieť mobilných telefónov.

V 60. rokoch tohto storočia vznikali základné pojmy ako sú pojem databáza, entita, atribút entity a väzba medzi entitami. Databáza je súbor dát, ktoré slúžia pre popis reálneho sveta. Entita je prvok reálneho sveta, ktorý je opísaný svojimi charakteristikami. Tie sa väčšinou považujú za atribút. Jednotlivé entity odpovedajúce prvkom z reálneho sveta majú medzi sebou určitý vzťah, ktorý sa označuje väzba medzi entitami.

V súčasnej dobe sa najčastejšie používajú relačné databázy. Je to predovšetkým preto, že sa ľahko ovládajú a dobre reprezentujú bežné dátové štruktúry. Relačný databázový systém umožňuje užívateľom vytvárať jednotlivé samostatné databázy, pričom každá databáza môže obsahovať jednu či viac tabuliek. Tabuľky sú tvorené riadkami a stĺpcami. Riadky odpovedajú jednotlivým záznamom, dátovým vetám, a stĺpce atribútom, jednotlivým prvkom týchto dátových viet. Teda všetky dátové vety v jednej tabuľke majú rovnakú štruktúra atribútu dané štruktúrou tabuľky.

Každý atribút má dopredu určený dátový typ, ktorého hodnoty môže nadobúdať. Vo všetkých databázových systémoch sú k dispozícii celé čísla (integer), reálne čísla (numeric alebo decimal), reťazce (char alebo varchar), dátum (date), booleovské hodnoty (boolean alebo logical), typy pre rozsiahle textové či binárne dáta (text, resp. blob).

1.6 MySQL

MySQL patrí k najrozšírenejším systémom pre riadenie dát používaným pre tvorbu dynamicky generovaných WWW stránok, ktorý bol vytvorený švédskou firmou MySQL AB. MySQL umožňuje pracovať na jednom počítači s viac databázami pomocou jazyka SQL. Každá databáza môže potom obsahovať niekoľko tabuliek, ktoré môžu byť navzájom poprepájané.

Ak chceme pracovať s databázou, musíme sa k nej z PHP pripojiť. Nato slúži funkcia mysql_connect(), ktorej pridáme tri parametre: adresu databázového serveru, ku ktorému sa chceme pripojiť, užívateľské meno a prístupové heslo. Po vytvorení pripojenia je potrebné vybrať databázu, s ktorou chceme pracovať. Nato sa používa príkaz mysql_select_db(), kde ako

parameter zadávame meno databázu. Pre optimálny výkon je vhodné používať všade rovnaké kódovanie (na servery, počas prenosu i u klienta)

```
<?php
mysql_connect("localhost", "root", "heslo");
mysql_select_db("databaza");
mysql_query("SET CHARACTER SET utf8");
?>
```

1.7 Jazyk SQL

História SQL jazyka sa datuje od 70. a 80. rokoch. Prvý štandard bol prijatý v roku 1986, ktorý sa označuje ako SQL86. Jeho nedostatky sa však objavili časom, preto opravená verzia je z roku 1992 označovaná ako SQL92. Ten je v oblasti relačných databáz štandardom dodnes. Skratka SQL znamená Structured Query Language a je to dotazovací jazyk, ktorý sa používa na manipuláciu s dátami v databázach.

SQL sa skladá z niekoľko častí, kde niektoré sú určené pre administrátorov a návrhárov databázových systémov, iné pre koncových užívateľov a programátorov. Prvá časť je jazyk DDL – Data Definition Language. Ide o jazyk pre vytváranie databázových schém a katalógov. Spôsob ukladania tabuliek definuje jazyk SDL – Storage Definition Language. Tretia časť je jazyk VDL – View Definition Language, ktorý je určený pre návrhárov a správcov. Poslednou časťou je jazyk DML – Data Manipulation Language, ktorý obsahuje základné príkazy INSERT, UPDATE, DELETE a najpoužívanejší príkaz SELECT. S týmto jazykom najviac pracujú koncoví užívatelia a programátori databázových aplikácií.

1.8 Vytvorenie tabuľky

Základným príkazom pre vytvorenie databázovej tabuľky je príkaz CREATE TABLE. Jeho syntax je nasledujúca:

```
CREATE TABLE meno_tabuľky
      (meno_stĺpca typ [integritné obmedzenie],
       ...
       ...
      )
```

1.9 Vkladanie záznamu

Pre vkladanie jedného riadku dát do tabuľky sa používa príkaz INSERT, ktorého syntax je nasledovná:

```
INSERT INTO meno_tabuľky [(mena stĺpcov)] VALUES (zoznam hodnôt)
```

1.10 Úprava záznamu

Pre úpravy záznamu tabuľky je určený príkaz UPDATE. Umožňuje zadaným záznamom zmeniť príslušné atribúty na nové hodnoty.

```
UPDATE tabuľka SET atribút = hodnota [ , attribute = hodnota ...] [ WHERE podmienka ];
```

1.11 Výber záznamu

Medzi najkomplexnejší príkaz jazyka SQL patrí výberový príkaz SELECT. Zároveň je to tiež najpoužívanejší príkaz zo všetkých, lebo ide o jediný príkaz, ktorý umožňuje výber z databázy a čítaní dát. Syntax tohto príkazu je nasledovná

```
SELECT zoznam atribútov FROM zoznam tabuliek [ WHERE podmienky ] [ GROUP BY atribút ]
```

1.12 Mazanie záznamov

Niekedy je treba z tabuľky nepotrebné záznamy vymazať. K tomu sa používa príkaz DELETE s nasledujúcou syntaxou

```
DELETE FROM tabuľka [ WHERE podmienka ];
```

Ak sa podmienka nezadá, budú vymazané všetky údaje tabuľky

2 PRAKTICKÁ ČASŤ

2.1 Vytvorenie databázy a tabuliek

Na vytvorenie databázy a tabuliek som použila program PhpMyAdmin. Potrebovala som vytvoriť databázu zahranicna_cesta (CREATE DATABASE zahranicna_cesta;) a nasledujúce tabuľky:

Tabuľka **obce**:

```
CREATE TABLE obce (
  id int(11) NOT NULL auto_increment,
  obec varchar(100) NOT NULL,
  okres varchar(100) NOT NULL,
  kraj varchar(100) NOT NULL,
  PRIMARY KEY  (id)
);
```

Tabuľka **krajina**:

```
CREATE TABLE krajina (
  id int(3) NOT NULL auto_increment,
  krajina varchar(50) NOT NULL,
  mena_kod varchar(3) NOT NULL,
  mena varchar(20) NOT NULL,
  sadzba int(5) NOT NULL,
  PRIMARY KEY  (id)
);
```

Tabuľka **hranice:**

```
CREATE TABLE hranice (
  id int(11) NOT NULL auto_increment,
  prechod varchar(100) NOT NULL,
  PRIMARY KEY  (id)
);
```

Tabuľka **cesta:**

```
CREATE TABLE cesta (
  id int(11) unsigned NOT NULL auto_increment,
 meno varchar(6) NOT NULL,
  volby int(20) NOT NULL,
  datum_od int(8) NOT NULL,
  datum_do int(8) NOT NULL,
  cesta_kam varchar(10) NOT NULL,
  diety decimal(8,2) NOT NULL,
  ubytovanie decimal(8,2) NOT NULL,
  cestovne decimal(8,2) NOT NULL,
  ostatne decimal(8,2) NOT NULL,
  vreckove varchar(10) NOT NULL,
  doprava1 varchar(10) NOT NULL,
  cena1 decimal(8,2) NOT NULL,
```

```
  doprava2 varchar(10) NOT NULL,
  cena2 decimal(8,2) NOT NULL,
  doprava3 varchar(10) NOT NULL,
  cena3 decimal(8,2) NOT NULL,
  doprava4 varchar(10) NOT NULL,
  cena4 decimal(8,2) NOT NULL,
  doprava5 varchar(10) NOT NULL,
  cena5 decimal(8,2) NOT NULL,
  doprava6 varchar(10) NOT NULL,
  cena6 decimal(8,2) NOT NULL,
  kam1 varchar(10) NOT NULL,
  kam2 varchar(10) NOT NULL,
  datum_prechod_od int(8) NOT NULL,
  datum_prechod_do int(8) NOT NULL,
  odkial1 int(11) NOT NULL,
  odkial2 int(11) NOT NULL,
  strava decimal(8,2) NOT NULL,
  ub1 decimal(8,2) NOT NULL,
  ub2 decimal(8,2) NOT NULL,
  ub3 decimal(8,2) NOT NULL,
  ov1 decimal(8,2) NOT NULL,
  ov2 decimal(8,2) NOT NULL,
  ov3 decimal(8,2) NOT NULL,
  ov4 decimal(8,2) NOT NULL,
  ov5 decimal(8,2) NOT NULL,
  datum_aktual int(8) NOT NULL,
  PRIMARY KEY  (id)
);
```

Tabuľka **doprava:**

```
CREATE TABLE doprava (
  id tinyint(10) NOT NULL,
  prostriedok varchar(20) NOT NULL
);
```

Tabuľka **oddelenie:**

```
CREATE TABLE oddelenie (
  kod int(3) NOT NULL,
  nazov varchar(255) NOT NULL,
  PRIMARY KEY  (kod)
);
```

Tabuľka **person:**

```
CREATE TABLE person (
  person_id int(10) unsigned NOT NULL default '0',
  person_passwd varchar(32) NOT NULL default '',
  person_name varchar(50) NOT NULL default '',
  person_surname varchar(50) NOT NULL default '',
  person_title_1 varchar(20) NOT NULL default '',
  person_title_2 varchar(10) NOT NULL default '',
  person_position varchar(50) NOT NULL default '',
```

```
  person_room varchar(20) NOT NULL default '',
  person_phone varchar(100) NOT NULL default '',
  person_fax tinytext NOT NULL,
  person_email varchar(50) NOT NULL default '',
  person_skype tinytext NOT NULL,
  person_icq tinytext NOT NULL,
  person_department varchar(6) NOT NULL default '',
  person_url tinytext NOT NULL,
  person_photo tinytext NOT NULL,
  person_active tinyint(1) unsigned NOT NULL default '0',
  person_perms varchar(20) NOT NULL default '',
  person_show int(6) unsigned zerofill NOT NULL,
  person_text_sk text NOT NULL,
  person_text_en text NOT NULL,
  PRIMARY KEY  (person_id)
);
```

Tabuľka **strava:**

```
CREATE TABLE strava (
  id int(11) NOT NULL auto_increment,
  strava varchar(100) NOT NULL,
  podiel decimal(3,2) NOT NULL,
  PRIMARY KEY  (id)
);
```

Tabuľka **vreckove:**

```
CREATE TABLE vreckove (
  id tinyint(2) NOT NULL auto_increment,
  percento varchar(30) NOT NULL,
  vypocet decimal(6,2) NOT NULL,
  hodnota tinyint(1) NOT NULL,
  PRIMARY KEY  (id)
);
```

Jednotlivé tabuľky obsahujú polia a tie svoje záznamy. Napríklad tabuľka oddelenie pozostáva z polí kod a nazov. Pole kod predstavuje primárny kľúč, ktorý zabezpečí rýchle vyhľadávanie čísla a následne priradenie príslušného oddelenia v tabuľke. Ako typ pola je nastavený INT (číslo), ktoré vyjadruje koľko ciferné je číslo. Pole nazov je typu varchar (slovo) obsahujúce 255 rôznych znakov. Jednotlivé záznamy som vkladala cez SQL a príslušný kód s údajmi, kódovanie som prednastavila utf8_slovak_ci.

2.2 Vytvorenie samostatného formulára

Na začiatku každého skriptu (stránky) som vytvorila spojenie s databázou:

```
<?php
if(!mysql_connect(SERVER,USER,PASSWORD))
```

```
{
   echo "Nepodarilo sa spojiť s databázovým serverom!";
   exit;
}
```

V prípade, že sa spojenie nepodarí naviazať, vypíše sa chybové hlásenie a ukončí sa skript. Podobným spôsobom sa vyberá aj databáza zahranicna_cesta:

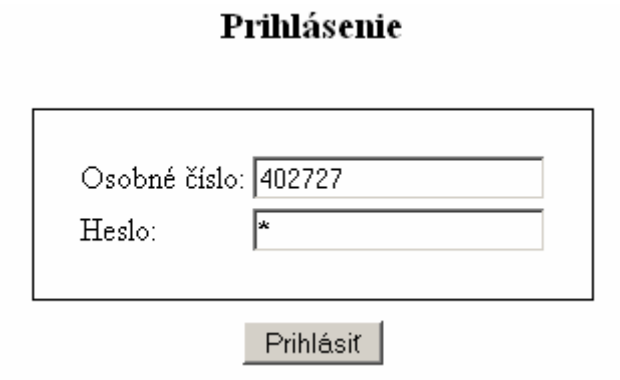
```
if(!mysql_select_db("zahranicna_cesta"))
{
   echo " Nepodarilo sa vybrať databázu zahranicna_cesta!";
   exit;
}
?>
```

Samotný program začína prihlásením užívateľa, ktorý zadá svoje osobné číslo a heslo. Prihlásenie na server sa vykoná pomocou premennej $_SESSION, ktorej hlavné využitie spočíva v tom, že ak prebehne prihlásenie, vytvorí sa session a uloží do nej jednoznačný identifikátor užívateľa. Počas celej práce užívateľa má každý php skript prístup k session a preto aplikácia vie, kto s ňou narába. Po dohlásení užívateľa sa session zruší (unset()). Po overení správnosti zadaných údajov sa užívateľovi zobrazia jeho vyplnené formuláre, alebo si môže zvoliť nový formulár, alebo sa odhlásiť. Týmto som zamedzila nepovoleným osobám, aby sa dostali k prístupu vyplňovania formulára.

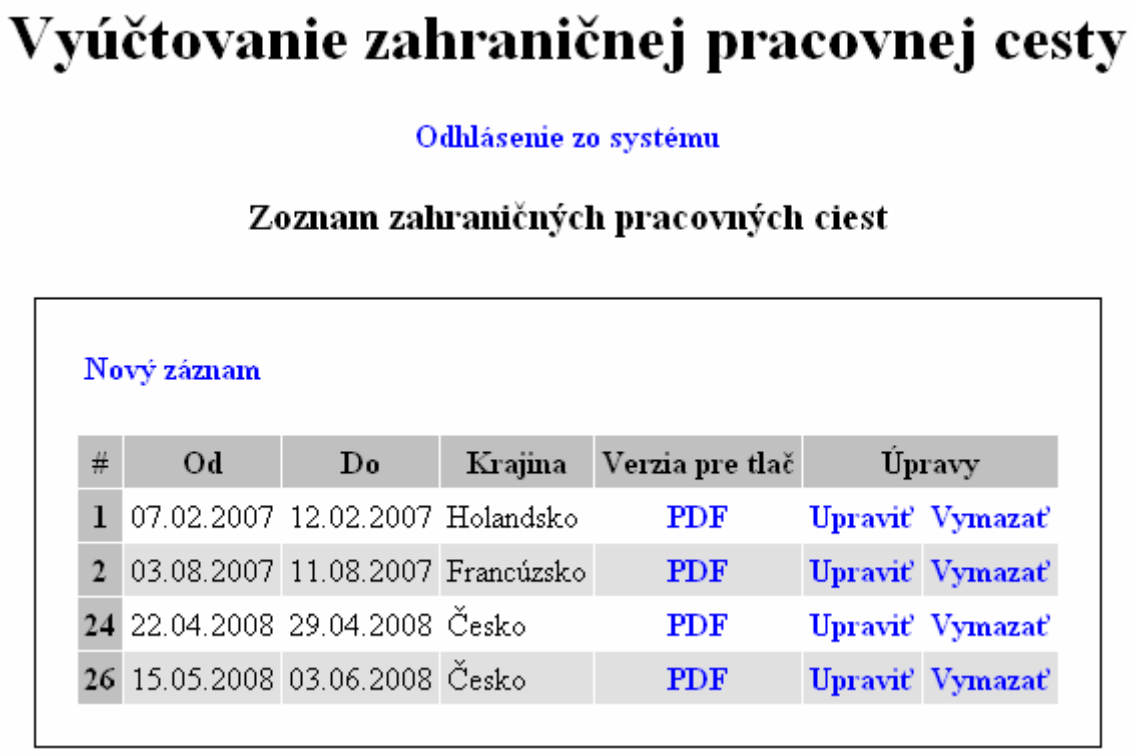
```
session_start();
if($_GET['logout'])
   unset($_SESSION['person_id']);

if($_SESSION['person_id'])
{
  if($_GET['page'] == 1)
    include "bp.php";
  elseif($_GET['page'] == 2)
    include "bp1.php";
  else
    include "edit.php";
  exit();
}
elseif(trim($_POST['person_id']) && trim($_POST['person_passwd']))
{
  $sql = "SELECT person_id, person_name, person_surname FROM person WHERE
    person_id='".trim($_POST['person_id'])."' AND
    person_passwd='".MD5(trim($_POST['person_passwd']))."'";
  //echo $sql."<br>";
  $vysledok = mysql_query($sql);
  $num_rows = mysql_num_rows($vysledok);
  if($num_rows)
  {
    $_SESSION['person_id'] = trim($_POST['person_id']);
    include "edit.php";
```

```
    exit();
  }
}
```



Obr. 3: Prihlásenie do systému



#	Od	Do	Krajina	Verzia pre tlač	Úpravy	
1	07.02.2007	12.02.2007	Holandsko	PDF	Upraviť	Vymazať
2	03.08.2007	11.08.2007	Francúzsko	PDF	Upraviť	Vymazať
24	22.04.2008	29.04.2008	Česko	PDF	Upraviť	Vymazať
26	15.05.2008	03.06.2008	Česko	PDF	Upraviť	Vymazať

Obr. 4: Zoznam zahraničných pracovných ciest

Nasleduje typ a štandard dokumentu, súlad s definíciou Strict DTD jazyka XHTML 1.0.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

V hlavičke je definovaný názov medzi tagmi <title>...</title>, ktorý sa zobrazí v okne prehliadača a meta značka, kde je nastavené kódovanie dokumentu.

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
```

Ďalej nasleduje telo so samostatnými príkazmi programu. Názov formulárovej stránky som vytvorila pomocou príkazu

```
<h1>Vyúčtovanie zahraničnej cesty</h1>
```

Každý formulár musí byť ohraničený prvkom**,** ktorý sa skladá z URL adresy, mena a metódy akou má byť spracovaný.

```
< form action="bp1.php#ofset" name="form" method="post">
```

Do blokového prvku divu som vložila návestia, ktoré zabezpečia po vložení údaja návrat stránky na aktuálnu tabuľku. Týmito návestiami som zjednodušila vyplňovanie formulára, ktoré by inak vracali užívateľa na začiatok stránky

```
<div><a name="<?php echo $nav[1];?>"></div>
```

Tabuľka sa vytvára pomocou tagu <table border="1">...</table>, kde som jej priradila orámovanie veľkosti 1px. Jednotlivé riadky sa v tabuľke definujú tagom <tr>...</tr> a samotné údaje pomocou <td>...</td>. Hneď prvá tabuľka obsahuje priezvisko a meno s titulmi, ktoré sa po prihlásení automaticky vyplnia. Oddelenie si môže vybrať z ponuky, ktorá sa načíta z hore spomínanej databázy oddelenie.

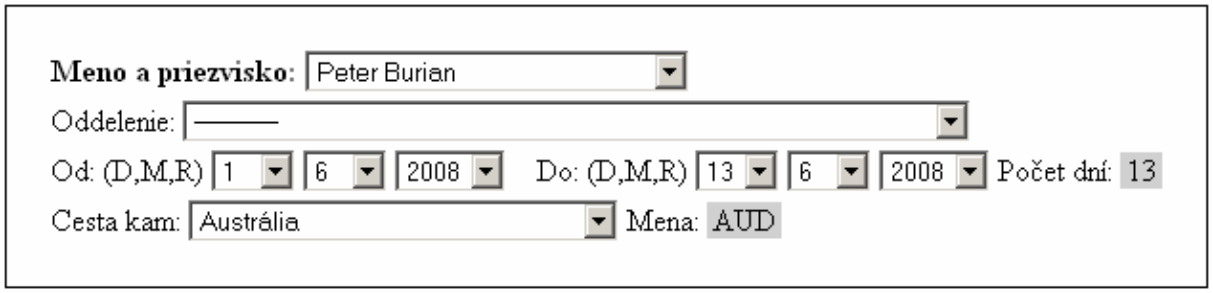
```
<select name="volby">
<?php
$sql = "SELECT * FROM oddelenie ORDER BY nazov";
$vysledok = mysql_query($sql);
while($riadok = mysql_fetch_assoc($vysledok))
  echo "\t<option value='".$riadok['kod']."'>".$riadok['kod']." - ".$riadok['nazov']."</option>\n";
?>
</select>
```

Dátum Od a Do sa vyberá po dni, mesiaci a roku, kde som použila funkciu for. Po zadaní dátumu sa následne automaticky vykoná prepočet dní trvania zahraničnej cesty

```
<select name="datum_od_den"
onchange="javascript:document.form.obnov.value=1;document.form.ofset.value=1;document.form.submit(
)">
<?php
for($i=1; $i<=31; $i++)
{
  $selected = ($i == date("d",$pole['datum_od'])) ? " selected='selected'" : "";
  echo "\t<option value='$i'".$selected.">$i</option>\n";
```

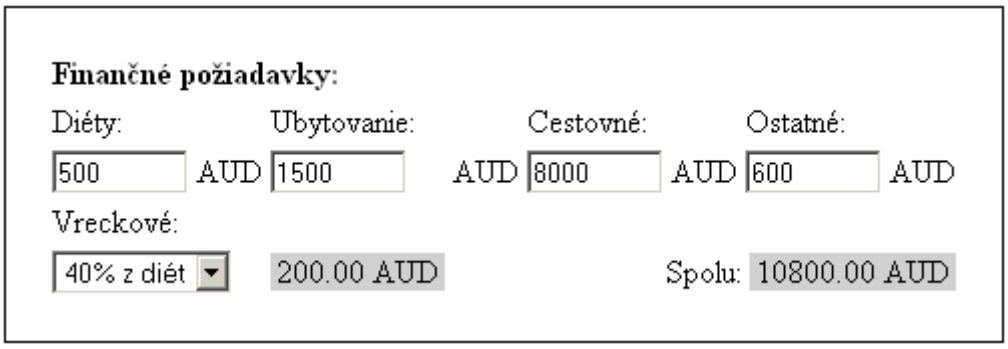
```
}
?>
</select>
<select name="datum_od_mesiac"
onchange="javascript:document.form.obnov.value=1;document.form.ofset.value=1;document.form.submit(
)">
<?php
for($i=1; $i<=12; $i++)
{
  $selected = ($i == date("m",$pole['datum_od'])) ? " selected='selected'" : "";
  echo "\t<option value='$i'".$selected.">$i</option>\n";
}
?>
</select>
<select name="datum_od_rok"
onchange="javascript:document.form.obnov.value=1;document.form.ofset.value=1;document.form.submit(
)">
<?php
for($i=2000; $i<=date("Y")+2; $i++)
{
  $selected = ($i == date("Y",$pole['datum_od'])) ? " selected='selected'" : "";
  echo "\t<option value='$i'$selected>$i</option>\n";
}
?>
</select>
```

Cestujúci si zvoli cieľ cesty z ponuky štátov, kde sa mu následne zobrazí mena zvolenej krajiny. Táto mena sa bude ďalej automaticky používať v celom programe, kde sa vyskytne formulárové pole pre zadávanie sumy.



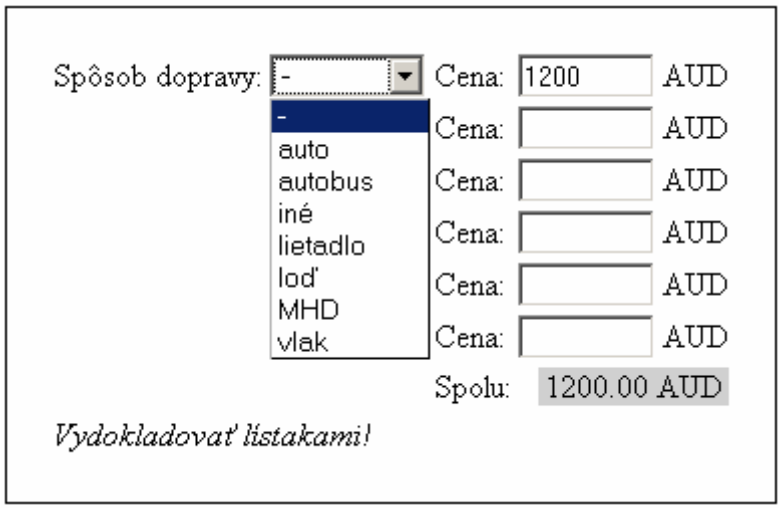
Obr. 5: Meno a priezvisko

Druhá tabuľka pozostáva z finančných požiadaviek, kde cestujúci zadáva do formulárového poľa sumu diét, ubytovania, cestovného a ostatných výdavkov. V nasledujúcom riadku si vyberie percento z diét, kde sa následne vykoná prepočet zo zadanej diéty. Ďalší údaj obsahuje sumu zadaných údajov v príslušnej mene.



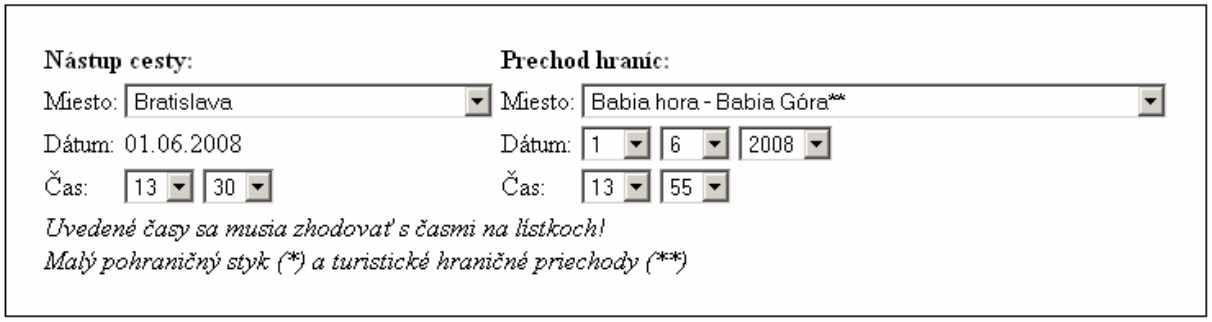
Obr. 6: Zoznam Finančné požiadavky

Tretia tabuľka predstavuje spôsob dopravy, kde sa dajú z databázy vybrať rôzne dopravné prostriedky, ktoré boli počas tejto zahraničnej cesty použité. Ku každému dopravnému prostriedku treba vyplniť sumu, ktorú musí cestujúci dokladovať lístkami. Posledný riadok tvorí suma zadaných jednotlivých dopravných prostriedkov.



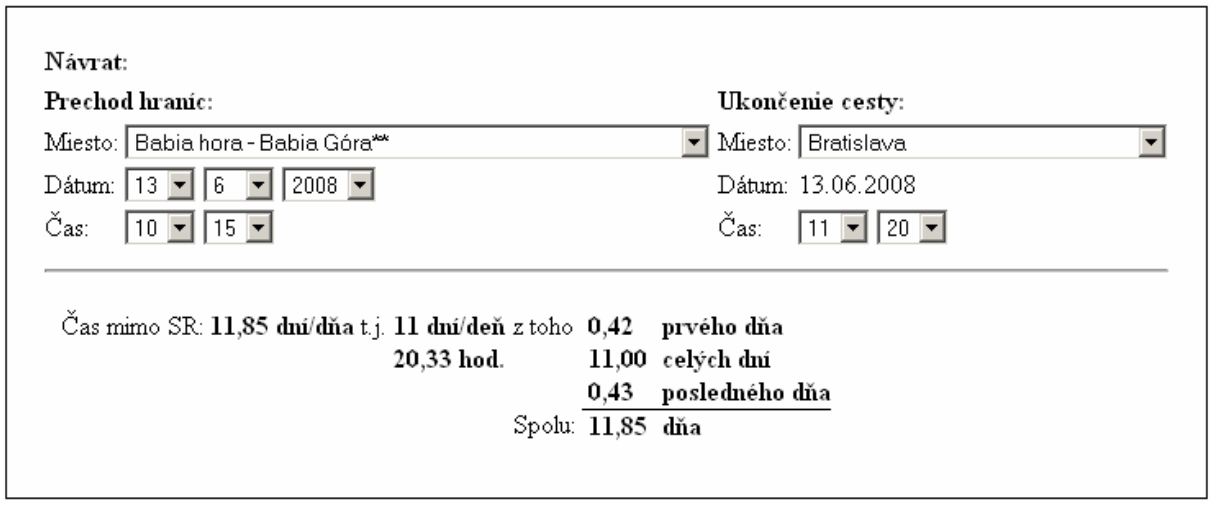
Obr. 7: Spôsob dopravy

Vo štvrtej tabuľke si cestujúci vyberie z databázy miesto nástupu a miesto prechodu hraníc. Dátum nástupu cesty sa vypíše automaticky, dátum prechodu hraníc cestujúci musí zadať po dni, mesiaci a roku. V nasledujúcom riadku sa nachádza čas nástupu cesty a prechodu hraníc, ktorý sa zadáva po hodine a minúte.



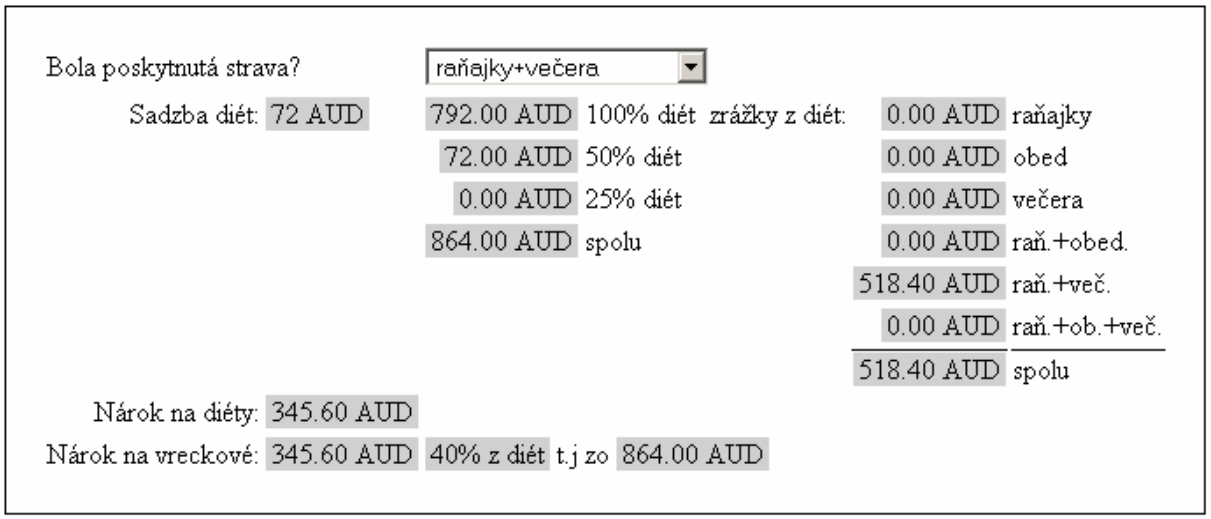
Obr. 8: Nástup cesty

Piata tabuľka sa týka návratu zahraničnej cesty. Opäť si cestujúci vyberie z databázy miesto prechodu hraníc a miesto ukončenia cesty. Dátum prechodu hraníc bude zadávať po dni, mesiaci a roku a dátum ukončenia cesty sa automaticky vypíše. V ďalšom riadku sa vyberajú časy prechodu hraníc a ukončenia cesty. Posledný riadok tabuľky zahŕňa, koľko z celého dňa bol cestujúci mimo Slovenskej republiky, koľko to je dní a hodín, koľko je to z prvého dňa, celých dní a koľko z posledného dňa. Konečné číslo udáva počet dní trvania zahraničnej cesty.



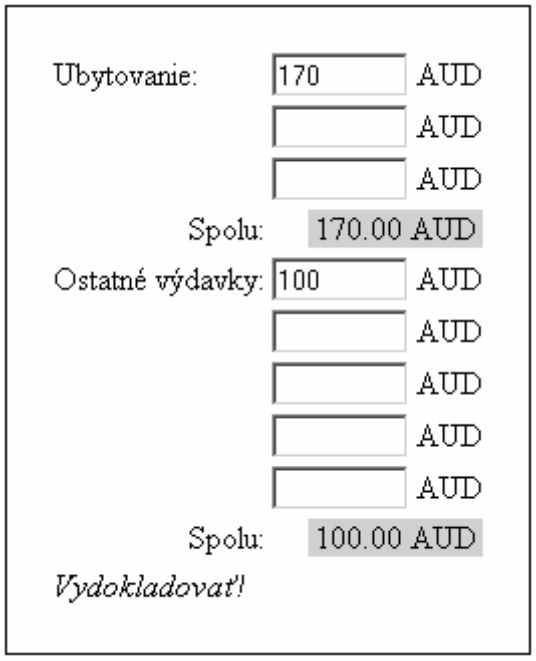
Obr. 9: Návrat cesty

Šiesta tabuľka je tvorená údajmi týkajúcimi sa poskytnutia stravy. Cestujúci si z ponuky vyberie aká strava bola poskytnutá. Následne sa zobrazí sadzba, percento, zrážka z diét s hodnotou v príslušnom políčku, nárok na diéty, nárok na vreckové a konečná suma v príslušnej mene.



Obr. 10: Strava

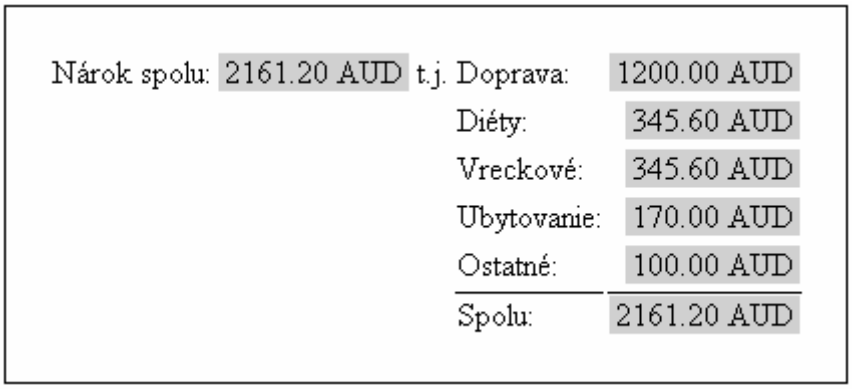
V siedmej tabuľke doplní cestujúci do formulárového poľa sumu za ubytovanie a ostatné výdavky. Pod týmito položkami sú zobrazené aj sumy v príslušnej mene, ktoré treba dokladovať pokladničnými lístkami.



Obr. 11: Ubytovanie a ostatné výdavky

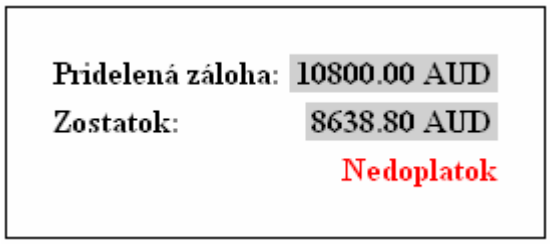
Nasledujúca tabuľka predstavuje zhrnutie vyplnených údajov, ktoré sa automaticky vyhodnotia po ich zadaní. Údaje pozostávajú z týchto položiek: nárok spolu, doprava, diéty,

vreckové, ubytovanie, ostatné výdavky a celková suma, ktorá bola použitá počas doby trvania zahraničnej cesty.



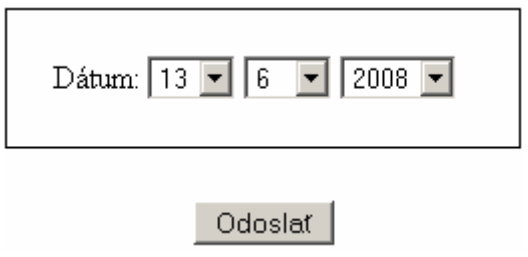
Obr. 12: Zhrnutie

Hneď pod touto tabuľkou nás program sám upozorní, či treba doplatiť, alebo nám musia vrátiť, alebo je to v poriadku. V ďalšom riadku je zobrazená suma pridelenej zálohy a zostatok, ktorý musím doplatiť, alebo mu majú vrátiť.



Obr. 13: Nedoplatok

V poslednej tabuľke vyplní cestujúci už iba dátum vyhotovenia dokladu o zahraničnej ceste.



Obr. 14: Dátum vyhotovenia

Po odoslaní vyplneného formulára sa zobrazia všetky uložené doklady o zahraničných cestách, ktoré sa dajú neskôr dopĺňať alebo meniť a tlačiť.

Verzia pre tlač je robená vo formáte PDF, čím som umožnila rovnaký vizuálny vzhľad stránky pri používaní rôznych prehliadačov. V PHP súbore som na začiatku vytvorila prepojenie s databázou. Príkazom require som vložila knižnicu FPDF, potom nasleduje výber z databázy. Na ďalšom riadku som vytvorila objekt $pdf. Aby sa mohol vkladať text, trebalo vytvoriť novú stránku. K tomu sa použila metóda AddPage(). Použitím SetXY() sa nastavujú pozície osi. Pomocou metódy SetFont() som vybrala typ písma - Arial, B (tučné), veľkosť 16 bodov. Vytváranie textu som vykonávala metódou Cell() - konkrétne som vytvorila textový objekt 190x10 a do neho vložila text "Vyuctovanie zahranicnej cesty", ktorý je zarovnaný na stred. Pomocou metódy Output() sa dokončí tvorba PDF a odošle výsledok prehliadaču. Takýmto spôsobom som vkladala nasledujúce riadky z formulára.

```
<?php
require('fpdf.php');
$sql = "SELECT
cesta.*,person.*,oddelenie.*,krajina.krajina,krajina.mena,krajina.sadzba,krajina.mena_kod FROM cesta
LEFT JOIN person ON meno=person_id LEFT JOIN oddelenie ON kod=volby LEFT JOIN krajina ON
cesta_kam=krajina.id WHERE cesta.id='".intval($_GET['id'])."'";
//echo $sql."<br>";
$vysledok = mysql_query($sql);
$num_rows = mysql_num_rows($vysledok);
if($num_rows)
{
  $row = mysql_fetch_assoc($vysledok);
  $pdf = new FPDF();
  $pdf->AddPage();
  $pdf->SetXY(10,10);
  $pdf->SetFont('Arial','B',16);
  $pdf->Cell(190,10,'Vyučtovanie zahraničnej cesty',0,0,'C');
...
  $pdf->Output();
}
?>
```

3 ZÁVER

Cieľom mojej bakalárskej práce bolo modernizovať zastarané používanie ručného vypisovania vyúčtovania zahraničnej pracovnej cesty. V nich sa pri chybnom vyplnení nesmie škrtať, a tak sa museli znova vypisovať. Tým vznikali väčšie náklady, strácal sa čas a bolo to neefektívne.

Moderný elektronický formulár ma preto v praxi oveľa väčšie využitie. Každý zamestnanec ÚIAM ho môže nájsť na webovej stránke ústavu.

Formulár bol vytvorený v programe PSPad, kde boli využité programovacie jazyky ako sú: HTML, PHP, SQL (MySQL). Najskôr bola vytvorená podľa vzoru vizuálna časť formulára v HTLM jazyku, pozostávajúca z jednotlivých tabuliek. Postupne boli pridávané do potrebných políčok PHP kódy. Ďalej bola vytvorená databáza zahranicna_cesta a v nej potrebné tabuľky, kde sa pracovalo s programom PhpMyAdmin. Tie sa navzájom poprepájali a spojili s formulárom. Ponuka jednotlivých údajov bola spracovávaná tiež cez databázu. Potom sa pracovalo na verzii pre tlač, ktorá bude pretransformovaná do formátu PDF kvôli používaniu rôznych internetových prehliadačov. Celé to bolo neskôr ošetrené proti získaniu údajov nepovoleným osobám prihlásením sa do systému. Nakoniec bol formulár validovaný pomocou testovania na internetovej stránke konzorcia W3C.

Tento elektronický formulár je dôkazom toho, ako sa dajú zjednodušiť a modernizovať administratívne doklady. Veľkou výhodou takéhoto formulára je, že ho môžete vyplniť na ktoromkoľvek počítači a je prístupný kedykoľvek. Časom by takto mohli byť robené všetky v praxi najbežnejšie používane doklady prístupné na internete.

LITERATÚRA

[1] *Historie Internetu* [on line].
Dostupné z: <http://www.webdesign.paysoft.cz/clanky/2006/historie-internetu/>

[2] *Jazyk PHP* [on line].
Dostupné z: <http://www.jazykphp.yw.sk/riadiacastruktura_cykluswhile.php>

[3] *Jazyk PHP* [on line].
Dostupné z: <http://www.jazykphp.yw.sk/riadiacastruktura_cyklusdo-while.php>

[4] Jiři Bráza: *PHP 5*. Grada Publishing, 2005

[5] *Databáze a jazyk SQL* [on line].
Dostupné z: <http://interval.cz/clanky/databaze-a-jazyk-sql/>

[6] *Základy jazyka PHP* [on line].
Dostupné z: <http://www.kirp.chtf.stuba.sk/~cirka/vyuka/php/kap3.php#sec5>