Internet programiranje

Programski jezik PHP i rad sa MySQL bazom

Predavači:
Dr Dražen Drašković, docent
master inž. Jelica Cinović, asistent

Šta je SQL?

- SQL (*Structured Query Language*) - strukturirani jezik za upite
- SQL je standardizovani jezik za pristupanje sistemima za upravljanje relacionim bazama podataka (RDBMS).
- SQL omogućava unošenje podataka u bazu i čitanje podataka iz nje.
- SQL se koristi u sistemima za upravljanje bazama podataka: MySQL, Oracle, PostgreSQL, Sybase, Microsoft SQL Server i drugi.

7) PHP i MySQL

- Ove funkcije omogućavaju pristup MySQL database serveru.
- Podela:
  - mysql funkcije (*MySQL functions*)
    - proceduralni pristup
  - mysqli funkcije (*MySQL Improved Extensions*)
    - proceduralni pristup
    - objektno-orjentisani pristup

MySQL funkcije

- Starija biblioteka funkcija za rad sa *MySQL*-om
- Ne podržava neka proširenja *MySQL*-a
  uvedena u verzijama 4.1 i novijim
- Proceduralni pristup
- *MySQL* funkcije nisu ugrađene u *PHP*.
  => Treba dodati odgovarajuće dinamičko proširenje
  (dinamic extension) u php.ini fajlu:

  extension=php_mysql.dll

MySQL Improved Extensions

- Novija biblioteka funkcija za rad sa *MySQL*-om
- Podržava proširenja *MySQL*-a
  uvedena u verzijama 4.1 i novijim
- Proceduralni ili objektno-orijentisani pristup
- *MySQL Improved* f-je nisu ugrađene u PHP.
  => Treba dodati odgovarajuće dinamičko proširenje (*dinamic extension*) u php.ini fajlu:

  extension=php_mysqli.dll

Čitanje podataka iz MySQL baze

Čitanje podataka iz baze

1. Uspostavljanje veze sa MySQL-om i označavanje trenutno aktivne baze podataka **mysqli_connect**
2. Izvršavanje upita i dobijanje skupa rezultata **mysqli_query**
3. Učitavanje reda rezultata iz skupa rezultata **mysqli_fetch_row**
4. Obrada dobijenih vrednosti za učitani red
5. Zatvaranje konekcije sa bazom podataka **mysqli_close**

Uspostavljanje veze sa bazom

mysqli mysqli_connect(

string host ,

string username ,

string password ,

string database)

- Uspostavllja vezu sa RDBMS-om,
  i označava aktivnu bazu podataka.
- Ova funkcija umesto objekta vraća rezultat tipa resurs (objekat tipa mysqli, odnosno konekciju na MySQL server) ili *false*, ako konekcija ne može da se uspostavi

Izbor baze podataka

bool mysqli_select_db
(mysqli connection , string database)

Označava BP zadatu parametrom *database* kao trenutno aktivnu (BP nad kojoj će se izvršavati upit koji se prosledi po ostvarenoj MySQL konekciji), u vezi sa MySQL-om zadatoj parametrom *connection*.

Funkcija vraća:

TRUE, u slučaju uspeha

FALSE, u slučaju neuspeha.

Zadavanje upita bazi podataka

mysqli_result mysqli_query
(mysqli connection , string SQL_command)

Izvršava SQL iskaz čiji je tekst zadat parametrom *SQL_command*.

Parametar *connection* je objekat tipa mysqli (konekcija sa MySQL-om) dobijen kao povratna vrednost funkcije mysqli_connect().

Ova funkcija vraća:

- objekat tipa mysqli_result (skup rezultat upita), za komande koje vraćaju rezultat SELECT, SHOW, DESCRIBE ili EXPLAIN
- TRUE, za ostale komande u slučaju uspeha
- FALSE, u slučaju neuspeha

mysqli_fetch_row()

array mysqli_fetch_row
(mysqli_result result_set)

- Učitava red po red rezultata iz objekta tipa mysqli_result , dobijenog kao povratna vrednost funkcije mysqli_query
- Funkcija vraća:
  - numerički niz koji predstavlja red rezultata (prva kolona - element sa indeksom 0 itd.)
  - NULL, kada nema više redova u skupu rezultata

mysqli_close()

bool mysqli_close (mysqli connection)

- Zatvara konekciju sa MySQL-om koja je prethodno uspostavljena pozivanjem funkcije mysqli_connect().

- Funkcija vraća:
  - TRUE, u slučaju uspeha
  - FALSE, u slučaju neuspeha

Ostale mysqli funkcije

- mysqli_free_result
- mysqli_num_rows
- mysqli_set_charset
- *Napomena*:
  Potpun spisak i detaljna objašnjenja
  svih mysqli funkcija u PHP dokumentaciji

mysqli_free_result

void mysqli_free_result
(mysqli_result result)

Oslobađa memoriju dodeljenu parametru *result*, dobijenim kao povratna vrednost f-je mysqli_query.

Bez poziva mysql_free_result f-je memorija dodeljena skupu rezultata bi se oslobodila tek pri završetku PHP skripta.

Obrada grešaka (#1)

- Pri upotrebi MySQL-ovih funkcija za rad s bazama podataka postoji verovatnoća da se pozivanje neke od navedenih funkcija završi neuspehom
- Uzroci neuspeha: RDBMS nije dostupan, zadat je pogrešan parametar, korisnik je uneo pogrešnu lozinku, upit je pogrešno sastavljen, ...
- U tom slučaju PHP će u većini slučajeva prikazati neku svoju grešku (najčešće na nivou Warning-a) i nastaviti sa izvršavanjem skripta

Obrada grešaka (#2)

- Cilj je izbeći ovakvo ponašanje,
  i obezbediti sledeće:
  - omogućiti korisniku da ispisuje greške kako on želi (umesto PHP-ovih grešaka) i
  - omogućiti prekid izvršavanja skripta u slučajevima kada je to potrebno (ako ne uspe konekcija sa BP nema smisla nastaviti dalje,...)

Funkcije za obradu MySQLi grešaka (#1)

- Grupa funkcija MySQL Improved sadrži i funkcije za obradu grešaka koje mogu nastati pri radu sa mysqli f-jama, koje omogućavaju otkrivanje i detaljno obaveštavanje o nastalim greškama

- Ove funkcije mogu omogućavaju programerima lakše otkrivanje i ispravljanje grešaka, one nisu od interesa krajnjim korisnicima

Funkcije za obradu MySQLi grešaka (#2)

int mysqli_connect_errno ( )

F-ja vraća:

broj (kod) greške nastale pri poslednjem pozivu f-je mysqli_connect()
0, ako nije došlo do greške

string mysqli_connect_error ( )

F-ja vraća:

tekst opisa greške nastale pri poslednjem pozivu f-je mysqli_connect()

prazan string, ako nije došlo do greške

Funkcije za obradu MySQLi grešaka (#3)

int mysql_errno (mysqli connection)

F-ja vraća:

broj (kod) poslednje greške nastale pri pozivu neke mysqli f-je u konekciji *connection*

0, ako nije došlo do greške

string mysqli_error (mysqli connection)

F-ja vraća:

tekst opisa poslednje greške nastale pri pozivu neke mysqli f-je u konekciji *connection*

prazan string, ako nije došlo do greške

Funkcije za obradu MySQLi grešaka (#4)

- Korisno bi bilo napisati sledeće korisnički definisane funkcije:

```
function showerror_connection(){
    die ("Greška ".mysqli_connect_errno()." :".
          mysqli_connect_error());
}

function showerror($connection){
    die ("Greška ".mysqli_errno($connection)." :".
          mysqli_error($connection));
}
```

- Ove funkcije će programeri pozivati u slučaju otkrivanja greške pri pozivu neke mysqli funkcije, kako bi dobili detaljne informacije o nastaloj grešci.

Operator @

- Operator @ predstavlja **operator za kontrolu prikaza standardnih PHP-ovih poruka o greškama** u veb pretraživaču
- Ako se operator @ stavi ispred nekog izraza, bilo koja PHP-ova standardna greška koja je eventualno nastala u toku izračunavanja izraza neće biti prikazana u veb pretraživaču (što bi se desilo ako bi izostavili @)
- Podsetnik: display_errors=On (php.ini)

Upotreba datoteka za umetanje (#1)

- Pogodno je parametre konekcije sa BP (host, username, password, database) čuvati u posebnoj datoteci

- Fleksibilnost podešavanja tih parametara na jednom, centralnom mestu olakšava:
  - testiranje sistema sa rezervnom ili udaljenom kopijom podataka, jednostavnim menjanjem imena baze podataka i/ili servera u datoteci.
  - testiranje sistema sa različitim kombinacijama korisničkih imena i lozinki, sa različitim ovlašćenjima

Upotreba datoteka za umetanje (#2)

- Pogodno je i korisnički definisane funkcije za obradu grešaka čuvati u posebnoj datoteci.
- Datoteke za umetanje najčešće imaju ekstenziju .inc ili .php

Upotreba datoteka za umetanje - bezbedonosni problem -

- Ako korisnik aplikacije zahteva datoteku za umetanje sa ekstenzijom .inc, njen sadržaj će se prikazati u veb pretraživaču, tako da se mogu videti parametri i lozinke za pristup RDBMS-u odnosno izvorni kodovi funkcija koji bi trebalo da ostanu skriveni

Upotreba datoteka za umetanje - rešenje problema: prvi način -

- **Umesto ekstenzije .inc koristiti ekstenziju .php** (najbolje rešenje)

- U tom slučaju PHP parser obrađuje sadržaj datoteke za umetanje, ali se u veb pretraživaču ništa ne prikazuje jer datoteka ne sadrži telo skripta

- Jedini nedostatak je što se takve datoteke ne mogu lako razlikovati od drugih datoteka
  => rešenje: sve datoteke za umetanje staviti u poseban folder (npr. inc)

Upotreba datoteka za umetanje - rešenje problema: drugi način -

- **Datoteke za umetanje smestiti negde izvan stabla dokumenta u vašoj instalaciji Apache veb servera**

- U tom slučaju se u direktivi include ili require navodi kompletna putanja do datoteke

Upotreba datoteka za umetanje - rešenje problema: treći način -

- **Podesiti Apache tako da se korisnicima ne dozvoljava učitavanje sadržaja datoteka sa ekstenzijom .inc**

Rad sa drugim bazama podataka

Rad sa drugim RDBMS

- Rad sa drugim RDBMS (*Relational Database Management System*) se odvija na sličan način kao sa MySQL-om (isti niz koraka).

- Za svaki od podržanih RDBMS-a postoji skup funkcija jako sličan onim za MySQL (PostgreSQL, Oracle, Microsoft SQL Server,...).

ODBC

- Za RDBMS koje PHP ne podržava direktno (kao što je npr. *Microsoft Access*) na raspolaganju su ODBC funkcije (*Open DataBase Conectivity* - otvoreni standard za povezivanje sa bazama podataka)

- ODBC funkcije se mogu koristiti i za RDBMS koje PHP podržava.

- ODBC omogućava zamenu RDBMS-a u sloju baze podataka veb aplikacije, bez potrebe za promenom PHP skripta.

ODBC klijent

- Kada se RDBMS-u pristupa pomoću ODBC funkcija neophodno je instalirati i ODBC klijent-a za dati RDBMS.

- ODBC klijenta treba da obezbedi proizvođač RDBMS-a, i on je za svaki RDBMS specifičan.

PHP & MySQL

Čitanje podataka na osnovu korisničkih unetih parametara

Primer

- Napraviti HTML formu za pretraživanje prodavnica po nazivu:
  - Korisnik unese parametre pretraživanja u HTML form element i po završetku pritisne dugme Submit.
  - Na osnovu unetih parametara pretraživanja vrši se čitanje zahtevanih podataka iz baze i isti se prikazuju.
  - Koraci 1 i 2 se mogu ponavljati veći broj puta.

Struktura skriptova

- Dva pristupa
  - Razdvojeni skriptovi:
    Sastoje se iz dve povezane komponente
    (dva odvojena fajla)

    Podvarijanta: sa korišćenjem frejmova.

  - Kombinovani skriptovi:
    Sastoje se jedne komponente (jednog fajla)

Razdvojeni skriptovi

- Sastoje se iz dve povezane komponente (dva odvojena fajla)
  - html (ili php) fajl koji sadrži HTML formu za unos parametara koje prosleđuje ovom drugom php fajlu
  - php fajl koji prihvata prosleđene parametre, obrađuje ih, čita podatke iz baze i prikazuje rezultate;
    po pravilu, ovaj fajl sadrži link na prvi html (ili php) fajl za ponovno zadavanje parametara pretraživanja

Kombinovani skriptovi #1

- Dve komponente razdvojenih skriptova, html (ili php) fajl koji sadrži HTML formu za unos parametara i PHP fajl koji vrši prijem, obradu i prikaz podataka, se nalaze u istom fajlu.
- Ove komponente su potpuno iste kao odgovarajuće komponente odvojenih skriptova.
- Treba dodati upravljački deo koji određuje kada se izvršava jedna, a kada druga komponenta.

Kombinovani skriptovi #2

- Kada je kombinovani (jedinstveni) skript pozvan bez parametara, treba da se izvrši komponenta koja predstavlja HTML formu za unos parametara.

- Kada je kombinovani (jedinstveni) skript pozvan sa parametrima, prosleđenim od samog sebe, tačnije od komponente koja predstavlja HTML formu, potrebno je izvršiti PHP komponentu koja prima, obrađuje i prikazuje podatke.

Struktura kombinovanih skriptova

```
<?php
    if (empty($_POST["poslato_iz_forme"])){
?>
    ... HTML (PHP) form komponenta sa hidden
elementom poslato_iz_forme=1


<?php
    }/* if */ else{
?>
    ... PHP komponenta za obradu i prikaz
<?php
    }//else
?>
```

Ugradnja parametara u hiperlinkove

Karakteristični use case

- Slučaj čitanja podataka iz BP na osnovu korisnički prosleđenih podataka ugrađenih u hiperlinkove (karekteristični slučaj korišćenja)
  - Korisniku se prikazuje spisak stavki, koje ujedno predstavljaju hiperlinkove
  - Kada korisnik odabere određenu stavku tj. prati odgovarajući link, vrši se čitanje iz baze i prikaz podataka vezanih za odabranu stavku
  - Koraci 1 i 2 se mogu ponavljati veći broj puta

Struktura skriptova

- Kod prosleđivanja parametara putem hiperlinkova, postoji obično dva povezana PHP fajla.

- Prvi fajl prikazuje sve stavke. Svaka stavka predstavlja hiperlink koji referencira drugi fajl, samo se za svaku stavku prosleđuje drugačija vrednost parametra.

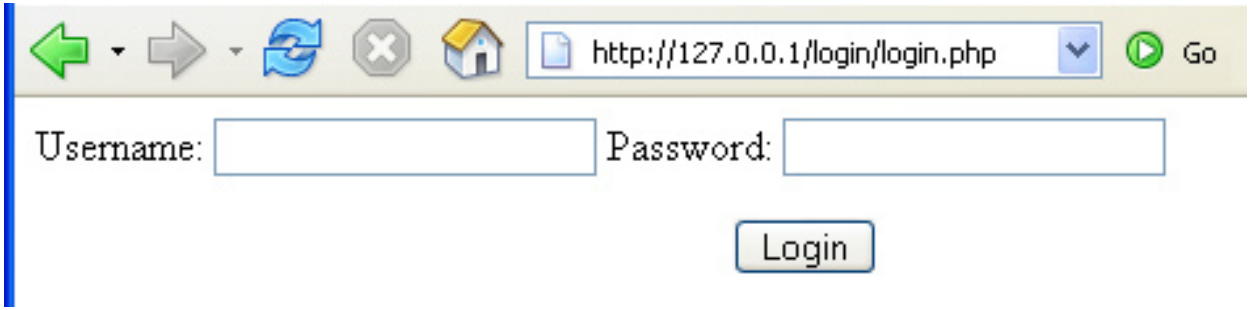
- Drugi fajl prihvata parametar prosleđen putem hiperlinka od prvog fajla, čita iz baze i prikazuje podatke o izabranoj stavki u okviru prvog fajla; po pravilu ovaj fajl sadrži povratni link na prvi fajl.

PHP i rad sa MySQL-om

Korisnička autentifikacija

Forma za prijavljivanje korisnika

- Tekstualna polja za unos:
  - korisničkog imena
  - lozinke



login.php (1)

- Na početku skripta stavljamo lokalne promenljive:

```
$uname = "";
$pword = "";
$errorMessage = "";
$num_rows = 0;
```

- **$errorMessage** ćemo koristiti, ako želimo da sačuvamo neku poruku koju ćemo vratiti login stranici.

login.php (2)

- Sledeći kod prikazuje SQL proveru, kako bi se sprečio SQL injection napad:

```
if ($_SERVER['REQUEST_METHOD'] = = 'POST')
{


}
```

- Nakon ovoga znamo da li je forma prosleđena ili nije (da li smo kliknuli na Submit)

login.php (3)

- Prvo ćemo prihvatiti podatke koji su prosleđeni i smestiti ih u lokalne promenljive:

```
$uname = $_POST['username'];
$pword = $_POST['password'];
```

- Nakon ovoga znamo da li je forma prosleđena ili nije (da li smo kliknuli na Submit)
- Zatim treba da uklonimo i neželjeni HTML kod (skript napad):

```
$uname = htmlspecialchars($uname);
$pword = htmlspecialchars($pword);
```

login.php (4)

- Pokušamo da ostvarimo konekciju sa bazom:

```
$user_name = "root";
$pass_word = "";
$database = "login";
$server = "127.0.0.1";


$db_handle = mysql_connect($server, $user_name,
$pass_word);
$db_found = mysql_select_db($database, $db_handle);
```

- Ako je baza pronađena, varijabla nazvana $db_found će biti true:

```
if ($db_found) {
}
else {
    $errorMessage = "Error logging on";
}
```

login.php (5)

- Sada treba tekst koji imamo da sredimo od neželjenih znakova:

```
$uname = quote_smart($uname, $db_handle);
$pword = quote_smart($pword, $db_handle);
```

- Funkcija quote_smart sprečava SQL injection :

```
function quote_smart($value)
{
    $value = stripslashes($value);
    if (!is_numeric($value))
    $value = "'" . mysql_real_escape_string($value) .
    "'";
    return $value;
}
```

PHP - stripslashes

- string **stripslashes** (string $str)
- Ima string kao argument,
  već smo rekli da radi suprotno od addslashes()
- Vraća string vrednost, ali bez backslash-eva.
  Na primer:
  - \' postaje samo '
  - dupli backslash (\\) postaje jedan (\)

PHP - is_numeric

- bool **is_numeric** (mixed $var)
- Argument $var može biti različitog tipa!
- Vraća **TRUE** ako je argument *var* ili broj ili numerički string, **FALSE** u drugim slučajevima.

```
<?php
$tests = array("42", 1337, "1e4", "not numeric",
        Array(), 9.1);

foreach ($tests as $element) {
    if (is_numeric($element)) {
        echo "'{$element}' je broj!", PHP_EOL;
    } else {
        echo "'{$element}' nije broj!", PHP_EOL;
    }
}
?>
```

PHP - mysql_real_escape_string

- Izbacuje specifične znakove u stringu, kako bi se string koji vrati funkcija koristio u SQL naredbi.
- Ova funkcija mora uvek (uz nekoliko izuzetaka) da se koristi za pravljenje “bezbednih podataka” pre slanja upita MySQL bazi!

login.php (6)

- Kada imamo korisničko ime i lozinku, možemo da izvršimo SQL upit:

```
$upit = "SELECT * FROM login WHERE kime =
$uname AND klozinka = $pword";
$rezultat = mysql_query($upit);
```

- Funkcija mysql_query izvršava upit na MySQL bazom. Šta vraća mysql_query?

mysql_query($upit)

- Vrednost u promenljivoj $rezultat će biti ili true (ako postoji neki red u bazi koji zadovoljava upit) ili false (ako nijedan red nije vratio upit). U drugom slučaju, treba stavi poruku o grešci:

```
if ($rezultat) {


}
else {
   $errorMessage = "Nije uspesno
   logovanje";
}
```

mysql_num_rows($ rezultat)

- Ako se SQL upit izvrši uspešno, baza može da vrati nekoliko redova.
- Tada koristimo mysql_num_rows( ) funkciju
- Ako nema redova, znači da korisnik sa tim korisničkim imenom i tom lozinkom, ne postoji u bazi:

```
$num_rows = mysql_num_rows($rezultat);
if ($num_rows > 0) {
   $errorMessage= "Ulogovani ste!";
}
else {
   $errorMessage= "Nije uspesno logovanje";
}
```

Username = primarni ključ

- U primeru, ako je broj redova veći od 1, to znači da 2 korisnika imaju isto korisničko ime i lozinku. Ako imate sistem (bazu) gde svaki korisnik ima jedinstveno korisničko ime, onda mora da bude $num_rows = 1.

PHP napredne tehnike

Zaštita aplikacije

Problem

- Nikada se ne pouzdajte ni u šta nad čim nemate potpunu kontrolu.
- Podaci korisničkog unosa se ne smeju prihvatiti bez provere (tj. upisati u BP ili neki fajl na serveru ili proslediti dalje sistemu pomoću neke izvršne komande).
- Problemi:
  - Zaštita baze podataka
  - Zaštita od zlonamernih komandi
  - Zaštita od zlonamernih skriptova

Zaštita baze podataka (#1)

- **P1**: Znakovi ", ' , \ , **NULL** mogu da izazovu probleme prilikom upisivanja u BP, zato što BP može da ih protumači kao upravljačke znakove.

- **R1**: Funkcija addslashes() pretvara sve ove znakove u izlazne sekvence dodajući ispred njih znak **** .
  Funkcija stripslashes() vrši inverznu obradu tj. uklanja znakove **** (vraća podatke u izvorni oblik).

Zaštita baze podataka (#2)

- Napomena: magic_quotes_gpc on (php.ini)
  - PHP automatski poziva f-ju addslashes() za sve ulazne GET, POST i COOKIE promenljive.
  - PHP automatski poziva funkciju stripslashes() pre slanja podataka na izlaz.
  - Opcija je po defaultu postavljena na vrednost on u novijim verzijama PHP-a.
  - Ne može biti postavljena u runtime-u.
  - Pozivom funkcije get_magic_quotes_gpc() dobija se status promenljive magic_quotes_gpc.

Zaštita baze podataka (#3)

- Napomena: magic_quotes_runtime on (php.ini)
  - PHP poziva funkciju f-ju addslashes() za sve podatke pročitane iz eksternih izvora (BP, tekst fajlovi, XML fajlovi).
  - Opcija je po defaultu postavljena na vrednost off u novijim verzijama PHP-a.
  - Može biti postavljena u runtime-u pozivom funkcije set_magic_quotes_runtime().
  - Pozivom funkcije get_magic_quotes_runtime() dobija se status promenljive magic_quotes_runtime.

Primer

```
$str1 = "Is your name O'reilly?";
$str2 = addslashes($str1);
echo($str2);   // Outputs: Is your name O\'reilly?
$str3 = stripslashes($str2);
echo($str3);   //Outputs: Is your name O'reilly?
```

Zaštita od zlonamernih komandi

- **P2**: Ako se podaci korisničkog unosa prosleđuju funkcijama system() ili exec() koje izvršavaju na veb serveru komandu koja im se prosledi kao parametar, pomoću određenih metaznakova zlonamerni korisnici mogu da izvrše proizvoljne naredbe na sistemu.

- **R2**: Funkcija escapeshellcmd() pretvara sve ove znakove u izlazne sekvence dodajući ispred njih znak **** .

Zaštita od zlonamernih komandi

- **P3**: Korisnici mogu da ugrade štetne skriptove (HTML i PHP oznake) u parametre koje prosleđuju što može dovesti do grešaka u prikazu veb stranica koje šalju te podatke na izlaz.
- **R3**: F-ja strip_tags() uklanja sve HTML i PHP oznake iz znakovnih podataka. F-ja htmlspecialchars() pretvara znakove koji u HTML-u nešto znače u odgovarajuće HTML specijalne znake tj. entitete(& => & , " => " , ' => ' , < => < , > => >)

Primer

```
$text = '<b>Hello</b> world';

 echo strip_tags($text);
// Output: Hello world

 echo strip_tags($text, '<b>');
// Output: <b>Hello</b> world
//  => Hello world (u browser-u)

 echo htmlspecialchars($text);
// Output: &lt;b&gt;Hello&lt;/b&gt; world
//  => <b>Hello</b> world (u browser-u)
```

Uklanjanje krajnjih belina

- **P4**: Vrednost korisnički unetog parametra sadrži krajnje leve i krajnje desne beline.

- **R4**: Pomoću funkcije trim() uklanjaju se krajnje beline sa obe strane stringa.

Ograničavanje dužine stringova

- **P5**: Dužina (broj karaktera) vrednosti nekog parametra korisničkog unosa ne zadovoljava ograničenja baze podataka.

- **R5**: Pomoću substr() ograničava se dužina korisničkog parametra odsecanjem karaktera sa desne strane ili se prekida izvršavanje i traži od korisnika ponovni unos podatka.

Značaj provere podataka

- Ključno je obezbediti da svi podaci zadovoljavaju:
  - Sve zahteve korisnika i sistema
  - Ograničenja definisana u bazi podataka

Mesta provere podataka

- Provera na klijentskoj strani - opciono
  Najčešće korišćenjem Java Script-a!

- Provere podataka u srednjem sloju
  (na serverskoj strani) - obavezno
  Korišćenjem PHP!

- Provera podataka u bazi (na serverskoj strani) - automatski DBMS vrši ovu proveru.

Provera podataka na klijentskoj strani

- Obavlja se u klijenskom veb pregledaču najčešće pomoću JavaScript-a.
- Obično se vrši provera vrednosti elemenata forme.
- Nedostatak:
  Zavisi od korisnika i njegovog okruženja - korisnik može da zabrani upotrebu JavaScript-a, kao i da namerno ili nenamerno zaobiđe proveru ispravnosti podataka.

Provera podataka u srednjem sloju

- Provera se vrši u skriptu srednjeg sloja (PHP skriptu) i predstavlja glavni način provere.
- Ovo se ne sme izostaviti ni u jednoj veb aplikaciji.

Provera podataka u bazi

- Provera se vrši automatski pri ažuriranju baze, a proverava se zadovoljenost ograničenja definisanih u bazi.
- Nije poželjno osloniti se na implicitnu proveru ispravnosti podataka u bazi iz sledećih razloga:
  - Presretanje grešaka koje MySQL javlja uz pomoć PHP funkcija je komplikovano
  - Nepotrebno se opterećuje mreža i DBMS
  - Teško je korisniku podatke i poruke o grešci predstaviti u razumljivom obliku

PHP f-je koje se koriste za proveru

- O nekim funkcijama je već bilo reči:
  - `bool empty( mixed var )`
  - `bool isset( mixed var,..., mixed var)`
  - `void var_dump ( mixed expression,..., mixed expression )`
- Funkcije za rad sa regularnim izrazima

Funkcija empty()

bool empty (mixed var)

- Utvrđuje da li se promenljiva može smatrati praznom
- Vraća TRUE ako se promenljiva može smatrati praznom, ili FALSE u suprotnom
- Promenljiva se smatra praznom ako nije definisana ili ima nultu vrednost. Sledeće vrednosti se smatraju praznim: **""** (prazan string), **0** (0 kao integer), **"0"** (0 kao string), **NULL**, **FALSE**, **array()** (prazan niz), **var $var;** (promenljiva tj. polje klase koja je definisana, ali joj nije dodeljena vrednost)

Funkcija isset()

bool isset (mixed var, ... ,mixed var)

- Funkcija vraća TRUE ako promenljiva var postoji, ili FALSE u suprotnom. U varijanti više promenljivih (poziv funkcije sa više argumenata) vraća se TRUE samo ako sve promenljive postoje.
- Smatra se da promenljiva $x ne postoji u sledeća tri slučaja:
  - Pre nego što joj je dodeljena neka vrednost
  - Ako joj je dodeljena vrednost null ($x=null)
  - Nakon poziva funkcije unset (unset($x))

Funkcija var_dump()

void var_dump (mixed expression, ... ,
mixed expression)

- Za izraze skalarnog tipa ispisuju se tip i vrednost izraza.
- Za nizove se ispisuje tip (array), broj članova niza, a zatim za svaki član niza indeks, tip i vrednost.
- Za objekte se ispisuje tip (Object), naziv klase, broj polja objekta/klase, a zatim za svako polje objekta naziv polja, tip i vrednost.
- Koristi se u procesu testiranja podataka prosleđenih skriptu, a ne za samu proveru.

Funkcije za rad sa regularnim izrazima

- Pronalaženje i izdvajanje vrednosti:

```
int preg_match ( string uzorak, string izvor
        [, array &$matches [, int $flags = 0
        [, int$offset = 0 ]]] )
```

- Zamena delova izvorne vrednosti:

```
mixed preg_replace( string pattern,
        string replacement, string string
        [, int $limit = -1 [, int &$count ]] )
```

- Razdvajanje izvorne vrednosti na elemente niza:

```
array preg_split ( string $pattern ,
        string $subject [, int $limit = 1
        [, int $flags = 0 ]] )
```

Funkcija preg_match() (#1)

```
int preg_match ( string uzorak, string izvor
          [, array &$matches [, int $flags = 0
          [, int$offset = 0 ]]]
```

- Proverava se da li se unutar stringa izvor nalazi deo koji odgovara regularnom izrazu uzorak (case sensitive).
- Ako se opcioni parametar *regs* ne koristi funkcija vraća vrednost 1 u slučaju da se unutar stringa *izvor* nalazi deo koji odgovara regularnom izrazu *uzorak*. U suprotnom vraća se false.

Funkcija preg_match() (#2)

- Parametar matches se koristi da se u njemu čuvaju delovi stringa *izvor* koji odgovaraju regularnom izrazu i delovima regularnog izraza (koji se u tom slučaju moraju nalaziti unutar malih zagrada).
- $regs[1] će sadržati podstring parametra *izvor* koji odgovara delu regularnog izraza koji počinje od prve leve zagrade; $regs[2] podstringu koji počinje od druge leve zagrade, itd.
  $regs[0] će sadržati koopiju kompletnog stringa koji odgovara regularnom izrazu.