MATEMATIČKI FAKULTET
BEOGRADSKI UNIVERZITET

MASTER TEZA

Korišćenje HTML-a, PHP-a i MySQL-a u izradi višeslojnih web aplikacija

Vanja Petković

Beograd, maj 2008

Sadržaj

1 Uvod

HTML, PHP i MySQL predstavljaju tehnologije otvorenog koda savršene za brz i efikasan razvoj web aplikacija.

HTML (*eng. Hypertext Markup Language*) , je jezik zasnovan na oznakama i koristi se za opisivanje strukture web dokumenata i neke karakteristike njihovog ponašanja. Nastanak HTML-a je inspirisan SGML-om (*eng. Standard Generalized Markup Language*), koji je takođe zasnovan na oznakama. HTML predstavlja način za prikazivanje teksta i njegovo povezivanje sa drugim vrstama resursa, poput zvučnih, grafičkih, ili datoteka drugih vrsta. HTML fajl je jedan tekst fajl koji sadrži ono što se želi prikazati i oznake (*eng. mrkup tag*). On opisuje strukturu web dokumenta i neke karakteristike njihovog ponašanja, ali pre svega služi za prikaz dokumenata.

Originalni HTML je stvorio Tim Berners Lee 1989. godine. U to vreme HTML nije bio precizno definisan, već je predstavljao skup elemenata za rešavanje problema komunikacije između Lee-a i njegovih saradnika. Prvi javni opis HTML-a je bio dokument pod nazivom HTML Tags. Taj dokument je opisivao 22 elementa koji su predstavljali prvobitni dizajn HTML-a. Trinaest od tih elemenata i danas postoje u HTML 4. Rane verzije HTML-a nisu bile definisane čvrstim sintaksičnim pravilima, dok su novije verzije definisane monogo strožijim pravilima i zahtevaju mnogo precizniji kod. Prva specifikacija HTML jezika je objavljena 1993. godine od strane IETF kao formala „aplikacija“ SGML-a. IETF 1995. godine objavljuje HTML 2.0, ali se dalji razvoj HTML obavlja pod okriljem W3C**.** U 2000 godini HTML postaje internacionalni standard zasnovan na XML-u(ISO/IEC 15445:2000). Poslednja specifikacija HTML jezika objavljenja od strane W3C je HTML 4.01, 1999 godine.

PHP je moćan skript jezik koji omogućava programerima da brzo prave složene web aplikacije. PHP je naslednik jednog starijeg proizvoda pod nazivom PHP/FI(Personal Home Page / Forms Interpreter). PHP/FI je kreiran od strane Rasmus Ledorf, 1995 godine. U početku je to prestavljao jednostavan skup Perl skriptova, koji je on nazvao ”Personal Home Page Tools”. Rasmus je te skriptove koristio da prati pristup njegovoj ličnoj stranici. Vremenom je Rasmus napisao mnogo veću C implementaciju, koja je imala mogćnost komunikacije sa bazama podataka i davala mogućnost korisniku da razvije jednostavnu dinamičku web aplikaciju. PHP je od samog početka bio vrlo funkcionalan, a tu osobinu je zadržao i danas. Rasmus je odabrao da izvorni kod PHP/FI projekta objavi tako da svako može da ga koristi, unapredi i popravi greške.

Druga implementacija PHP/FI, PHP/FI 2.0, je zvanično objavljena u novembru 1997. godine. Imala je zajednicu od nekoliko hiljada korisnika širom sveta, a oko 50000 domena je imalo instaliranu ovu implementaciju.

PHP 3.0 je prva verzija koja odgovara onom što je danas PHP. Kreirana je od strane Andi Gutmans-a i Zeev Suraski-a 1997. godine. Radeći na jednoj složenoj web aplikaciji, zaključili su da je implementacija PHP/FI 2.0 neefikasna tako da su je ponovo napisali i objavili kao PHP 3.0. PHP 3.0 je zvanično objavljen u junu 1998 godine.

Ubrzo nakon što je PHP 3.0 zvanično objavljen, Andi Gutmans i Zeev Suraski su počeli sa radom na ponovnom pisanju PHP implementacije kako bi poboljšali performance PHP u složenijim aplikacijama. Nova implementacija nazvana „Zend Engine“ je uspešno ispunila zacrtane ciljeve u poboljšanju performansi. Zvanično je objavljena u maju 2000. godine pod nazivom PHP 4.0. Pored poboljšanja performansi, PHP 4.0 je doneo i podršku za mnogo više web servera, HTTP sesije i nekoliko novih jezičkih konstrukcija.

PHP 5 je objavljen u julu 2004. posle nekoliko test verzija. Ova verzija je donela kompletno novu podršku za objektno orijentisano programiranje kao i SimpleXML omogućavajući jednostavan rad sa podacima u XML formatu.

MySQL je „Open Source“ baza podataka koja je, prema podacima švedske firme MySQL AB, instalirana na više od 10 miliona računara širom sveta. Pogodna je, kako za opsluživanje potreba manjih

internet prezentacija, tako i za velike korporacije koje opslužuju internet stranice gustog saobraćaja. MySQL je trenutno u vlasništvu firme MySQL AB koja poseduje sva izdavačka prava na MySQL server izvorni kod. Ona ga lincencira za upotrebu u komercijalnim aplikacijama koje koriste MySQL. U trenutku pisanja ovog dokumenta Sun Microsystems Inc. je objavila da namerava da otkupi prava na MySQL od firme MySQL AB za 1 000 000 000 $.

Danas PHP i MySQL koristi nekoliko hiljada programera za razvoj web aplikacija. Prema nekim podacima više od 20% internet sajtova koristi PHP i MySQL.

2 HTML

2.1 Elementi i oznake

HTML dokumenti su tekstualni dokumenti koji se sastoje od elemenata. Elementi su definisani upotrebom oznaka (*eng. tags*). U HTML-u oznake su oivčene streličastim zagradama < >. Većina HTML oznaka javlja se u parovima npr. <HEAD> i </HEAD> gde je < ?> početak (*start tag*), a < /?> označava kraj (*end tag*) nekog HTML elementa. Tekst između početne i krajnje oznake predstavlja sadržaj (*content*) tog HTML elementa. Neki HTML elementi nemaju sadržaj i oznaku za kraj kao npr.
-oznka za novi red. HTML elementi nisu osetljivi na mala i velika slova tako da je svejedno da li pišemo
 ili
.

Često korišćene oznake u HTML dokumentima su oznake za naslove, paragrafe i novi red. Oznake za naslove su od <h1> </h1> do <h6> </h6> gde <h1> definiše naslov najvećeg nivoa a <h6> najmanjeg nivoa. Oznaka za paragraf su . Dok HTML elementi za naslov i paragraf spadaju u skup elemenata koji imaju početnu i krajnju oznaku i sadržaj, element koji označava novi red
 ima samo početnu oznaku. Svaki HTML dokument počinje oznakom <HTML>, a završava se sa </HTML>.

HTML oznake mogu imati i atribute. Oni se koriste za „proširenje" samih HTML elemenata. Ako je prisutan atribut (proširenje) kada web pretraživač obrađuje oznaku on će takođe potražiti i atribute koji su pridruženi toj oznaci, i u zavisnosti od oznake i njenih atributa, odrediti na koji način će prikazati sadržaj tog HTML elementa.

2.2 Zaglavlje i metainformacije

Zaglavlje (*HEAD*) je element HTML dokumenta za definisanje informacija o tom dokumentu, uključujući sam naziv dokumenta, meta informacije, ukazivač sledeće strane, kao i veze do drugih HTML elemenata. Za svaki HTML-dokument poželjno je ima naslov (*TITLE*) unutar zaglavlja. Autori HTML dokumenata bi trebalo da koriste naslov dokumenta kako bi opisali sadržaj samog dokumenta. Unutar zaglavlja se mogu nalaziti i metainformacije, odnosno <META> element. Metainformacije predstavljaju „informacije o informacijama u dokumentu". Te informacije se najčešće koriste od strane internet pretraživača kako bi se poboljšali rezultati pretrage. Sledi primer zaglavlja HTML dokumnta:

```
<HEAD>
  <TITLE>Ovo je HTML dokument</TITLE>
  <META name="author" content="Vanja Petkovic">
  <META name="keywords" content="HTML,Zaglavlje,Metainformacije">
  <META name="date" content="2007-5-17">
</HEAD>
```

2.3 Telo HTML dokumenta

Telo HTML dokumenta se nalazi između oznaka <BODY> i </BODY>. To je ustvari deo HTML dokumenta koji sadrži sve što se prikazuje u radnom prozoru internet pretraživača. Kako telo dokumenta sadrži ceo sadržaj samog dokumenta, to telo ima veliki broj atributa koji kontrolišu celokupni izgled dokumenta. Neki od tih atributa su:

- BGCOLOR – postavlja boju pozadine dokumenta
- BACKGROUND – postavlja sliku kao pozadinu dokumeta
- TEXT – postavlja boju teksta u dokumentu
- LINK – boja veza (*link*) koje još nisu upotrebljene
- VLINK – boja veza koje su upotrebljene
- TOPMARGIN – veličina gornje i donje ivice

- LEFTMARGIN – veličina leve ivice
- onLoad – definiše skriptu koja će se pokrenuti nakon učitavanja dokumenta
- onFocus – definiše skriptu koja će se pokrenuti kada telo dokumenta dobije fokus
- onBlur - definiše skriptu koja će se pokrenuti kada telo dokumenta izgubi fokus

itd.

2.4 Formatiranje teksta

HTML definiše veliki broj elemenata koji se koriste za formatiranje teksta:

- <B> </B> - podebljani tekst
- <I> </I> - iskošeni tekst
- <U> </U> - podvučeni teks
- - tekst ispisan u indeksu
- - tekst ispisan u eksponentu
- <STRONG> </STRONG> - jače naglašen tekst
- <SMAL> </SMAL> - sitniji tekst
- <EM> </EM> - naglašeno ispisivanje teksta, obično kurzivom

itd.

2.5 Sidra

Sidra (*anchors*) su veze (*links*) koje se koriste kako bi se povezali razni resursi na web-u ili različite celine unutar jednog HTML dokumenta. Sidro se nalazi unutar oznaka <A> </A> i može da bude veza ka HTML strani, slici, zvuku, filmu ili nekoj drugoj vrsti resursa. Atribut **HREF** se koristi kako bi se navela lokacija na koju pokazuje sidro:

<A href=”http://www.google.com”>

Atribut **NAME** se koristi kako bi se odredilo odredište veze unutar dokumenta.

2.6 Okviri

Sa okvirima (*frames*) se može prikazati više od jednog HTML dokumenta unutar istog prozora internet pretraživača. Svaki HTML dokument se naziva okvirom i svi okviri su međusbno nezavisni. Da bi se definisala podela prozora na okvire, koristi se element <FRAMESET> </FRAMESET>. Atribut **COLS** se koristi kako bi se dokument podelio u okvire po kolonama. Ako želimo da podelimo dokument u okvire po redovima, onda koristimo atribut **ROWS.** Dimenzije kolona ili redova se zadaju u procentima, broju tačaka ili relativnoj veličini.

```
<frameset cols="25%,75%">
  <frame src="frame_a.htm">
  <frame src="frame_b.htm">
</frameset>
```

Bitno je napomenuti da se <FRAMESET> koristi umesto <BODY> elementa, tj. da se ne mogu koristiti zajedno <FRAMESET> i <BODY> element.

2.7 Tabele

HTML dozvoljava kreiranje tabela. To se postiže ozanakama <TABLE> </TABLE>. Tabela je prazna dok se u njoj ne kreiraju redovi i ćelije. Unutar tabele se formiraju redovi koristeći oznake <TR> </TR>, dok se unutar redova formiraju ćelije oznakama <TD> </TD>. U svakoj ćeliji se nalazi sadržaj koji može predstavljati tekst, slike, liste, paragrafe, tabele itd. Zaglavlja reda ili kolone tabele se definišu upotrebom oznaka <TH> </TH>. <TABLE> oznaka ima sledeće atribute:

- ALIGN – definiše poravnanje tabele u odnosu na dokument. Moguće su sledeće vrednosti atributa: LEFT (poravnanje uz levu stranu dokumenta), RIGHT (desnu stranu) i CENTER (centriranje tabele u odnosu na dokument)
- BGCOLOR – podešava boju pozadine tabele. Boja se zadaje kao heksadecimalna vrednost u RGB sistemu boja ili se koriste neka od imena koja su unapred definisana.
- BORDER – podešava veličinu okvira tabele izraženu u tačkama. Podrazumevana vrednost je 1.
- CELLPADDING – podešava količinu prostora, izraženu brojem tačaka, između ivica ćelije i njene sadržine.
- CELLSPACING – podešava količinu prostora, izraženu brojem tačaka, između spoljašnjeg okvira tabele i ćelija u tabeli, kao i prostor između samih ćelija.
- COLS – podešava broj kolona u tabeli. Zadavanjem ovog atributa može se ubrzati rad internet čitača sa tabelama, posebno onim većim.
- WIDTH – podešava širinu tabele izraženu brojem piksela ili u procentima širine prozora. Ako se zadaje procentualna vrednost, broj treba da se završava znakom za procenat %.
- RULES – određuje koje će od unutrašnjih ivica biti prikazane. Moguće su sledeće vrednosti:
- NONE – uklanja sve unutrašnje ivice.
- GROUPS – prikazuje horizontalne ivice između grupa u sastavu tabele. Grupe se definišu pomoću oznaka <THEAD>, <TBODY>, <TFOOT> i <COLGROUP>.
- ROWS – prikazuje horizontalne ivice između svih redova u tabeli.
- COLS – prikazuje vertikalne ivice između svih kolona u tabeli.
- ALL – prikazuje sve ivice između kolona i redova u tabeli.

Oznaci za red tabele, <TR>, možemo zadati sledeće atribute:

- ALIGN – definiše horizontalno poravnanje teksta u ćelijama posmatranog reda. Moguća je jedna od sledećih vrednosti atributa: LEFT (tekst se poravnava ulevo), RIGHT (tekst se poravnava udesno) , CENTER (tekst se centrira), JUSTIFY (tekst se poravnava i sa leve i sa desne strane) i CHAR (tekst se poravnava prema znaku definisanom atributom CHAR)
- CHAR – definiše znak prema kome će se tekst poravnavati kada se koristi atribut ALIGN =CHAR.
- CHAROFF – definiše za koliko će tačaka tekst u ostatku reda biti pomeren pd prvog pojavljivanja znaka za poravnanje koji je definisan atributom CHAR.
- VALIGN – definiše način vertikalnog poravnanja teksta u ćelijama. Moguća je jedna od vrednosti:
- TOP – tekst se poravnava prema vrhu ćelije
- MIDDLE – tekst se poravnava prema sredini ćelije
- BOTTOM – tekst se poravnava prema dnu ćelije
- BASELINE – tekst se poravnava prema težišnoj liniji zajedničkoj za sve ćelije u redu.

Neki od atributa koji se mogu koristiti sa oznakama za ćeliju tabale, <TD>, i zaglavlje kolone ili reda tabele, <TH>, su:

- ALIGN – definiše način horizontalnog poravnanja teksta u ćeliji. Moguća je jedna od sledećih vrednosti: LEFT (tekst se poravnava u levo), RIGHT (tekst se poravnava u desno), CENTER (tekst se centrira), JUSTIFY (tekst se poravnava i sa leve i sa desne strane), CHAR (tekst se poravnava prema znaku definisanom atributom CHAR).

- CHAR – definiše tekst prema kojem će se izvršiti poravnanje.
- CHAROFF – definiše za koliko će tačaka tekst u ostatku reda biti pomeren od prvog pojavljivanja znaka za poravnanje koje je definisano atributom CHAR.
- COLSPAN – definiše broj kolona preko kojih se prostire posmatrana ćelija
- ROWSPAN – definiše broj redova preko kojih se prostire posmatrana ćelija

itd.

Opis tabele:

TABELA	Kolona 1	Kolona 2
Red 1	(1,1)	(1,2)
Red 1	(2,1)	(2,2)

U HTML jeziku, izgleda ovako:

```
<TABLE BORDER=2 COLS = 2>
            <TR>
                  <TH>TABELA</TH>
                  <TH>Kolona 1</TH>
                  <TH>Kolona 2</TH>
            </TR>
            <TR>
                  <TH ALIGN = LEFT>Red 1</TH>
                  <TD ALIGN=CENTER>(1,1)</TD>
            <TD ALIGN=CENTER>(1,2)</TD>
            </TR>

            <TR>
                  <TH ALIGN = LEFT>Red 1</TH>
                  <TD ALIGN=CENTER>(2,1)</TD>
                  <TD ALIGN=CENTER>(2,2)</TD>
            </TR>
 </TABLE>
```

2.8 Liste (spiskovi)

HTML podržava nenumerisane i numerisane liste, kao i listu definicije pojmova. Nenumemerisana lista se definiše oznakama <UL> </UL> i svaki element u listi je označen najčešće crnom tačkom. Numerisana lista se definiše oznakama <OL> </OL> i predstavlja listu u kojoj su elementi numerisani prema redosledu pojavljivanja. Atribut TYPE menja stil numeracije liste spiska i može imati jednu od sledećih vrednosti: 1, a, A, i, I. Vrednost od koje počinje numerisanje se definiše atributom START. Element numerisane ili ne numerisane liste se zadaje oznakama <LI> </LI>.

Pored nenumerisanih i numerisanih lista, HTML podržava i liste definicije pojmova. Lista definicije pojmova se zadaje oznakama <DL> </DL>. Svaki pojam koji se definiše se zadaje oznakama <DT> </DT>, dok se definicija nalazi unutar oznaka <DD> </DD>.

2.9 Forme (obrazci)

Forme ili obrasci se definišu oznakama <FORM> </FORM>. One predstavljaju oblasti koje sadrže objekte koji zahtevaju ulazne podatke od korisnika. Najčešće korišćena oznaka unutar forme je <INPUT>. Tip <INPUT> forme je zadat atributom TYPE i HTML prepoznaje sledeće tipove <INPUT> oznake:

- TEXT – definiše polje za unos teksta.
- PASSWORD – isto kao i TEXT samo što je tekst koji unosi korisnik skriven znakom „*“.
- RADIO – definiše polje tipa radio-dugmeta koje se može izabrati pritiskom miša. Polja ovog tipa uvek se javljaju u grupama, pri čemu svi elementi grupe imaju isto ime (zadaje se atributom NAME) i različite vrednosti (zadaje se atributom VALUE)
- CHECKBOX – formira polje koje može biti izabrano pritiskom miša.
- SUBMIT – formira dugme kojim se sadržaj forme šalje serveru na obradu.
- RESET – formira dugme kojim se forma vraća u svoje prvobitno stanje.
- HIDDEN – omogućava da se u obrazac uključe informacije koje ne treba da budu menjane. To se koristi u slučajevima kada je dokument formiran od strane skripta i treba da sadrži statičke informacije.
- IMAGE – funkcioniše slično kao dugme SUBMIT, osim što umesto dugmeta upotrebljava sliku koja se zadaje atributom SRC (SRC=”url slike”).
- BUTTON – formira dugme koje poziva skriptu.

Pored oznake <INPUT>, forme podržavaju i korišćenje oznaka <SELECT> </SELECT> -za definisanje padajućih listi mogućih vrednosti koje korisnik može da odabere. Svaka mogućnost (vrednost) padajuće liste se zadaje pomoću oznaka <OPTION> </OPTION>.

Oznakom <TEXTAREA> </TEXTAREA> se definiše polje za unošenje teksta u HTML formu. Ovo polje (koje se često naziva zona) je predviđeno za tekst od više redova. Atributi vezani za ovu oznaku su: COLS (definiše broj kolona za svaki red teksta u polju), NAME (ime polja koje će u paru sa vrednošću koja je uneta u polje biti dostavljeno skriptu), ROWS (određuje broj redova teksta koji se mogu smestiti u polje) itd. Unutar obrazaca se mogu koristiti još i oznake <LABEL></LABEL> (predstavlja naslov polja unutar frome), <FIELDSET></FIELDSET> (predstavlja skup polja u formi) i <LEGEND></LEGEND>(predstavlja potpis za skup polja određen oznakom <FIELDSET>).

2.10 Slike

HTML dokument dozvoljava i prikaz slika. To se postiže upotrbom oznake <IMG>, koja sadrži informacije o izvoru slike, načinu njenog prikazivanja i njnom ponašanju. Atributi vezani za ovu oznaku su:

- ALT – predstavlja alternativni tekst koji će biti ispisan ako korišćeni čitač nema mogućnost prikazivanja grafike.
- ALIGN – podešava način poravnanja slike i teksta koji je okružuje. Može imati jednu od vednosti: LEFT (slika se iscrtava na levoj strani dok je tekst raspoređen oko nje), RIGHT (slika se iscrtava na desnoj strani dok je tekst raspoređen oko nje), TOP (vrh teksta koji okružuje sliku poravnat je s vrhom slike), MIDDLE (težišna linija teksta koji okružuje sliku poravnata je sa sredinom slike), BOTTOM (težišna linija teksta koji okružuje sliku poravnata je sa dnom slike)
- BORDER – određuje veličinu okvira koji će biti iscrtan oko slike.
- HEIGHT i WIDTH – određuju veličinu na koju će slika biti skalirana.
- HSPACE i VSPACE – određuju dodatni prazni prostor tj. margine oko slike.
- SRC – je obavezan atribut i predstavlja **url** datoteke koja sadrži sliku.

2.11 Boje

U HTML dokumentu boje se zadaju u RGB sistemu kao heksadecimalni broj koji predstavlja kombinaciju crvene, zelene i plave boje. Najmanji broj koji može biti zadat jednoj komponenti boje je 0 (heksadecimalno #00) a najveći 255 (heksadecimalno #FF). Sledeća slika daje primere kombinacija crvene, zelene i plave komponente:

Boja	Boja HEX	Boja RGB
	#000000	rgb(0,0,0)
	#FF0000	rgb(255,0,0)
	#00FF00	rgb(0,255,0)
	#0000FF	rgb(0,0,255)
	#FFFF00	rgb(255,255,0)
	#00FFFF	rgb(0,255,255)
	#FF00FF	rgb(255,0,255)
	#C0C0C0	rgb(192,192,192)
	#FFFFFF	rgb(255,255,255)

W3C je predefinisao 16 standardnih imena boja koja se mogu koristiti umesto heksadecimalnih brojeva. To su: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white, i yellow.

3 MySQL

3.1 MySQL monitor i administrativni alati

MySQL monitor je aplikacija komandne linije, koja dopušta da se povežemo na MySQL server, da izdajemo upite i da vidimo njihove rezultate. Pokreće se izdavanjem komande mysql. Bitno je napomenuti da se svaka sql naredba završava znakom „ ; “ .

Pored MySQL monitora, MySQL distribucija sadrži brojne administrativne aplikacije. Ove aplikacije se nalaze u MySQL bin direktorijumu , kao i MySQL monitor, predstavljaju aplikacije komandne linije.

Neki od važnijih administrativnih alata su: mysqladmin i mysqldump.

Aplikacija mysqladmin se upotrbljava za izvršavanje administrativnih operacija na MySQL serveru, poput kreiranja, ili uklanjanja baza podataka i pregleda važećeg statusa servera. Za kreiranje baze podataka upotrebom mysqladmin alata sintaksa komande je sledeća:

```
mysqladmin create databsename
```

gde databazename predstavlja ime baze koju kreiramo. Slično predhodnoj komandi, da bi se izbrisala baza podataka, koristi se komanda:

```
mysqladmin drop databasename .
```

Kompletnu listu mysqladmin komandi možemo dobiti izdavanje sledeće komande:

```
mysqladmin –help .
```

Aplikacija mysqldump je veoma korisna za kreiranje kopija struktura tabela kao i sadržaja istih. Da bi se napravila datoteka koja sadrži sql komande za kreiranje tabela i popunjavanje sadržaja istih, ako one već sadrže podatke, mysqldump aplikacija se koristi na sledeći način:

```
mysqldump databasename .
```

3.2 MySQL tipovi podataka

Tipovi podataka koji se mogu koristiti u MySQL bazi podataka se mogu podeliti u tri kategorije:

- Numerički tipovi podataka
- Tipovi podataka koji predstavljaju datum i tačno vreme
- Tipovi stringova

MySQL koristi sve standardne ANSI SQL numeričke tipove podataka. To su:

- TINYINT – veoma mali ceo broj koji može biti označen ili neoznačen. Ako je broj označen onda se nalazi u opsegu od -128 do 127, a ako je broj neoznačen onda je u opsegu od 0 do 255.
- SAMLLINT – mali ceo broj koji može biti označen ili neoznačen. Ako je označen onda se nalazi u opsegu od -32.768 do 32.767, a ako je neoznačen, onda je dopušten opseg od 0 do 65.535.
- MEDIUMINT – ceo broj srednje veličine koji takođe može biti označen ili neoznačen. Ako je označen, onda je u opsegu od -8.388.608 do 8.388.607, a neoznačen je u opsegu od 0 do 16.777.215.

- INT – ceo broj koji je u opsegu od -2.147.483.648 do -2.147.483.647 ako je označen ili u opsegu od 0 do 4.294.967.295 ako je neoznačen.
- BIGINT – veliki ceo broj koji je u opsegu od -9.223.372.036.854.775.808 do -9.223.372.036.854.775.808 ako je označen, ili u opsegu od 0 do 18.446.744.073.709.551.615
- FLOAT (A, B) – predstavlja broj u pokretnom zarezu kod kojega se mogu definisati dužina prikaza (A), i broj decimala (B). Ako se ne navede dužina prikaza i broj decimala, onda se podrazumeva da je dužina prikaza 10, a broj decimala je 2. Decimalna tačnost za tip FLOAT ide do 24 decimale.
- DOUBLE (A, B) – predstavlja broj u pokretnom zarezu dvostruke preciznosti. Kao i kod FLOAT može biti definisana dužina prikaza i broj decimala, gde ako se pomenute vrednosti ne definišu, njihova podrazumevana vrednost je 16 za dužinu prikaza i 4 za broj decimala. Decimalna tačnost ide do 53 decimale. Sinonim za DUBLE je REAL.
- DECIMAL (ili NUMERIC) – predstavlja nezapakovan broj u pokretnom zarezu. Pre MySQL 5.03 verzije DECIMAL vrednosti su se čuvale kao stringovi dok od 5.03 verzije čuvaju se u binarnom formatu.

MySQL ima nekoliko dostupnih tipova padataka za smeštanje datuma i tačnog vremena. Ovi tipovi su fleksibilni pri unošenju, što znači da se mogu uneti i podaci koji nisu stvarni kao npr. 30. februar. Ova osobina za sobom povlači da odgovornost za greške prilikom provere datuma snosi osoba zadužena za razvoj aplikacije. MySQL samo proverava da je mesec u opsegu od 0 do 12, i da je dan između 0 i 31.

- DATE – datum je u formatu YYYY-MM-DD, u opsegu od 1000-01-01 do 9999-12-31. Tako npr. 1. junu 2006 godine će biti zapisan kao 2006-06-01.
- DATETIME – kombinacija datuma i tačnog vremena je u formatu YYYY-MM-DD HH:MM:SS, u opsegu od 1000-01-01 00:00:00 do 9999-12-31 23:59:59.
- TIME – smešta tačno vreme u formatu HH:MM:SS.
- YEAR – smešta godinu u format sa 2 ili 4 cifre. Ako je dužina naznačena kao 2, YEAR može biti od 1970 do 2069, a ako je dužina naznačena kao 4, onda je YEAR između 1901 do 2155. podrazumevana veličina je 4.

MySQL podržava sledeće tipove stringova:

- CHAR – string fiksne dužine između 1 i 255 karaktera. Ako se ne navede podrazumevana veličina je 1
- VARCHAR - string promenljive dužine. Dužina može biti navedena u opsegu od 0 do 65535. Za razliku od CHAR vrednosti, string čuvaju u onoliko bajtova memorije kolika je dužina stringa plus bajt za dužinu stringa.
- BLOB (Binary Large Objects) – upotrebljava se za smečtanje velike količine binarnih podataka, poput slika i drugih fajlova.
- TEXT – slično kao i BLOB, s tim da je razlika između ova dva tipa podataka u tome što je poređenje podataka sačuvanih kao blob „osetljivo pitanje“, dok je za TEXT podatke „neosetljivo“.
- ENUM – definiše listu članova za koju vrednost mora biti selektovana, tj. ako definišemo ENUM kao ENUM(‘A’, ‘B’, ‘C’) onda vrednost tipa ENUM može biti ‘A’, ‘B’ ili ‘C’.

3.3 Kreiranje tabela

Kreiranje tabela zahteva tri bitne informacije:

- naziv tabele
- naziv polja
- definicije za svako polje

Sintaksa naredbe za kreiranje tabele je sledeća

```
CREATE TABLE ime_tabele (ime_kolone tip_kolone);
```

Pored ovih osnovnih podataka koji su potrebni da bi se tabela kreirala, prilikom kreiranja tabele mogu se zadavati i ključevi i indexi. MySQL koristi dva tipa ključeva. To su primarni i jedinstveni ključevi. Primarni ključevi su ujedno i jedinstveni ključevi i u svakoj tabeli bi trebalo da postojji primarni ključ, ako ni zbog čega drugog, bar da bi bile olakšane međusobne veze između tabela. Primarni ključ ne može imati vrednost NULL, tako da njemu treba, prilikom kreiranja, zadati i neku podrazumevanu vrednost:

```
CREATE TABLE ime_tabele(ime_kolone tip_kolone PRIMARY KEY NOT NULL DEFAULT
'vrednost');
```

Ako je primarni ključ ceo broj, na njega se može primeniti i klauzula AUTO_INCREMENT. To znači da ako je polje koje koristi AUTO_INCREMENT prazno, MySQL će popuniti to polje sledećim najvećim celim brojem:

```
CREATE TABLE ime_tabele(ime_kolone tip_kolone PRIMARY KEY NOT NULL
AUTO_INCREMENT);
```

Dodavanje jedinstvenih ključeva je slično dodavanju primarnih ključeva. Razlika je u tome što polja definisana kao jedinstveni ključevi mogu sadržati NULL vrednost:

```
CREATE TABLE ime_tabele(
ime_kolone tip_kolone PRIMARY KEY NOT NULL AUTO_INCREMENT
ime2_kolone tip_kolone UNIQUE
);
```

Indekeks baze podataka je sličan indeksu na kraju knjige – pomaže da se brže pronađu termini. Ako tabela ima primarni ključ, ta kolona je automatski indeksirana. Ostali indeksi mogu se definisati u toku naredbe kreiranja tabele na sledeči način:

```
CREATE TABLE ime_tabele(
ime_kolone tip_kolone PRIMARY KEY NOT NULL AUTO_INCREMENT,
ime2_kolone tip_kolone UNIQUE,
ime3_kolone tip_kolone,
INDEX index_kol3 (ime3_kolone)
);
```

Još jedan način za dodavanje indeksa je upotrebom komande CREATE INDEX koja ima sledeću sintaksu:

```
CREATE INDEX index_kol ON ime_tabele (ime_kolone, ime2_kolone);
```

3.4 Popunjavanje Tabela

SQL komanda za dodavanje novih slogova u bazu podataka je INSERT. Osnovna sintaksa INSERT komande je:

```
INSERT INTO ime_tabele ( lista kolona) VALUES (lista vrednosti kolona);
```

U okviru liste vrednosti kolona stringovi se moraju ograditi navodnicima. SQL standard su jednostruki navodnici, ali MySQL dopušta upotrebu i jednostrukih i dvostrukih navodnika. Na primer, ako imamo tabelu **customer** (mušterija) sa kolonama **firstname** i **lastname,** ubacivanje jednog imena bi izgledalo kao:

```
INSERT INTO customer  (firstname) VALUES (“Vanja”);
```

ili

```
INSERT INTO customer (firstname) VALUES ('VANJA');
```

Ako se ne navede ni jedna kolona u listi kolona MySQL očekuje da sve kolone budu u listi vrednosti:

```
INSERT INTO customer VALUES("Vanja",  "Petkovic");
```

U okviru INSERT naredbe mogu se koristiti i rezultati funkcija, kao npr. funkcije za datum i tačno vreme.

3.5 Selektovanja podataka iz tabela

SQL naredba za selektovanje slogova iz tabele je SELECT. Njena sintaksa je:

```
SELECT lista_izraza_i_kolona FROM ime_tabele
[WHERE uslovi]
[ORDER BY lista_kolona [ASC | DESC]]
[LIMIT offset, broj_slogova]
```

Da bi se selektovale sve kolone iz tabele **ime_tabele** (a da se ne navode sve kolone u listi kolona), može se upotrebiti specijalni znak *****, koji zamenjuje sve elemente. Da bi se u SELECT naredbi zadali neki uslvovi koje zadovoljavaju slogovi u tabeli, koristi se klauzula WHERE. Operatori poređenja su:

- = - jednako
- != - različito
- <= - manje ili jednako
- < - manje
- >= - veće ili jednako
- > - veće
- BETWEEN – se koristi za poređenje celih brojeva jer pretražuje rezultate između zadatih vrednosti
- LIKE - je operator koristan za poređenje stringova koji ima dva džoker znaka:
- % - odgovara višestrukim karakterima
- _ - odgovara tačno jednom karakteru

Rezultati SELECT upita su poređani onako kako se pojavljuju u tabeli. Ako želimo rezultate da poređamo na neki specifičan način, npr. po datumu, imenu, ID-u, onda svoje zahteve naznačavamo upotrebom klauzule ORDER BY.

Upotrebom LIMIT klauzule možemo ograničiti broj slogova koji se dobija kao rezultat nekog SELECT upita. Postoje dva zahteva za LIMIT klauzulu: offset – početna pozicija i broj slogova.

MySQL unutar SELECT naredbe dopušta korišćenje i funkcije prikupljanja. Neke od njih su COUNT(), MIN() i MAX(). Na primer ako želimo da dobijemo koliko ima slogova u nekoj tabeli možemo zadati sledeći SQL upit:

```
SELECT COUNT(ime_kolone) FROM ime_tabele;
```

Ako bismo željeli da dobijemo koliko različitih slogova, prema nekoj koloni, ima u nekoj tabeli, onda bismo rezultat dobili upotrebom ključne reči DISTINCT na sledeći način:

```
SELECT COUNT(DISTINCT ime_kolone) FROM ime_tabele.
```

SELECT naredba dozvoljava i upotrebu klauzule GROUP BY. Ona koncentriše elemente rezultujućeg skupa. WHERE klauzula nije dozvoljena kada koristimo GROUP BY klauzulu, ali se uslovi tada zadaju upotrebom klauzule HAVING.

SELECT naredbu možemo koristiti i u okviru INSERT naredbe i na taj način možemo kopirati sadržaj jedne tabele u drugu. Sintaksa ove naredbe je:

```
INSERT INTO table_name (lista_kolona) SELECT_UPIT;
```

3.6 UPDATE komanda

UPDATE je SQL komanda koja se upotrebljava za izmenu sadržaja jedne, ili više kolona u postojećem slogu. Najosnovnija sintaksa UPADTE naredbe je:

```
UPADTE ime_tabele
SET ime_kolone1 = 'nova vrednost',  ime_kolone2 = 'nova vrednost'...
[WHERE uslovi];
```

Dakle, da bismo izmenili neki slog u tabeli, treba da za svaku kolonu, čiju vrednost želimo da izmenimo, navedemo novu vrednost. Ako želimo da se izmene odnose samo na slogove koji zadovoljavaju neke uslove, onda koristimo WHERE kaluzulu kao i u SELECT naredbi. Kao i u okviru INSERT naredbe, i u UPDATE naredbi se mogu koristiti rezultati funkcija koji se dodeljuju kolonama.

3.7 REPLACE komanda

MySQL podržava i upotrebu REPLACE naredbe koja je slična INSERT naredbi:

```
REPLACE INTO ime_tabele (lista_kolona) VALUES (lista_vrednosti);
```

Efekat REPLACE naredbe je sledeći: ako slog koji umećemo u tabelu sadrži vrednost primarnog ključa, koja se podudara sa vrednošću primarnog ključa sloga koji je već u tabeli, slog u tabeli će biti obrisan, a novi će biti umetnut na njegovo mesto.

3.8 DELETE komanda

DELETE komanda se upotrebljava za brisanje slogova i tabela. Njena osnovna sintaksa je:

```
DELETE FROM ime_tabele
[WHERE uslovi];
```

Ako želimo da obrišemo samo slogove koji zadovoljavaju neke uslove, onda koristimo WHERE kaluzulu kako bismo zadeli te uslove. Inače, ako zadamo DELETE naredbu bez WHERE klauzule, ona će ukloniti sve slogove iz tabele. Da bismo optimizovali uklanjanje svih slogova iz tabele, umesto DELETE naredbe bez WHERE klauzule, možemo da koristimo TRUNCATE naredbu:

```
TRUNCATE TABLE ime_tabele;
```

Ovom naredbom je MySQL odbacio tabelu ime_tabele i ponovo je kreirao bez slogova.

3.9 SHOW i DROP komanda

SHOW naredba može biti upotrebljena za izlistavanje svih baza podataka na serveru:

```
SHOW DATABASES;
```

ili za izlistavanje svih tabela u okviru neke baze podataka:

```
SHOW TABLES ;
```

DROP naredba se takođe može primeniti na bazu podataka na serveru:

```
DROP DATABASE [IF EXIST] ime_baze;
```

ili na tabelu u okviru neke baze podataka:

```
DROP TABLE [IF EXIST] ime_tabele.
```

Ovim naredbama se brišu cele baze podataka na serveru ili tabele u okviru neke baze podataka. IF EXIST klauzula služi za proveru da li imenovana baza, odnosno tabela postoji pre brisanja.

3.10 Izmena strukture tabele

Da bismo videli opis postojeće tabele, možemo upotrebiti naredbu DESCRIBE. Pomoću nje možemo da vidimo sledeće informacije o postojećoj tabeli:

- Nazive polja
- Tipove polja
- Da li kolona može sadržati NULL vrednost
- Da li je kolona ključ i kog je tipa
- Bilo koju podrazumevanu vrednost
- Dodatnu informaciju, kao što je auto_increment.

Sintaksa naredbe DESCRIBE je:

```
DESCRIBE ime_tabele [ime_kolone];
```

Ako ne navedemo ime kolone čiji nam opis treba, onda ćemo dobiti opis svih kolona u navedenoj tabeli.

Naredbom RENAME možemo da na brz način preimenovati jednu ili više tabela. Njena sintaksa je:

```
RENAME TABLE staro_ime TO novo_ime [, staro_ime2 TO novo_ime2];
```

Ako preimenujemo više tabela u toku jednog upita, akcija se dešava s leva na desno, pa tako upitom:

```
RENAME TABLE tabela TO tabela2, tabela2 TO tabela;
```

postižemo da **tabela** dobije novo ime **tabela2**, da bismo joj opet vratili staro ime. Takođe, naredbom RENAME možemo premestiti tabelu iz jedne baze podataka u drugu, sve dok su obe baze podataka na istom fajl sistemu:

```
RENAME TABLE  stara_baza.ime_tabele TO nova_baza.ime_tabele;
```

ALTER je naredba koja ima više varijacija od bilo koje druge SQL naredbe. Njenom upotrebom možemo izvršiti nekoliko različitih tipova operacija: dodavanje, izmenu ili brisanje iz tabela. Kada se izdaje ALTER naredba, MySQL kreira privremenu kopiju tabele nad kojom se vrše izmene. Zatim se operacije izmena obavljaju nad kopijom tabele, i kad su one izvršene briše se originalna tabela a kopija dobija ime originalne tabele. Dok se vrše izmene na kopiji tabele, iz originalne tabele se mogu selektovati podaci, ali se ne može vršiti nikakva izmena nad podacima. Osnovna sintaks ALTER naredbe je:

```
ALTER TABLE  ime_tabele  alter_speciikacija;
```

Alter specifikacije se mogu podeliti u tri grupe: dodavanje, izmena i brisanje elemenata u okviru tabele, poput polja, kjlučeva, indeksa i slično.
Alter specifikacije za dodavanje struktura tabelama su:

- ADD COLUMN (definicija) [FIRST | AFTER ime_kolone]
- ADD INDEX [ime_indeksa] (ime_indeks_kolone)
- ADD PRIMARY KEY (ime_kolone)
- ADD UNIQUE [ime_indeksa] (ime_indeks_kolone)

Alter specifikacije za izmenu struktura postojeće tabele su:

- ALTER COLUMN ime_kolone {SET DEFAULT podrazumevana_vrednost DROP|DEFAULT}
- CHANGE COLUMN stara_kolona definicija_kreiranja [FIRST|AFFTER ime_kolone]
- MODIFY COLUMN definicija_kreiranja [FIRTST|AFTER ime_kolone]

Alter specifikacije za uklanjanje su:

- DROP COLUMN ime_kolone
- DROP INDEX ime_indeksa
- DROP PRIMARY KEY

3.11 MySQL string funkcije

MySQL poseduje ugrađene funkcije za rad sa stringovima. One se mogu podeliti u 5 grupa: funkcije dužine i povezivanja, funkcije dodavanja i uklanjanja , funkcije podstringova, funkcije lociranja i pozicioniranja i funkcije modifikacije stringova.

Funkcije koje se odnose na dužinu uključuju LENGTH(), OCTET_ LENGTH(), CHAR_ LENGTH() i CHARACTER_LENGTH() koje broje koliko ima karaktera u zadatom stringu. Funkcija nadovezivanja dva ili više stringova je CONCAT(). Ako je potrebno nadovezati stringove sa nekim separatorom, onda se koristi funkcija CONCAT_WS().

U funkcije dodavanja i uklanjanja spadaju funkcije za uklanjanje dodatnih karaktera iz stringova. RTRIM() i LTRIM() uklanjaju praznine sa desne, odnosno leve strane stringa. Za uklanjanje karaktera koji su praznine koristi se TRIM() funkcija. Ako želimo da uklonimo karaktere **X** sa početka stringa korističemo TRIM funkciju na sledeći nači TRIM(LEADING X from neki_string), a ako želimo da uklonimo **X** sa kraja stringa korističemo TRIM na sledeći način TRIM(TRAILING X from neki_string). Ako ne koristimo ključne reči LEADING ili TRAILING, onda će zadati karakter biti uklonjen i sa početka i sa kraja stringa. Funkcije RPAD() i LPAD() se koriste za dodavanje karaktera sa desne, odnosno leve strane.

Ako želimo iz nekog stringa da izdvojimo neki podstring, možemo koristiti funkciju SUBSTRING(), kojoj kao parametre zadajemo string iz kojeg tražimo podstring, poziciju odakle počinje podstring i dužinu podstringa. Ako želimo da izdvojimo nekoliko karaktera sa desne ili leve strane stringa, koristićemo funkciju RIGHT() ili LEFT().

Ukoliko želimo da pronađemo neki string, u okviru nekog stringa, možemo koristiti funkciju LOCATE() na sledeći način LOCATE(podstring, neki_string).

U funkcije modifikacije stringa spadaju funkcije LCASE(), UCASE() i REPLACE(). Funkciom LCASE transformiše string u mala slova, dok UCASE transformiše string u velika slova. Funkcija REPLACE se upotrebljava kada želimo da svako pojavljivanja nekog podstringa u stringu zamenimo nekim drugim podstringom.

3.12 MySQL numeričke funkcije

U MySQL-u dostupne su osnovne aritmetičke operacije: sabiranje, oduzimanja, množenje i deljenje. Tako na primer, ako izvršimo upit SELECT 10+20; , dobićemo rezultat 30. Pored ovih osnovnih aritmetičkih operacija, postoji i funkcija MOD() čijom se upotrebom dobija ostatak pri deljenju dva broja. Pored aritmetičkih funkcija MySQL podržava i još neke matematičke funkcije. Među njima su:

- SIGN() – vraća znak broja
- ABS() – vraća apsolutnu vrednost broja
- LOG10() – izračunava logaritam osnove 10 nekog broja
- LOG() – izračunava prirodni logaritam nekog broja
- POW() ili POWER() – koristi se za stepenovanje brojeva
- SQRT() – vraća koren broja
- SIN(), COS(), TAN() – izračunavaju sinus, kosinus i tangens broja
- ASIN(), ACOS(), ATAN() - izračunavaju arkus sinus, arkus kosinus i arkus tangens broja
- DEGREES() – prevodi radijane u stepene
- RADIANS() – prevodi stepene u radijane
- ROUND() – funkcija zaokruživanja

3.13 MySQL funkcije vezane za datum i vreme

MySQL poseduje skup veoma korisnih funkcija za rad sa datumom i tačnim vremenom. Neke od njih su:

- DAYOFWEEK() – vraća indeks dana u sedmici. Prvi dan u sedmici je nedelja sa indeksom 1, a poslednji dan u sedmici je subota sa indaksom 7.
- WEEKDAY() – kao i DAYOFWEEK vrača indeks dana u sedmici, s tim što je ovde prvi dan u sedmici ponedeljak sa indeksom 0, a poslednji dan u sedmici je nedelja sa indeksom 6.
- DAYOFMONTH() – vraća indeks dana u mesecu, koji je u opsegu od 1 do 31.
- DAYOFYEAR() – vraća indeks dana u godini, koji je u opsegu od 1 do 366.
- DAYNAME() – vraća naziv dana u sedmici za konkretan datum.
- MONTH() – vraća redni broj meseca u godini za konkretan datum.
- MONTHNAME() – vraća naziv meseca za konkretan datum.
- WEEK() – vraća redni broj sedmice u godini za konkretan datum.
- HOUR(), MINUTE(), SECOND() – vraćaju sate, minute i sekunde za konkretno vreme
- DATE_ADD() – sabira zadati datum sa nekim vremenskim intervalom.
- DATE_SUBB() – oduzima od zadatog datuma neki vremenski interval.
- CURDATE() – vraća važeći datum u formatu YYYY-MM-DD.
- CURTIME() – vraćaju tačno vreme u formatu HH:MM:DD.

3.14 Transakcije

Transakcija predstavlja sekvencijalnu grupu operacija manipulisanja bazom podataka koja je izvršena kao jedna radna celina. Drugim rečima transakcija će se uspešno izvršiti samo ako se uspešno izvrše sve operacije unutar nje. Ako bilo koja operacija unutar transakcije ne uspe, onda je cela transakcija neuspešna.

Transakcije imaju četiri standardne osobine koje se označavaju skraćenicom ACID. Te osobine su:

- Atomičnost (atomicity) – odnosi se na mogućnost baze podataka da se ili sve operacije u okviru transakcije uspešno izvrše ili se nijedna operacije transakcije ne izvrši.
- Konzistentnost (consistency) – baza podataka uspešno menja stanja sve dok se transakcija ne izvrši.
- Izolacija (isolation) – transakcije su međusobno nezavisne i transparentne.
- Izdržljivost (durability) – osigurava da rezultat, ili efekat izvršene transakcije, opstaju u slučaju pada sistema.

U MySQL-u transakcije započinju naredbom BEGIN WORK i završavaju se naredbama COMMIT ili ROLLBACK. Kada je transakcija uspešno završena, treba izdati naredbu COMMIT da pi operacije izvršene u okviru transakcije imale efekta. Ako dođe do greške unutar transakcije, treba izdati naredbu ROLLBACK kako bi se svaka tabela vratila u stanje pre početka transakcije.

4 PHP

4.1 Tipovi podataka i promenljive

PHP podržava osam osnovnih tipova podataka. To su četiri skalarna tipa podataka:

- boolean – logički tip podataka koji može imati vrednosti TRUE ili FALSE
- integer – celobrojni tip podataka
- float (double) – tip podataka u pokretnom zarezu dvostruke preciznosti
- string – znakovni tip podataka

dva složena tipa podataka:

- array – nizovni tip podataka
- object – objekat, koristi se za čuvanje instanci klasa

i dva specijalna tipa podataka:

- NULL – promenljive kojima nije dodeljena vrednost imaju vrednost NULL
- resource – sadrži referencu ka spoljnom resursu (npr. veza sa bazom podataka)

U većini programskih jeika promenljive mogu da sadrže samo jedan tip podataka koji se pre upotrebe promenljive mora deklarisati kao što je u C-u. U PHP-u se tip promenljive određuje na osnovu vrednosti koja joj je dodeljena. Na primer ako napravimo dve promenljive a i b na sledeći način:

```
$a = 0;
$b = 0.00;
```

onda je promenljivoj $a dodeljena vrednost 0 celobrojnog tipa pa je i $a celobrojnog tipa.
Analogno tome je promenljiva $b tipa float (double). Ako bismo sada vrednost promenljive $a promenili na sledeći način:

```
$a = 'Neki Text';
```

promenljiva $a bi u tom slučaju postala tipa string. Ovo znači da PHP menja tip promenljive prema onome što se u njoj nalazi, tj. PHP nije strogo tipiziran jezik.

PHP poseduje skup funkcija za manipulisanje promenljivim. Neke od njih su:

- string gettype (var) – vraća tip promenljive var. Moguće vrednosti vraćenog stringa su boolean, integer, double, string, array, object, resource, NULL ili unknown type.
- bool is_bool (mixed var) - proverava da li je promenljiva var tipa boolean.
- bool is_double (mixed var) - proverava da li je promenljiva var tipa double.
- bool is_integer (mixed var) - proverava da li je promenljiva var tipa integer.
- bool is_array (mixed var) - proverava da li je promenljiva var tipa array.
- bool is_resource (mixed var) - proverava da li je promenljiva var tipa resource.
- bool is_array (mixed var) - proverava da li je promenljiva var tipa array.
- bool is_object (mixed var) - proverava da li je promenljiva var tipa object.
- bool is_null (mixed var) - proverava da li je promenljiva var tipa NULL.
- float floatval (mixed var) – vraća float vrednost promenljive.

4.2 Operatori i izrazi

Operatori su simboli koji u izrazima određuju neke operacije. Operatori imaju jedan, dva ili tri argumenta(operanda). Operatori u PHP-u se mogu podeliti u nekoliko grupa: aritmetički operatori, operatori za znakovne vrednosti, operatori dodele, prefiksno i sufiksno uvećanje i umanjenje, operator reference, operatoti poređenja, logički operatori, operatori nad bitovima, uslovni operator, operator zanemarivanja greške, operator izvršenja, operatori za rad sa nizovima i operator za utvrđivanje tipa.

Aritmetički operatori

Operator	Ime	Primer
+	sabiranje	$a + $b
-	oduzimanje	$a - $b
*	množenje	$a * $b
/	deljenje	$a / $b
%	modulo	$a % $b

Sabiranje, oduzimanje, moženje i delenje su operacije koje se odvijaju i kao istoimene matematičke operacije. Ove operacije uzimaju po dva argumenta, ali se oduzimanje može koristiti i kao unarni operator za negativne brojeve, npr.:

```
$a = -1;
```

Operator modulo vraća ostatak celobrojnog delenja promenljive $a sa $b.

Operatori za znakovne vrednosti

Operator	Ime	Primer
.	spajanje stringova	$a . $b
.=	spajanje i dodelivanje stringova	$a .= $b

Oba ova operatora za rad sa stringovima su binarna i levo asocijativna.

Operatori dodele

Operator	Upotreba	Ekvivalentan izrazu
+=	$a += $b	$a = $a + $b
-=	$a -= $b	$a = $a - $b
*=	$a *= $b	$a = $a * $b
/=	$a /= $b	$a = $a / $b
%=	$a %= $b	$a = $a % $b
.=	$a .= $b	$a = $a . $b
=	$a = X (X – neka vrednost)	

Prvih pet operatora dodele u tabeli se nazivaju kombinovanim operatorima dodele i svaki od njih je skračenica za izvršenje određene operacije nad promenljivom i upisivanja rezultata u tu promenljivu.

Operatori za prefiksno i sufiksno uvećanje i umanjenje

Operatori za uvećanje i umanjenje su ++ i --. Ako npr. imamo izraze $a = 4; i $b = ++$a; . U drugom izrazu ++ predstavlje prefiksni operator uvećanja i nakon njegovog izvršenja promenljive $a i $b imaju vrednost 5. Ako bismo u drugom izrazu umesto prefiksnog operatora koristili sufiksni operator na sledeći način $b = $a++, onda bi nakon njegovog izvršenja promenljiva $a imala vrednost 5, dok promenljiva $b vrednost 4.

Operator reference

Operator referenca se predstavlja znakom &. Kada se nekoj promenljivoj dodeli neka druga promenljiva obično se pravi kopija prve promenljive koja se smešta negde u memoriju. Na primer

```
$a = 5;
```

```
$b = $a;
```
Na ovaj način je vrednost promenljive $a kopirana u promenljivu $b. Ako bi se sada vrednost promenljive $a postavila na 7, promenljiva $b bi i dalje zadržala vrednost 5. Kopiranje može da se izbegne upotrebom operatora reference na sledeći način:
```
$b =  &$a;
```
Sada promenljiva $b ima vrednost 5 baš kao i promenljiva $a, ali sada se svaka izmena izvršena na promenljivu $a odnosi i na promenljivu $b. To je zato što sada obe promenljive i $a i $b pokazuju na isti blok memorije.

Operatori poređenja

Operator	Ime	Upotreba
= =	jednako	$a = = $b
= = =	identično	$a = = = $b
!=	različito	$a != $b
!= =	nije idntično	$a != = $b
<>	različito	$a <> $b
<	manje od	$a < $b
>	veće od	$a > $b
<=	manje ili jednako	$a <= $b
>=	veće od ili jednako	$a >= $b

Operatori poređenja se koriste za poređenje dve vrednosti. Izrazi u kojima se upotrebe ovi operatori vraćaju, u zavisnosti od rezultata poređenja, vrednost *true* ili *false*.

Logički operatori

Operator	Ime	Upotreba	Rezultat
!	negacija	!$a	Vraća true ako je $a false i obrnuto
&&	konjunkcija	$a && $b	Vraća true ako su $a i $b true, u protivnom vraća false
||	disjunkcija	$a || $b	Vraća true ako su $a ili $b ili oba true, u protivnom vraća false
and	konjunkcija	$a and $b	Isto kao &&, ali s nižim prioritetom
or	disjunkcija	$a or $b	Isto kao ||, ali s nižim prioritetom

Logički operatori se koriste za kombinovanje rezultata logičkih izraza. Na primer ako treba proveriti da li je vrednost promenljive $a između 0 i 100, treba proveriti da li su ispunjena oba uslova $a >= 0 i $a <= 100. To nam omogućava operator logičke konjunkcije:

```
$a >= 0 && $a <= 100;
```

Operatori nad bitovima

Operator	Ime	Upotreba	Rezultat
&	konjunkcija	$a & $b	Bitovi koji su 1 u $a i $b oni su 1 i u rezultatu
|	disjunkcija	$a | $b	Bitovi koji su 1 u $a ili $b oni su 1 i u rezultatu
~	negacija	~$a	Bitovi koji su 1 u $a nisu 1 u rezultatu i obrnuto
^	isključiva disjunkcija	$a ^ $b	Bitovi koji su 1 u $a ili $b ali ne u oba, oni su 1 i u rezultatu
<<	pomeranje ulevo	$a << $b	Pomera bitove u $a ulevo za $b mesta
>>	pomeranje udesno	$a >> $b	Pomera bitove u $a udesno levo za $b mesta

Operatori nad bitovima omogućavaju da se ceo broj obrađuje kao grupa bitova koja ga predstavlja u memoriji.

Uslovni operator

Uslovni operator ? ima istu sintaksu i deluje identično kao i uslovni oprator u C-u:

uslov ? vrednost ako je uslov true : vrednost ako je uslov false

Operator zanemarivanja greške

U PHP-u postoji i operator zanemarivanja greške, @, može se uportrbiti u svakom izrazu. Npr.:

$a = @(57/0);

Bez operatora @, izvršno okruženje bi generisalo upozorenje „delenje nulom“, ali sa operatorom @ greška se zanemaruje.

Operator izvršenja

Operator izvršenja je predstavljen parom inverznih apostrofa ``. Sve što se nalazi između inverznih apostrofa, PHP pokušava da izvrši kao komandu zadatu na komandnoj liniji servera. Rezultat komande predstavlja vednost izraza. Na pimer, u operativnim unix-olokim sistemima možemo upotrebiti

```
$out = `ls –al`;
echo ’<pre>’.$out.’</pre>’
```

kako bismo generisali spisak datoteka u direktorijumu i dodelili ga promenljivoj $out.

Operatori za rad sa nizovima

Operator	Ime	Upotreba	Rezultat
+	unija	$a + $b	Vraća niz koji se sastoji od svih elemenata niza $a i $b
= =	jednako	$a = = $b	Vraća true ako $a i $b imaju jednake elemente
= = =	identično	$a = = = $b	Vraća true ako $a i $b imaju jednake elemente u jednakom redosledu
!=	različito	$a != $b	Vraća true ako je $a različito od $b
<>	različito	$a <> $b	Vraća true ako je $a različito od $b
!= =	nije identično	$a != = $b	Vraća true ako $a nije identičan $b

Pored opearatora za rad sa nizovima, koji su navedeni u predhodnoj tabeli, postoji i operator []. Ovo je operator koji omogućava da se pristupi pojedinim elementima niza.

Operator za utvrđivanje tipa

U objektno orijentisanom programiranju postoji jedan operator za utvrđivanje tipa objekta. To je operator *instanceof* . Ovaj operator omogućava da se ispita da li je neki objekat instanca neke zadate klase:

```
class nekaKlasa();
objekat = new nekaKlasa();
if(objekat instanceof nekaKlasa)
      //URADITI NEŠTO
```

4.3 Kontrola toka programa

Kontrola toka programa se vrši preko upravljačkih struktura. Možemo ih grupisati u uslovne ili strukture sa grananjem i strukture sa ponavljanjem ili petlje.

Kada god treba da se donese neka odluka, tada se koriste uslovne strukture. U uslovne strukture spadaju iskazi *if, else, elseif* i *switch.*

Iskazi if

Iskaz if je jedan od najvažnijih iskaza mnogih programskih jezika pa tako i PHP-a. Njegova sintaksa je:

```
if ( uslov )
      iskaz
```

Ako je uslov zadovoljen onda se izvršava iskaz. Često je unutar jednog uslovnog iskaza potrebno izvršiti više iskaza. Tada se ti iskazi grupišu u *blok*. Da bi se grupa iskaza deklarisala kao blok, neophodno je upotrebiti vitičaste zagrade.

Iskazi else

Često je potrbno izvršiti neki alternativni iskaz u slučaju da uslov iz if iskaza nije ispunjen. To se postiže iskazom else:

```
if ( uslov )
      iskaz1
else
      iskaz2
```

Iskazi elseif

Za mnoge odluke koje treba da se donesu postoji više od dve opcije. Pomoću iskaza elseif može se napraviti redosled izvršavanja tih opcija. Sintaksa elseif iskaza je sledeća:

```
if ( uslov1 )
      iskaz1
elseif  ( uslov2 )
      iskaz2
      .
      .
      .
```

Ako uslov1 iz if iskaza nije ispunjen, onda se prelazi na izvršenje iskaza elseif i ako je njegov uslov (uslov2) ispunjen onda se izvršava iskaz2. Reč elseif se može kucati i sa razmakon kao else if.

Iskaz switch

Iskaz switch deluje slično kao iskaz if, ali omogućava da iskaz ima više od dve vrednosti. U iskazu switch uslov može imati više različitih vrednosti, koje moraju biti skalarnog tipa (integer, float ili string).

```
switch ( a ) {
      case a1:
            iskaz1
            break;

      case a2:
            iskaz2;
            break;

      .
      .
      .

      default:
            iskaz3;
            break;
}
```

Kada je potrebno da se neki deo koda ponovi više puta to se može izvesti upotrebom petlji.

Petlja while

Petlja while kao i iskaz if zavisi od uslova. Razlika između if iskaza i petlje while je u tome što iskaz if izvršava blok nardbi kad god je uslov izvršen a petlja while izvršava blok naredbi dokle god je taj uslov ispunjen:

```
while ( uslov)
      iskaz
```

Petlja do ... while

Sintaksa do while petlje je:

```
do
      iskaz
while ( uslov )
```

Razlika između petlje while i do while je u tome što se uslov druge petlje ispituje na kraju. To znači da se iskaz unutar do while petlje izvršava sigurno jedanput, čak i kad uslov nije ispunjen.

Petlje for i foreach

Sintaksa for petlje je:

```
for ( izraz1; uslov; izraz2 )
      izraz3;
```

Izraz1 se izvršava jednom na početku petlje. U njemu se obično zadaje početna vrednost brojača u petlji. Uslov se ispituje pre svake iteracije i ako nije zadovoljen izvršavanje petlje prestaje. Ako je uslov ispunjen, izraz3 se izvršava po jednom u svakoj iteraciji. I na kraju svake iteracije izvršava se izraz2. On obično menja vrednost brojača u petlji.

Počev od verzije PHP 4, postoji i petlja foreach. Ona predstavlja jednostavnu strukturu za iteraciju kroz nizove. Petlja foreach predstavlja jednostavniji oblik for petlje za iteraciju kroz nizove. Njena sintaksa je:

```
foreach (array_expression as $value)
      izraz
```

4.4 Funkcije

Kao i u većini programskih jezika i u PHP-u moguća je upotreba funkcija. Deklaracija funkcije započinje rezervisanom rečju *function* iza koje slede ime funkcije i parametri, a zatim i kod koji se izvršava kada je funkcija pozvana. Jedan jednostavan proimer funkcije bi bio:

```
function my_function( ) {
      echo ‘MY FUNCTION’;
}
```

Funkcije se pozivaju upotrebom njihovih imena, npr.:

```
my_function();
```

Bitno je naglasiti da PHP ne pravi razliku između malih i velikih slova prilikom pozivanja funkcija.

Za povratak iz funkcije koristi se ključna reč *return.* Upotrebom ove ključne reči moguće je i vratiti neku vrednost iz funkcije.

PHP podržava prenošenje parametara funkcijama po vrednosti i po referenci. Tako, ako nije potrebno sačuvati promene izvršene nad nekom promenljivom unutar funkcije, onda tu promenljivu prosleđujemo funkcije kao parametar po vrednosti, a ako je pak potrebno sačuvati promene izvršene u funkciji nad nekom promenljivom, onda je neophodno da se taj parametar funkcije prosledi po referenci.

4.5 Stringovi

PHP poseduje bogat skup funkcija za rad sa stringovima. Koristeći ove funkcije moguće je štampati, formatirati, odsecati, skraćivati, spajati, rastavljati, određivati dužinu stringa kao i izvršiti mnoge druge operacije nad stringovima.

Za štampanje stringova koriste se funkcije print i echo. Tehnički, print nije funkcija već predstavlja jezičku konstrukciju. To znači da se sa komandom print ne moraju koristiti zagrade, već sve što se nađe iza ključne reči print pa do tačke zarez biće odštampano. Prototip „funkcije“ print je:

```
print (string);
```

Za štampanje jednog ili više stringova može se koristiti i ključna reč echo. Kao i print i echo je konstrukcija samog jezika, a ne funkcija. Kad se prosledi više argumenata u konstrukciju echo, oni moraju biti razdvojeni zarezima. Sintaksa je sledeća:

```
echo string1[, string2…]
```

Pored ovih jezičkih konstrukcija za štampanje stringova, postoje i funkcije za formatirano štampanje stringova. To su funkcije sprintf i printf i one su, po koncepciji, iste kao iste funkcije u programskom jeziku C. Prototipovi ovih funkcija izgledaju ovako:

```
string sprintf( string format [, mešani argumenti] );
void printf( string format [mešani argumenti]);
```

Za formatiranje stringova postoji i funkcija nl2br(), koja uzima string kao ulazni parametar, i sve znakove za novi red u njemu zamenjuje odgovarajućim HTML oznakama za novi red
. Kako HTML zanemaruje znakove za novi red u stringu, ako string ne obradimo funkcijom nl2br(), on će biti prikazan u jednom redu web čitača. Prototip ove funkcije je:

```
string nl2br( string a);
```

PHP poseduje i funkcije za pretvaranje malih slova u velika i obrnuto. To su funkcije strtoupper(), strtolower(), ucfirst() i ucwords(). Funkcije strtoupper() i strtolower() se koriste za pretvaranje svih slova znakova u stringu u velika slova, odnosnu za pretvaranje svih slova u stringu u mala slova. Funkcija ucfirst() menja u veliko slovo prvi znak ulaznog argumenta, a funkcija ucwords() menja prvi znak svake reči koja počinje alfabetskim znakom. Prototipovi ovih funkcija su:

```
string strtoupper( string a);
string strtolower( string a);
string ucfirst( string a);
string ucwords( string a);
```

Pored funkcija za vizuelno formatiranje stringova, postoje i funkcije za formatiranje stringova radi unošenja u bazu podataka. Određeni znakovi u stringovima mogu da izazovi probleme prilikom upisa tih stringova u

bazu, jer baza može da ih protumači kao kontrolne znakove. Ti znakovi su navodnici, obrnuta kosa crta i znak NULL. Pre upisa znakovnog podatka u bazu, trebalo bi taj podatak formatirati funkcijom addslashes() koja svaki zanak ‘ zamenjuje znakom \’, a svaki znak \ se zamenjuje znakom \\. Njen prototip je:

```
string addslashes( string a);
```

Funkcija sa suprotnim dejstvo od addslashes() je funkcija stripslashes() koja uklanja kose crte. Njen prototip je:

```
string stripslashes( string a );
```

Ponekad je potrebno analizirati znakovni podatak, naprimer razdvojiti ime domena ili e-pošte na sastavne delove. U tu svrhu može se upotrebiti funkcija explode(). Njen prototip je:

```
array explode( string graničnik, string tekst [, int broj] );
```

Ova funkcija deli string tekst na mestima gde naiđe na string graničnik. Izdvojene delove funkcija vraća u obliku niza. Ukupan broj delove se može zadati neobaveznim argumentom broj.

Suprotne funkcije funkciji explode() su implode() i join(), koje su međusobno identične. Prototip funkcije implode je:

```
string implode( string g,  array delovi);
```

Ova funkcija nadovezuje delove i između njih se umeće string g.

Funkcija za izdvajanje podstringa nekog stringa je substr(). Ona izdvaja deo stringa između zadate početne vrednosti i dužine podstringa. Njen prototip je:

```
stirng substr( string tekst, int početak [, int dužina]);
```

Za utvrđivanje dužine stringa koristi se funkcija strlen():

```
int strlen(string a);
```

Funkcije za pronalaženje stringova unutar stringova su strstr(), strchr(), strrchr(), stristr(). U PHP-u funkcije strstr() i strchr() su identične i služe za pronalaženje znaka ili stringa unutar drugog stringa:

```
string strstr( string tekst, string uzorak );
```

Funkciji se prosleđuje argument tekst u kojem treba pronaći uzorak. Rezultat funkcije je deo vrednosti parametra tekst koji počinje od prvog pojavljivanja parametra uzorak. Ukoliko se uzorak ne pojavljuje u tekstu, rezultat funkcije je false. Funkcija stristr() je skoro identična sa funkcijom strstr() s tim što stristr() ne pravi razliku između velikih i malih slova. Funkcija strrchr() vraća deo izvornog stringa od poslednjeg pojavljivanja uzorka pa nadalje.

4.6 Nizovi

Niz je promenljiva koja sadrži skup vrednosti u nekom redosledu. Jedan niz može imati više elemenata, a svaki element sadrži po jednu vrednost, poput teksta, brojeva ili drugog niza. Niz čiji elementi su drugi nizovi, naziva se višedimenzioni niz. PHP podržava numerički indeksirane i asocijativne nizove. U PHP-u indeksi uvek počinju nulom, ali je umesto nje moguće zadati i neku drugu vrednost.

Da bi se niz inicijalizovao treba upotrebiti jezičku konstrukciju array() na sledeći način:

```
$niz = array( [ključ =>] vrednost, . . .  );
```

gde je ključ vrednost indeksa niza koja može biti tip integer ili string, a vrednost je bilo koja vrednost elementa.

Niz ne mora da se inicijalizuje samo upotrbom array(). Ako se potrebni podaci nalaze u nekom drugom nizu, moguće je kopirati jedan niz u drugi upotrebom operatora = .

Pristupanje elementima niza se vrši pomoću operatora [], u kojem se navodi indeks (ključ), preko kojeg se određuje element niza kojem se pristupa. Na primer.:

```
niz[0]; - za pristupanje nultom elementu niza.
```

PHP poseduje skup funkcija za sortiranje nizova. Da bismo sortirali niz u rastućem redosledu, može se upotrebiti funkcija sort. Njen prototip je:

```
bool sort( array niz[, flags]);
```

Funkcija sort() se može upotrebiti za sortiranje i numeričkih podataka i podataka string. Prvi argument funkcije je niz koji treba sortirati dok je drugi argument flag opcioni i može biti jedna od sledećih konstanti: SORT_REGULEAR, SORT_NUMERIC ili SORT_STRING.

Slične funkcije funkciji sort() su asort() i ksort(). One se koriste kada želimo da vrednost i ključ(indeks) ostanu povezani i posle sortiranja. One se najčešće koriste kada nizovi imaju asocijativne(opisne) indekse. Funkcija asort() se upotrebljava kada je potrebno sortirati niz prema vrednostima svakog elementa, a funkcija ksort() se koristi kada treba sortirati niz prema asocijativnim indeksima.

Funkcije sort(), asort() i ksort() sortiraju nizove u rastućem redosledu. Za svaku od njih postoji i odgovarajuća funkcija koja sortira niz u opadajućem redosledu. To su funkcije rsort(), arsort() i krsort().

Slične funkcijama sort(), asort() i ksort() su funkcije usort(), uasort() i uksort(). One sortiraju niz prema zadatoj funkciji za poređenje. Prototip funkicije usort() je:

```
bool usort( array niz, callback cmp_function);
```

Za navigaciju unutar niza mogu se koristiti funkcije: current() – vraća tekući element niza, next() – pomera pokazivač na elemente niza unapred za jedan element i vraća novi tekući element, each() – vraća tekući element i pomera pokazivač na sledeći element u nizu, reset() – vraća prvi element u nizu, end() – vraća poslednji element u nizu, prev() – pomera pokazivač unatrag za jedno mesto i zatim vraća nov tekući element.

Ako na svaki element nekog niza treba izvršiti neku operaciju, može se upotrbiti funkcija array_walk(). Njen prototip je:

```
bool array_walk( array &array, callback funcname[, userdata]);
```

Prvi argument funkcije array_walk() je niz koji treba obraditi, drugi je ime funkcije koja će biti izvršena na za svaki element niza. Treći argument je opcioni i ako je zadat biće prosleđen funkciji funcname kao parametar.

Da bi se prebrojao broj elemenat u nizu, mogu se upotrbiti funkcije count() i sizeof(). Obe ove funkcije vraćaju ukupan broj elemenata u nizu koji im je prosleđen. Složenija funkcija od ove dve za

prebrojavanje je funkcija array_count_values(). Ova funkcija za zadati niz vraća tabelu učestalosti svih vrednosti elemenata niza. Njen prototip je:

```
array array_count_values ( array input );
```

4.7 Rad sa datotekama

Proces upisivanja, odnosno čitanja podataka u datoteku se sastoji iz tri koraka:

- Otvaranje datoteke
- Upisivanje/čitanje podataka iz datoteke
- Zatvaranja datoteke

Otvaranje datoteke

Za otvaranje datoteke u PHP-u postoji funkcija fopen(). Njen prototip je:

```
resource fopen (string filename, string mode [, bool use_include_path [,
resource context]] );
```

Prvi parametar funkcije fopen() treba da bude putanja datoteke koju je potrebno otvoriti. Ako je putanja u obliku „schema://…“ onda PHP smatra da je putanja ka datoteci ustvari njena URL adresa. Ukoliko je u pitanju putanja oblika ftp://... , biće otvorena FTP veza sa navedenim serverom u pasivnom režimu, a pokazivač će biti vraćen na početak datoteke. U slučaju da ime putanja počinje sa http://..., biće otvorena HTTP veza sa navedenim serverom, a pokazivač će biti postavljen na tekst odgovora sa servera.

Drugi parametar funkcije fopen() je režim u kojem se datoteka otvara. To je znakovna vrednost kojom se određuje šta treba raditi sa datotekom. Režimi otvaranja su nabrojani u sledećoj tabeli:

Režim	Ime	Opis
r	Read	Otvara datoteku za čitanje, od početka.
r+	Read	Otvara datoteku za čitanje i pisanje, od početka.
w	Write	Otvara datoteku za pisanje od početka. Ukoliko datoteka postoji, postojeći sadržaj se briše. Ako datoteka ne postoji, sistem pokušava da je napravi.
w+	Write	Otvara datoteku za pisanje i čitanje od početka. Ukoliko datoteka postoji, postojeći sadržaj se briše. Ako datoteka ne postoji, sistem pokušava da je napravi.
x	Cautious write	Otvara datoteku za pianje, od početka. Ukoliko datoteka ne postoji, sistem je ne otvara, fopen() vraća vrednost false, a PHP generiše upozorenje.
x+	Cautious write	Otvara datoteku za pianje i čitanje, od početka. Ukoliko datoteka ne postoji, sistem je ne otvara, fopen() vraća vrednost false, a PHP generiše upozorenje.
a	Append	Otvara datoteku za pisanje od kraja postojećeg koda. Ako datoteka ne postoji sistem pokušava da je napravi.
a+	Append	Otvara datoteku za pisanje i čitanje od kraja postojećeg koda. Ako datoteka ne postoji sistem pokušava da je napravi.
b	Binary	Koristi se u kombinaciji sa nekim od ostalih režima, i upotrbljava se za otvaranje binarnog fajla.
t	Text	Koristi se u kombinaciji sa nekim od ostalih režima, i upotrbljava se za otvaranje tekstualnog fajla.

Pisanje u datoteku

Za piasanje u datoteku mogu se koristiti funkcije fwrite() ili fputs(). Funkcija fputs() je drugo ime za funkciju fwrite(). Prototip funkcije fwrite() je:

```
int fwrite ( resource handle, string string [, int length] );
```

Prvi parametar je pokazivač ka fajlu u koji se vrši upisivanje, drugi parametar je string koji se upisuje a treći parametar je opcioni i ako je zadat, predstavlja broj bajtova koje će fwrite() upisati.

Čitanje iz datoteke

Čitanje iz datoteke se može obaviti na više načina. Može se čitati red po red i to pomoću funkcija fgets(), fgetss(), fgetcsv().

Funkcija fgets() čita red teksta sve dok ne naiđe na znak za novi red ili za kraj datoteke ili ne pročita 998 bajtova. Njen prototip je:

```
string fgets ( resource handle [, int length] );
```

Prvi parametar je pokazivač na fajl iz kojeg se čita , a drugi parametar je opcioni i predstavlja maksimalnu dužinu reda.

Funkcija fgetss() je varijacija funkcije fgets() sa sledećim prototipom:

```
string fgetss( resource handle [, int length [, string allowable_tags]] );
```

Ona, kao i fgets, čita jedan red teksta, s tim što iz teksta uklanja sve PHP i HTML oznake. Ako je potrebno zadržati neku oznaku, onda se ona zadaje u argumentu allowable_tags.

Funkcija fgetcsv() rastavlja na pojedinačna polja redove koje čita iz datoteke i to čini na mestima gde naiđe na zadati graničnik. Njen prototip je:

```
array fgetcsv( resource handle [, int length [, string delimiter [, string
enclosure]]] );
```

Parametar enclosure predstavlja znak kojim počinje i završava svako polje u redu. Ako se ništa ne navede podrazumeva se znak “.

Datoteka se može pročitati i cela pomoću funkcija readfile(), file() i fpassthru().
Funkcije readfile() i file() prvo otvaraju datoteku, zatim čitaju njen sadržaj i na kraju zatvaraju datoteku. Jedina razlika između ove dve funkcije je što readfile() prosleđuje sadržaj datoteke na standardni izlaz (Web čitač), a file() datoteku učitava u niz. Funkcija fpassthru(), se koristi tako što se prvo otvori datoteka pa se zatim pokazivač na tu datoteku prosledi funkciji fpassthru() koja čita sadržaj otvorene datoteke od mesta na kom je pokazivač i prosleđuje ga na standardni izlazni tok. Po pročitanom sadržaju datoteke fpassthru() zatvara fajl. Prototipovi ovih funkcija su:

```
int readfile ( string filename [, bool use_include_path [, resource
context]] );

array file ( string filename [, int flags [, resource context]] );

int fpassthru ( resource handle );
```

Za čitanje datoteke znak po znak koristi se funkcija fgetc(). Parametar ove funkcije je pokazivač na otvoreni fajl, a njena povratna vrednost je znak koji je pročitan iz datoteke.

Zatvaranje datoteke

Za zatvaranje datoteke koristi se funkcija fclose(). Njen prototip je:

```
bool fclose ( resource handle );
```

Njen parametar je pokazivač ka otvrenoj datoteci, a povratna vrednost je true ako je datoteka uspešno zatvorena odnosno false ako zatvaranje nije uspelo.

Druge korisne funkcije za rad sa datotekama su feof() – proverava da li se stiglo do kraja datoteke, file_exists() – proverava da li neka datoteka postoji, filesize() – vraća veličinu datoteke u bajtovima, unlink() – briše datoteku sa sistema.

Za kretanje po datoteci koriste se funkcije rewind() – pomera pokazivač na početak datoteke, ftell() – daje tekući položaj pokazivača u datoteci. Funkcija fseek() pomera pokazivač za zadati broj bajtova u odnosu na referentni položaj. Njen prototip je:

```
int fseek ( resource handle, int offset [, int whence] );
```

Moguće vrednosti referentnog položaja (whence) su SEEK_SET – početak datoteke, SEEK_CUR – tekući položaj pokazivača i SEEK_END kraj datoteke.

4.8 Objektno orijentisani PHP

Iako PHP 4 donekle podržava objektno orijentisan kocept, ovaj koncept je dosledno realizovan tek u PHP 5. To je urađeno u cilju poboljšanja preformansi i omogućavanja više osobina objektnog programiranja.

Svaka definicija klase započije ključnom reči *class* iza koje sledi ime klase. Unutar definicije klase se navode atributi i metode klase. Atributi se definišu tako što se unutar definicije klase deklariše promenljiva ispred koje stoji rezervisana reč *var*. Metode se deklarišu kao funkcije unutar definicije klase.

Većina klasa ima specijalnu metodu koja se zove *konstruktor*. Konstruktor se poziva prilikom pravljena objekta date klase i obično obavlja korisne početne poslove kao što je dodeljivanje početnih vrednosti atributima klase. Konstruktor se deklariše kao i druge metode klase, ali on ima specijalno ime *construct*. To je novina koju uvodi PHP 5. U predhodnim verzijama konstruktor je imao isto ime kao i klasa. Kompatibilnost sa predhodnim verzijama čuva se na sledeći način: ako PHP ne nađe unutar klase funkciju koja se zove *construct*, potražiće funkciju koja ima isto ime kao i klasa.

Metoda *destruktor*, je takođe novina koju uvodi PHP 5. Ona ima suprotno dejstvo od metode *constructor*. Ona omogućava da se izvrše neke akcije pre uništavanja objekta.

Instanciranje objekata klase se vrši pomoću rezervisane reči *new*. To se radi tako što se posle reči *new* navodi ime klase i zadaju parametri konstruktora.

Unutar klase postoji specijalni pokazivač *$this*, koji upućuje na tekući objekat. Ako tekući objekat ima atribut koji se zove $atribt, njemu se, unutar metoda klase, može pristupiti pomoću konstrukcije $this->atribut.

PHP 5 uvodi i modifikatore pristupa (*eng. Access modifiers*), koji određuju vidljivost atributa i metoda klase. Oni se zadaju na početku deklaracije atributa ili metoda. PHP podržava sledeće modifikatore pristupa:

- public – objekat klase je javni, tj. njemu mogu pristupati i kôd unutar i kôd izvan klase. Ovo je podrazumevana opcija.
- private – objekat klase je privatni, tj. njemu može pristupati samo kôd koji se nalazi unutar klase. Elementi klase označeni kao privati se ne mogu nasleđivati

- protected – objekat klase je zaštićen, tj. njemu može pristupati samo kôd unutar klase, ali se taj element nasleđuje u svim potklasama.

PHP podržava nasleđivanje osobina klasa. Rezervisana reč *extends* označava da je klasa potklasa druge klase. Višestruko nasleđivanje nije podržano u PHP-u, tj. klasa može da nasleđuje samo osobine jednog direktnog pretka.

Ponekad je potrebno u novoj klasi redefinisati iste atribute i klase kao u natklasi. To se može uraditi da bi se nekom atributu u potklasi dala drugačija podrazumevana vrednost u odnosu na isti atribut u natklasi, ili da bi se promenila funkcionalnost metoda u potkalsi. Ova pojava se naziva *overriding* i podržana je u PHP-u. Pomoću rezervisane reči *final* može se sprečiti nasleđivanje i overriding metoda natklase.

Još jedna od novina koja je uvedena u PHP 5 jesu interfejsi. Jedna od njihovih upotreba jeste zaobilaženje ograničenja jednostrukog nasleđivanja. Suština interfejsa je u tome da se deklariše određen broj funkcija koje moraju biti realizovane u svim klasama koje realizuju taj interfejs. Interfejs se definiše slično kao klasa, sem što se umesto rezervisane reči *class* upotrbljava rezervisana reč *interface*. Da bi se označilo da neka klasa implementira interfejs, koristi se ključna reč *implements.*

4.9 Obrada izuzetaka

Izuzeci su nova i važna mogućnost koju uvodi PHP 5. Oni predstavljaju jednobrazan mehanizam obrade grešaka pomoću objektno orjentisanog koda koji se lako proširuje i održava.

Osnovna ideja na kojoj se zasniva obrada izuzetaka jeste da se kod izvršava unutar *try* bloka. Ako unutar *try* bloka nešto krene naopako, može se preuzeti akcija generisanja izuzetka. Generisanje izuzetaka se u PHP izvršava na sledeći način:

```
throw  new Exception('message', error_code);
```

Iza svakog *try* bloka mora da sledi bar jedan *catch* blok. Zadavanje više *catch* blokaova ima smisla ako svaki blok catch treba da presretne drugu vrstu izuzetka. Objekat koji se prosleđuje bloku *catch* jeste objekat prosleđen iskazu *throw* koji je generisao izuzetak. Izuzetak može biti bilo kog tipa, ali je korisno da to bude instanca klase *Exception* ili instanca klase koja nasleđuje klasu Exception.

Klasa *Exception* je ugrađena u PHP 5. Konstruktor te klase prihvata dva parametra. To su tekst poruke i kod(broj) greške. Osim konstruktora klasa Exception stavlja na raspolaganje i sledeće metode:

- getCode() – vraća kod greške koji je bio prosleđen konstruktoru.
- getMessage() – vraća tekst poruke koja je bila prosleđena konstruktoru.
- getFile() – pozivajućem kodu vraća punu putanju datoteke u kojoj je generisan izuzetak.
- getTrace() – vraća niz sa podacima o stablu pozivanja, koji omogućava utvrđivanje mesta na kome je generisan izuzetak.
- getTraceAsString() – vraća iste podatke kao i getTrace(), ali formatirane kao znakovne podatke.
- _toString() – omogućava da se iskazu echo direktno prosledi ceo sadržaj objekta klase Exception, sa svim podacima koje daju navedene metode.

4.10 Pristup MySQL bazi podataka pomoću PHP-a

PHP 5 ima novu biblioteku funkcija za uspostavljanja veze s bazama podataka, koja se zove *mysqli*. Ova biblioteka podržava upotrebu i objektno orijentisane i proceduralne sintakse.

Povezivanje sa MySQL serverom obavlja se na sledeći način:

```
$db = new mysqli('host', 'username', 'pasword');
```

Predhodni red koda pravi instancu klase *mysqli* i uspostavlja vezu sa serverom ‘host’ pod korisničkim nalogom ‘username’ i sa lozinkom ‘pasword’. Ovaj objektno orijentisani oblik sintakse omogućava da se sa bazom podataka obavlja komunikacija tako što se pozivaju metode objekta $db. Da bi se na proceduralni način uspostavila veza sa bazom, treba upotrebiti sledeću funkciju:

```
mysqli mysqli_connect (‘host’, ‘username’, ‘pasword’ );
```

Umesto objekta, ova funkcija vraća rezultat tipa resurs, koji predstavlja vezu sa bazom podataka.

Većina funkcija iz mysqli biblioteke ima i objektno orijentisani interfejs i proceduralni interfejs. Razlikuju se uglavnom po tome što imena iz proceduralne verzije započinju s *mysqli_* i tim funkcijama se mora proslediti resurs koji je dobijen pozivanjem funkcije mysqli_connect().

Kao parametar konstruktora mysqli ili funkcije mysqli_connect(), potrebno je navesti i sa kojom bazom se uspostavlja veza. Ako je potrebno da se iz tekuće baze podataka pređe u drugu bazu podataka, to se može uraditi na jedan od sledeća dva načina:

```
$db->select_db(name);
```

ili

```
mysqli_select_db($db, name);
```

Da bi se izvršio upit nad bazom podataka, potrebno je upotrebiti metodu:

```
$result  =  $db->query($query);
```

u objektnom interfejsu, ili :

```
$result = mysqli_query($db, $query);
```

u proceduralnom interfejsu. U oba slučaja parametar $query predstavlja tekst SQL upita koji treba izvršiti.

Objektno orijentisana verzija funkcije vraća objekat koji omogućava čitanje rezultata, a za istu namenu proceduralna verzija vraća identifikator tipa resurs.

Za prebrojavanje redova koje upit vraća, u proceduralnom interfejsu, koristi se funkcija *mysqli_num_rows()*. Kada se koristi objektni interfejs, broj redova koje je upit vratio smešta se u svojstvo *num_rows* objekta $result, kojem se može pristupiti na sledeći način:

```
num_rows  = $result->num_rows;
```

Funkcijama $result->fetch_assoc() ili mysqli_fetch_assoc() se čita svaki red iz skupa rezultata i vraćaju podaci u obliku niza, u kojem je svaki ključ ime jednog atributa, a svaka vrednost je vrednost učitana iz baze.

Upotrebom funkcije fetch_row(), odnosno mysqli_fetch_row(), takođe se dobija rezultat u obliku niza, samo što se sada umesto ključeva, koji predstavljaju ime atributa u bazi podataka, dobijaju numerički ključevi.

Skup rezultata se može osloboditi ako se pozove metoda *free()* ili funkcija *mysqli_free_result()*. Kada se oslobodi skup rezultata, može se upotrebiti metoda *close()* ili funkcija *mysqli_close()* da bi se prekinula veza s bazom podataka. Izričito pozivanje ovog metoda, odnosno funkcije nije obavezno jer će se veza sa bazom podataka prekinuti po izvršenju skripte.

5 Opis implementacije prodavnice „pametnih telefona”

Izrada aplikacije koja predstavlja Internet prodavnicu pametnih telefona može se podeliti u tri celine:

- Instalacija razvojnog okruženja
- Izrada baze podataka koja sadrži podatke vezane za prodavnicu mobilnih telefona
- Izrada kataloga pametnih telefona
- Izradu korpe za kupovinu

5.1 Instalacija razvojnog okrženja

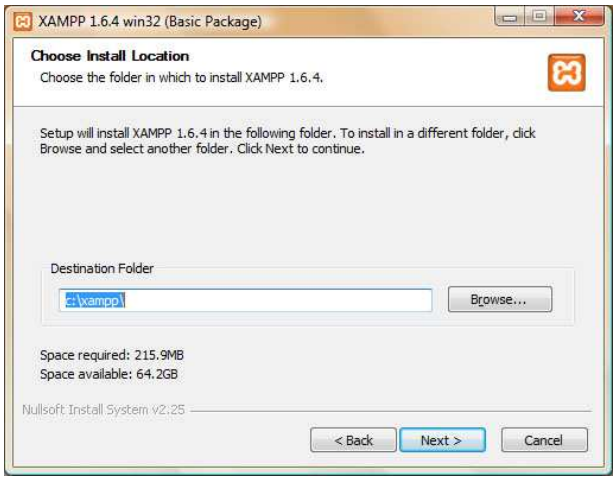
Prvi korak koji treba napraviti ka izradi prodavnice pametnih telefona jeste instalacija HTTP servera, PHP-a i MySQL-a. Dva najpoznatija HTTP servera koji podržavaju PHP su Microsoftov IIS (*Internet Information Services*) i projekat otvorenog koda Apache koji postoji u verzijama kako za Unix-olike tako i za Windows operativne sisteme. Za izradu ove aplikacije koristiće se Apache web server koji se može skinuti sa internet adrese www.**apache**.org.

Najlakši način da se instaliraju Apache, PHP i MySQL jeste upotrebom aplikacije XAMPP. XAMPP je distribucija Apache-a koja je veoma jednostavna za instalaciju i sadrži MySQL, PHP i Perl. Postoje različite distribucije XAMPP-a. One se razlikuju po tome za koji su operativni sistem namenjene. Trenutno su podržani Linux, Windows od verzije 98 pa sve do Viste, Mac OS kao i Slolaris 8 i 9. XAMPP se može preuzeti sa web stranice http://www.apachefriends.org/en/xampp.html.

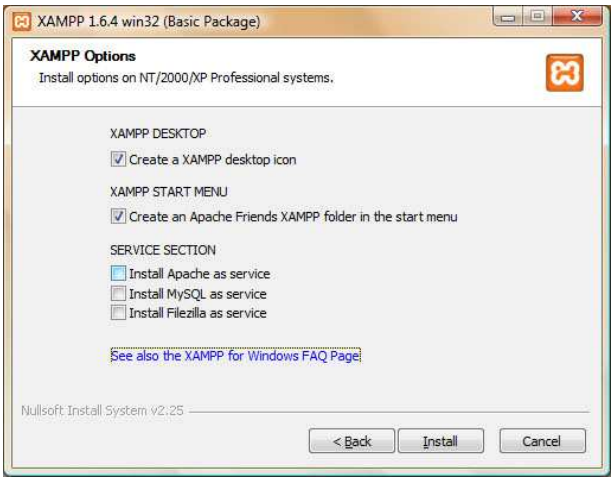
U verziju XAMPP-a za Windows su uključeni:

- HTTP server Apache
- PHP verzije 4 i 5
- FTP server otvorenog koda FileZilla
- E-mail server Mercury Mail Transport System

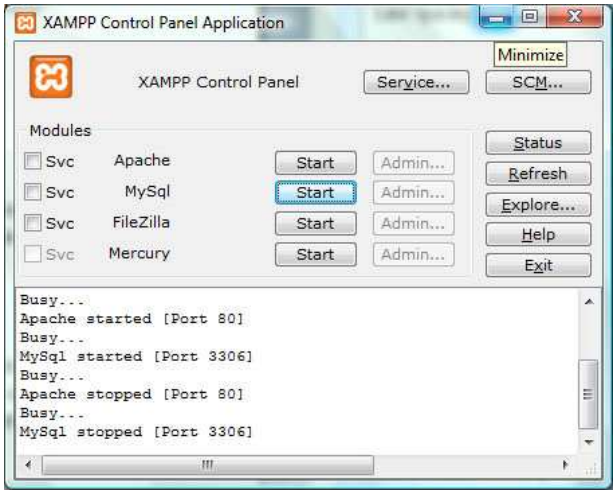
Sam proces instalacije XAMPP-a na Windows operativni sistem je krajnje jednostavan. Po dovlačenju XAMPP-a sa odgovarajuće internet stranice potrebno je pokrenuti program xampp-win32-installer.exe, nakon cega će početi proces instalacije.



Kliktanjem na dugme „Next“ otvara se przor u kom je moguće izabrati lokaciju na lokalnom računaru gde će XAMPP biti instaliran. Na sledećem prozoru se može izabrati da li će Apache, MySQL i FileZilla biti instalirani kao Windows servisi.



Odabirom dugmeta „Install“ proces instalacije se započinje. Po završetku aplikacije, da bi se Apache i MySQL serveri koristili, potrebo ih je pokrenuti. Veoma zgodan način da se to uradi jeste uz pomoć aplikacije xampp-control.exe koja se nalazi u C:\xampp (lokacija gde je xampp instaliran).



Sa slike se može videti da je pokretanje i zaustavljanje servera instaliranih sa XAMPP-om veoma intuitivno i praktično.

Da bi neki dokument bio vidljiv u web čitaču potrebno ga je smestiti u datoteku htdocs koja se nalazi u instalacionom dirktorijumu XAMPP-a i zatim u web čitaču otkucati adresu: http://127.0.0.1/<ime.dokumenta>. Tako je za potrebe „prodavnice pametnih telefona“ kreiran direktorijum „smartphone.com“ u kojem će biti smeštene sve PHP skripte potrebne za funkcionisanje aplikacije. To znači da se početnoj stranici „prodavnice pametnih telefona“ može pristupiti (sa lokalnog računara) na adresi http://127.0.0.1/smartphone.com/index.php.

5.2 Izrada baze podataka

Osnovni sloj web aplikacije je baza podataka u kojoj će se nalaziti svi podaci vezani za „prodavnicu pametnih telefona.

Da bi se kreirala baza podataka, prvo se treba prijaviti na MySQL server. To se može uraditi tako što se iz komandne linije pokrene MySQL monitor izdavanjem nardbe *mysql* sa komandne linije, na sledeći način:

```
mysql –h host –u username –p
```

Opcijom –h se zadaje ime računara na kojem se nalazi MySQL server sa kojim želimo da se povežemo. Opcijom –u zadaje se korisničko ime pod kojim se prijavljujemo na MySQL server. Opcjija –p obaveštava server da će biti unešena lozinka. Ako korisnički nalog nema lozinku, opcija –p može da se izostavi.

Sada je potrebno kreirati bazu podataka koja će biti korišćena za smeštanje podataka vezanih za prodavnicu mobilnih telefona. Ako je ime te baze „smartPhoneShop“, to se može uraditi izdavanjem sledeće komande u MySQL monitoru:

```
CREATE DATABASE smartPhoneShop;
```

Sledeći korak u izradi baze podataka jeste kreiranje tabela. Prilikom projektovanja baze podataka treba voditi računa da sve što se nalazi u bazi podataka zauzima što manje mesta. To se može obezbediti pomoću projektantskih rešenja koja minimizuju redundantnost sadržaja i tako što će se koristiti najmanji mogući tipovi podataka za kolone. Kompletan kod za izradu tabela baze podataka može se pogledati u fajlu „smartPhoneDBTables.sql“ a ovde ce kao primer biti prikazana naredba za kreiranje tabele „phones“ koja će da sadrži spisak svih telefona:

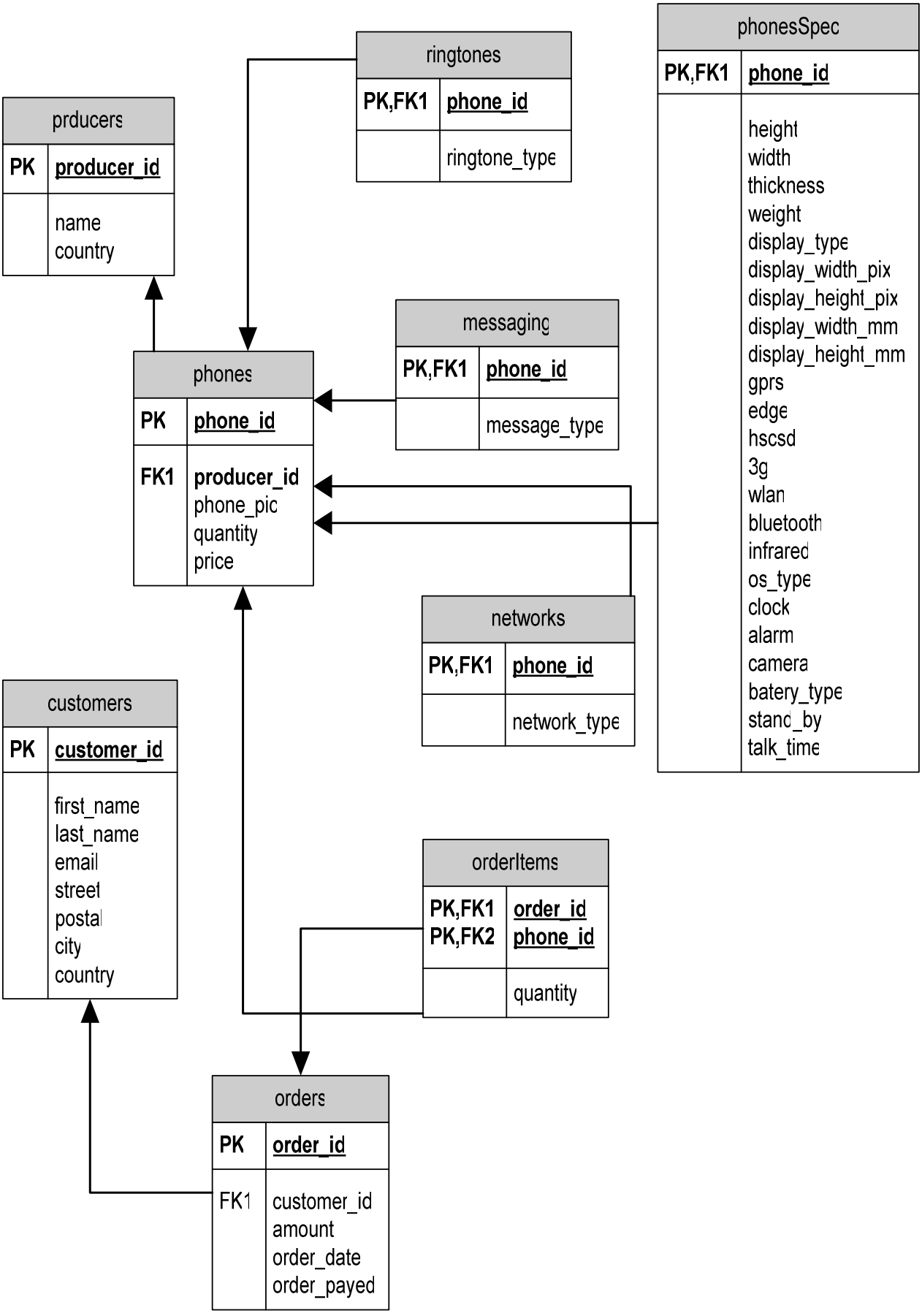
```
CREATE TABLE phones (
      phone_id varchar(20) PRIMARY KEY NOT NULL,
      producer_id SMALLINT,
      phone_pic   varchar(256),
      quantyty SMALLINT,
      price float(7,2) );
```

Interesantna kolona u ovoj tabeli je kolona „phone_pic“. Ova kolona sadrži putanju u fajl sistemu do slike telefona. Iako MySQL podržava skladištenje podataka multimedijskog tipa, najčešće je bolje rešenje da se prosleđene slike čuvaju u sistemu datoteka. Upotreba podataka tipa BLOB u MySQL-ovoj bazi podataka može da pogorša preformanse kompletne aplkacije.

Kako se sve komande za kreiranje nalaze u datoteci „smartPhoneDBTables.sql“, sve tabele „smartPhoneShop“ baze podataka će se najlakše kreirati „upotrebom SQL komandi iz spoljnjih fajlova“ na sledeći način:

```
mysql –u username –p password </path/to/smartPhoneDBTables.sql
```

Sledećom šemom su predstavljene sve tabele i veze između njih potrebne za funkcionisanje internet prodavnice „pametnih telefona“:


prducers
PK producer_id
name
country
ringtones
PK,FK1 phone_id
ringtone_type
phonesSpec
PK,FK1 phone_id
height
width
thickness
weight
display_type
display_width_pix
display_height_pix
display_width_mm
display_height_mm
gprs
edge
hscsd
3g
wlan
bluetooth
infrared
os_type
clock
alarm
camera
batery_type
stand_by
talk_time
messaging
PK,FK1 phone_id
message_type
phones
PK phone_id
FK1 producer_id
phone_pic
quantity
price
networks
PK,FK1 phone_id
network_type
customers
PK customer_id
first_name
last_name
email
street
postal
city
country
orderItems
PK,FK1 order_id
PK,FK2 phone_id
quantity
orders
PK order_id
FK1 customer_id
amount
order_date
order_payed

Radi bezbednosti podataka u bazi podataka preporučljivo je definisati barem tri korisnika te baze. Jedan korisnik baze bi trebalo da bude administrator same te baze podataka, znači neko ko poznaje MySQL i ko će raditi na održavanju baze podataka. Takav korisnik bi trebao da ima sva ovlaščenja nad tom bazom podataka. Drugi korisnik baze podataka je „vlasnik“ internet prodavnice, koji pored ovlašćenja za čitanje redova baze podataka, treba da ima i ovlaščenja za unošenje, brisanje i menjanje redova u tabeli, kako bi bio u mogućnosti da dodaje novi sadržaj svojoj web prodavnici. Treči korisnik bi bio potencijalni kupac koji bi trebalo da ima ovlašćenja za pregledanje, unošenje i menjanje redova u svim tabelam baze podataka koje su vezane za kupca, tj. Kupac ne bi smeo da ima ovlaščenje da menja ili unosi podatke u tabele koje čuvaju podatke o telefonima koji se prodaju, ali su svakako takvom korisniku potrebna ovlašćenja za unošenje podataka u tabele koje skladište porudbine i podatke o kupcima.

Ova tri korisnika se mogu kreirati izdavanjem odgovarajućih GRANT naredbi. Takva naredba za administratora baze podataka bi izgledala ovako:

```
GRANT ALL ON smartPhoneShop TO admin IDENTIFIED BY ‘adm123’;
```

5.3 Izrada kataloga pametnih telefona

Izrada kataloga pametnih telefona može da se podeli u dve faze. Prva faza bi bila kreiranje PHP skripti koje „vlasniku“ prodavnice omogućavaju da unosi nove i ažurira postojeće telefone i podatke vezane za njih. Druga faza bi bila izrada skripti koje omogućavaju „kupcu“ telefona da pregleda katalog pametnih telefona.

Skripte koje se odnose na ažuriranje sadržaja prodavnice ne bi smele da budu dostupne običnim korisnicima, tj. pristup stranama (skriptama) za ažuriranje podataka bi trebalo da bude zaštičen proverom identiteta korisnika. PHP skriptovi obično ne zavise od platforme, ali se u osnovnoj proveri identiteta koriste vrednosti promenljivih okruženja na serveru. Da bi skript mogao da proveri identitet pomoću HTTP protokola na Apache web serveru, na kojem radi PHP, ili na IIS web serveru gde PHP radi kao ISAPI modul, skripta mora da prepozna tip servera. Sledeća skripta može da radi na oba ova servera:

```
//provera identiteta
//ako se radi o IIS, potrebno je setovati $PHP_AUTH_USER I $PHP_AUTH_PW
      if (substr($SERVER_SOFTWARE, 0, 9) == 'Microsoft' &&
            !isset($_SERVER['PHP_AUTH_USER']) &&
            !isset($_SERVER['PHP_AUTH_PW']) &&
            substr($HTTP_AUTHORIZATION, 0, 6) == 'Basic ') {

                  list($PHP_AUTH_USER, $PHP_AUTH_PW) =
            explode(':', base64_decode(substr($HTTP_AUTHORIZATION, 6)));
      }
      if ($_SERVER['PHP_AUTH_USER']!=$user || $_SERVER['PHP_AUTH_PW']!=$psw) {
            //pogrešna lozinka
            header('Authenticate: Basic realm="Smart Phone Shop"');
            if (substr($SERVER_SOFTWARE, 0, 9) == 'Microsoft')
                  header('Status: 401 Unauthorized');
            else
                  header('HTTP/1.0 401 Unauthorized');

            echo '<h1>Go Away!</h1>';
            echo 'You are not authorized to view this resource.';
      }
      else {
            //provera je uspela, prikaži stranicu
      }
```

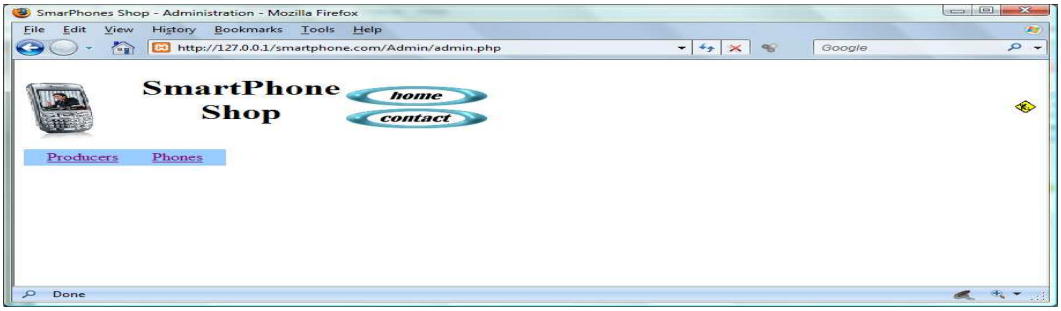
PHP skripte koje se koriste za unošenje i ažuriranje podataka se oslanjaju na metode iz klasa Page, PageAdmin, Phone i Producer. Klase Producer i Phone sadrže metode koje služe za upravljanje sadržajima u odgovarajućim tabelama baze podataka. Na primer, metodama iz klase Producer mogu se dodavati novi proizvođači, vršiti izmene na podacima postojećih proizvođaća ili se mogu pročitati svi proizvođaći iz baze podataka:

```
<?php
class Producer {
    var $name;
    var $country;
    var $db;

    function __construct($pdb) {
        if(!$pdb) {
           echo 'There is no db handler. </br>'
                .'Please go back and try agin';
            exit;
        }

        $this->db = $pdb;
    }

    function setNameCountry($pName, $pCountry){
        if(!$pName) {
            echo 'You have not entered name for this producer. </br>'
                .'Please go back and try agin';
            exit;
        }

        if(!$pCountry) {
            echo 'You have not entered country for this producer. </br>'
                .'Please go back and try agin';
            exit;
        }

        $this->name = $pName;
        $this->country = $pCountry;
    }

    function setToDB() { //update == 0 for new item
        //formate data for data base
        if(!get_magic_quotes_gpc()) {
            $this->name = addslashes($this->name);
            $this->country = addslashes($this->country);
        }

        //db query
            $query = "insert into smartPhoneShop.producers values(null,
'".$this->name."', '".$this->country."');";

        $result = $this->db->query($query);

        return $result;
    }

    function updateInDB($id) {
        //formate data for data base
        if(!get_magic_quotes_gpc()) {
            $this->name = addslashes($this->name);
```

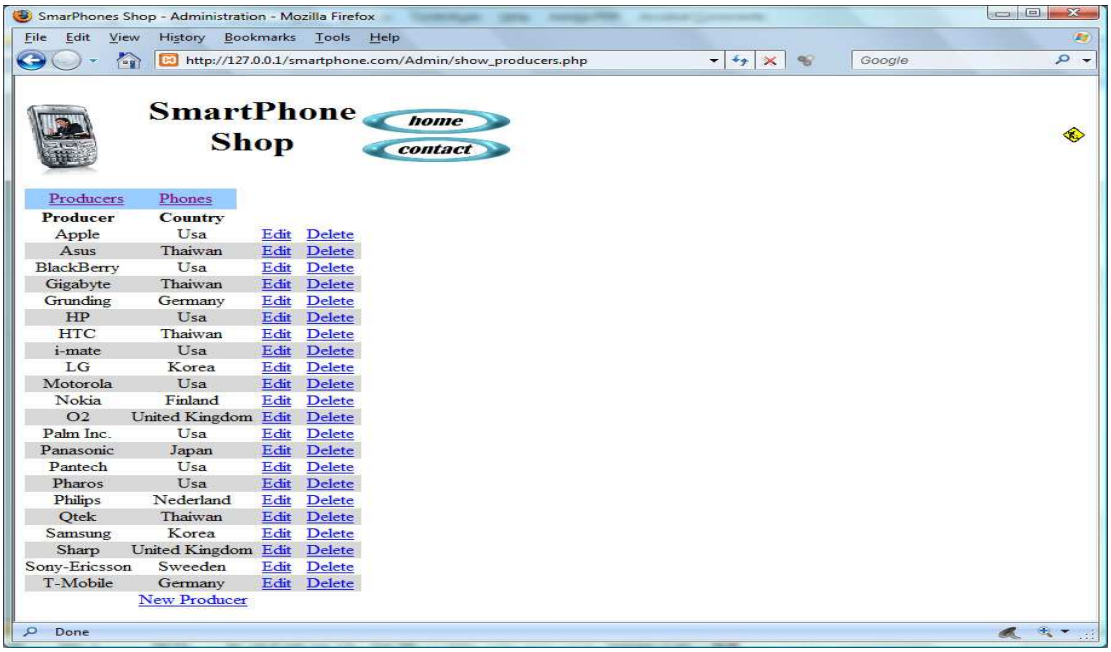
```
            $this->country = addslashes($this->country);
        }

$query = "update smartPhoneShop.producers set name = '".$this->name."', country
= '".$this->country."' where producer_id = $id;";
        $result = $this->db->query($query);

        return $result;
    }

    //return all producers from DB
    function getFromDB() {
        $query = "select * from smartPhoneShop.producers order by name";
        $result = $this->db->query($query);
        if(!$result) {
            exit;
        }

        return $result;
    }

    function getFromDBByID($id) {
        $query = "select * from smartPhoneShop.producers where producer_id =
$id";
        $result = $this->db->query($query);
        if(!$result) {
            exit;
        }

        return $result;
    }

    function deleteFromDBByID($id) {
        $query = "delete from smartPhoneShop.producers where producer_id = $id";
        $result = $this->db->query($query);
        if(!$result) {
            exit;
        }

        return $result;
    }

}
?>
```

Klase Page i PageAdmin sadrže metode za prikazivanje sadržaja na web stranicama. Metode klase Page se koriste za prikazianje sadržaja i na stranicama kojima može slobodno da se pristpa (bez provere identiteta), dok je klasa PageAdmin potklasa klase Page i, pored nasleđenih metoda, poseduje i metode koje su karakteristične samo za prikazivanje sadržaja na stranicama za izmenu sadržaja u bazi podataka.

Da bi ažurirao podatke u prodavnici pametnih telefona, „vlasnik“ prodavnice mora da ode na administrativni deo web aplikacije na adresi http://127.0.0.1/smartphone.com/Admin/admin.php.



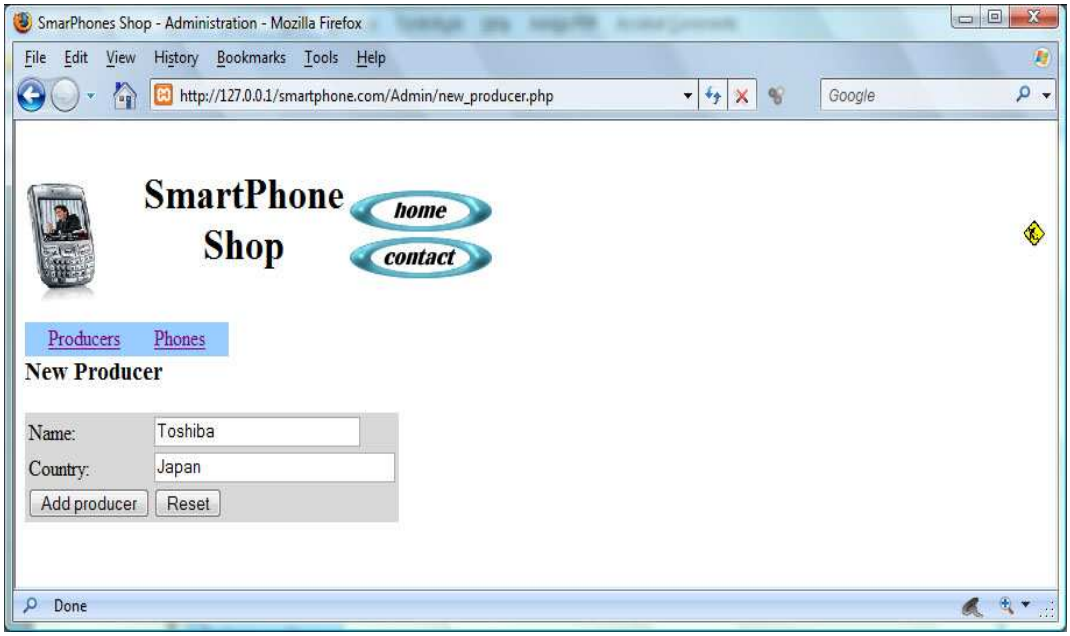
Na ovoj stranici mogu se ažurirati podaci vezani za proizvođaće (Producers) mobilnih telefona i podatke vezane za same mobilne telefone (Phones). Klikom na link „Producers“ otvara se stranica na kojoj će biti prikazana lista svi proizvođaća koji se nalaze u bazi podataka.



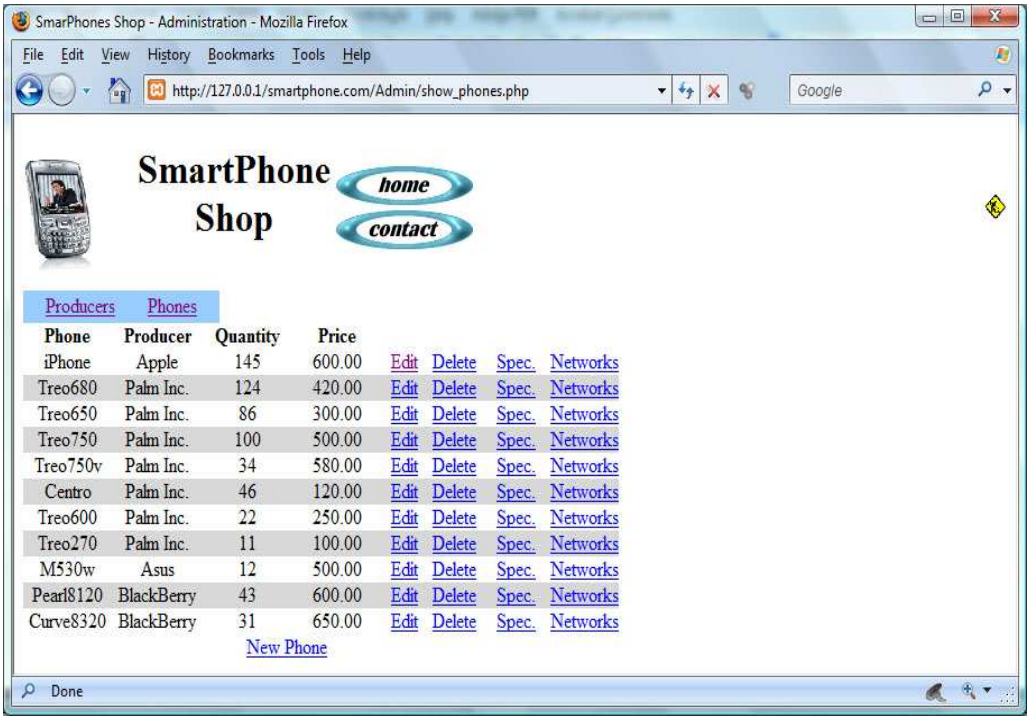
U svakom redu liste se nalaze naziv proizvođača, zemlja iz koje je proizvođač i linkovi ka skriptama za editovanje i brisanje proizvođača iz baze podataka. Odabirom linka „Edit“, pored imena nekog proizvođača, otvara se stranica na kojoj se mogu izmeniti podatci vezani za tog proizvođača.

Nakon što su podaci o proizvodžaču izmenjeni, izmene se mogu „snimiti“ u bazi klikom na dugme „Change“.

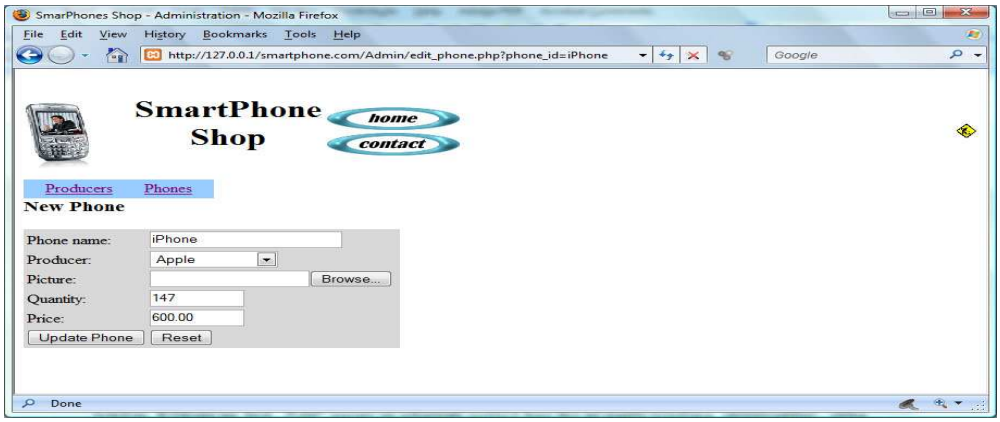
Na dnu liste svih proizvodžača se nalazi link „New Producer“ koji se može iskoristiti za dodavanje novog proizvodžaća mobilnih telefona u bazu.



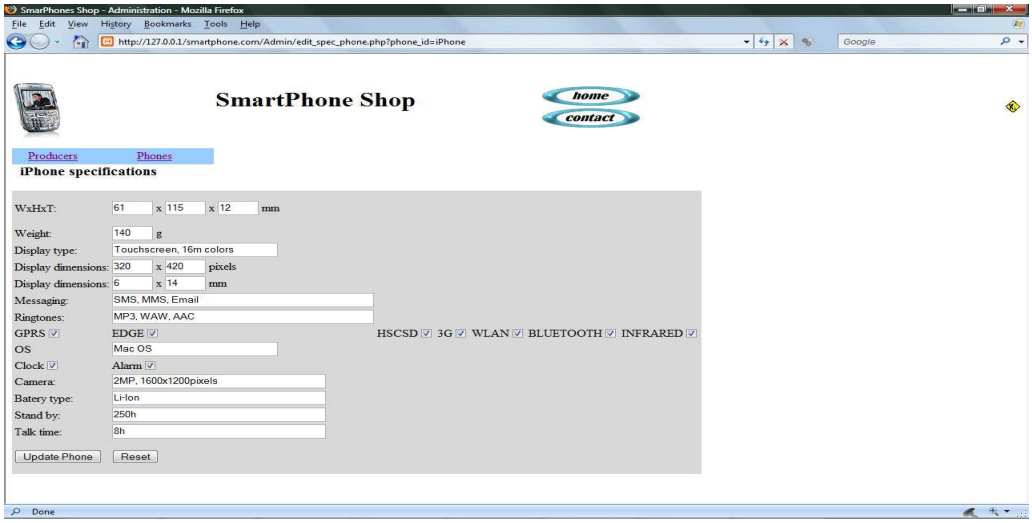
Pored ažuriranja podataka vezanih za proizvodžače, na stranici „admin.php“ može se izabrati i link „Phones“ ako je potrebno ažurirati podatke vezane za telefone. Odabirom linka „Phones“ otvara se strana sa listom svih mobilnih telefona koji se nalaze u bazi podataka.



Slično kao i u listi svih proizvodžača i ovde se nalaze linkovi za ažuriranje podataka vezanih za određeni telefon. Link „Delete“ služi za brisanje telefona iz baze podataka. Klikom na link „Edit“ mogu se ažurirati podaci kao što su naziv telefona, proizvodžač , slika telefona, količina telefona u prodavnici kao i njegova cena.



Link „Spec.“ služi za otvaranje stranice na kojoj je moguće ažuriranje specifikacija samog telefona.

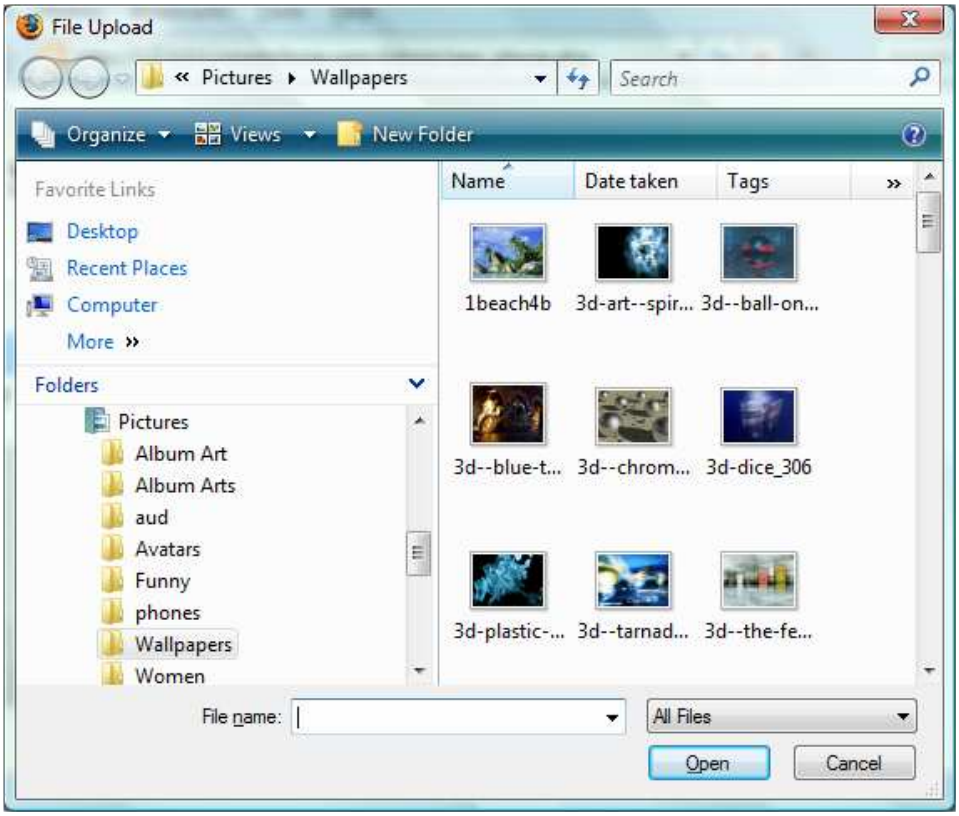


Kao što se na slici može videti, na ovoj stranici je moguće ažurirati pdatke kao što su: dimenzije telefona, njegova težina, operativni sistem na kom telefon radi i ostali podaci karakteristični za taj mobilni telefon. Klikom na dugme „Update Phone“ u bazu se pohranjuju svi ovi podaci.

Klikom na link „Networks“ mogu se izmeniti podati o mrežama na kojim funkcioniše taj telefon.

Na dnu liste svih telefona, kao i kod liste svih proizvodžača, nalazi se link „New Phone“ preko kojeg se dodaje novi telefon u bazu podataka.

Klikom na link „New Phone“ otvara se stranica na kojoj je moguće Uneti podatke o novom telefonu. Za sliku telefona potrebno je u odgovarajuće polje uneti putanju do sike na lokalnom računaru. To je najjednostavnije uraditi pritiskom na dugme „Browse...“ posle čega se otvara prozor „Upload File“ u kojem je moguće pronaći sliku na lokalnom računaru.



Kada je odgovarajuća slika pronađena na lokalnom računaru potrebno je pritisnuti dugme „Open“ i putanja do te slike na lokalnom računaru će biti ispisana u odkovarajućem tekst polju. Pritiskom na dugme „Add

Phone“ na stranici za dodavanje novog telefona u bazu podataka, ti podaci se zaista ubacuju u bazu podataka. Fajl koji je dodat kao slika za taj telefon, je potrebno prvo prebaciti (*upload*) na server i zatim smestiti u odgovarajući direktorijum na serveru. Prebacivanje fajla na privremenu lokaciju na server računaru se obavlja automatski. Zatim je taj fajl potrebno pomeriti na odgovarajuću lokaciju. Taj posao obavlja metoda uploadPicture() iz klase Phone:

```
private function uploadPicture($pic) {
        //upload picture
        if ($pic['error'] > 0) {
            //echo 'Problem: ';
            switch ($_FILES['picture']['error']) {
                case 1:
                  echo 'File exceeded upload_max_filesize';
                  break;
                case 2:
                  echo 'File exceeded max_file_size';
                  break;
                case 3:
                  echo 'File only partially uploaded';
                   break;
                case 4:
                  echo 'No file uploaded';
                  break;
            }
        }

        // put the file where we'd like it
        $upfile = '../'.$this->phone_pic;

        if (is_uploaded_file($pic['tmp_name'])) {
            if (!move_uploaded_file($pic['tmp_name'], $upfile)) {
              echo 'Problem: Could not move file to destination directory';
            }
        }

    }
```

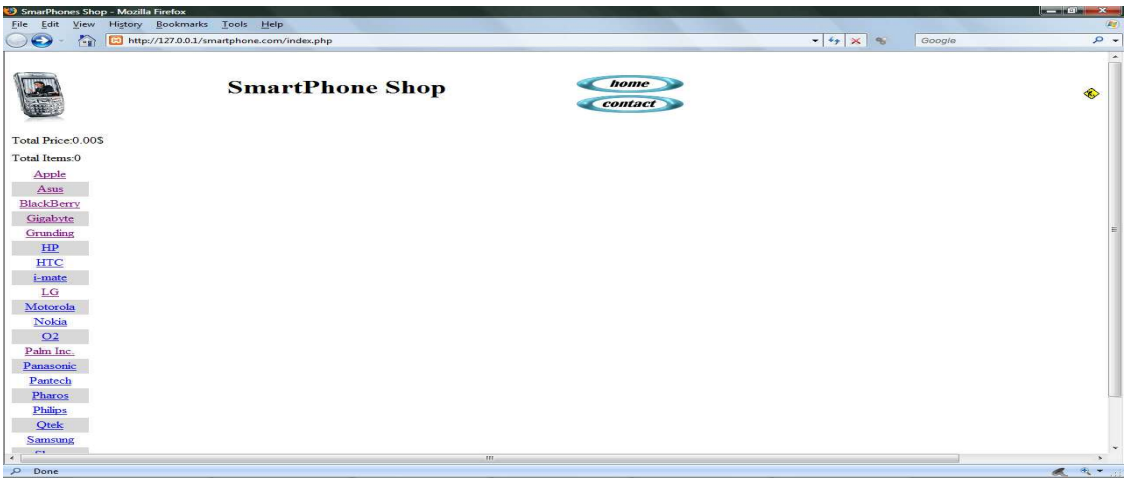
Funkcija is_uploaded_file() proverava da li je fajl prebačen na server računar preko HTTP POST mehanizma za uploadovanje. Ako jeste, onda se fajl pomera na odgovarajuću lokaciju na server računaru pomoću metoda move_uploaded_file().

Metode klase Page se koriste za prikazivanje sadržaja na svim web stranicama prodavnice. Tako, klasa Page ima metode koje se korite na svakoj strani web prodavnice. Na primer, metoda:

```
function DisplayTop($dir, $session);
```

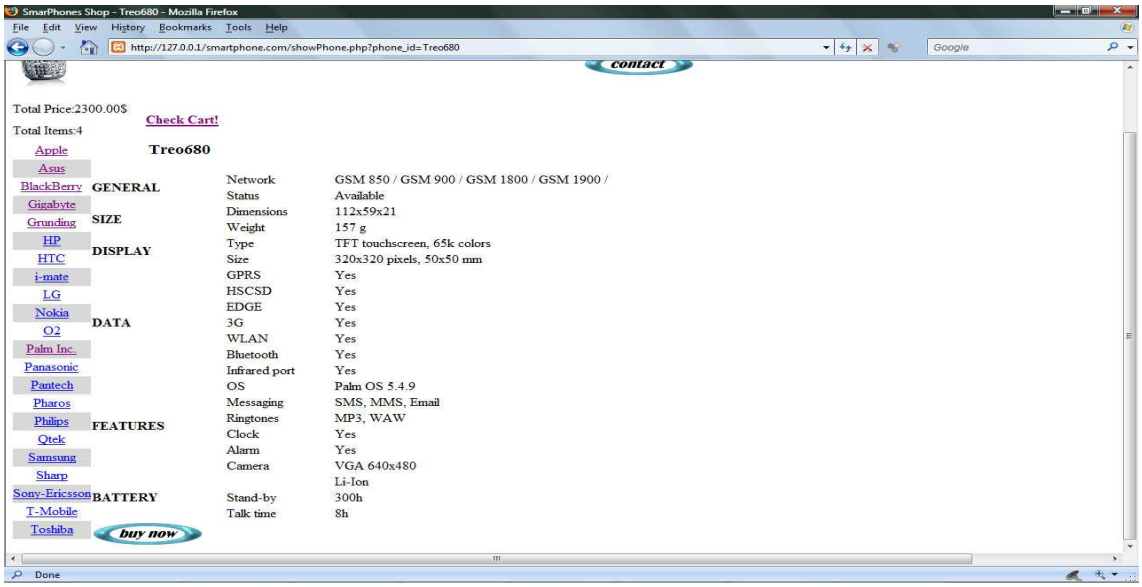
služi za ispisivanje zaglavlja HTML strane, postavljanja naslova stranice, zadavanja ključnih reči, kao i za iscrtavanje zaglavlja same web stranice. Pored ove metode tu je i metoda za prikazivanje liste svih proizvođača i druge.

Kada „kupac“ poseti prodavnicu mobilnih telefona, prva strana koju vidi (http://127.0.0.1/smartphone.com/index.php) je strana na kojoj se (sa leve strane) nalazi spisak svih proizvođača.



Klikom na neki naziv proizvođača otvara se stranica sa svim telefonima tog proizvođača koji se trenutno nalaze u prodavnici (bazi podataka).

Klikom na sliku određenog telefona otvara se stranica sa njegovim specifikacijama.



Ispod specifikacija telefona se nalazi dugme „Buy now“ čijim pritiskom se izabrani telefon dodaje u korpu za kupovinu. Na stranici je korpa za kupovinu pretstavljena labelama „Total Pice“ i „Total Items“ tako da u svakom trenutku „kupac“ može da vidi koliko je telefona dodao u korpu i kolika je cena svih telefona u korpi zajedno. Klikom na link „Check Cart!“ može se videti kompletan spisak telefona koji se nalaze u korpi.

Link „Continue Shopping“ služi za nastavljanje kupovine, dok se odabirom linka „Purchase“ otvara stranica na kojoj se nalazi formular za kupovinu na kojem je potrebno da kupac unese svoje podatke i pritiskom na dugme „BUY“ kupi telefone koje je dodao u korpu.

Nakon pritiska na dugme „BUY“ treba obaviti procesiranje platne kartice i obavestiti kupca o uspešnosti naplate (kupovine).


Purchase - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://
Google
SmartPhone Shop
home
contact
Apple
Asus
BlackBerry
Gigabyte
Grunding
HP
Thankyou for shopping with us. Your order has been placed.
Done
Purchase - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://
Google
SmartPhone Shop
home
contact
Apple
Asus
BlackBerry
Gigabyte
Grunding
Could not process your card. Please contact the card issuer or try again.
Done

5.4 Izrada korpe za kupovinu

Kada su kreirani baza podataka i katalog proizvoda, implementcija korpe za kupovinu je prilično jednostavna.

Najjednostavniji način za kreiranje korpe za kupovinu u PHP-u jeste upotreba sesija. Svakoj PHP sesiji dodeljuje se jedinstven identifikator, što je nasumično izabran i šifrovan broj. PHP generiše identifikator sesije i čuva ga na klijentskom računaru dok traje sesija.

Osnovni koraci postupka upotrebe sesija su:

- Otvaranje sesije
- Registrrovanje promenljivih sesije
- Upotreba promenljivih sesije
- Poništavanje promenljivih i uništavanje sesije

Otvaranje ili započinjanje sesije može da se uradi na dva načina. Najjednostavniji način je da se na početku skripte u kojoj je potrebna sesija pozove funkcija

```
session_start();
```

Ova fukcja ispituje da li već postoji započeta sesija. Ako ne postoji, funkcija započinje novu sesiju i time omogućava pristup superglobalnom nizu $_SESSION. Ako postoji već započeta sesija , onda funkcija session_start() učitava tekuće vrednosti registrovanih promenljivih sesije da bi se one mogle koistii u skriptu.

Drugi način da se započne sesija jeste da se PHP podesi tako da automatski započinje sesiju čim neko stigne na web lokaciju. To se može zadati pomoću opcije session.auto_start u datoteci php.ini.

Da bi se napravila (registrovala) promenljiva sesije, potrebno je zadati vrednost nekom elementu super globalnog niza $_SESSION. Na primer, pomoću sledeća tri reda :

```
$_SESSION['cart'] = array();
$_SESSION['items'] = 0;
$_SESSION['total_price'] ='0.00';
```

kreirane su tri promenljive koje predstavljaju korpu za kupovinu . $_SESSION['cart'] – ova promenljiva predstavlja asocijativni niz u kojem su ključevi identifikatori izabranih telefona, a vrednosti su količine tih telefona. $_SESSION['items'] – predstavlja ukupan broj izabranih telefona i SESSION['total_price'] – predstavlja ukupuan iznos cene za sve telefone dodate u korpu.

Da bi određena promenljiva sesije postala upotrebljiva, sesija najpre mora biti započeta pomoću funkcije session_start(), i zatim se toj promenljivoj može pristupiti preko super globalnog niza $_SESSION. U sledećem primeru je prikazano kako se dodaje novi telefon u korpu:

```
function add_phone_id($phone_id, $db) {
      //add session var
    if($phone_id) {

          if(!isset($_SESSION['cart']))
          {

            $_SESSION['cart'] = array();
            $_SESSION['items'] = 0;
            $_SESSION['total_price'] ='0.00';
          }

            if(isset($_SESSION['cart'][$phone_id]))
                  $_SESSION['cart'][$phone_id]++;
```

```
            else {
                  $_SESSION['cart'][$phone_id] = 1;

            }
            $_SESSION['total_price'] = calculate_price($_SESSION['cart'], $db);
            $_SESSION['items'] = calculate_items($_SESSION['cart']);
      }
}
```

U ovom primeru se prvo proverava isset() funkcijom da li je promenljiva $_SESSION['cart'] registrovana i ako nije, onda se registruju sve promenljive potrebne za korpu. Zatim se dodaje novi telefon u korpu ili se, ako taj telefon već postoji , njegova količina uvećava za 1. Na kraju rutine se izračunavaju promenljive $_SESSION['items'] i $_SESSION['total_price'] koje se prikazuju u zaglavlju html stranice kao ”Total Price” i ”Total Items”.

Kada neka promenljiva sesije više nije potrebna, ona se može poništiti. To se može učiniti direktno poništavanjem odgovarajućeg elementa niza $_SESSION, npr.:

```
unset($_SESSION['items']);
```

Da bi se istovremeno poništili svi elementi niza $_SESSION, potrebno je upotrebiti sledeći iskaz:

```
$_SESSION = array();
```

Kada sesija nije više potrebna, tj. kada se završi sa njenom upotrebom, potrebno je poništiti sve njene promenljive i zatim pozvati funkciju

```
session_destroy();
```

koja će osloboditi i identifikator sesije.

5.5 Proširenja aplikacije

Za aplikaciju koja je opisana u predhodnim poglavljima može se reći da je „minimalna“ celina koju je trebalo napraviti kako bi se pokazalo kako se PHP i MySQL mogu iskoristiti za kreiranje web aplikacija. Da bi ova aplikacija bila prihvatljivija, u komercijalnom smislu, moguće je doraditi i neka proširenja.

Tako je, na primer, moguće napraviti stranicu preko koje bi „kupac“ mogao da pronađe odgovarajući pametni telefon zadavajući parametre koje bi određeni telefon morao da ima. Takođe je moguće u delu aplikacije, koji je rezervisan za „vlasnika“ prodavnice, napraviti stranice na kojima bi mogle da se pregledaju neke statistike vezane za prodaju telefona i slično.

Interesantno proširenje aplikacije bi bilo i omogućavanje da se šalju e-mail poruke. Koristeći ovu mogućnost „vlasnik“ prodavnice bi mogao da na jednostavan nači obaveštava svoje kupce o novim telefonima ili nekim popustima i slično. Ovu opciju relativno lako napraviti korišćenjem PHP-a. Serveri za elektronsku poštu najčešće podržavaju jedan ili oba najpoznatija protokola za čitanje sadržaja elektronskih sandučića. To su POP3 (*Post Office Protocol version 3*) i IMAP (*Internet Message Access Protocol*). U PHP-u je ugrađena odlična podrška i za POP3 i IMAP protokol, koja je obezbeđena preko biblioteke funkcija IMAP.

Pored IMAP biblioteke, u PHP-u postoje i razne druge biblioteke od kojih su najkorisnije: PDFlib – biblioteka funkcija za generisanje PDF dokumenata i GD – biblioteka za obradu slika u formatima JPEG, PNG, WBMP.

6 Zaključak

Glavni konkurenti PHP-a su Perl, Microsoftov ASP.NET, JSP (Java Server Pages) i ColdFusion. Osobine po kojima se PHP ističe u odnosu na svoju konkurenciju su: velika PHP zajednica, visoke prefomanse, lakoća učenja i upotrebe, ugrađene biblioteke za obavljanje velikog broja poslova koji su uobičajeni u web aplikacijama, izvorni kod je dostupan svima, prenosivost, dobra podrška za objektno orijentisano programiranje, povezivanje sa velikim brojem sistema za upravljanje bazama podataka i cena.

Velika PHP zajednica
Oko PHP-a je stvorena veoma velika zajednica bez koje PHP verovatno ne bi predstavljao ono što predstavlja danas. Prednosti PHP zajednice su dobijanje pomoći, veliki broj skriptova čiji je izvorni kod dostupan svima. Postoji veliki broj foruma na kojima se može zatražiti pomoć u rešavanju problema, veliki broj sajtova sa uputstvima i primerima kako se određeni problemi rešavaju u PHP-u.

Visoke prefomanse
PHP je veoma efikasan. Jedan relativno jeftin server može da obradi više miliona zahteva dnevno. Ako se aplikacija izvršava na više spregnutih računara, onda kapacitet obrade postaje praktično neograničen.

Lakoća učenja i upotrebe
Lakoća učenja PHP-a se ogleda u njgovoj sintaksi koja je jako slična sintaksi programskog jezika C. Tako, ako neko već poznaje programske jezike C, C++ ili Javu, gotovo odmah će moći da koristi PHP na produktivan način.

Ugrađene biblioteke
PHP u sebi ima veliki broj funkcija koje omogućavaju obavljanje velikog broja poslova koji su potrebni u web aplikacijama. Tako npr. mogu se generisati GIF slike tokom rada aplikacije, uspostavljati veze sa web servisim, koristiti XML dokumenta, slati poruke elektronskom poštom, itd.

Izvorni kod je dostupan svima
Izvorni kod PHP-a je javno dostupan. Za razliku od komersijalnih proizvoda zatvorenog koda, ako je potrebno nešto izmeniti ili dodati PHP-u, može se slobodno učiniti.

Prenosivost
PHP je na raspolaganju za mnoge operativne sisteme: Linux, FreeBSD, komercijalne verzije UNIX-a i razne verzije Microsoftovog Windowsa. Kod koji se napiše u PHP pod jednim operativnim sistemom najćešće će raditi i pod drugim operativnim sistemima.

Dobra podrška za objektno orijentisano programiranje
PHP-ova verzija 5 donosi objektno orijentisane mogućnosti: nasleđivanje, privatni i zaštičeni atributi i metode, abstraktne klase i metode, interfejsi, konstruktori i destruktori.

Povezivanje sa velikim broju sistema za upravljanje bazama podataka
PHP standardno omogućava uspostavljanje veza s više sistema za upravljanje bazama podataka. Osim sa MySQL-om, moguće je uspostavljanje direktnih veza sistemima PostgreSQL, mSQL, Oracle, FilePro, Informix, InterBase i Sybase. U PHP 5 ugrađen je i interfejs za pristupanje SQLite bazama podataka. Pomoću standarda ODBC(Open Database Connectivity) mogu se uspostaviti veze sa svakom bazom podataka za koju postoji ODBC upravljački program.

Cena
PHP je besplatan i njegova najnovija verzija se uvek može preuzeti sa adrese www.php.net.

Najvažniji konkurentni MySQL-a su PostgreSQL, takođe besplatan, i komercijalni Microsoftov SQL i Oracle. Od konkurencije ga izdvajaju sledeće osobine: visoke performanse, cena, prenosivost, lako se uči, izvorni kod je javno dostupan kao i podrška u slučaju problema.

Visoke performanse
Na web lokaciji www.mysql.com mogu se naći rezultati uporednih testova. Mnogi od njih pokazuju da je MySQL brži od konkurentskih programa. Godine 2002, časopis eWeek objavio je rezultate uporednih testova pet baza podataka koje su bile izvor podataka za istu web aplikaciju. Isti, najbolji rezultat, postigli su MySQL i znatno skuplji Oracle.

Cena
MySQL se može nabaviti potpuno besplatno po uslovima GPL (General Public License), a na raspolaganju je i komercijalna verzija licence. Komercijalna verzija je neophodna samo u slučaju ako se MySQL distribuira kao sastavni deo aplikacije koja nije aplikacija otvorenog koda.

Prenosivost
MySQL se može koristiti na mnogim verzijama UNIX-a kao i na Microsoftovom Windowsu.

Lako se uči
Ako programer poznaje neki drugi sistem za upravljanje bazama podataka koji kao upitni jezik koristi SQL, onda neće imati većih potečkoća da pređe na MySQL.

Izvorni kod je javno dostupan
Kao i za PHP, izvorni kod MySQL-a može se besplatno nabaviti i slobodno menjati.

Podrška u slučaju problema
Podrška, obuka, konsultantske usluge i sertifikovanje znanja mogu se dobiti od firme MySQL AB (www.mysql.com).

PHP i MySQL zbog ovih svojih karakteristika predstavljaju, ako ne najbolji, onda jedan od najboljih izbora u izradi složenih i komercijalnih web aplikacija.

7 Literatura

- Jeremy Allen - „PHP 4.1“
- Luke Welling, Laura Thomson – „PHP and MySQL Web Development“
- Tim Converse – „PHP 5 and MySQL Bible“
- Julie C. Meloni – „Sams Teach Yourself MySQL in 24 hours“
- http://www.php.net – „Zvanični PHP internet sajt“
- http://www.mysql.com/ - „Zvanični MySQL internet sajt“
- http://www.w3schools.com/html/default.asp - „HTML tutorial“
- http://www.apachefriends.org/en/xampp.html - „XAMPP aplikacija“

8 Napomena

Kompletan „source code“ potreban za kreiranje web prodavnice pametnih telefona koja je u ovom dokumentu opisana, kao i XAMPP aplikacija nalaze se CD koji je priložen uz ovu master tezu.