**ELEKTROTEHNIČKI FAKULTET UNIVERZITETA U BEOGRADU**


# REALIZACIJA VEB KVIZA

–Diplomski rad–

Kandidat:

Ilija Đurić 2008/424

Mentor:

doc. dr Zoran Čiča

Beograd, Septembar 2016.

# SADRŽAJ


**SADRŽAJ** ..... **2**

**1. UVOD** ..... **3**

**2. KORIŠĆENI ALATI** ..... **4**

2.1. NOTEPAD++ ..... 4
2.2. HTML ..... 4
2.3. CSS ..... 4
2.4. PHP ..... 4
2.5. JAVASCRIPT ..... 5
2.6. WAMPSERVER ..... 5
2.7. MYSQL ..... 5
2.8. PHOTOSHOP ..... 5

**3. PROJEKTNI ZAHTEVI** ..... **6**

3.1. ZAHTEVI ADMINISTRATORA ..... 6
3.2. ZAHTEVI KORISNIKA ..... 6

**4. UPUTSTVO ZA KORIŠĆENJE SAJTA** ..... **7**

4.1. UPUTSTVO ZA ADMINISTRATORA ..... 7
4.2. UPUTSTVO ZA KORISNIKA ..... 8

**5. KOD SAJTA** ..... **10**

5.1. HTML KOD ..... 10
5.2. CSS KOD ..... 13
5.3. PHP KOD ..... 14
*5.3.1. Kreiranje baze* ..... *14*
*5.3.2. Navigacioni bar* ..... *15*
*5.3.3. Manipulacija bazom* ..... *16*
*5.3.4. Kreiranje naloga i kviz* ..... *18*
5.4. JAVASCRIPT KOD ..... 20

**6. ZAKLJUČAK** ..... **21**

**LITERATURA** ..... **22**

# 1. UVOD

Internet je jedan od najvažnijih alata današnjice, sa ogromnim brojem korisnika i kompanija koji imaju potrebu za izradom veb sajtova prilagođenih njihovim potrebama. Posedovati makar i elementarno znanje iz veb programiranja i veb dizajna je izuzetno poželjna veština.

U ovom radu će biti prezentovano u kratkim crtama i sa par primera osnove veb programiranja u HTML (*Hyper Text Markup Language*), CSS (*Cascading Style Sheets*), PHP (*Hypertext Preprocessor*) i JavaScript kodom, kao i rad sa MySQL (*My Structured Query Language*) i Wamp serverom koji služe za rad sa bazom podataka.

Veb sajt koji je izrađen za demonstraciju koda bavi se uvodom u Wi-Fi tehnologiju, ali mogao je biti na bilo koju drugu temu. Sam deo o tome zauzima manji deo programskog koda, jer mnogo veći deo je namenjen funkcionisanju kviza u okviru sajta. O ovome će biti više detalja u delu koji opisuje samu realizaciju koda kviza.

Ono što je takođe važno napomenuti je da je učenje veb programiranja ograničeno najviše poznavanjem engleskog jezika. Na Internetu postoji velik broj tutorijala, primera i foruma na kojima ljudi aktivno razmenjuju znanje, savete i rešenja problema u veb programiranju. Ovo naravno uključuje i materijale za apsolutne početnike koji tek počinju sa radom.

# 2. KORIŠĆENI ALATI

U ovome poglavlju biće dat kratak opis programa i programskih kodova koji se koriste prilikom veb programiranja.

## 2.1. Notepad++

Jedan od najpopularnijih alata za editovanje koji se koristi za veb programiranje, ali i programske jezike drugačijih namena. Notepad++ je besplatan program otvorenog koda koji svako može da preuzme sa **notepad-plus-plus.org** i koristi i u komercijalne i u nekomercijalne svrhe. Notepad++ editor podržava preko 50 programskih jezika i nastao je 2003. godine. Od tada se razvija i dostupan je u velikom broju jezika zahvaljujući korisnicima koji su pomagali u prevodu.

## 2.2. HTML

HTML (*Hyper Text Markup Language*) je osnova svake veb stranice. U njemu se nalazi izgled, struktura i sadržaj samog sajta. HTML omogućava ubacivanje teksta, slika, video snimaka i ostalih sadržaja. On takođe omogućava pozivanje i CSS, PHP i JavaScript koda u okviru stranice.

HTML kod je definisan standardima koje definiše W3C konzorcijum (*World Wide Web Consortium*). Standardi su neophodni da bi kreiran kod bio razumljiv veb pretraživačima koji koriste različiti korisnici.

## 2.3. CSS

CSS (*Cascading Style Sheets*) se koristi da se odredi stil veb stranice. Iako je moguće odraditi ovaj deo posla u HTML kodu, poželjno je premestiti deo koda zadužen za ovo u zaseban fajl. Glavni razlozi su uniformnost izgleda svih stranica na veb sajtu koji kreirate, kao i mogućnost lakše izmene stila stranica. Umesto ulaženja u svaku pojedinačnu stranicu, dovoljno je samo promeniti sadržaj CSS fajla.

## 2.4. PHP

PHP (Hypertext Preprocessor) je programski jezik namenjen pre svega izradi dinamičkog dela veb strane. On predstavlja serversku stranu sajta i omogućava čak i kreiranje HTML strane, pri čemu se ne vidi PHP kod.

Ovaj jezik uglavnom obavlja funkcije u pozadini sajta, od najprostijih (na primer, jednostavne računske operacije) do onih komplikovanijih (baratanje podacima iz baze podataka). Sličan je po sintaksi programskom jeziku C, čak ima i neke iste naredbe.

Kompajliranje koda nije potrebno za PHP, već se koristi interpeter koji izvršava PHP kod svaki put kada ga pozove korisnička strana.

## 2.5. JavaScript

JavaScript je dinamički, netipovan i direktni programski jezik, standardizovan u ECMAScript jezičkim specifikacijama. Njegove funkcije su slične PHP kodu, ali JavaScript se izvršava na korisničkoj strani tj. klijentu.

On sa HTML i CSS predstavlja osnovu za veb programiranje. Nalazi se u svakom većem veb sajtu, a podržan je i od strane svih veb pretraživača kao i velikog broja dodatnih manjih programa za njih. Korisiti se i u ne-veb sredinama, kao recimo u PDF i programima za desktop.

Iako ima sličnosti sa Javom u imenu, sintaksi i standardnoj biblioteci, oni su ipak su dva različita jezika.

## 2.6. WampServer

WampServer je skup više programa u jednom paketu. Njegova skraćenica označava Windows, Apache, MySQL, a P se može odnositi na PHP, Python ili Perl.

WampServer simulira rad Internet veb servera na računaru i omogućava da se kod testira pre nego što se primeni u praksi. U okviru ovog paketa se često nalaze dodatni programi kao phpMyAdmin koji omogućava rad sa bazom podataka preko PHP koda.

## 2.7. MySQL

MySQL (*My Structured Query Language*) je jezik za upite koje upućujemo relacionim bazama podatka. Preko njega nam je omogućen višekorisnički pristup bazi, kao i mogućnost dodele privilegija pristupa podacima (administratori i korisnici).

Pomoću MySQL naredbi možemo ubacivati, brisati i menjati sadržaj podataka u bazi, kao i stvarati nove baze i veze između baza. Ukratko, moguć je visok stepen manipulacije nad podacima, ali i kontrola pristupa u zavisnosti od dodeljenih privilegija.

## 2.8. Photoshop

Iako nije programski jezik, Photoshop ili bilo koji drugi program za crtanje je neophodan pri izradi modernih veb strana. Iako je moguće izraditi veb sajt isključivo od teksta, gotovo svi sajtovi danas u sebi imaju neku sliku.

Te slike je moguće donekle obraditi i u HTML kodu (skaliranje slike i dr.), ali je praktičnije izvršiti bržu obradu u nekom programu za crtanje nego gubiti vreme sa podešavanjem koda. Pored toga Photoshop i slični programi nude mnogo veće i bolje opcije za obradu slika za veb sajt.

# 3.PROJEKTNI ZAHTEVI

Prilikom izrade veb stranice u obzir se mora uzeti više faktora. Funkcionalnost samog koda je najvažnija, ali ništa manje nije važniji korisnički interfejs. Sajt bi trebalo da je pregledan i lepo dizajniran, tako da je prosečnom korisniku odmah jasno gde se šta nalazi na sajtu i šta radi.

Ukoliko sajt treba da sadrži samo neke informacije, njegova izrada je vrlo jednostavna. Nakon što se napravi jedna stranica, dovoljno je samo nju kopirati i promeniti joj sadržaj (slike i tekst u njoj) i brzo ćete imate mnogo stranica uz malo kodovanja. Međutim, ukoliko je potrebno da sajt radi sa ljudima, programiranju se mora pristupiti mnogo ozbiljnije.

Neće svi korisnici sajta biti vešti. Neki će recimo zaboraviti ili prevideti da unesu ime u obrazac kada se prijavljuju na sajt ili će varati na kvizu tako što će se vratiti korak unazad ukoliko pogreše na kvizu. Valjalo bi, u granicama mogućnosti, sprečiti ovo neželjeno ponašanje korisnika. Moguće je opomenuti korisnika da je zaboravio dodati ime u formular, ali gotovo je nemoguće da ga blokirate da se vrati korak unazad u kvizu. Bolje je računati na to da korisnik zbilja želi testirati svoje znanje nego uzaludno potrošiti vreme na veoma komplikovan kod, osim ako se ne pravi kviz koji će se koristiti u procesu ocenjivanja i gde je veoma bitno sprečiti sve vidove varanja.

Pored svega ovoga, postoje i zahtevi u zavisnosti od nivoa privilegija koje su dodeljene korisnicima sajta. Ovde ćemo obraditi zahteve specifično za realizovani sajt koji predstavlja tutorijal za Wi-Fi.

## 3.1. Zahtevi administratora

Administrator sajta je zadužen za bazu podataka sa pitanjima za kviz. On mora imati sledeće mogućnosti:

- da vidi sva pitanja u bazi
- da briše ili menja po želji pojedinačno pitanje
- da dodaje nova pitanja u bazu

Kao i za korisnika, i administratora treba opomenuti u slučaju da je napravio grešku.

## 3.2. Zahtevi korisnika

Obični korisnici sajta moraju imati sledeće mogućnosti i ograničenja:

- mogućnost da naprave nalog na sajtu i da se uloguju
- mogućnost da rade kviz i da im se pamti najbolji rezultat
- da ne vide administratorski deo sajta
- da im se napomene kada nije moguće raditi kviz (npr. kada nema pitanja u bazi)

# 4. UPUTSTVO ZA KORIŠĆENJE SAJTA

Pre nego što se upustimo u sam kod sajta, prvo ćemo posmatrati kako sajt izgleda u akciji. Dok je moguće pokrenuti HTML deo sajta bez dodatnih programa, PHP deo zahteva WAMP server da bi radio.

## 4.1. Uputstvo za administratora

Da bi se razlikovao od običnog korisnika, administratoru se prilikom logovanja dodeljuje tip 'admin' i u navigaciji dobija posebne tabove pomoću kojih može da manipuliše nad bazom. Nije moguće preko sajta stvoriti nalog sa ovom privilegijom već je neophodno direktno pomoću phpMyAdmin u bazi podataka ovo odraditi.

U elektronski priloženom kodu i sadržaju baze već postoji nalog za administratora. E-mail je **admin@admin**, a šifra je **123.** Nakon logovanja administrator vidi sledeće tabove.


**Slika 4.1.1 Tabovi za administratora**

U sekciji **Baza pitanja** administrator može da vidi sva pitanja koje se nalaze u bazi, kao i tačan i netačne odgovore. Ovde ima i mogućnost da obriše ili izmeni svako pitanje.


**Slika 4.1.2 Opcije za brisanje i izmenu pitanja**

U sekciji **Dodavanje pitanja** administrator može da ubaci nova pitanja u bazu. Ako zaboravi da unese neku od stavki u obrazac, sajt će ga opomenuti i zaustaviti dodavanje pitanja.


**Slika 4.1.3 Sekcija za dodavanje novog pitanja**

Po završetku rada administrator se može izlogovati sa sajta pritiskom na odgovarajuće dugme koje prekida trenutnu sesiju.

## 4.2. Uputstvo za korisnika

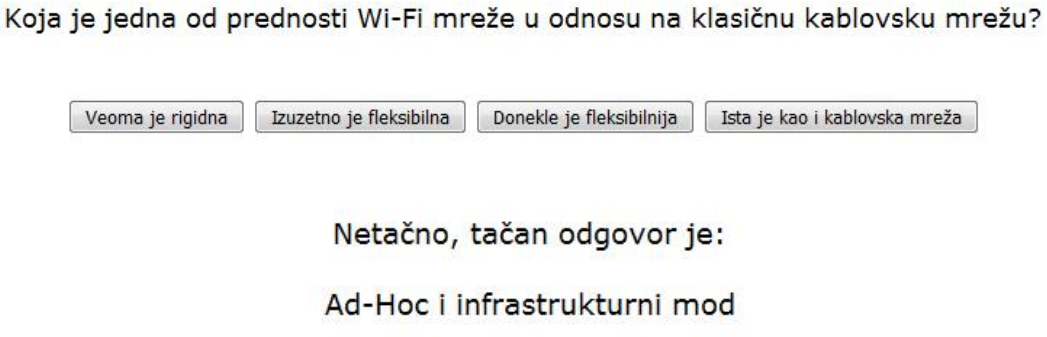
Korisnik sajta koji nema nalog ne mora uopšte biti upoznat sa bilo kakvim posebnim pravilima. Ukoliko pokušava da radi kviz, dobiće obaveštenje da nije ulogovan i link ka stranici gde se može ili ulogovati ili kreirati svoj nalog.


**Slika 4.2.1 Meni za logovanje i registraciju**

Nakon toga kada se vrati na stranicu sa kvizom biće mu ponuđeno da bira na koliko pitanja želi da odgovara.

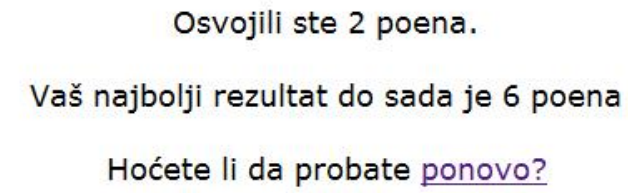

**Slika 4.2.2. Izbor broja pitanja**

Kako bude prolazio kroz kviz dobijaće i obaveštenja da li je tačno ili netačno odgovorio na pitanje, i nakon toga će automatski biti prebačen na sledeće pitanje sve dok ne bude odgovorio na sva pitanja. Kviz svaki put nasumično bira pitanja iz baze i meša tačne i netačne odgovore kada se pokrene kviz, tako da ne postoji mogućnost da svi tačni odgovori budu u istoj koloni.




**Slika 4.2.3. Primer pitanja i neispravnog odgovora**

Na kraju korisnik dobija obaveštenje koliko je tačnih odgovora ima. Pomoću baze podataka vrši se poređenje sa njegovim najboljim rezultatom do tada i ukoliko je bio bolji ovaj put taj rezultat se unosi u bazu.




**Slika 4.2.4. Rezultat kviza**

Po završetku rada korisnik se može izlogovati sa sajta pritiskom na odgovarajuće dugme koje prekida trenutnu sesiju.

# 5. KOD SAJTA

U ovom poglavlju ćemo videti delove koda korišćene za izradu veb sajta. Poseban osvrt će biti na PHP kod jer on obavlja najveći deo funkcija. Videćemo i pozadinsku mapu sajta koja pokazuje i kako pojedini fajlovi utiču na druge.

## 5.1. HTML kod

HTML je osnova svakog veb sajta i sastoji se iz dva glavna dela: glave (*head*) i tela (*body*). Svaki od ovih delova se otvara sa **< >**, a zatvara sa **</ >.** Između ovoga nalazi se glavni kod stranice. Ceo fajl je uokviren sa **<html>** i **</html>** oznakama. U nastavku možemo videti primer koda početne stranice.

```
<!doctype html>
<html>

<head>

<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
<link rel="stylesheet" type="text/css" href="STILOVI/STILSTRANE.css" />

</head>
<body>

<div id="a">
<?php include "navigacija.php"; ?>
</div>

<div id="okvir1">

<div id="b" align="center">
<font size="5px" face="verdana">
<br><br>
  Dobrodošli. <br><br><br>Na ovom sajtu možete naći osnovne informacije u vezi
  funkcionisanja, osobina i istorije Wi-Fi.
  U izvorima imate linkove ka dodatnim sajtovima i materijalima. U sekciji kviz
  možete testirati vaše znanje o Wi-Fi nakon što proučite sajt.
  <br><br><br>
  Želim vam srećan rad.
  <br><br><br><br><br>
  Projekat Ilije Đurića, 2008/0424
</div></div>
</body>
</html>
```

Zaglavlje HTML stranice se sadrži **meta** i **link** podatke. Prvi omogućavaju da HTML kodira sa tekstom u srpskoj latinici, a drugi pozivaju stil stranice preko CSS fajla. U ovom delu se može dodati i JavaScript kod, o čemu će biti reči kasnije,

U telu HTML stranice postoji potreba da podelimo sadržaj na odvojene celine pomoću ** ** naredbe. Ovo nam omogućava veću preglednost koda i omogućava da se celine bolje stilizuju u CSS fajlu. Na početnoj stranici postoje dve glavne celine odvojene: ** ** i ** ** u kojoj se nalazi još jedan ** **, kojem će kasnije biti dodat ** **, ** ** itd.

U ** ** se nalazi pozivanje na PHP kod pomoću naredbe **<?php include "navigacija.php"; ?>** koja poziva zajednički tab sa linkovima kao ostalim stranicama. Iako je moguće uneti u HTML deo stranice sve linkove ka drugim stranicama, mnogo je lakše pozivati navigaciju preko jednog PHP fajla. Ukoliko kasnije postoji potreba da se promeni neki link ili ispravi stranica, jednostavnije je to promeniti samo u jednom zajedničkom fajlu.

U priloženom kodu možemo videti kako je isti kod HTML ponovno korišćen, a samo je proširen sa ** ** i ** ** i promenjen je tekst stranice i dodata slika.

```
<!doctype html>
<html>
<head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
<link rel="stylesheet" type="text/css" href="../STILOVI/STILSTRANE.css" />
</head>
<body>
<div id="a">
<?php include "navigacija.php"; ?>
</div>
<div id="okvir1">
<font size="4px" face="verdana">
<div id="b">
<b>Arhitektura i osnovne komponente</b>
<br><br>
Arhitektura je definisana standardom 802.11 koji opisuje strukturu i
organizaciju mreže.
Ovo omogućava izvršavanje različitih zadatka, kao što je biranje pravilne tačke
pristupa ili moda
rada koji korisniku najviše odgovara.
<br><br>
Osnovne komponente jedne Wi-Fi mreže su bežične stanice (STA). To mogu biti
personalni računari, laptopovi,
telefoni ili neki drugi uređaji. STA ima pristup mreži preko pristupne tačke
(AP).
Kada su dve STA spojene bežično, one formiraju bazični servisni set (BSS). Ovo
je najosnovnija jedinica Wi-Fi mreže.
<br>
<img align="left" src="slika1.png">
</div>
<br><br>
<div id="c">
<b>Modovi rada</b>
<br><br>
Postoje dve operativna moda opisana standardom IEEE 802.11: Ad-Hoc mod i
infrastrukturni mod.
Oba koriste BSS, ali daju drugačije mrežne topologije. Sve stanice moraju
izabrati mod da bi se mogla stvoriti mreža.
<br><br>
Ad-Hoc je najprostiji tip mreže. Stanice međusobno komuniciraju, čak i kada nisu
u dometu komunikacija će se odvijati
preko posrednika. Pošto se ovaj sistem može instalirati bilo gde, vrlo je
koristan u situacijama kada je neophodno
```

```
brzo podizanje mreže.
<br>
<img align="left" src="slika2.png">
      </div>
<div id="d">
<br>
Infrastrukturni mod zahteva da BSS ima jedan bežični AP. Glavna uloga njega je
da proširi pristup fiksnoj mreži
klijentima bežične mreže. Svi oni moraju da izvrše asocijaciju sa distributivnim
sistemom (DS) da bi dobili pristup
mrežnim resursima.
<br>
<img align="left" src="slika3.png"></div>
</font>
      </div>
```

Na slici 5.1.1 možemo videti kako kod iskorišćen dva puta sa manjim izmenama izgleda. Odlaskom na sajt i prolaskom kroz sve stranice, kao i posmatranjem njihovog koda, može se videti velika sličnost u HTML delu koda.


**Slika 5.1.1 Rezultat HTML kodiranja**

## 5.2. CSS kod

Kao što je rečeno u poglavlju 5.1 o HTML kodu, postoji potreba da izdvojimo stil stranice iz koda same stranice u poseban fajl. Želja da sajt bude stilski uniforman je važna zbog dobrog izgleda, kao i zbog lakšeg programiranja u budućnosti. Na delu koda datom u nastavku možemo videti kako svaki je deo stranice zasebno stilizovan.

**body** sadrži najopštije informacije o strani i bez **div** njegov stil će dominirati.

```
body{
      font-family:Comic Sans;
      font-size:10pt;
      background-color:grey;
}
```

**#a** sadrži podatke o ** ** iz HTML koda (boju, veličinu, poziciju itd).

```
#a{
    vertical-align:top;
    width:93%;
    position:center;
    margin:0 auto;
    z-index: 10;
    background-color:black;
  }
```

Deo **nav** sadrži ceo niz naredbi o stilu navigacije. Pošto je prilično obiman kod, ovde je prikazan samo deo koda.

```
nav ul ul {
          display: none;
          vertical-align:center;
          z-index: 10;
         }

nav ul li:hover > ul {
          display: block;
                 }

nav ul {
          background: black;
          padding: 0 17px;
          list-style: none;
          position: relative;
          display: inline-table;
           }
```

U nastavku možete videti i kod za ** **, a za ** ** i ostale **div** možete videti kod u fajlu **STILSTRANE.php**.

```
#okvir1{
        vertical-align:top; width:93%;
        height:800px; position:relative;
        margin:0 auto;  background-color:white;
        border-top: 10px;
        border-top-color: white;
       }
```

## 5.3. PHP kod

Ovo je najkompleksniji deo koda sajta jer obavlja mnogo funkcija:

- kreira bazu
- stvara navigacione tabove u zavisnosti od toga ko je ulogovan
- omogućava administratoru da dodaje, menja i briše pitanja u bazi
- omogućava korisniku da napravi nalog i da radi kviz

Da bi ovo bilo moguće kodirati, nije dovoljno samo dobro poznavanje sintakse programerskog jezika, već i dobra logika. Ne postoje naredbe koje mogu sve da reše, već pri izradi koda mora da se smisle i originalna rešenja. Ovo ćemo posebno videti malo kasnije na primeru kako sajt bira nasumična pitanja za korisnika i kako ih pamti dok korisnik ne odgovori na svako pitanje.

### *5.3.1. Kreiranje baze*

Pokretanjem fajla **kreiranjeBaze.php** se stvara prazna baza podataka. U nastavku možemo videti i kod koji to radi.

**$sql** naredbe brišu postojeću bazu ako postoji i stvaraju praznu bazu **kviz.**

```
<?php
$veza=mysqli_connect ('localhost','root','');
if ($veza){echo "uspeh";} $sql="DROP DATABASE IF EXISTS kviz";
if (mysqli_query($veza,$sql)) {echo "uspeh";};
$sql="CREATE DATABASE IF NOT EXISTS kviz DEFAULT CHARACTER SET utf8
COLLATE utf8_general_ci";
mysqli_query($veza,$sql);
```

Sa **EOT CREATE TABLE** naredbom se stvara prazna tabela za podatke.

```
$sql = <<<EOT CREATE TABLE korisnici (
```

Tabela se puni redovima koji će sadržati imena, prezimena i ostale informacije kao i važnu **id** kolonu koja će sadržati samopovećavujuću vrednost.

```
id_korisnika INT(9) UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
 me VARCHAR(20) CHARACTER SET utf8  NOT NULL,
prezime VARCHAR(20) CHARACTER SET utf8 NOT NULL,
email VARCHAR(50) CHARACTER SET utf8  NOT NULL,
lozinka VARCHAR(20) CHARACTER SET utf8 NOT NULL,
bestscore INT,
temp INT,
tip VARCHAR(100) CHARACTER SET utf8  NOT NULL)
EOT;
```

Sa **mysqli_select_db** i **mysqli_query** stvorene tabele se ubacuju u bazu.

```
mysqli_select_db($veza,'kviz'); mysqli_query($veza,$sql);
```

Kreiranoj bazi se može pristupiti i direktno iz **phpMyAdmin** gde imamo mogućnost da ubacujemo i brišemo baze, tabele i kolone po želji, kao i da manipulišemo sadržajem svih njih. Malo kasnije biće i reči o tome kako administrator preko sajta menja sadržaj baze, kao i kako korisnici kreiraju svoje naloge u bazi.

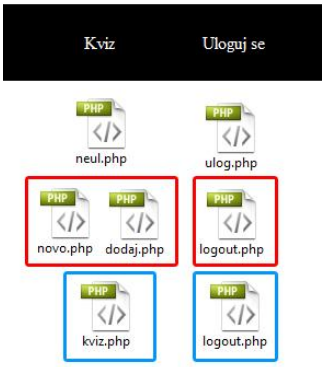
### *5.3.2. Navigacioni bar*

Izdvojen u zaseban PHP fajl, navigacioni bar vrši nekoliko funkcija, od kojih je najbitnija da prikazuje linkove ka stranicama u zavisnosti ko je ulogovan.

Ispod svih **<ul>** i **<li>** delova koda koji su po prirodi HTML kod i koji omogućuju odlazak na ostale stranice nalazi se PHP deo koda. Sa **session_start()** on započinje sesiju i proverava da li uopšte postoji ulogovani korisnik sa **isset($_SESSION['tip'])**.

```
<li><a> Istorija </a>
    <ul>
      <li><a href="poce.php"> Početak </a></li>
      <li><a href="rast.php"> Rast </a></li>
      <li><a href="dana.php"> Danas </a></li>
    </ul>
   </li>
  <li><a href="stan.php"> Standardi </a></li>
  <li><a href="izvo.php"> Izvori </a></li>
  <?php
     session_start();
     if (isset($_SESSION['tip']))
        {  if($_SESSION['tip']=='admin'){
           echo '<li><a> Kviz </a>  <ul>
                <li><a href="prom.php"> Baza pitanja </a></li>
                <li><a href="novo.php"> Dodavanje pitanja </a></li>
           </ul>  </li>  <li><a href="logout.php"> Izloguj se </a></li>';
  }else if($_SESSION['tip']=='user'){
    echo     '<li><a href="kviz.php"> Kviz </a></li>
              <li><a href="logout.php"> Izloguj se </a></li>'; }
                    } else { echo
                              '<li><a href="neul.php"> Kviz </a></li>
                             <li><a href="ulog.php"> Uloguj se </a></li>';  }
```

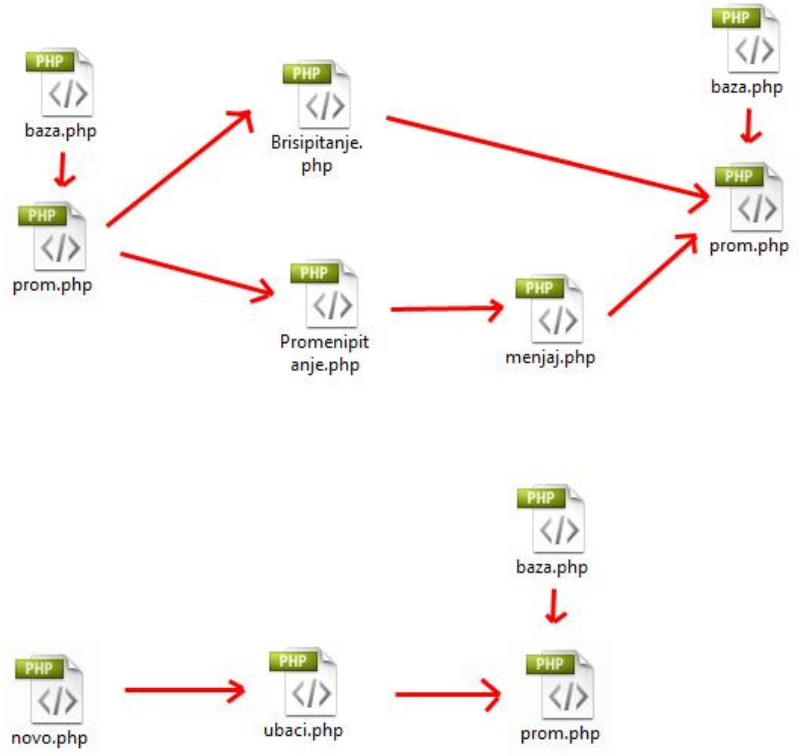
Ukoliko se niko nije ulogovao na sajt, navigacioni bar će dati link ka stranici za logovanje pomoću **echo** naredbe. Ako se neko već ulogovao, vratiće linkove ka odgovarajućim stranicama u zavisnosti da li ulogovan administrator (srednje crvene kockice) ili korisnik (donje plave kockice).




**Slika 5.3.2.1 Navigacioni bar u zavisnosti od logovanja**

### *5.3.3. Manipulacija bazom*

Na slici 5.3.3.1 možemo videti kompleksnost koja je potrebna da bi administrator imao mogućnosti da uređuje pitanja u bazi podataka. Gornji dijagram prikazuje koji se sve PHP pozivaju kada treba da se obriše ili promeni pitanje, dok donji dijagram prikazuje šta se koristi pri unošenju novog pitanja u bazu.




**Slika 5.3.3.1 PHP fajlovi uključeni u rad sa podacima u bazi u slučaju administratora**

U ovom delu ćemo razmotriti korak po korak kako **baza.php** uzima podatke iz baze podataka i vraća ih na HTML stranicu, kao i kako pravi linkove napravljene da brišu ili menjaju svako pitanje pojedinačno.

Prvi deo uspostavlja vezu sa bazom pomoću **$veza** i govori koji sve podaci su potrebni administratoru pomoću **$upit**: pitanje, tačan odgovor, opcione odgovore, kao i identifikacioni broj svakog pitanja, koji je unikatan i koji će nam biti neophodan svuda.

```
<?php

$veza=new mysqli('localhost','root','');
mysqli_set_charset($veza,"utf8");
$upit="SELECT id_pitanja, pitanje, odgovor, opcija1, opcija2, opcija3 FROM
kviz.pitanja";
$rez=$veza->query($upit);
```

Nakon što je dobio naredbu šta da uzima pomoću **$rez**, počinjemo da sa **fetch_assoc()** uzimamo i stavljamo podatke u **$row** sve dok ima podataka u bazi da se uzmu.

```
if ($rez->num_rows > 0) {
    while($row = $rez->fetch_assoc())
```

Sada kada imamo aktuelne podatke iz baze možemo da ispišemo šta je potrebno administratoru. Pre svega mu prikazujemo pitanja i odgovore iz baze:

```
   {    echo '<br>';
        echo $row['pitanje'].'<br>Tačan odgovor: '.$row['odgovor'].'<br>Netačni
odgovori:<br>1. '.$row['opcija1'].'<br>2. '.$row['opcija2'].'<br>3.
'.$row['opcija3'];
            echo '<br>';
```

Potom stvaramo i linkove koji su potrebni za brisanje i modifikaciju pitanja. Dodavanjem nastavka **?ID='.$row["id_pitanja"].'** na kraj linka dobićemo da prenosimo jedinstveni identifikacioni broj pitanja u PHP formular, što će nam omogućiti da pomoću njega specifično obrišemo ili izmenimo željeno pitanje:

```
            echo ' <a href="Brisipitanje.php?ID='.$row["id_pitanja"].'">
Izbrišite pitanje </a>';
            echo '<br>';
        echo ' <a href="Promenipitanje.php?ID='.$row["id_pitanja"].'"> Promenite
pitanje </a>';
            echo '<br>';
```

Na kraju je neophodno i napomenuti administratoru ukoliko je baza prazna, što činimo jednom **else** naredbom.

```
}} else {
echo 'U bazi trenutno ne postoje pitanja';}

?>
```

Logika i u ostalim PHP fajlovima je slična. Potrebno je ostvariti vezu sa bazom, poslati joj upit i najčešće ili naći red sa određenim identifikacionim brojem ili češljati celu bazu i vraćati podatke iz nje. Ovo su primeri nekih od upita koji se koriste:

**- $sql = "DELETE FROM kviz.pitanja WHERE id_pitanja=$IDpitanja";**

briše pitanje iz baze, sa sve odgovorima.

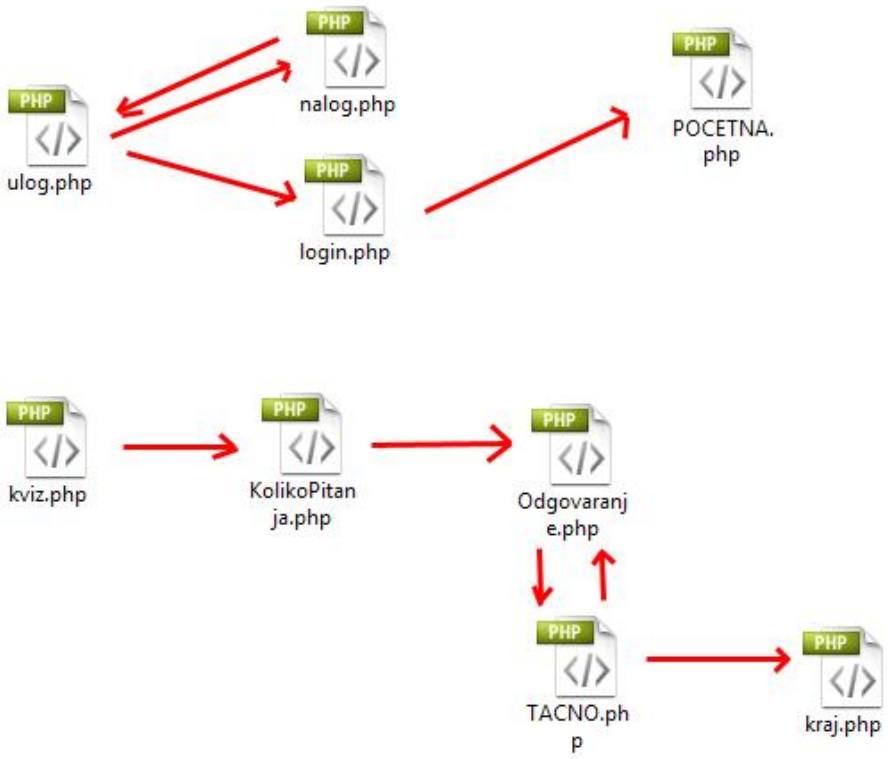
**- $upit1="INSERT INTO kviz.pitanja(id_pitanja, pitanje) VALUES ('$id_pitanja','$pitanje') ON DUPLICATE KEY UPDATE pitanje='$pitanje'";**

specifično menja tekst samog pitanja u bazi, a ne menja ništa drugo.

### *5.3.4. Kreiranje naloga i kviz*

Slično kao i za administratora, i za korisnika je potrebno više fajlova da bi mu bilo omogućeno da radi ono što mu je potrebno. Gornji dijagram prikazuje proceduru koja se dešava kada korisnik kreira nalog i kada se uloguje, a donji dijagram prikazuje kako funkcioniše pokretanje samog kviza (slika 5.3.4.1).

Može se primetiti postojanje petlji u oba slučaja. Prva petlja vraća korisnika na logovanje nakon što je kreirao nalog, a druga petlja se vrti između postavljanja pitanja i provere odgovora sve dok korisnik ne odgovori na poslednje pitanje.




**Slika 5.3.4.1 PHP fajlovi uključeni u rad sa podacima u bazi u slučaju korisnika**

Mi ćemo se više posvetiti PHP fajlu **KolikoPitanja.php** koji je mnogo važniji jer on sastavlja nasumičan niz pitanja koje korisnik odgovara. Na početku se započinje sesija. Ovde je to neophodno jer **KolikoPitanja.php** ne sadrži u sebi navigacioni tab. Nakon toga gledamo na koliko je pitanja korisnik hteo da odgovara.

```
<?php
session_start();
if (isset($_POST['Tri'])) {$BR=3;};
if (isset($_POST['Pet'])) {$BR=5;};
if (isset($_POST['Sedam'])) {$BR=7;};
if (isset($_POST['Deset'])) {$BR=10;};
if (!isset($_SESSION['BROJ']) && empty($_SESSION['BROJ'])) $BROJ='';
```

Slično kao i ranije, uspostavljamo vezu sa bazom i uzimamo identifikacione brojeve pitanja, ali ih uzimamo nasumično pomoću **RAND()** naredbe i ograničavamo na broj pitanja na koja korisnik želi da odgovara pomoću **LIMIT $BR**.

```
$conn = mysqli_connect ('localhost','root','');
$sql="SELECT id_pitanja FROM kviz.pitanja ORDER BY RAND() LIMIT $BR";
$result = $conn->query($sql);
```

Ovde je isto važno videti da li u bazi uopšte ima dovoljno pitanja na koje korisnik želi da odgovara. Ukoliko nema, dobija poruku da pokuša kviz sa manje pitanja.

```
if ($result->num_rows < $BR) {   echo    '<br>';     echo    '<br>';     echo
'<br>';     echo    '<br>';
                                    echo 'U bazi nema dovoljno pitanja za vaš
test. <br>Molimo vas odaberite manje pitanja.';
                                header("refresh:4; url=kviz.php"); }
```

Ukoliko ima dovoljno pitanja u bazi **KolikoPitanja.php** prelazi na sledeći važan korak. Potrebno je da se nekako prenese ostalim PHP fajlovima na koja će pitanja korisnik da odgovara. Ovde je moguće više rešenja, od pravljenja nove baze koja će držati nasumično odabrana pitanja do pravljenja niza pitanja koji će se prosleđivati.

U našem slučaju mnogo je jednostavije da postoji samo jedan string koji se sastoji od niza brojeva razdvojenih jednim znakom (u ovom slučaju donja crta _) koji će prosleđivati kao **$_SESSION['BROJ']** i izgledaće recimo ovako: 3_5_7_2.

```
else {if ($result->num_rows > 0)
{
    while($row = $result->fetch_assoc()) {

                if ($BROJ=='') {$BROJ=$row["id_pitanja"];}
else
                        {   $temp=$BROJ;
                        $BROJ=$temp.'_'.$row["id_pitanja"];}
                                                }
} else {
    echo "0 results";
}
$_SESSION['BROJ']=$BROJ;
header("Location: Odgovaranje.php");    }
?>
```

Ostali PHP fajlovi onda mogu da koriste ovaj string koji će posle razbijati naredbom **explode** i praviti novi niz 3,5,7,2. Pomoću njega će prvo korisniku dati da odgovara na pitanje broj 3, a od ostalih brojeva će napraviti novi niz 5_7_2 koji će postati vrednost u **$_SESSION['BROJ']**. Kada korisnik bude odgovrio na pitanje broj 3, čiji su odgovori promešani, PHP fajl će opet koristiti identifikacioni broj da uporedi odgovor korisnika sa tačnim odgovorom pitanja u bazi.

Na ovaj način prilično jednostavno možemo omogućiti da korisnik radi kviz i bez mnogo komplikacija možemo proveravati da li je tačno odgovorio.

## 5.4. JavaScript kod

Na sajtu JavaScript je zadužen samo za jednu funkciju, ali on ima još mnogo primena. Za potrebe sajta realizovanog u okviru ove teze specifično je zadužen da proveri da li su administrator prilikom unosa pitanja ili korisnik prilikom stvaranja naloga neko polje ostavili prazno.

JavaScript kod je izmešten u zaglavlje HTML koda i on se poziva kao funkcija nakon popunjavanja formulara. Uzima vrednost svakog od polja i proverava da li je prazno.

```
<head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
<link rel="stylesheet" type="text/css" href="../STILOVI/STILSTRANE.css" />
<script>
function validateForm() {
    var x = document.forms["myForm"]["ime"].value;
    var y = document.forms["myForm"]["prezime"].value;
    var z = document.forms["myForm"]["email"].value;
    var k = document.forms["myForm"]["lozinka"].value;
    if (k == null || k == "" || x == null || x == "" || y == null || y == "" ||
z == null || z == ""  )
        {
        if ( x == null || x == "") alert("Niste uneli ime");
        if ( y == null || y == "") alert("Niste uneli prezime");
        if ( z == null || z == "") alert("Niste uneli email");
        if ( k == null || k == "") alert("Niste uneli lozinku");
        return false;    }   }
```

Kada se pokrene neki od formulara za unos, on će prvo pozvati JavaScript kod pomoću **onsubmit** naredbe.

```
<form name="myForm" action="Nalog.php" onsubmit="return validateForm()"
method="post">
Ime <input type="text" name="ime" /> <br />
Prezime <input type="text" name="prezime" /> <br />
E-mail <input type="text" name="email" /> <br />
Lozinka <input type="text" name="lozinka" /> <br />
<input type="submit" name="upis" value="Prijavite se" />
      </form>
```

Ukoliko je korisnik ili administrator neko polje ostavio prazno dobiće obaveštenje da je načinio grešku i biće onemogućen da nastavi rad dok je ne ispravi.


**Slika 5.4.1 Generisanje obaveštenja**

# 6. ZAKLJUČAK

Iako deluje komplikovano na prvi pogled, kodiranje veb stranice sa vremenom i praksom postaje lakše. Od prvih koraka sa najprostijim HTML kodom lako se dođe do osnovna PHP programiranja, a odatle sledi put do kompleksnijih stvari.

Prilikom kodiranja, neizostavna je pomoć drugih ljudi na Internetu i poznavanje engleskog. Svaki programerski problem može biti rastavljen na manje celine koje je lakše rešiti, a i ako se zapne na nekima od njih, velika je verovatnoća da se neko već mučio sa sličnim problemom, i da je rešenje objavljeno na Internetu.

U ovom radu je realizovan kviz koji omogućava kvalitetno proveravanje znanja tako što se nasumično biraju pitanja, a sami odgovori nikad nisu istog redosleda. Kao sledeći korak u usavršavanju kviza bi bilo uzimanje u obzir aspekat bezbednosti sa stanovišta sprečavanja varanja na samom kvizu. Napomenimo da realizovani kod za kviz može da se primeni za bilo koju oblast, a u ovoj tezi je izabran WiFi.

# LITERATURA


[1] Slajdovi Dr Aleksandre Smiljanić: http://home.etf.rs/~aleksandra/IP.html

[2] Tutorijali sa sajta W3Schools: http://www.w3schools.com/

[3] Postovi sa sajta stackoverflow: http://stackoverflow.com

[4] Tutorijali sa sajta php.net: http://php.net

[5] http://ftp1.digi.com/support/documentation/0190170_b.pdf

[6] https://en.wikipedia.org/wiki/Wi-Fi

[7] https://en.wikipedia.org/wiki/IEEE_802.11

[8] https://en.wikipedia.org/wiki/Wi-Fi_Alliance

[9] https://en.wikipedia.org/wiki/Institute_of_Electrical_and_Electronics_Engineers

[10] https://www.tnooz.com/article/global-wifi-growth-track-one-hotspot-every-eight-people

[11] http://www.flashrouters.com/blog/2015/08/07/what-is-wi-fi-router-wireless-router-faq

[12] https://www.ipass.com/press-releases/the-global-public-wi-fi-network-grows-to-50-million-worldwide-wi-fi-hotspots

[13] http://www.economist.com/node/2724397