Izrada web aplikacije za analizu rezultata usporedbenih mjerenja etalona duljine

Braun, Teo

Undergraduate thesis / Završni rad

2015

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Mechanical Engineering and Naval Architecture / Sveučilište u Zagrebu, Fakultet strojarstva i brodogradnje**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:235:911505

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-25**

Repository / Repozitorij:

Repository of Faculty of Mechanical Engineering and Naval Architecture University of Zagreb

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

ZAVRŠNI RAD

Teo Braun

Zagreb, 2015.

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

ZAVRŠNI RAD

Mentor:

Prof. dr. sc. Dubravko Majetić

Student:

Teo Braun

Zagreb, 2015.

Izjavljujem da sam ovaj rad izradio samostalno koristeći stečena znanja tijekom studija i navedenu literaturu.

Zahvaljujem se svima koji su mi pomogli oko završnog rada. Prvenstveno svom mentoru, prof. dr. sc. Dubravku Majetiću, njegom razumijevanju i strpljivosti, te svima sa Katedre za mjerenje i kontrolu posebice prof. dr. sc. Biserki Runje, dr.sc. Vedranu Šimunoviću i Tomislavu Habeku koji su mi pomagali u izradi web aplikacije sa svojom stručnošću i susretljivošću.

Teo Braun

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE
Središnje povjerenstvo za završne i diplomske ispite
Povjerenstvo za završne ispite studija strojarstva za smjerove:
proizvodno inženjerstvo, računalno inženjerstvo, industrijsko inženjerstvo i menadžment, inženjerstvo materijala i mehatronika i robotika

Sveučilište u Zagrebu Fakultet strojarstva i brodogradnje	
Datum	Prilog
Klasa:	
Ur.broj:	

ZAVRŠNI ZADATAK

Student: **Teo Braun** Mat. br.: 0035177714

Naslov rada na hrvatskom jeziku: **Izrada web aplikacije za analizu rezultata usporedbenih mjerenja etalona duljine**

Naslov rada na engleskom jeziku: **Web Application for Analysis of Length Standards Measurement Results Obtained by Differential Method**

Opis zadatka:

U postupku mjerenja duljine etalona usporedbenom metodom duljina umjeravanog etalona određena je mjerenjem razlike između referentnog i umjeravanog etalona iste nazivne duljine. U tu se svrhu u Laboratoriju za precizna mjerenja dužina FSB-a koristi elektronički komparator. Dobiveni rezultati mjerenja prikazuju se u digitalnom obliku na ekranu elektronskog komparatora i ručno upisuju u Radnu podlogu. Potrebna analiza rezultata uključuje izračune statističkih veličina koje opisuju centralnu tendenciju i rasipanje rezultata mjerenja, izračune varijacije duljine te potrebne korekcije uslijed utjecaja temperature i mjerne sile. Iz svega proizlazi potreba za izradom internetske aplikacije za pohranu, obradu i analizu rezultata usporedbenih mjerenja etalona duljine.

U radu treba načiniti slijedeće:

1. Razviti sučelje za unos podataka.
2. Izgraditi bazu podataka sukladnu postojećoj Radnoj podlozi.
3. Ugraditi algoritme za statističku obradu podataka.
4. Generirati dokument potvrde o umjeravanju.
5. Razviti preglednik izvršenih mjerenja.
6. Tijekom izrade programske podrške koristiti MySql i PHP.
7. Izvesti zaključke rada.

Zadatak zadan:
25. studenog 2014.

Rok predaje rada:
1. rok: 26. veljače 2015.
2. rok: 17. rujna 2015.

Predviđeni datumi obrane:
1. rok: 2., 3., i 4 . ožujka 2015.
2. rok: 21., 22., i 23. rujna 2015.

Zadatak zadao:

Prof. dr. sc. Dubravko Majetić

Predsjednik Povjerenstva:

Prof. dr. sc. Zoran Kunica

SAŽETAK

Web stranice postoje već 25. godina. U ovom vremenu su iz običnih tekstualnih dokumenata s poveznicama sazrile u web aplikacije koje mogu apsolutno sve što i obične aplikacije. Za razliku od običnih aplikacija nije im potrebna instalacija, dostupna je na svakom uređaju sa web preglednikom i uvijek možete biti sigurni da vaši korisnici koriste najnoviju verziju.

U sklopu završnog rada je moderniziran posao koji se gotovo isto radio još od 1896. godine kada je Carl Edvard Johansson napravio prvu garnituru etalona duljine No1 – Mjerenje planparalelnih graničnih mjerki.

Kako ovaj posao zahtjeva ispunjavanje puno različitih obrazaca te računanje dosta veličina ovisno o vrsti etalona, logično je da se posao može olakšati digitalizacijom.
Ovaj problem riješen je izradom web stranice korištenjem skupom web alata: HTML, CSS, JavaScript, Bootstrap, HTML DOM, PHP, MySQL, PHP PDO, AJAX, FPDF i FPDI.

Kao rezultat imamo modernu interaktivnu web stranicu koja komunicira sa bazom podataka, automatski računa potrebna polja, izvlači podatke iz DIN 861 standarda, te nudi mogućnosti lakog ažuriranja podataka i konačno generiranje PDF certifikata.

Ključne riječi:
Etalon duljine, Internet, web programiranje, web aplikacija, baza podataka, HTML, CSS, PHP, MySQL, JavaScript, PDO, DOM, AJAX, Bootstrap, PDF

ABSTRACT

Web sites have existed for 25 years. In this time period we have seen them mature from simple text documents with hyperlinks to complex web applications that have all the power and potential of regular platform based applications. There are many advantages in using web applications over platform based ones: web applications don't have to installed, they can be run on any device that has a web browser installed and the end-user is always using the latest version of your web application without the need to update.

As part of this bachelor's thesis we have brought a 100 year old job to the digital age. A job that has not changed since its introduction in 1896 when Carl Edvard Johansson first made the Gauge block set No 1 – Measuring gauge blocks.

This job requires filling out endless amounts of measurement data into forms followed by tedious calculations. The amount of forms to fill will vary by the type of gauge block being measured. It's natural to optimize this kind of work by bringing it to the digital age. This problem has been solved by making a website using the following web technologies: HTML, CSS, JavaScript, Bootstrap, HTML DOM, PHP, MySQL, PHP PDO, AJAX, FPDF and FPDI.

As a result we have built a modern interactive website that can communicate and relay information to a database, easily update data, do instant calculations, cross-reference data according to DIN 861 standards and generate PDF certificates.

Keywords:

Gauge block, Internet, web programming, web application, database, HTML, CSS, PHP, MySQL, JavaScript, PDO, DOM, AJAX, Bootstrap, PDF

POPIS SLIKA

Slika 2.1 Tablica tolerancija za etalone duljine po DIN 861 2
Slika 2.2 Shematski prikaz elektronskog komparatora 3
Slika 2.3 Mjerne točke etalona klase 00 i 0 4
Slika 3.1 Server Klijent - prikaz jednostavne internetske mreže 7
Slika 4.1 Bez(lijevo) i sa Bootstrap-om(desno) 13
Slika 4.2 Prikaz prilagodljivog kostura Bootstrap radnog okruženja 14
Slika 5.1 Nazivni obrazac radne podloge za umjeravanje etalona duljine 23
Slika 5.2 Obrazac za rezultate mjerenja etalona duljine klase 00 i 0 24
Slika 5.3 Obrazac za rezultate mjerenja etalona duljine klase 1 i 2 24
Slika 5.4 phpMyAdmin sučelje 26
Slika 5.5 Prikaz strukture podataka tablice *nazivna, kalsa12* i *klasa000* 27
Slika 5.6 Prikaz strukture podataka tablice *dopusteno* i *user* 28
Slika 6.1 Sučelje za unos podataka nazivne stranice 31
Slika 6.2 Sučelje za unos mjerenja etalona duljine klase 1 i 2 32
Slika 6.3 Sučelje za unos mjerenja etalona duljine klase 00 i 0 33
Slika 8.1 Prikaz pregleda svih podataka iz tablice *nazivna*, sortirane silazno po vremenu kreiranja 40
Slika 9.1 Implementacija kalendara pomoću bootstrap-datepicker biblioteke 43
Slika 9.2 Upozorenje uslijed prevelikog odstupanja temperatura 46

POPIS TABLICA

Tablica 4.1 Klase prilagodljivost kostura ($-broj stupaca) 14
Tablica 5.1 Tipovi podataka korišteni u bazi 26
Tablica 7.1 Navigacija URL adresom kroz web stranicu 36

POPIS OZNAKA

Oznaka	Jedinica	Opis
ln	mm	Nazivna duljina etalona
δLi	µm	izmjerena razlika duljine umjeravanog i referentnog etalona
$\overline{\delta Li}$	µm	srednja vrijednost izmjerenih razlika etalona
ΔL_s	µm	odstupanje mjere iz certifikata etalona
ΔL_c	µm	odstupanje duljine u sredini etalona
te	µm	dopušteno odstupanje duljine u sredini po DIN 861
tv	µm	maksimalno dopuštena varijabilnost po DIN 861
fv	µm	najveći raspon dopuštenih odstupanja duljine kod etalona klase 00 i 0

SADRŽAJ

1 Uvod 1

2 Etaloni duljine 2

2.1 Opće informacije 2

2.2 Usporedbena metoda mjerenja kratkih etalona duljine 3

2.3 Radna podloga za umjeravanje planparalelnih graničnih mjerki usporednom metodom . 4

3 Internet 6

3.1 Klijent 6

3.2 Server 6

3.3 HTTP protokol 7

4 Web alati 9

4.1 HTML 9

4.2 CSS 11

4.3 JavaScript 12

4.3.1 HTML DOM 12

4.4 Bootstrap 13

4.5 PHP 15

4.5.1 POST i GET 16

4.5.1.1 GET 16

4.5.1.2 POST 17

4.6 MySQL 17

4.6.1 PHP PDO 19

4.7 AJAX 20

4.8 FPDF 21

4.9 FPDI 22

5 Kreiranje baza podataka 23

5.1 Papirnati obrasci 23

5.2 Izrada baze podataka 25

6 Izrada grafičkog korisničkog sučelja 30

7 Navigacija 34

8 Korištenje baze podataka 37

8.1 Spajanje na bazu podataka 37

8.2 Spremanje u bazu podataka 37

8.3 Pregled baze podataka 39

8.4 Ažuriranje baze podataka 40

8.5 Brisanje baze podataka 41

9 Automatizacija unosa podataka 42

9.1 Kalendar 42

9.2 Izračuni 43

9.3 Dohvaćanje tablice tolerancija DIN 861 44

9.4 Izbor mjeritelja i provjeravača 45

9.5 Upozorenje 46

9.6 Zaštita 47

10 Kreiranje PDF-a 48

11 Zaključak 49

Literatura 50

Resursi 50

1 UVOD

Etaloni duljine su referentna tvar ili mjerni sustav namijenjen za određivanje, ostvarivanje, čuvanje ili obnavljanje mjernih jedinica. Najčešće se koriste za umjeravanje mjernih uređaja unutar poduzeća, tvornica, proizvodnih pogona, te svima kojima su precizne veličine bitne. Bez postojanja etalona duljine, različita poduzeća koja zajedno rade na proizvodu, zbog razlika u mjerama, ne bi mogla surađivati jer im dosjedi ne bi odgovarali.

Kako bi etaloni duljine bili referentni moraju se umjeravati od strane ovlaštenog laboratorija za mjerenja. Umjeravanje se provodi usporednom metodom referentnog i umjeravanog etalona.

Ova metoda je ostala gotovo nepromijenjena od same pojave prve garniture etalona duljine No1 koju je napravio Šveđanin Carl Edvard Johansson 1896. godine. Jedina modernizacija od tada do danas je korištenje elektronskog komparatora, dok je količina potrebne papirologije ostala nepromijenjena. S obzirom da je papirologija još uvijek ručno pisana, repetitivna i dugotrajna, te često zahtjeva prepisivanje u računalo, optimizacija digitalizacijom je vrlo poželjna, što je ujedno i tema ovog rada.

Kroz daljnja poglavlja objašnjeni su razni web alati potrebni za izradu web stranica, praćeno izradom same web stranice koja zamjenjuje svu papirologiju vezanu uz umjeravanje dužine etalona.

2 ETALONI DULJINE

2.1 Opće informacije

Etaloni duljine [1], u smislu utjelovljenja dužinske mjere između krajnjih površina, postojali su već i kod starih kulturnih naroda. U današnjem obliku ti su se etaloni počeli razvijat i primjenjivati krajem devetnaestog stoljeća, pod nazivom planparalelne granične mjerke i prihvaćeni su kao primjenjiv materijaliziran etalon duljine širom svijeta.

Danas su etaloni duljine normirani. U Europi je prihvaćena i trenutno važeća norma ISO 3650: Length standards-Gauge blocks iz 1998. godine, dok u Americi postoje dvije norme koje se odnose na normizaciju etalona duljine: Federal Specification GGG-G-15C i ANSI/ASME B89.1.9.M. Razlike među njima su gotovo zanemarive. Normama ISO 3650 i ANSI/ASME B89.1.9.M propisane su osnovne definicije i zahtjevi na kvalitetu mjernih površina, te zahtjevi na oblik i dimenzije etalona.

U ovom radu će se koristiti njemačka norma DIN 861. koja je nastala prije ali je kompatibilna sa standardom ISO 3650. Prema normi DIN 861 etaloni duljine su bili gradirani u sljedeće klase točnosti: 00, 0, 1 i 2. (od najfinije do najgrublje).

(correspond with ISO 3650 and SS 3348)

Nominal length mm	Degree of precision 00		Degree of precision 0		Degree of precision 1		Degree of precision 2	
	Length tolerance *) µm	Max length variation width**) µm	Length tolerance *) µm	Max length variation width**) µm	Length tolerance *) µm	Max length variation width**) µm	Length tolerance *) µm	Max length variation width**) µm
-10	±0.06	0.05	±0.12	0.10	±0.20	0.16	±0.45	0.30
(10)-25	±0.07	0.05	±0.14	0.10	±0.30	0.16	±0.60	0.30
(25)-50	±0.10	0.06	±0.20	0.10	±0.40	0.18	±0.80	0.30
(50)-75	±0.12	0.06	±0.25	0.12	±0.50	0.18	±1.00	0.35
(75)-100	±0.14	0.07	±0.30	0.12	±0.60	0.20	±1.20	0.35
(100)-150	±0.20	0.08	±0.40	0.14	±0.80	0.20	±1.60	0.40
(150)-200	±0.25	0.09	±0.50	0.16	±1.00	0.25	±2.00	0.40
(200)-250	±0.30	0.10	±0.60	0.16	±1.20	0.25	±2.40	0.45
(250)-300	±0.35	0.10	±0.70	0.18	±1.40	0.25	±2.80	0.50
(300)-400	±0.45	0.12	±0.90	0.20	±1.80	0.30	±3.60	0.50
(400)-500	±0.50	0.14	±1.10	0.25	±2.20	0.35	±4.40	0.60
(500)-600	±0.60	0.16	±1.30	0.25	±2.60	0.40	±5.00	0.70
(600)-700	±0.70	0.18	±1.50	0.30	±3.00	0.45	±6.00	0.70
(700)-800	±0.80	0.20	±1.70	0.30	±3.40	0.50	±6.50	0.80
(800)-900	±0.90	0.20	±1.90	0.35	±3.80	0.50	±7.50	0.90
(900)-1000	±1.00	0.25	±2.00	0.40	±4.20	0.60	±8.00	450

*) = Permitted ± deviation at arbitrary point in relation to nominal dimension.
**) = Permitted length variation (depending on flatness and parallelism error).

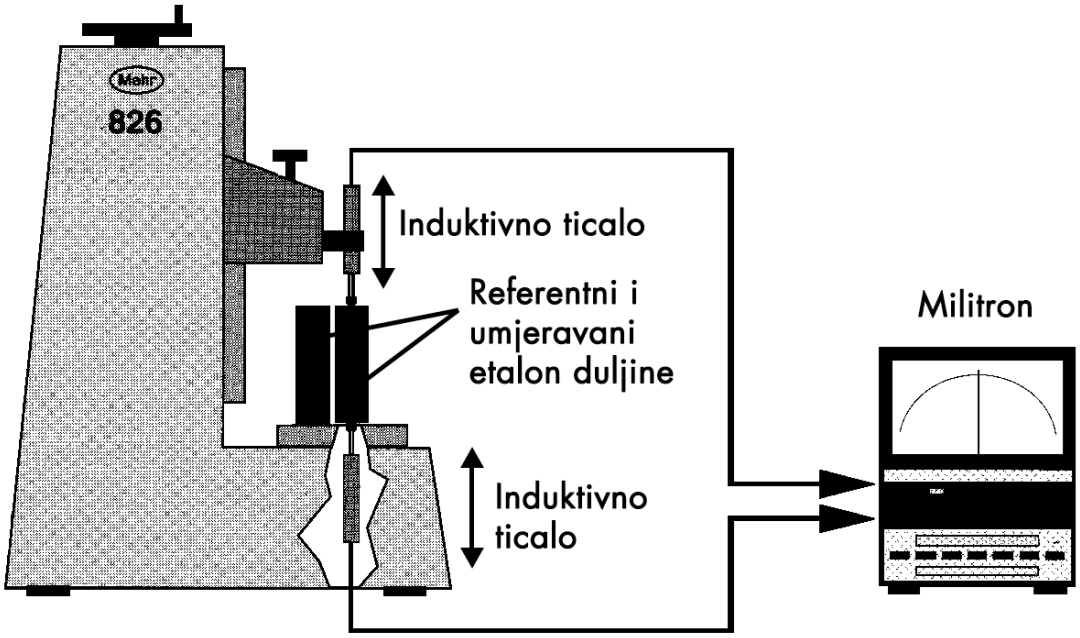
Slika 2.1 Tablica tolerancija za etalone duljine po DIN 861

Prema zahtjevima norma etaloni duljine moraju biti izrađeni iz visoko kvalitetnog čelika ili nekog drugog odgovarajućeg materijala otpornog na trošenje. Pri tome stabilnost materijala ne smije prijeći propisana dopuštena odstupanja, a mjerne površine moraju se

moći tako obraditi da mogu osigurati prianjanje kako bi se etaloni mogli slagati jedan na drugi uz minimalne pogreške.

2.2 Usporedbena metoda mjerenja kratkih etalona duljine

Pri mjerenju duljine etalona usporedbenom metodom duljina umjeravanog etalona određena je mjerenjem razlike između referentnog etalona i umjeravanog etalona iste nazivne duljine. U tu svrhu koristi se elektronski komparator, koji nam daje razliku dviju dužina.



Slika 2.2 Shematski prikaz elektronskog komparatora

Duljina etalona *Ln* [mm] u bilo kojoj točci mjerne površine, definirana je kao okomita udaljenost između te točke i ravne površine mjerne podloge izrađene od istog materijala i iste kvalitete površine, na koju je prionula druga mjerna površina etalona.

Mjerenje se mora obavljati u standardnim mjeriteljskim uvjetima:

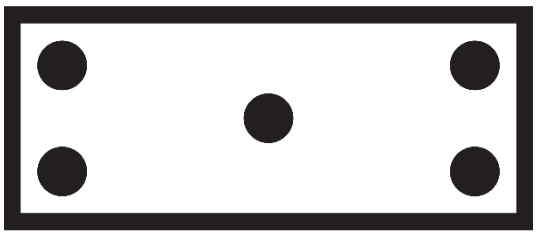
- temperature zraka od 20 °C
- tlaka zraka od 101325 Pa
- tlaka vodene pare u zraku 1333 Pa

2.3 Radna podloga za umjeravanje planparalelnih graničnih mjerki usporednom metodom

Prije početka samog mjerenja ispunjava se nazivna strana radne podloge. Ona sadrži opće podatke o mjerenju koje će se izvršiti.

To su veza sa certifikatom, proizvođač, mjerno područje, broj komada u garnituri, serijski broj garniture, klasa etalona, model, oznaka, naslovnik, oznaka mjernog uređaja, oznaka referentnog etalona te tko je vršio umjeravanje, provjeru i datum mjerenja.

Temeljem klase etalona odabire se odgovarajući obrazac za unos rezultata mjerenja. Etaloni klase 00 i 0, mjere se 25 puta. Pet puta u svakoj točci. Imaju poseban obrazac.



Slika 2.3 Mjerne točke etalona klase 00 i 0

Etaloni klasa 1 i 2 mjere se pet puta u središnjoj točci. Imaju poseban obrazac.
Količina potrebnih obrazaca ovisi o broju komada unutar jedne garniture.

Prilikom mjerenja vodi se računa o uvjetima u sobi te se zapisuju odstupanja od standardne temperature (20 °C) mjernog uređaja, etalona i okoliša. Ukoliko jedna navedenih odstupanja temperatura prelazi 0,5 °C ili ako je razlika između međusobno izmjerenih odstupanja veća od 0,1 °C, mjerenje je nevažeće, te treba čekati bolje uvijete i pokušati ponovno.

Osim unošenja rezultate mjerenja (δLi) i zapažanja u obrasce, potrebno je izračunati određene vrijednost.

δLi [µm] - izmjerena razlika duljine umjeravanog i referentnog etalona

$$\overline{\delta Li} = \sum_{i=1}^{n} \frac{\delta Li}{n} \tag{2.1}$$

$\overline{\delta Li}$ [µm] - srednja vrijednost izmjerenih razlika

ΔL_s [µm] - odstupanje mjere iz certifikata (uzima se iz certifikata vezanog za referentni etalon)

$$\Delta L_c = \overline{\delta Li} + \Delta L_s \tag{2.2}$$

ΔL_c [µm] - odstupanje duljine u sredini

te [µm] - dopušteno odstupanje duljine u sredini (uzima se iz Tablica tolerancija za etalone duljine po DIN 861)

tv [µm] - maksimalno dopuštena varijabilnost (uzima se iz Tablica tolerancija za etalone duljine po DIN 861)

fv [µm] - najveći raspon između pet dopuštenih odstupanja duljine kod etalona klase 00 i 0

Kada se sve izmjeri i izračuna, radna podloga se šalje na provjeru, nakon koje se sastavlja certifikat za pripadajuću garnituru planparalelnih graničnih mjerki.

3 INTERNET

Internet je globalna računalna mreža koja povezuje računala i računalne mreže korištenjem internet protokola (IP) u jednu cjelinu s namjerom razmjene podataka i korištenja raznih sadržaja, usluga i servisa kao što su web stranice, elektronička pošta i mnogi drugi.

Dvije glavne komponente interneta su serveri i klijenti. Svako računalo može uz odgovarajući softver biti server ili klijent ili oboje.

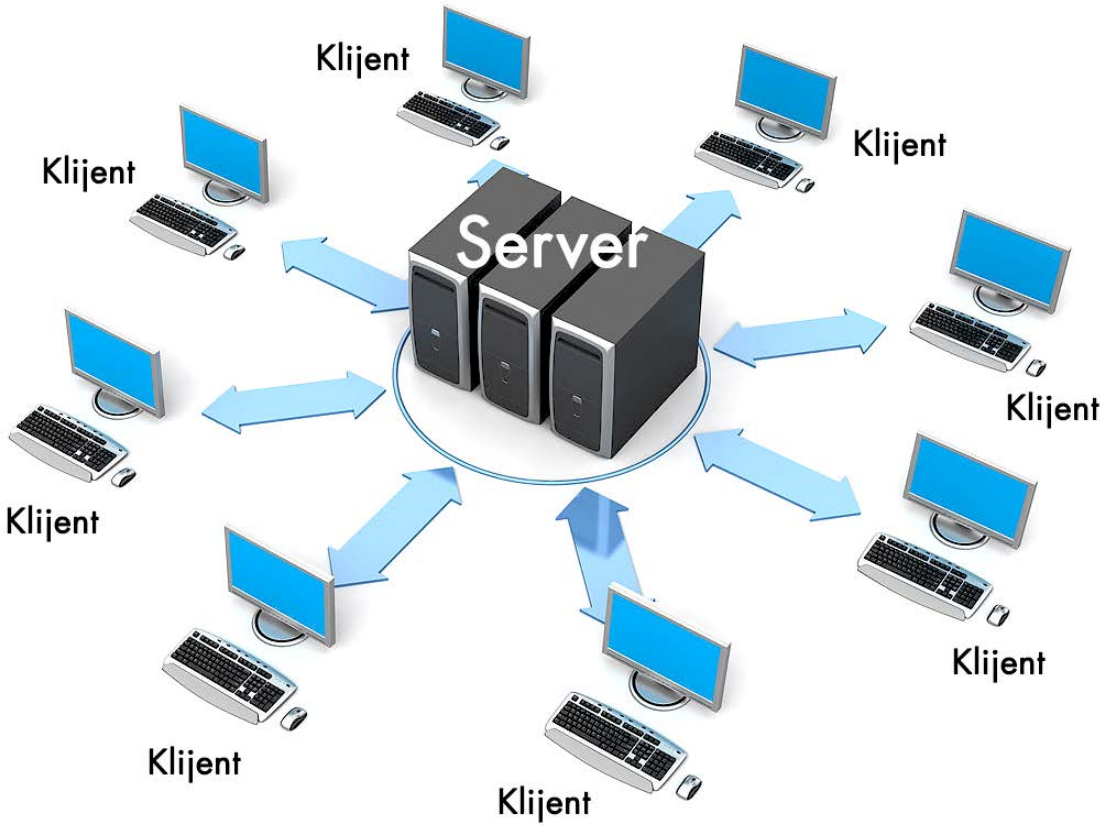
3.1 Klijent

Da bi računalo bilo klijent mora imati web preglednik i biti spojeno na mrežu sa serverom. U današnjem svijetu to nije problem jer gotovo svi uređaji sa ekranom imaju web preglednik, od računala, mobitela do televizora. Web preglednik je program koji korisniku omogućuje pregled web stranica i multimedijalnih sadržaja vezanih uz njih.

3.2 Server

Računalo može biti server ako je na njemu instaliran i omogućen serverski softver kao što je Apache HTTP server, koji je trenutno instaliran na 60% svih servera diljem svijeta. On omogućuje podršku izvođenja PHP, Perl, Python i Tcl skripti potrebnih za rad kompleksnijih web stranica. Za mogućnost korištenja baza podataka mora biti instaliran posebni softverski paket za to kao što je MySQL.

U konačnici se na serveru nalaze web stranice kojima klijenti pristupaju. Kako je pristup svakom računalu određen portovima koji su predstavljeni brojevima, tako je i pristup web serveru određen portom. Port za pristup web serveru je **80**. Svaki server ima svoju statičnu IP adresu, dok klijenti najčešće imaju dinamične IP adrese prilikom pristupa internetu. Kao primjer sljedeći niz brojeva **161.53.116.12:80** je stvarna URL adresa (eng. Uniform Resource Locator) na koju smo preusmjereni kada idemo na **www.fsb.unizg.hr**.



Slika 3.1 Server Klijent - prikaz jednostavne internetske mreže

3.3 HTTP protokol

Najpoznatiji i najkorišteniji internet servis je web, koji koristimo za pristupanje i prikazivanje web stranica pomoću HTTP protokola.

HTTP je kratica za (eng.) HyperText Transfer Protocol. To je zahtjev/odgovor protokol za komunikaciju između servera i klijenta. HTTP klijent inicira prijenos podataka nakon što uspostavi TCP (eng. Transmission Control Protocol, vrsta komunikacije) vezu s udaljenim web serverom na određenom portu.

Server konstantno osluškuje zahtjeve, čekajući da klijent pošalje niz znakova, kao što je "GET / HTTP/1.1" - kojim će zahtijevati uspostavljanje komunikacije, a nakon toga i poruku koja sadrži nekoliko slovnih nizova (zaglavlje) koji određuju aspekte zahtjeva, te paket neobaveznih podataka. Zahtjev klijenta će rezultirati slanjem odgovora sa servera, kao što je "200 OK" - nakon čega će server poslati i svoj paket podataka koji najčešće sadrži traženu datoteku ili poruku o grešci. Odmah po ispunjenju zahtjeva klijenta, server će prekinuti komunikaciju.

HTTP se razlikuje od ostalih TCP protokola kao što je npr. FTP (eng. File Transfer Protocol), po tome što se konekcija i komunikacija sa serverom prekida odmah nakon izvršenja zahtjeva klijenta. Zbog ovoga je HTTP idealan za Web, gdje je web stranica obično povezana s drugim stranicama na drugim serverima.

4 WEB ALATI

Od začeća Interneta do danas, prošlo je samo 25. godina. U tih 25. godina se Internet drastično promijenio, od samih web stranica do načina izrade web stranica. Iako postoji standardizacija izrade web stranica koju provodi World Wide Web Consortium (skr. W3C) koju je osnovao izumitelj interneta Tim Berners-Lee, te standardizacije imaju funkciju smjernica i preporuka. Kao rezultat te fleksibilnost postoji ogroman broj alata za izradu web stranica i protokola koji se mijenjaju iz godine u godinu.

Kao najpopularniji i vremenski najizdržljiviji alati su se pokazali oni besplatni sa otvorenim kodom te je danas veći dio Interneta napravljen upravo tim alatima.

Da bi se napravila moderna i interaktivna web stranica koja ima mogućnosti kakve današnji korisnici smatraju svakidašnjima potrebno je savladati HTML, CSS, JavaScript, Bootstrap, HTML DOM, PHP, MySQL, PHP PDO, AJAX, FPDF i FPDI.

Za web programiranje nije potreban poseban programski paket, jer su sve izvršne datoteke zapravo obične tekstualne datoteke sa drugačijim ekstenzijama koje možete pisati u tekstualnom editoru po vašem izboru (npr. Notepad).

Za testiranje jednostavnih web stranica potrebno ih je otvoriti u web pregledniku. Ukoliko web stranica sadrži PHP kod ili koristi usluge MySQL-a, potrebno je web stranicu prije otvaranja u web pregledniku staviti na server ili serverski softver instalirati lokalno na istom računalu. XAMPP je jedan od paketa koji omogućuje upravo to.

4.1 HTML

HTML je kratica za (eng.) HyperText Markup Language, što znači prezentacijski jezik za izradu web stranica [2]. Izvršava se u web pregledniku. Datoteka ima ekstenziju ***.html** ili ***.htm**

HTML čini osnovu svih web stranica jer definira elemente koji će se prikazivati kao što su naslovi, tekstovi, poveznice, slike, video, forme, tablice i mnogi drugi elementi. Bez tih elemenata naš web preglednik ne bi imao što za prikazati.

Kako bi definirali elemente koriste se posebne oznake (tag-ovi) koje svi web preglednici prepoznaju jednako, kako bi doživljaj i namjena koju je autor web stranice zamislio uvijek

bila jednaka. Te oznake se koriste na jednostavni način. Svaki element ima svoje jedinstveno ime kojim se označava početak i kraj elementa.

```
Ova riječ je puno <b>deblja</b> od drugih.
```

Što bi dalo za rezultat: „Ova riječ je puno **deblja** od drugih.“

Cijela sintaksa se zapravo svodi na **<ime elementa>** za označavanje početka elementa, i **</ime elementa>** za označavanje kraja elementa. Ostatak je samo učenje imena svih elemenata.

```
<ime elementa>Tekst koje je pod utjecajem elementa</ime elementa>
```

Iako je HTML osnova svih web stranica, to nije programski jezik. Njime se ne mogu napraviti ni obične operacije poput zbrajanje. Najviše što se s njim može postići su statični dokumenti kojima su vrhunac interakcije poveznice koje vode na druge web stranice.

Da bi se napravila ispravna HTML datoteka, unutar dokumenta potrebno je napisati napisati tri oznake, **<html> <head>** i **<body>**, i posebni prvi red sa sljedećom oznakom **<!DOCTYPE html>** koju ne treba zatvarati. **<html>** oznakom se uokviruje cijeli dokument, kako bi web preglednik znao da se radi o HTML kodu. Unutar njega mora biti **<head>** element i **<body>** element. U **<head>** element se stavljaju opće informacije o stranici, kao što su naslov **<title>**, te veze sa drugim dokumentima koje želimo očitati prije otvaranja stranice. Taj dio stranice se ne prikazuje u web pregledniku. **<body>** označava masu stranice, tj. dio koji zapravo vidimo. U taj element stavljamo sve elemente koje želimo prikazati kao što su paragrafi, slike i drugi. Prilikom spremanja umjesto uobičajne ***.txt** ekstenzije, datoteku spremimo sa ***.html** ekstenzijom i otvorimo je u web pregledniku kako bi vidjeli što smo postigli.

U primjeru ćemo pokazati najjednostavniju web stranicu sa njenim najmoćnijim elementom – poveznicom, koja se označava sa **<a>** oznakom, te će voditi na stranicu FSB-a, kada se klikne na tekst „Svi na FSB“.

```
<!DOCTYPE html>
<html>
     <head>
          <title>Fakultet strojarstva i brodogradnje</title>
     </head>
     <body>
          <a href=“http://www.fsb.unizg.hr/“>Svi na FSB</a>
     </body>
</html>
```

4.2 CSS

CSS je kratica od (eng.) Cascading Style Sheets [2]. Radi se o stilskom jeziku, koji se koristi za opis prezentacije dokumenta napisanog pomoću HTML jezika. Izvršava se u web pregledniku. Datoteka ima ekstenziju ***.css**

Kako se HTML razvijao počele su se koristiti elementi koji bi definirali font, veličinu teksta, boju teksta, širinu tablica i mnoge drugi elementi koji su zapravo stilske prirode, iako je HTML originalno namijenjen sadržaju. To je predstavljalo problem jer je u HTML datotekama bile sve teže čitati i editirati sadržaj zbog velike količine stilskih naredbi za svaki element.

To se riješilo pojavom CSS-a, koje se kao i do sada može koristiti unutar HTML datoteke sa oznakom **<style>** ili u zasebnoj datoteci što bi bilo poželjnije jer se na taj način može jednom definirati stil za sve iste elemente na stranici.

Npr. imamo stranicu sa 10 paragrafa. Da bi promijenili font svih paragrafa bi morali za svaki paragraf posebno dodati stilsku oznaku kojim je definiran font. To možemo pojednostaviti na način da napravimo zasebnu CSS datoteku u kojem se jednom definira stil svih paragrafa. Jedini zahtjev da se to postigne je dodavanje pozivnog elementa u HTML koji javlja pregledniku da primjeni CSS na toj stranici.

```
<link type="text/css" rel="stylesheet" href="dizajn.css" />
```

CSS ima nešto drugačiju sintaksu od HTML, gdje se navodi ime elementa i unutar vitičastih zagrada se definiraju parametri, završavajući svaki parametar sa točka-zarezom.

Za primjer ćemo napisati datoteku **dizajn.css** koji će iz prethodnog poglavlja za HTML, poveznicu „Svi na FSB“ pretvoriti u crvenu boju.

```
a{
  color: red;
}
```

4.3 JavaScript

JavaScript je skriptni programski jezik [3]. Izvršava u web pregledniku. Datoteka ima ekstenziju ***.js**

Za razliku od CSS-a i HTML-a, JavaScript je programski jezik koji dodaje web stranicama interaktivnost i dinamičnost. S njim možemo deklarirati varijable, zbrajati, množiti, raditi funkcije, nizove, komparacije i sve ostale mogućnosti na koje smo navikli u drugim programskim jezicima. Najčešće se koristi u obliku funkcije koja se aktivira pomoću nekog događaja u HTML djelu stranice, npr, na klik.

Za korištenje JavaScript-a, kod se mora ubaciti u HTML datoteku unutar elementa **<script>** ili posebnom naredbom ako je želimo pozvati iz zasebne datoteke (slično kao i CSS datoteku).

```
<script src="js/skripta.js"></script>
```

Sintaksa JavaScripta kao što i samo ime implicira, je vrlo nalik Java programskom jeziku, ali tu sve sličnosti nestaju, jer je Java objektno orijentiran jezik, dok JavaScript nije.

Za primjer ćemo napraviti funkciju običnog zbrajanja:

```
<script>
function zbrajanje(){
  var x=2;
  var y=2;
  var sum=x+y;
}
</script>
```

4.3.1 HTML DOM

Nakon što se cijela stranica učita, web preglednik stvori DOM (eng.) Document Object Model [3]. To je skup svih elemenata stanice pretvorenih u stablo objekata koji pokazuje gdje se na stranici elementi nalazi, uključujući njihove parametre i sadržaj.

Ovo je vrlo bitno jer se JavaScript-u time omogućuje pristup cijeloj stranici kako bi je učinio dinamičnom i promjenjivom. Uz uvjet da smo taj element nekako jedinstveno označili npr. sa parametrom **id**.

Za primjer naše funkcije **zbrajanje()**, koja nije ništa vidljivo na stranici u prethodnom primjeru napravila, iskoristiti ćemo je da u paragrafu **<p>** nam pošalje rezultat zbroja **2+2**.

Koristimo oznaku **id=“zbrojime“** za prepoznatljivost, te događaj **onclick=“zbrajanje()“** kojim se zove funkcija zbrajanja.

```
<p id=“zbrojime“ onclick=“zbrajanje()“>Zbroj 2+2</p>

<script>
function zbrajanje(){
  var x=2;
  var y=2;
  var sum=x+y;
  document.getElementById("zbrojime").innerHTML = sum;
}
</script>
```

4.4 Bootstrap

Bootstrap [4] [5] je radno okruženje koji koristeći HTML, CSS i JavaScript, ima unaprijed definirane elemente kao što su razne tipke, fontovi, simboli, navigacijska sučelja, padajući izbornici, forme i mnogi drugi elementi. To postiže na način da se smjeste već gotove CSS datoteke u mapu sa našom stranicom i prizovu u HTML datoteku.

Sve to je organizirano tako da pozivom na određenu klasu (**class**) možemo postići određeni učinak na naše elemente.

Za primjer ćemo poveznicu „Svi na FSB“ pretvoriti u tipku dodavanjem posebne klase.

```
<a class="btn btn-primary" href=“http://www.fsb.unizg.hr/“>Svi na FSB</a>
```

Slika 4.1 Bez(lijevo) i sa Bootstrap-om(desno)

Osim brzog uređivanja, Bootstrap-ova glavna funkcija je prilagodljiv kostur stranice, organiziran u 12 stupaca. To znači da stranice napravljene sa Bootstrap radnim okruženjem će dinamično mijenjati svoju širinu i preslagivati elemente na web stranici kako bi bilo bolje prilagodilo sadržaj veličini zaslona na kojem se gleda web stranica. Korištenjem Bootstrap-a možete biti sigurni da će vaša stranica uvijek jednako dobro izgledati i funkcionirati, bez razlika da li se web stranica koristi na TV-u ili mobitelu.

Tablica 4.1 Klase prilagodljivost kostura ($-broj stupaca)

Klase	Veličina uređaja	Aktivan kod širine ekrana
col-xs-$	Izrazito mali	0-768px
col-sm-$	Mali	768-992px
col-md-$	Srednji	992-1200px
col-lg-$	Veliki	1200px<

Da bi se iskoristila moć prilagodljivog kostura, moraju se u naše elemente dodati klase iz tablice 4.1. Dodavanjem sve 4 klase u neki element **class="col-xs-12 col-sm-12 col-md-6 col-lg-6"** možemo kontrolirati kako će se elementi ponašati na svim rezolucijama jer iako ih je četiri, aktivana je samo jedna klasa ovisno o širini zaslona na kojo se web stranica prikazuje.

Kako je ukupna širina 12 stupaca, dva elementa sa klasom kao u navedenom primjeru klase bi se u velikoj i srednjoj rezoluciji pokazivali jedan pokraj drugog, a u maloj i izrazito maloj rezoluciji bi se prikazivali jedan ispod drugog.

Slika 4.2 Prikaz prilagodljivog kostura Bootstrap radnog okruženja

4.5 PHP

PHP je kratica od (eng.) Hypertext preprocessor [2]. Izvršava se na serveru. Datoteka ima ekstenziju ***.php**

To je jedna od najnaprednijih i najkorištenijih server-side skriptnih tehnologija danas u upotrebi. To znači da za razliku od HTML, CSS i JavaScripta koji rade na računalu na kojem je web preglednik, PHP radi na serveru. Kada otvaramo PHP web stranicu, prvo se kod obradi i interpretira svu logiku na serveru i tek nakon obrade dolazi do našeg web preglednika u obliku običnog HTML-a koji onda naš web preglednik prikaže. Zbog toga mi kao korisnik ne možemo znati PHP kod stranice na kojoj se nalazimo, dok za HTML, CSS i JavaScript možemo uvijek pogledati izvorni kod.

Svojom sintaksom PHP je sličan mnogim drugim programskim jezicima, čak i ima istoznačne (iste po sintaksi i funkcionalnosti) funkcije kao i neki drugi jezici kao što su C ili Perl. To znači da jednu radnju možete izvesti korištenjem više različitih funkcija.

Kako bi napravili PHP stranicu, **<body>** elementa koristimo posebnu **<?php** oznaku za otvaranje i **?>** za zatvaranje PHP koda. Ako koristimo PHP kod unutar web stranice potrebno je promijeniti ekstenziju ***.html** u ***.php**.

```
<html>
<body>
<?php
   echo “Ovo je PHP kod.“;
?>
</body>
</html>
```

U primjeru smo otvorili PHP kod, te naredbom **echo** ispisali **Ovo je PHP kod.** Kao rezultat na našoj stranici će se prikazati taj tekst. Ovdje smo samo ispisali rečenicu, ali smo mogli koristiti i varijable koje se označavaju dolar oznakom.

```
<html>
<body>

<?php
   $x=2;
   $y=2;
   $sum=$x+$y;
   echo “Zbroj je “.$sum;
?>
</body>
</html>
```

Kao i svaki programski jezik PHP ima i sve naredbe kao što su **if**, **then**, **else**, **for**, **while** i mnoge druge, što nam daje veliku kontrolu za konačan prikaz naše web stranice.

4.5.1 POST i GET

JavaScript trenutno izgleda jako slično PHP-u po mogućnostima, ali PHP se izvršava na serveru prije pokretanja stranice, dok se JavaScript izvršava u web pregledniku. Iako je PHP moćan, zapravo stvara statične nepokretne stranice, jer se sva logika događa na serveru prije nego što nam generira HTML. Da bi došlo do neke promjene na web stranici korisnik treba svojom interakcijom sa web stranicom poslati nove podatke serveru i otvoriti novu stranicu temeljem tih podataka kako bi server ponovno obradio PHP datoteku.

Komunikacija između dviju stranica se ostvaruje na dva načina. POST i GET metodom.

4.5.1.1 GET

GET metodom [3] se šalju podaci koristeći URL vezu].
Obično kada pristupamo web stranici koristimo URL adresu sa imenom datoteke npr. **index.php**
Koristeći GET metodu kroz URL adresu možemo poslati podatke određenoj stranici. Npr. **indeks.php** čemo poslati vrijednosti x=2 i y=2:

```
Indeks.php?x=1&y=1
```

Nakon otvaranja stranice s takvom URL adresom, ti podaci se mogu dohvatiti sljedećim naredbama:

```
<?php
if (isset($_GET['x']) AND isset($_GET['y'])){
   $x=$_GET['x'];
   $y=$_GET['y'];
}
?>
```

Naredbom **isset** možemo provjeriti da li su opće zadani x i y unutar adrese, ako jesu dohvatimo vrijednosti koristeći naredbe **$_GET['x']** i **$_GET['y']** kako bi te vrijednosti koristili dalje u kodu ili pridodali novoj varijabli.

GET metoda se najčešće koristi za navigaciju kroz sadržajno generirane stranice koje se mijenjaju temeljem podataka koje dobivaju kroz URL adresu.

4.5.1.2 POST

POST metodom [3] se podaci, za razliku od GET metode, šalju skriveno, nedostupni su kroz adresu, nisu zabilježeni u povijesti web preglednika, nemaju limit na dužini, te se ne spremaju u predmemoriju. Ne postoji način da mu se pristupi i aktivira osim kroz kod stranice, dok smo GET metodu mogli aktivirati jednostavnim korištenjem adresne linije u web pregledniku. Kao rezultat navedenoga, POST je puno sigurniji način prijenosa podataka između web stranica.

Najčešće se koristi za prijenos korisnički unesenih podataka ili podataka osjetljive prirode.

Za primjer imamo dvije web stranice **ulazna.php** na kojoj je forma sa jednim poljem za **ime**, te **izlazna.php** koja će dohvatiti podatke iz ulazne koristeći **$_POST** naredbu i osobno vas pozdraviti.

ulazna.php

```
<html>
<body>

<form action="izlazna.php" method="POST">
Ime: <input type="text" name="ime">
<input type="submit">
</form>

</body>
</html>
```

izlazna.php

```
<html>
<body>

Pozdrav kolega <?php echo $_POST["ime"]; ?><br>
</body>
</html>
```

4.6 MySQL

MySQL je besplatan, open source sustav za upravljanje bazom podataka [3]. Nalazi se i izvršava na serveru. Temeljen je na SQL (eng.) Structured Query Language, računalni jeziku za izradu, traženje, ažuriranje i brisanje podataka iz relacijskih baza podataka.

Baza podataka se organizira u određeni broj tablica koje se koriste za pohranjivanje podataka. Podatci se unose u tablice kao redovi jer želimo određene podatke vezati jedan uz druge kao skup podataka, npr. kao varijabla rječnika jer svaki podatak ima svoj specifični ključ kako bi došli do njega.

MySQL sustav i baze podataka koje stvara se nalaze na serveru. Što znači da web stranice moraju ostvariti komunikaciju sa bazom podataka kako bi ih mogla koristiti.

Sintaksa za korištenje SQL-a je vrlo jednostavna i prilagođena običnom jeziku, tj naredbama za ono što najbolje radi: stvaranje tablica, ubacivanje, ažuriranje, brisanje i pretraživanje podataka.

Prvo moramo stvoriti bazu podataka sa podrškom za hrvatske znakove u kojoj ćemo kasnije stavljati sve tablice.

```
CREATE DATABASE baza CHARACTER SET utf8 COLLATE utf8_bin;;
```

Moramo stvoriti i korisničko ime i lozinku za pravo pristupa našoj bazi:

```
CREATE USER 'pero'@'%' IDENTIFIED BY '1234';
GRANT ALL PRIVILEGES ON baza.* TO 'pero'@'%';
FLUSH PRIVILEGES;
```

Sljedeće što treba je stvoriti tablicu. Za primjer ćemo stvoriti novu tablicu **korisnik** sa poljima **id**, **ime** i **prezime**. Za svako polje u tablici moramo definirati njeno ime i tip podatka. Tip podataka može biti **INT** (broj), **FLOAT** (decimalni broj), **VARCHAR**(tekst), **ENUMSET**(predefinirane opcije), DATETIME (vrijeme i datum) i mnogi drugi. **id** je oznaka koja će se automatski dodjeljivati i rasti, kako bi svaki korisnik imao jedinstveni broj. **ime** i **prezime** su tekstualna polja sa maksimalnom duljinom od 30 znakova.

```
CREATE TABLE korisnik (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
ime VARCHAR(30) NOT NULL,
prezime VARCHAR(30) NOT NULL,
)
```

Dodati podatke u tablicu činimo na sljedeći način. **id** je bitno naznačiti kao **NULL** jer će se on sam dodijeliti, kako bi svaki korisnik imao svoj jedinstveni broj.

```
INSERT INTO korisnik SET id=NULL, ime=“Pero“, prezime=“Perić“;
```

Nakon što imamo nekoliko korisnika trebamo selektirati podatke kako bi ih koristili u web stranici ili u neke druge svrhe. U tu svrhu moramo raditi pretragu. Kada radimo pretrage uvijek koristimo naredbu **WHERE**, praćenu sa uvjetima kao što su **=**, **<**, **>**, **<=**, **>=**, te logičkim operatorima **AND** kojim možemo biti vrlo specifični kod pretrage.
Za primjer ćemo pronaći korisnika referencirajući se na **id**=**3**.

```
SELECT FROM korisnik WHERE id=3;
```

Ukoliko želimo doći do podataka svih korisnika sa imenom **Pero** i prezimenom **Perić**, to možemo učiniti na sljedeći način. Rezultat će biti ili jedan ili više njih ili nijedan ovisno koliko ih nađe u tablici korisnik.

```
SELECT FROM korisnik WHERE ime="PERO" AND prezime="Perić";
```

Ažurirati podatke činimo referencirajući se na **id** korisnika koji znamo da je jedinstveni pokazatelj određenog korisnika. Npr, umjesto **Pero** se korisnik zapravo zove **Petar**.

```
UPDATE korisnik SET ime="Petar" WHERE id=1;
```

Kako bi izbrisali tog korisnika, opet se referenciramo pomoću **id** polja.

```
DELETE FROM korisnik WHERE id=1
```

4.6.1 PHP PDO

Postoji više načina kako ostvariti komunikaciju sa MySQL bazom podataka [6] [7]. Za MySQL su najpopularniji PHP MySQL ekstenzija, PHP MySQLi ekstenzija i PHP PDO.

PDO je kratica za (eng.) PHP Data Objects. PHP PDO je najnoviji način komunikacije sa MySQL bazama i prvi put se pojavio 2005., izlaskom PHP 5.1 (danas je PHP 5.6.4).

PDO je trenutno najefikasniji jer nudi najveću stopu sigurnosti protoka podataka do servera sa gotovo nemogućim injektiranjem malicioznog koda unutar putujućih podatkovnih paketa. Osim sigurnosti ima vrlo čistu i metodičnu sintaksu te je neovisan o vrsti SQL servera dok su druge metode specifične za MySQL. Time možemo uz minimalne promjene u kodu koristiti bilo koji SQL server, bez razlike da li se radi MySQL-u, Oracle-u ili Microsoft SQL Server-u.

Da bi koristili bazu koju smo kreirali u prethodnim primjerima, prvo moramo ostvariti konekciju sa bazom podataka imena **baza**.

```
<?php
$host="localhost";
$db_name="baza";
$username= "pero";
$password="1234";
try{
   $con=new PDO("mysql:host={$host};dbname={$db_name}",$username, $password);
   $con->exec("set names utf8"); //potrebno za hrvatske znakove
}
catch(PDOException $exception){
   echo "Connection error: ".$exception->getMessage();
}
?>
```

Kako web stranice nemaju kompajler koji bi nam javljao greške moramo se osigurati na druge načine. Često koristimo naredbu **try** iako nije nužna, jer ukoliko je izvođenje zbog nekog razloga nemoguće, na taj način možemo doznati da je bio problem.

Nakon povezivanja sa bazom sve SQL operacije se izvode na sličan način, pa ćemo prikazati primjer odabira imena sa **id=1** u tablici **korisnik**.

```
<?php
   $id=1;
   $q = $con->query("SELECT ime FROM korisnik WHERE id={$id}");
   $f = $q->fetch();
   $ime=$f["ime"];
   echo “Pozdrav “.$ime;
?>
```

4.7 AJAX

AJAX je kratica za (eng.) asynchronous JavaScript and XML [3]. To nije programski jezik niti radno okruženje, već poseban način korištenja JavaScripta, HTML-a, PHP-a, HTML DOM naredbi, te XMLHttpRequest kojima se šalju informacije između stranica.

AJAX omogućuje komuniciranje sa serverom u pozadini bez prekida rada ili obnavljanja stranice na kojoj se nalazimo. Time postižemo da web stranice postanu uistinu interaktivne i dinamične, sa svim mogućnostima koje nude razne web tehnologije spojene zajedno u jednu cjelinu čineći Internet onakav kakvim ga danas poznajemo.

AJAX se esencijalno sastoji od tri dijela. Od HTML elementa na početnoj stranici, izdvojene PHP datoteke koja će u pozadini komunicirati sa serverom i JavaScript koji će to sve povezati u cjelinu.

Za primjer ćemo napraviti **index.php** i **pozadina.php**. **indeks.php** će sadržavati dva polja za unos podataka. U prvo ćemo unijeti **id**, a u drugo polje kada ga kliknemo će se automatski dobaviti ime korisnika iz MySQL talbice **korisnik** temeljem upisanog **id**-a.

indeks.php

```
<html>
<body>

<form>
id: <input id=“id“ type="text" name="ime">
ime: <input id=“ime“ onclick=“baza()“ type="text" name="ime">
</form>

<script>
function baza() {
   var id =document.getElementById('id').value;
```

```
  if (window.XMLHttpRequest) {
    // code for IE7+, Firefox, Chrome, Opera, Safari
    xmlhttp = new XMLHttpRequest();
  } else {
    // code for IE6, IE5
    xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
  }
  xmlhttp.onreadystatechange = function() {
    if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
      document.getElementById("ime").value =xmlhttp.responseText;
    }
  }
  xmlhttp.open("GET","pozadina.php?id="+id,true);
  xmlhttp.send();
}
</script>
</body>
</html>
```

pozadina.php

```
<?php
$id = $_GET['id'];
$host="localhost";
$db_name="korisnik";
$username= "pero";
$password="1234";
$con=new PDO("mysql:host={$host};dbname={$db_name}",$username, $password);
$con->exec("set names utf8");
$q = $con->query("SELECT ime FROM korisnik WHERE id={$id}");
$f = $q->fetch();
$ime=$f["ime"];

echo $ime;
?>
```

4.8 FPDF

FPDF (eng.) Free PDF je skup PHP klasi koji služi za generiranje PDF dokumenata [8].

Za ispravno korištenje potrebno je **include** naredbom pozvati **fpdf.php** datoteku unutar stranice gdje koristimo njene mogućnosti.

Sljedeći primjer pokazuje izradu jednostavnog PDF dokumenta koji će ispisati riječ „Pozdrav“, veličine 16, stila bold. Nakon izvršavanja koda generira se datoteka **bok.pdf.**

```
<?php
include('fpdf.php');

$pdf = new FPDF();
$pdf->AddPage();
$pdf->SetFont('Arial','B',16);
$pdf->Cell(40,10,'Pozdrav');
$pdf->Output('bok.pdf','D');
?>
```

4.9 FPDI

FPDI (eng.) Free PDF Document Importer je skup PHP klasi koji služi za korištenje već postojećih PDF dokumenata u svrhu predložaka iznad kojih možemo pisati [9].

Za ispravan rad potrebno je već imati učitanu FPDF biblioteku te **include** naredbom pozvati **fpdi.php** datoteku.

Sljedeći primjer ispisuje riječ „Pozdrav“ iznad predloška na koordinatama **x=25mm**, **y=25mm** gledajući od gornjeg lijevog kuta u desno, odnosno dolje. Nakon izvršavanja koda generira se datoteka **bok.pdf.**

```
<?php
include('fpdf.php');
include('fpdi.php');

// initiate FPDI
$pdf =& new FPDI();
$pdf->AddPage();
$pdf->setSourceFile('predlozak.pdf');
$tplIdx = $pdf->importPage(1);
$pdf->useTemplate($tplIdx, 0, 0);

$pdf->SetFont('Arial');
$pdf->SetTextColor(255,0,0);
$pdf->SetXY(25, 25);
$pdf->Write(0, "Pozdrav");

$pdf->Output('bok.pdf', 'D');
?>
```

5 KREIRANJE BAZA PODATAKA

Rješavanje papirologije je glavni pokretač i smisao web stranice koju radimo, pa je logično da proces izrade započnemo organizacijom baze podataka gdje će se spremati svi rezultati mjerenja i izračuni.

5.1 Papirnati obrasci

Imamo tri različita obrasca zavisna jedan o drugom ovisno o tipu klase etalona koji mjerimo i broju komada u garnituri.

RADNA PODLOGA
ZA UMJERAVANJE PLANPARALELNIH GRANIČNIH MJERKI USPOREDBENOM METODOM

list 1 listova

Broj: _____

Datum: _____

Naziv:	
Proizvođač:	
Mjerno područje:	
Broj komada u garnituri:	
Serijski broj garniture:	
Klasa (ISO 3650):	
Model:	
Oznaka:	
Korisnik:	
Oznaka uređaja za diferencijsko mjerenje:	
Oznaka referentnih p.g.m:	

Za mjerilo izdati: ***Potvrdu o umjeravanju*** ***Izvješće o mjerenju***
(nepotrebno precrtati)

Mjerenja izvršio : Provjerio:

_____ _____

LFSB-WR 010

Slika 5.1 Nazivni obrazac radne podloge za umjeravanje etalona duljine

Tablica 1: Rezultati mjerenja dužine

Oštećenja na mjernim površinama:							Oštećenja na mjernim površinama:						
Redni broj	L_n mm	Serijski broj	Odstupanje temperature od 20 °C				Redni broj	L_n mm	Serijski broj	Odstupanje temperature od 20 °C			
			Θ_s, °C							Θ_s, °C			
			Θ, °C							Θ, °C			
			Θ_e, °C							Θ_e, °C			
Mjerenje broj	Izmjerena razlika duljine δL_i i=1	i=2	i=3	i=4	i=5		Mjerenje broj	Izmjerena razlika duljine δL_i i=1	i=2	i=3	i=4	i=5	
1							1						
2							2						
3							3						
4							4						
5							5						
$\overline{\delta L_i}$							$\overline{\delta L_i}$						
ΔL_s							ΔL_s						
ΔL_i							ΔL_i						
Promjenljivost duljine f_v, µm	t_v, µm	Odstupanje duljine u sredini ΔL_c, µm	t_e, µm				Promjenljivost duljine f_v, µm	t_v, µm	Odstupanje duljine u sredini ΔL_c, µm	t_e, µm			

list listova

Slika 5.2 Obrazac za rezultate mjerenja etalona duljine klase 00 i 0

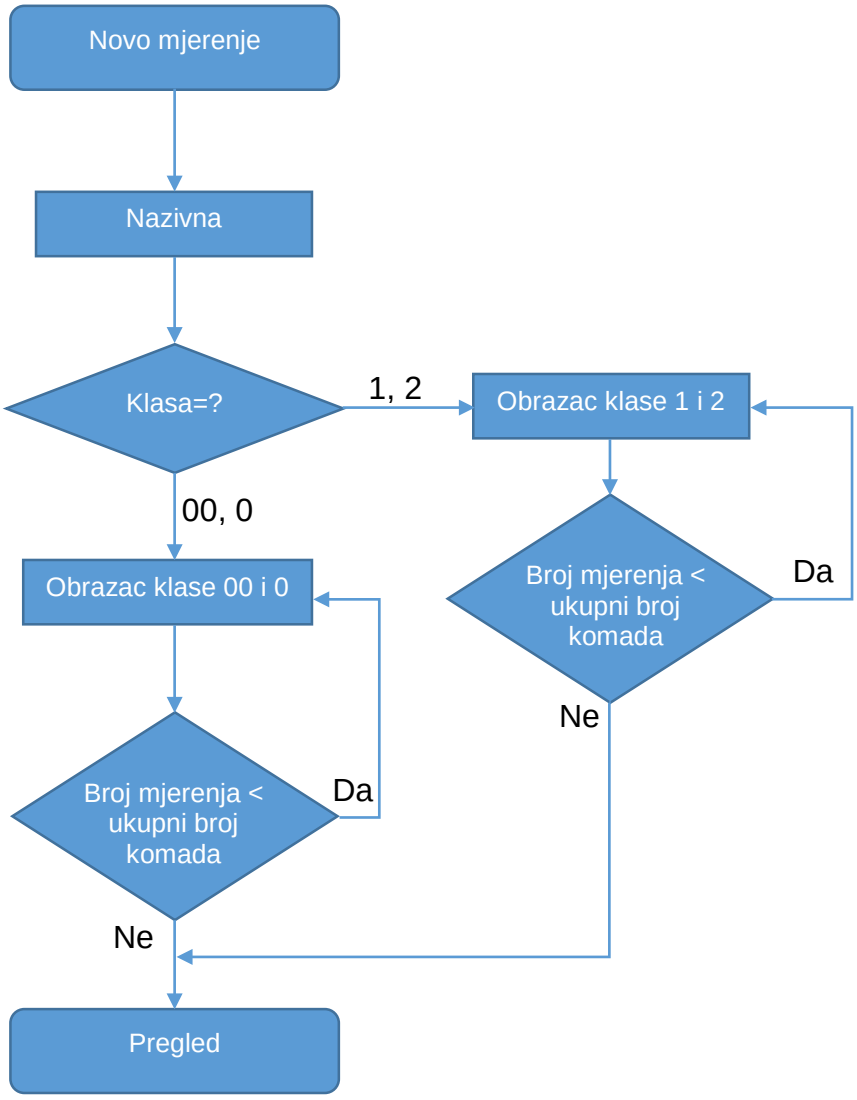
Tablica 1: Rezultati mjerenja dužine

Oštećenja na mjernim površinama												
Redni broj												
L_n, mm												
Serijski broj												
Mjerenje broj												
1												
2												
3												
4												
5												
$\overline{\delta L_i}$												
ΔL_s												
Odstupanje duljine u sredini ΔL_c, µm												
Dopušteno odstupanje duljine u sredini te, µm												
Odstupanje temperature od 20 °C Θ_s, °C												
Θ, °C												
Θ_e, °C												

list listova

Slika 5.3 Obrazac za rezultate mjerenja etalona duljine klase 1 i 2

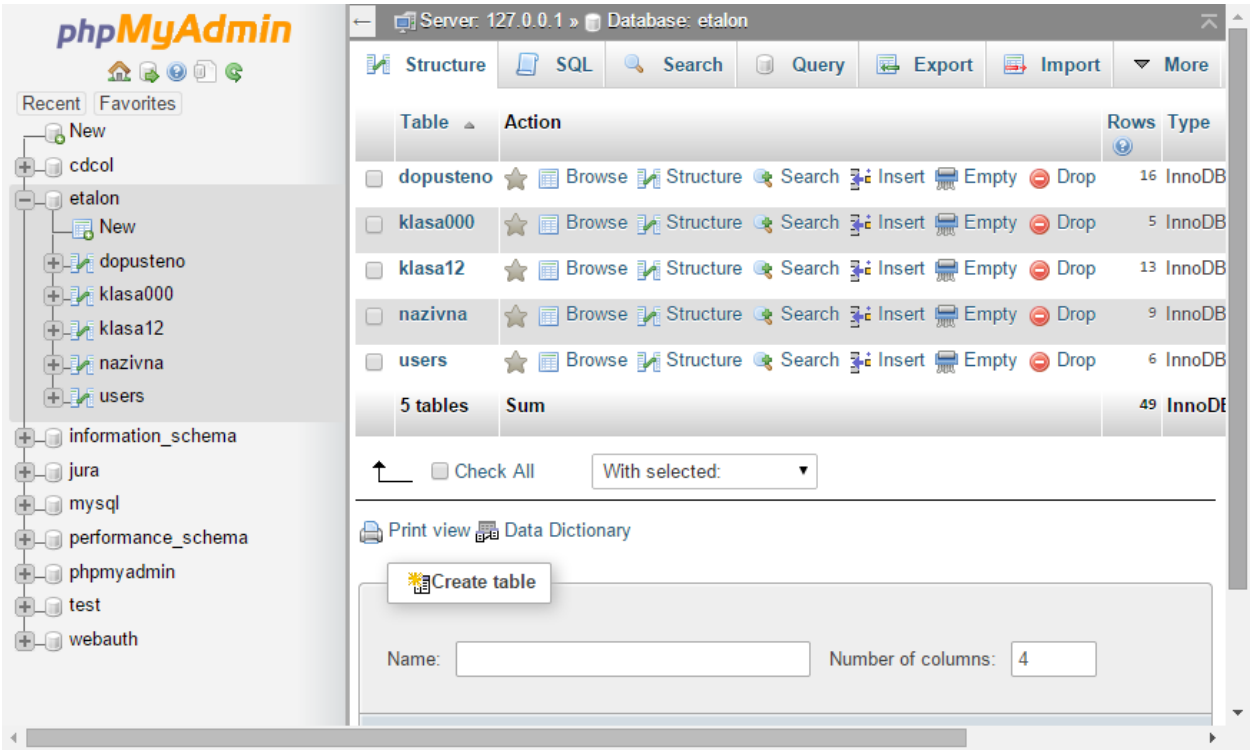
Procedura ispunjavanja obrazaca se vrši na sljedeći način, što ujedno čini i organizacijski temelj web stranice koju gradimo.



5.2 Izrada baze podataka

Da bi mogli napraviti bazu podataka potreban nam je MySQL server. U svrhu izrade koristio se XAMPP koji u sebi sadrži sve serverske mogućnosti koje su nam potrebne.

Unutar XAMPP-a postoji grafičko sučelje za rad s bazama podataka koji se zove phpMyAdmin.



Slika 5.4 phpMyAdmin sučelje

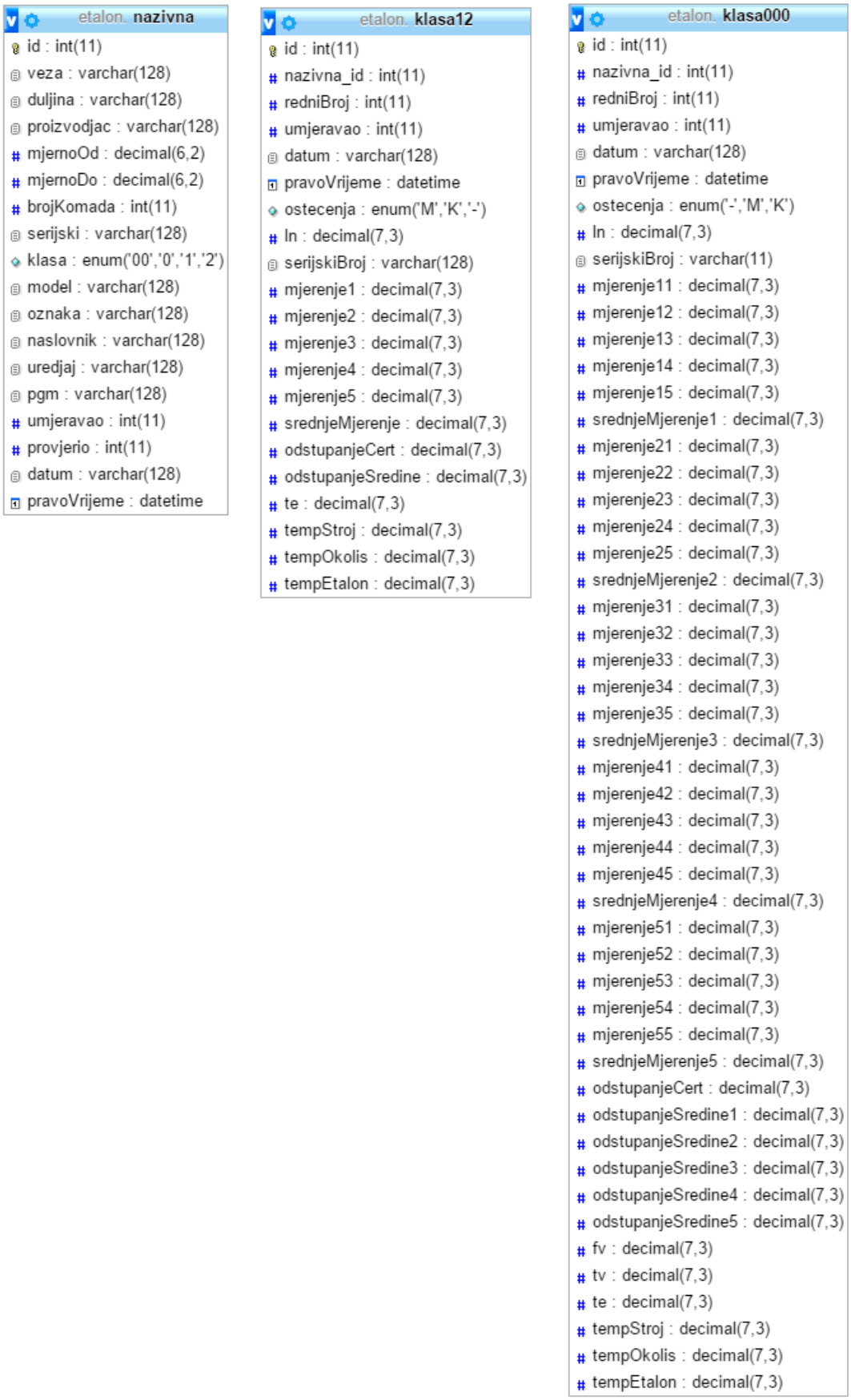
Koristeći phpMyAdmin rekreirali smo obrasce unutar baze **etalon** kao tablice sa nazivima **nazivna**, **klasa000** i **klasa12**.

Svako polje u tablici ima definirano ime i tip podatka koji će se spremati.
Potrebe naše stranice zadovoljava 5 tipa varijabli.

Tablica 5.1 Tipovi podataka korišteni u bazi

Tip podatka	**Opis**
int(11)	cijeli brojevi s maksimalno 11 znamenki
decimal (7,3)	decimalni brojevi s ukupno 7 znamenki od kojih je 3 iza decimalne točke
varchar (128)	obični znakovi, tj tekst do duljine od 128 znakova
enum('x','y','z')	limitiran izbor koji je definiran opcijama u zagradi (u ovom slučaju između x, y, i z)
datetime	zapis datuma i vremena

Tablice **nazivna, klasa000 i klasa12** kreirane su na način prikazan slikom 5.5



Slika 5.5 Prikaz strukture podataka tablice *nazivna, kalsa12* i *klasa000*

Osim podataka koje smo imali na običnim papirnatim obrascima, imamo još nekoliko novih polja korištenih u svrhu organizacije.

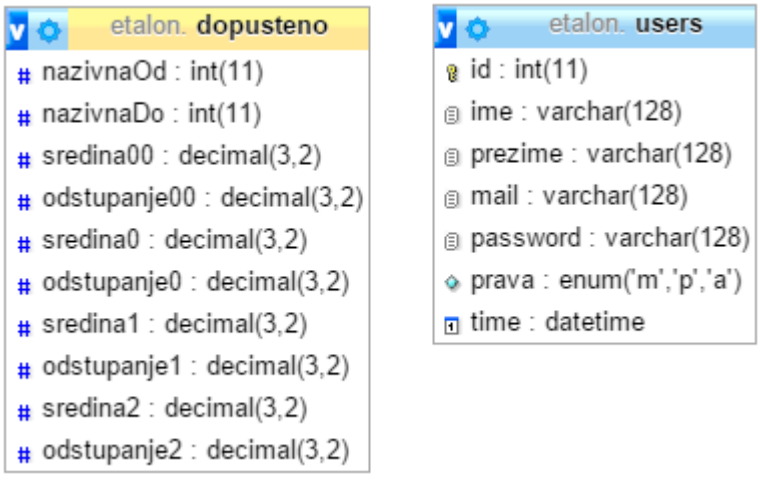
Unutar **nazivna** imamo sljedeća specifična polja:

- **id** – jedinstveni broj retka koji unosimo u tablicu. Automatski se dodjeljuje koristeći opciju **AUTOINCREMENT** te nam služi za raspoznavanje točno određenih redaka u tablici.
- **pravoVrijeme** – automatski bilježi vrijeme i datum unošenja korištenjem opcije **TIMESTAMP.** Iako već imamo polje **datum**, bitno je naglasiti da proizvoljno izabrani datum nije uvijek prikaz stvarne situacije. **pravoVrijeme** npr. koristimo za sortiranje podataka po vremenu stvaranja.
- **umjeravao** i **provjerio** – koristi **int** kao tip podataka. Iako neobično, to činimo jer imamo još jednu tablicu imena **user** (slika 5.6) gdje pišu podaci svih mjeritelja i provjeravača. Kako bi izbjegli pisanje istih podataka na više mjesta, pišemo samo **id** iz **user** tablice kojom se referenciramo na određenog mjeritelja ili provjeravača.

Tablice **klasa12** i **klasa000** međusobno nisu previše različite, a razlikuju se samo u količini mjerenih podataka. Oboje imaju ista specifična polja kao i **nazivna,** uz dodatak jednog novog specifičnog polja:

- **nazivna_id** – slično kao i kod polja **umjeravao,** koristi **int** tip podatka kojim se referenciramo na **id** specifičnog retka u **nazivna** tablici. Time ostvarujemo vezu između tablica kako bi znali koja mjerenja se odnose na koju nazivnu stranicu.

Uz spomenute tablice koristimo još tablicu **user** i **dopusteno**.



Slika 5.6 Prikaz strukture podataka tablice *dopusteno* i *user*

Tablicu **user** koristimo za spremanje korisničkih podataka, te poljem **prava** određujemo da li je korisnik mjeritelj (m), provjeravač (p) ili administrator (a).

Tablica **dopusteno** je savršena kopija dopuštenih tolerancija za etalone duljine po DIN 861, prikazanu na slici 2.1. Ona se koristi prilikom određivanja dopuštenog odstupanja u duljini (te) i maksimalno dopuštene varijabilnosti (tv).

6 IZRADA GRAFIČKOG KORISNIČKOG SUČELJA

Grafičko korisničko sučelje podrazumijeva korištenje HTML-a, CSS-a i Bootstrap-a.

HTML koristimo za stvaranje i organizaciju elemenata web stranice. Jedan od najčešće korištenih elemenata je ** ** koji dijeli stranicu na sekcije, što je danas temeljni oblik organizacije elemenata. Koristimo i element **<form>** koji definira obrazac unutar kojeg su **<input>** i **<select>** elementi kojima definiramo unosna polja. **<input>** još možemo parametrom **type="radio"** pretvoriti iz običnog unosnog polja u kružiće za odabir što koristimo za izbor klase, mjernog uređaja i referentnog etalona.

Bootstrap koristimo pomoću klasa kao parametre unutar HTML elemenata kako bi izgledali ugodno oku i imali mogućnosti prilagodljivosti bilo kojoj veličini uređaja.

Često je potrebno pomoću CSS-a nadglasati bootstrap-ove klase specificiranjem novih pravila kao što su boja, veličina i slično.

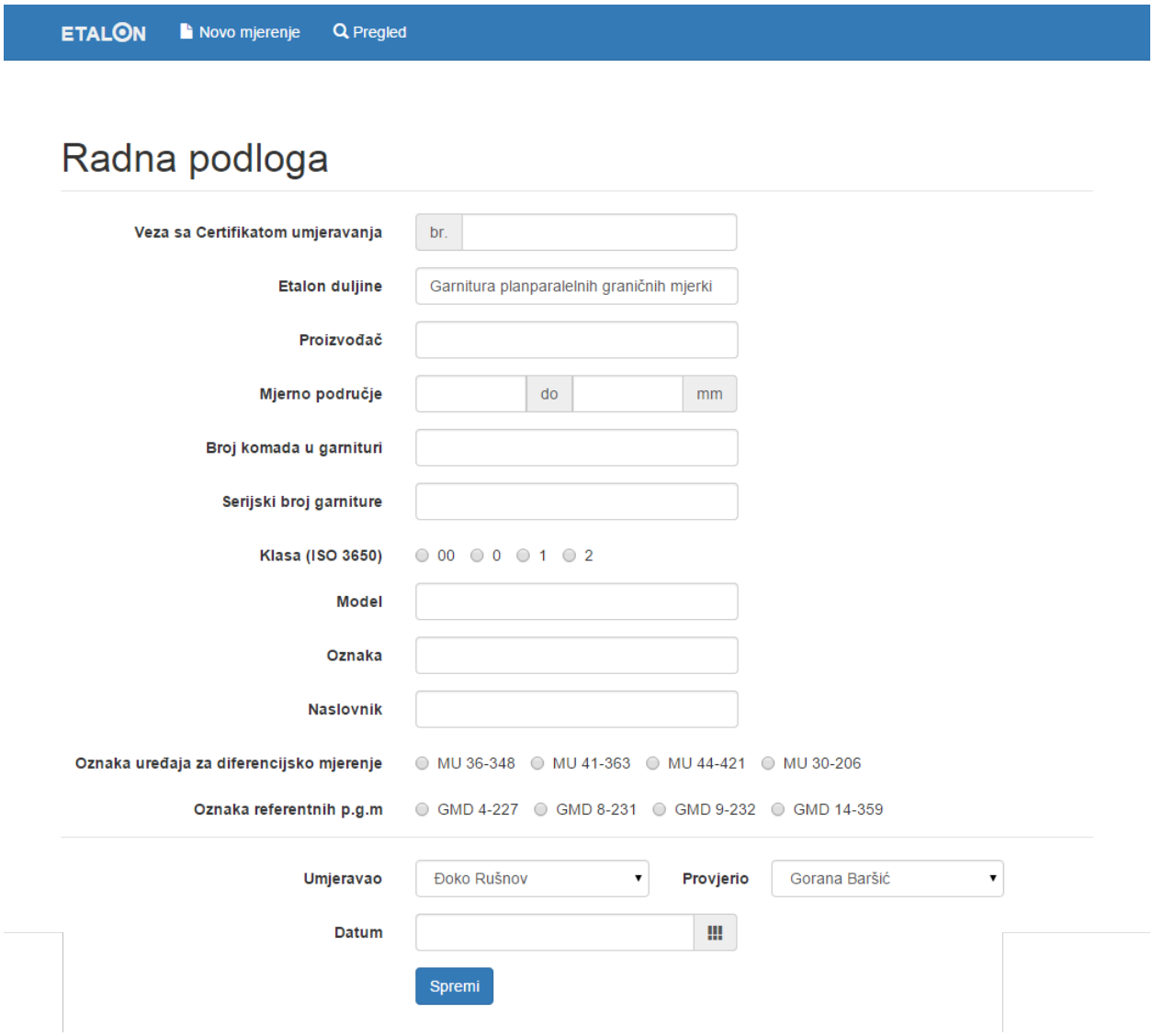
Kod unosnog polja za Serijski broj etalona:

```
<div class="form-group">
  <label class="col-md-4 control-label" for="serijski">Serijski broj
garniture</label>
  <div class="col-md-4">
    <input id="serijski" name="serijski" type="text" class="form-control
input-md">
  </div>
</div>
```

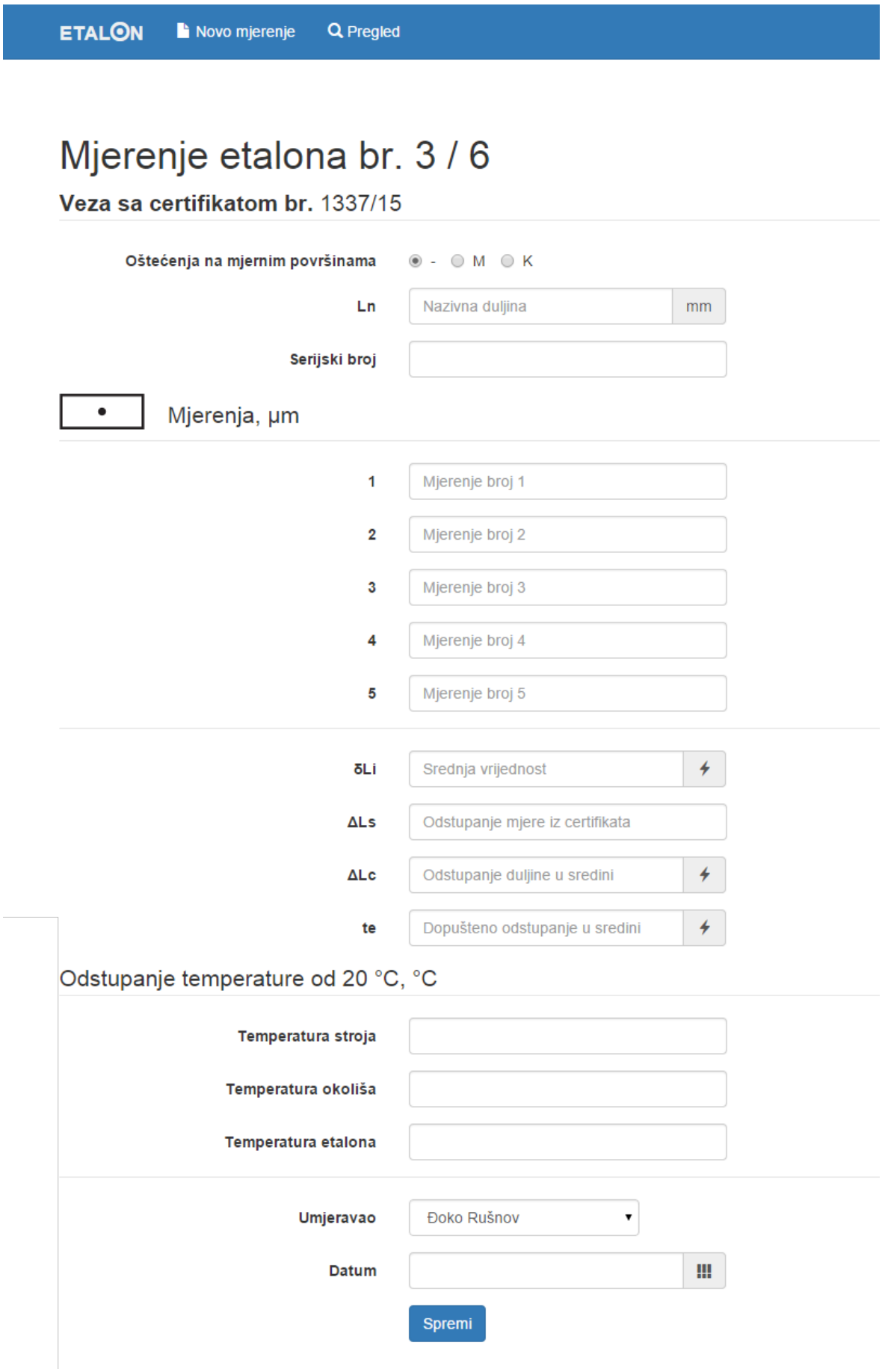
Kod kružića za odabir Klase etalona:

```
<div class="form-group">
  <label class="col-md-4 control-label" for="radios">Klasa (ISO 3650)</label>
  <div class="col-md-4">
    <label class="radio-inline" for="klasa1">
      <input type="radio" name="klasa" id="klasa1" value="00">
      00
    </label>
    <label class="radio-inline" for="klasa2">
      <input type="radio" name="klasa" id="klasa2" value="0">
      0
    </label>
    <label class="radio-inline" for="klasa3">
      <input type="radio" name="klasa" id="klasa3" value="1">
      1
    </label>
    <label class="radio-inline" for="klasa4">
      <input type="radio" name="klasa" id="klasa4" value="2">
      2
    </label>
  </div></div>
```

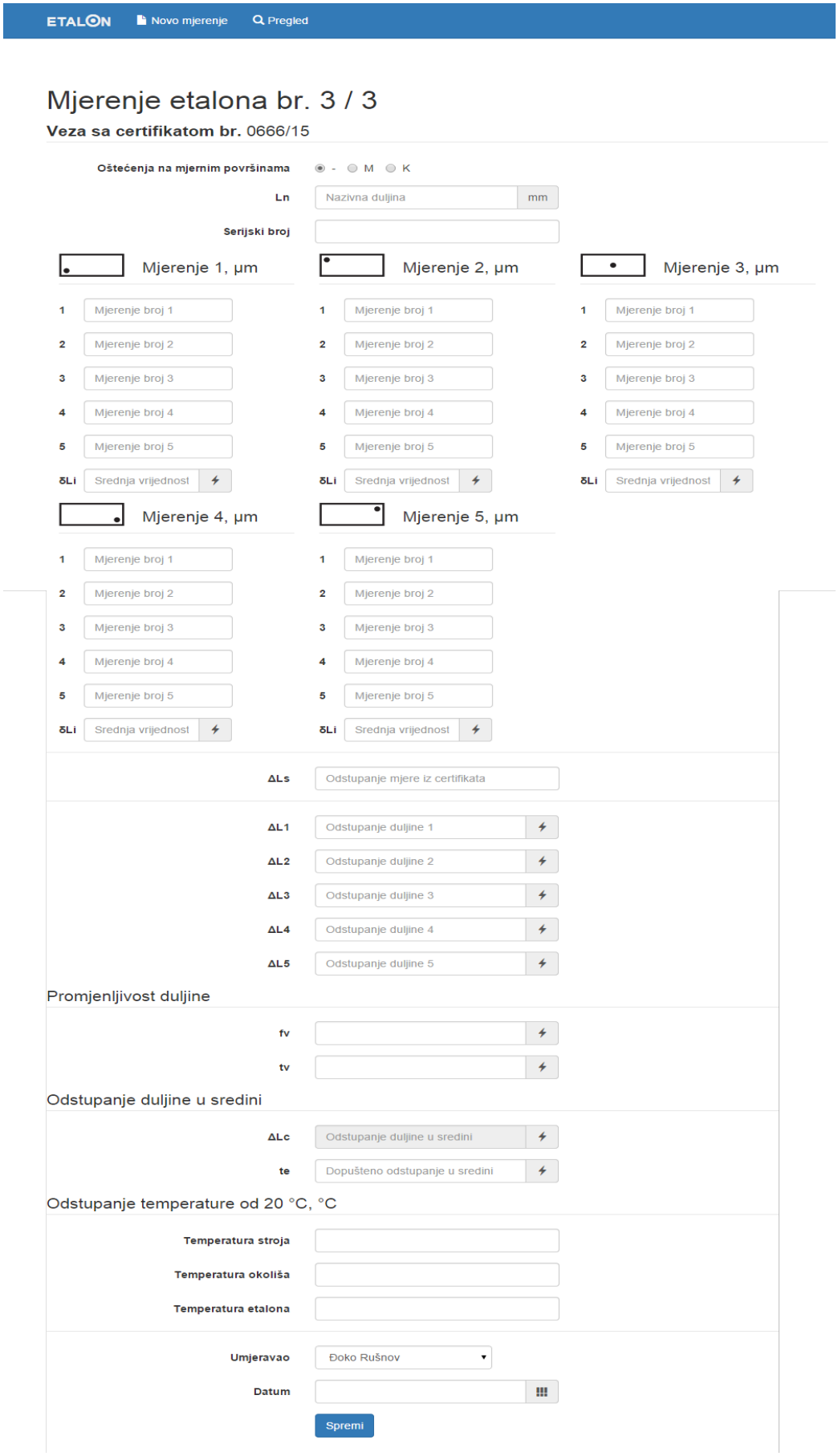
Svi ostali elementi rađeni su na istom principu dok u konačnici nismo rekreirali sve obrasce za unošenje podataka.



Slika 6.1 Sučelje za unos podataka nazivne stranice



Slika 6.2 Sučelje za unos mjerenja etalona duljine klase 1 i 2



Slika 6.3 Sučelje za unos mjerenja etalona duljine klase 00 i 0

7 NAVIGACIJA

Stranica je osmišljena na način da se ponaša kao jedna stranica **indeks.php**, iako se korisniku to tako ne čini. To postižemo PHP-om.

Najprije napravimo jednostavnu stranicu kao što je opisano u poglavlju 4.1, sa **<html>**, **<header>** i **<body>** elementima.
Unutar **<header>** elementa definiramo određene parametre kao što su naslov **<title>** i podršku za hrvatska slova **<meta charset="utf-8">**, te pozovemo sve vanjske datoteke potrebne za ispravan rad naše stranice koje trebaju biti učitane prije otvaranja same stranice.

```
<link href="css/bootstrap.min.css" rel="stylesheet">
<link href="css/etalon.css" rel="stylesheet">
<link rel="stylesheet" href="css/datepicker3.css">
<script src="js/respond.js"></script>
```

U **<body>** stavljamo samo navigacijsku traku i podnožje, tj elemente koji će se prikazivati na svakoj stranici.

Prije zatvaranja **<body>** elementa stavljamo pozive za preostale javascript datoteke koje su nam potrebne za rad stranice, ali koje nisu nužno potrebne prije učitavanja stranice.

```
<script src="js/jquery.js"></script>
<script src="js/bootstrap.min.js"></script>
<script src="js/bootstrap-datepicker.js"></script>
<script src="js/bootstrap-datepicker.hr.js"></script>
```

Svaka vanjska datoteka ima svoju funkciju:

- **bootstrap.min.css** – potreban za korištenje bootstrap klasa na HTML elementima
- **etalon.css** – naši specifični stilovi koji nadglasavaju bootstrap-ove stilove
- **datepicker3.css** – potreban za definiranje izgleda kalendara
- **respond.js** – potreban za dinamično prilagođavanje kostura web stranice i preslagivanje elemenata
- **jquery.js** – javascript biblioteka potrebna za rad drugih javascript datoteka
- **bootstrap.min.js** – omogućava rad pojedinih bootstrap klasa
- **bootstrap-datepicker.js** – omogućuje rad kalendara
- **bootstrap-datepicker.hr.js** – hrvatski jezik za rad prethodne datoteke

Kada bi učitali web stranicu kakva je sada, u njoj osim navigacijske trake i podnožja ne bi bilo ikakvog sadržaja. To ćemo riješit sljedećim PHP kodom koji ubacujemo između navigacijske trake i podnožja.

```
        <?php
if (!isset($_GET['s'])){
  include "stranice/pocetna.php";
}
else if ($_GET['s']==0){
  include "stranice/nazivna.php";
}
else if ($_GET['s']==1){
  include "stranice/klasa12.php";
}
else if ($_GET['s']==2){
  include "stranice/klasa000.php";
}
else if ($_GET['s']==3){
  include "stranice/pregled.php";
}
else if ($_GET['s']==10){
  include "stranice/edit_nazivna.php";
}
else if ($_GET['s']==11){
  include "stranice/edit_klasa12.php";
}
else if ($_GET['s']==12){
  include "stranice/edit_klasa000.php";
}
else{
    include "stranice/nope.php";
}
?>
```

Naredba **include** ubacuje sadržaj druge datoteke u mjesto gdje je naredba napisana.

Koristeći **isset** i **$_GET** naredbu možemo vrlo lako organizirati navigaciju. Naredba **$_GET['s']** dohvaća parametar **s** iz URL adrese, a naredba **isset** provjerava je li parametar uopće zadan u URL adresi, tj **!isset** provjerava nepostojanje parametra.

Tablica 7.1 Navigacija URL adresom kroz web stranicu

URL adresa	include	Opis ubačene stranice
index.php	pocetna.php	Opis završnog zadatka
index.php?s=0	nazivna.php	Obrazac za nazivnu stranicu
index.php?s=1	klasa12.php	Obrazac za etalon klase 1 i 2
index.php?s=2	klasa000.php	Obrazac za etalon klase 00 i 0
index.php?s=3	pregled.php	Pregled svih zapisa
index.php?s=10	edit_nazivna.php	Ažuriranje nazivne stranice
index.php?s=11	edit_klasa12.php	Ažuriranje obrasca za etalon klase 1 i 2
index.php?s=12	edit_klasa000.php	Ažuriranje obrasca za etalon klase 00 i 0
index.php?s=XX	nope.php	Ispiše „Greška, stranica ne postoji.“

Svaki puta kada poželimo otvoriti točno određenu stranicu, navigacija se ostvaruje korištenjem URL adresa iz tablice 7.1.

Ovime postižemo brži rad web stranice jer sve potrebne resurse mora učitati samo jednom, time drastično smanjujući potrebni prijenos podataka sa servera do klijenta.

8 KORIŠTENJE BAZE PODATAKA

8.1 Spajanje na bazu podataka

Spajamo se s bazom podataka koristeći PHP i PDO naredbe. Kod s kojim se spajamo ćemo staviti u zasebnu datoteku **db_connect.php** te je ubaciti u **indeks.php** naredbom **include,** kako bi se ostvarila stalna konekcija sa serverom.

db_connect.php

```
<?php
$host="localhost";
$db_name="etalon";
$username= "root";
$password="";

try{
   $con=new PDO("mysql:host={$host};dbname={$db_name}",$username, $password);
     $con->exec("set names utf8");
   }

// to handle connection error

catch(PDOException $exception){
   echo "Connection error: ".$exception->getMessage();
}
?>
```

8.2 Spremanje u bazu podataka

Spremanje podataka radimo pomoću PHP-a, PDO-a i POST metode.

Za **nazivna.php** stranicu unutar **<form>** oznake mora biti parametar **method="post"**, dok unutar samog **<form>** elementa mora biti **<button>** element s parametrom **type="submit"**. Ako ne stavimo parametar **action** unutar **<form>** oznake, stranica šalje podatke sama sebi, što nama odgovara.

Sljedeći PHP kod stavljamo prije **<form>** elementa. On služi da prepozna **POST** podatke i spremi ih na server, nakon čega temeljem izabrane klase etalona otvara sljedeći obrazac za mjerenja kojem prosljeđuje **id** putem URL-a.

```
<?php
if($_POST){
    try{
        //write query
        $query="INSERT INTO nazivna set id=NULL, veza=?, duljina=?,
proizvodjac=?, mjernoOd=?, mjernoDo=?, brojKomada=?, serijski=?, klasa=?,
model=?, oznaka=?, naslovnik=?, uredjaj=?, pgm=?, umjeravao=?, provjerio=?,
datum=?";
        //prepare query for execution
        $stmt=$con->prepare($query);
```

```
        //bind the parameters
        $stmt->bindParam(1, $_POST['veza']);
        $stmt->bindParam(2, $_POST['duljina']);
        $stmt->bindParam(3, $_POST['proizvodjac']);
        $stmt->bindParam(4, $_POST['mjernoOd']);
        $stmt->bindParam(5, $_POST['mjernoDo']);
        $stmt->bindParam(6, $_POST['brojKomada']);
        $stmt->bindParam(7, $_POST['serijski']);
        $stmt->bindParam(8, $_POST['klasa']);
        $stmt->bindParam(9, $_POST['model']);
        $stmt->bindParam(10, $_POST['oznaka']);
        $stmt->bindParam(11, $_POST['naslovnik']);
        $stmt->bindParam(12, $_POST['uredjaj']);
        $stmt->bindParam(13, $_POST['pgm']);
        $stmt->bindParam(14, $_POST['umjeravao']);
        $stmt->bindParam(15, $_POST['provjerio']);
        $stmt->bindParam(16, $_POST['datum']);

        //execute the query
        if($stmt->execute()){
            echo "<div class='alert alert-success'
role='alert'><b>Spremljeno.</b></div>";
            $id = $con->lastInsertId();
            $klasa=$_POST['klasa'];
            echo $klasa;
            if ($klasa=="1" or $klasa=="2"){
                header ("Location:index.php?s=1&id={$id}");}
            else if ($klasa=="0" or $klasa=="00"){
                header ("Location:index.php?s=2&id={$id}");}
            }else{
                echo "<div class='alert alert-warning' role='alert'>";
                    echo "<b>Neuspješno spremanje!</b> Ponovno unesite
podatke i pokušajte ponovno.";
                    echo "<button type='button' class='close' data-
dismiss='alert' aria-label='Close'>";
                        echo "<span aria-hidden='true'>&times;</span>";
                    echo "</button>";
                echo "</div>";
            }
        }catch(PDOException $exception){ //to handle error
            echo "Error: ".$exception->getMessage();
            }
    }
?>
```

Na stranici **klasa12.php** ili **klasa000.php** gdje nas je poslao **nazivna.php** prvo **GET** metodom čitamo **id** i **redniBroj** ako je dostupan. Pomoću **id** dohvaćamo **brojKomada** etalona duljine i provjeravamo na kojem smo upravo broju. Temeljem **redniBroj<brojKomada** odlučujemo da li ćemo ponoviti postupak prenoseći u sljedeći obrazac putem URL-a osim **id**-a i **redniBroj** ili otići na pregled zapisa ako su sva mjerenje izvršena.

PHP kod smještamo iznad **POST** metode.

```
<?php
if (isset($_GET['id'])){
    $nazivna_id=$_GET['id'];
    $q = $con->query("SELECT veza, brojKomada FROM nazivna WHERE
id=$nazivna_id");
    $f = $q->fetch();
    $brojKomada=$f["brojKomada"];
    $veza=$f["veza"];
}
if (!isset($_GET['n'])){
    $redniBroj=1;
}
else if (isset($_GET['n'])){
    $redniBroj=$_GET['n'];
}
```

8.3 Pregled baze podataka

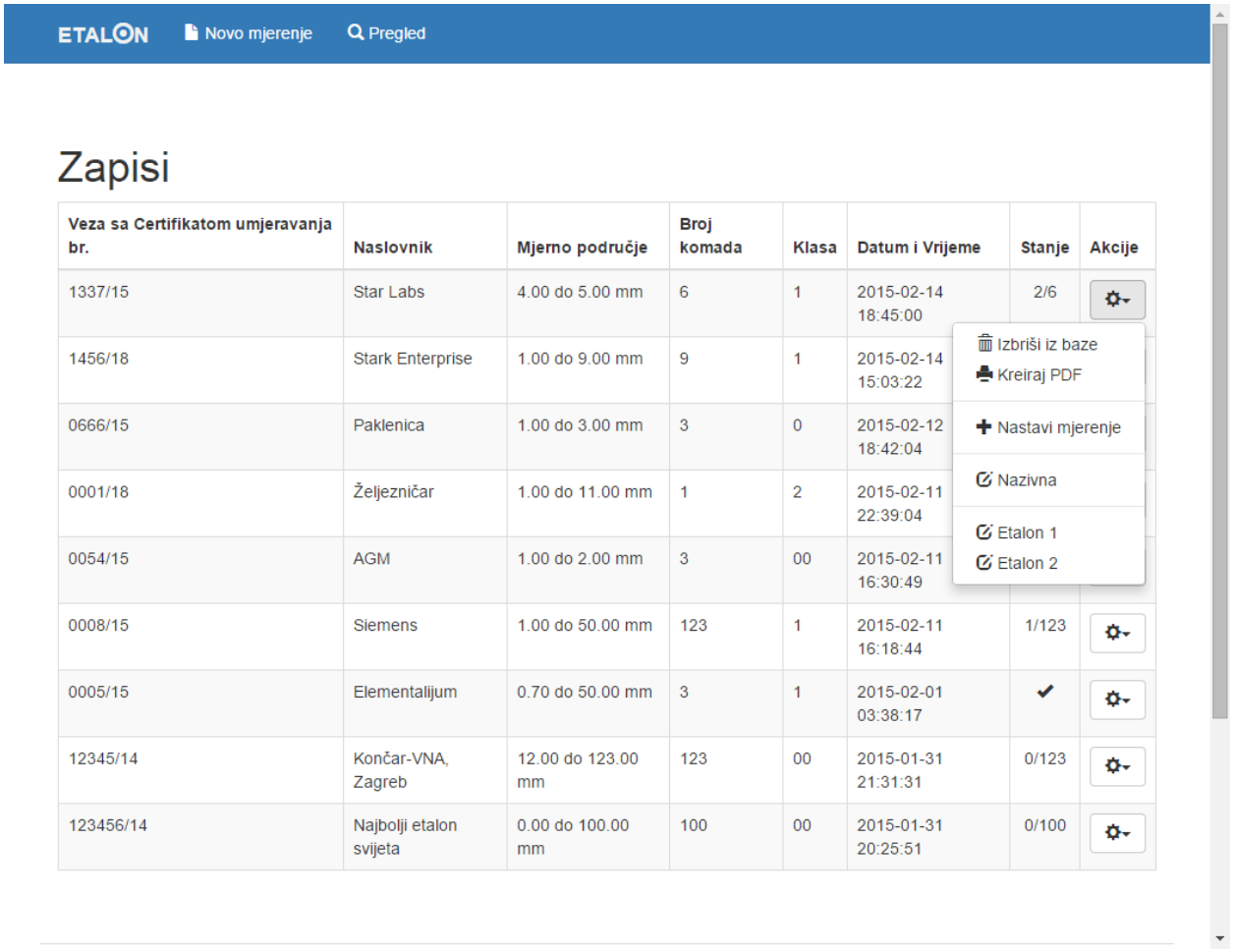
Za pregled baze podataka koristimo PHP, PDO i HTML za formiranje tablice u kojoj se ispisuje.

Unutar **pregled.php** selektirali smo sve podatke iz tablice **nazivna** i koristimo ih redak po redak. Unutar **while** petlje naredbom **extract** deklariramo sve varijable jednog retka unutar PHP-a ali samo u trajanju jedne **while** petlje, u kojoj ispisujemo potrebne podatke i generiramo sve potrebne poveznice sa **id**-ovima vezanim za taj redak tj. garnituru etalona dužine.

```
<?php

     //select all data
     $query="SELECT id, veza, naslovnik, pravoVrijeme, mjernoOd, mjernoDo,
klasa, brojKomada FROM nazivna ORDER BY pravoVrijeme DESC";
     $stmt=$con->prepare($query);
     $stmt->execute();

     $num=$stmt->rowCount();
     if($num>0){
        while ($row=$stmt->fetch(PDO::FETCH_ASSOC)){
             extract($row);
               }
     }
?>
```



Slika 8.1 Prikaz pregleda svih podataka iz tablice *nazivna*, sortirane silazno po vremenu kreiranja

8.4 Ažuriranje baze podataka

Ažuriranje podataka se provodi na gotovo isti način kao i spremanje podataka opisano u poglavlju 8.2.

No, postoji nekoliko razlika. Unutar **edit_klasa12.php** prije korištenja **POST** metode, treba slično kao i u poglavlju 8.3 dohvatiti sve podatke pomoću **id**-a vezane za podatke koje ažuriramo. Nakon što provedemo **extract** naredbu trebamo u svaku **<input>** oznaku dodati parametar **value="<?php echo $XYZ; ?>"**, gdje **$XYZ** označava odgovarajuću varijablu za odgovarajuće polje. Time postižemo da su sve vrijednosti već unesene kada otvorimo web stranicu.

Postoji i razlika u samoj **$query** varijabli kojom šaljemo serveru naredbe, dok je ostatak gotovo isti kao u poglavlju 8.2.

```
$query="UPDATE klasa12 set umjeravao=?, datum=?, ostecenja=?, ln=?,
serijskiBroj=?, mjerenje1=?, mjerenje2=?, mjerenje3=?, mjerenje4=?,
mjerenje5=?, srednjeMjerenje=?, odstupanjeCert=?, odstupanjeSredine=?, te=?,
tempEtalon=?, tempStroj=?, tempOkolis=? WHERE nazivna_id={$nazivna_id} AND
redniBroj={$redniBroj}";
```

8.5 Brisanje baze podataka

Brisanje podataka provodimo javascript-om, PHP i PDO-om unutar **pregled.php**.

Javascript ovdje služi kako bi imali skočni prozor koji nas upozorava na brisanje podataka te da sama URL adresa ne bude direktno dostupna kroz stranicu, već je preusmjeravamo iz javascript-a sljedećom naredbom:

```
window.location="config/delete.php?id="+id;
```

Sam **delete.php**, **GET** metodom uzima **id** garniture etalona i briše sve asocirane retke iz tablica **nazivna**, **klasa12** i **klasa000**. Bitno je napomenuti da je potrebno ponovno izvršiti spajanje na bazu kada koristimo zasebne PHP datoteke koje nisu direktno prikazane kroz **indeks.php**. Opet koristimo naredbu **include "db_connect.php";** za spajanje na bazu.

```
<?php
include "db_connect.php";

$query="DELETE FROM nazivna WHERE id = ?";
$query2="DELETE FROM klasa12 WHERE nazivna_id = ?";
$query3="DELETE FROM klasa000 WHERE nazivna_id = ?";
//prepare the sql statement
try{
    $con->beginTransaction();
  $stmt=$con->prepare($query);
  //bind the id of the record to be deleted
  $stmt->bindParam(1, $_GET['id']);
  $stmt->execute();

    $stmt=$con->prepare($query2);
    $stmt->bindParam(1, $_GET['id']);
  $stmt->execute();

    $stmt=$con->prepare($query3);
    $stmt->bindParam(1, $_GET['id']);
  $stmt->execute();
  //execute the delete statement
  if ($con->commit()){
     //redirect to index page
     //parameter "action=deleted" is used to show that something was deleted
     header ('Location: ../index.php?s=3&action=deleted');

  }else{
     die("Ne mogu zbrisati.");
  }
}catch(PDOException $exception){ //to handle error
     echo "Error: ".$exception->getMessage();
     }
?>
```

9 AUTOMATIZACIJA UNOSA PODATAKA

9.1 Kalendar

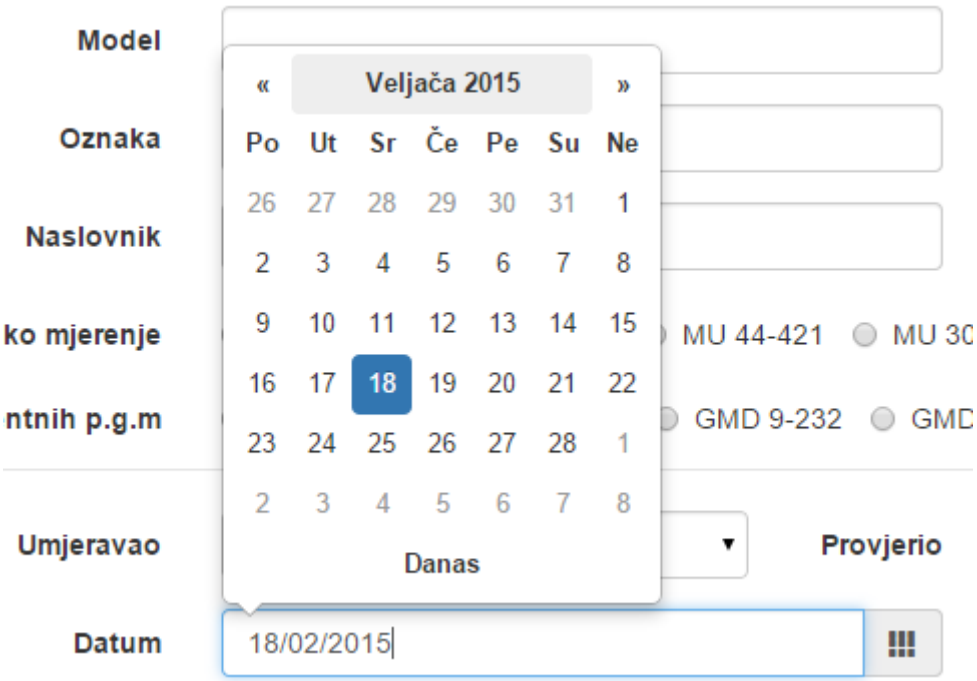
Kalendar je implementiram u polje za unos datuma radi bržeg i lakšeg korištenja.

Da bi to omogućili koristili smo gotovu biblioteku **bootstrap-datepicker** [10], što je nadogradnja na bootstrap. Ona se sastoji od **datepicker3.css**, **bootstrap-datepicker.js** i **bootstrap-datepicker.hr.js**, a čiji je način implementacije objašnjen u poglavlju 7.

bootstrap-datepicker.hr.js nam definira samo jezik korištenja, ali ne i format datuma koji će biti prilagođen našoj državi. To definiramo slijedećim kodom u kojem ponedjeljak definiramo kao prvi dan u tjednu, oblik datuma, jezik i način prikazivanja.

```
<script type="text/javascript">
    $(function () {
    $('#datum').datepicker({
        format: "dd/mm/yyyy",
        weekStart: 1,
        language: "hr",
        orientation: "bottom left",
        todayHighlight: true,
        todayBtn: "linked",
        autoclose: true
         });
    })
</script>
```

Kod ćemo ubaciti pred sam kraj **<body>** elementa, unutar **index.php** datoteke, a jedini uvjet uspješnog rada je korištenje parametra **id=“datum“** unutar **<input>** elementa kojem želimo dodijeliti ovu mogućnost.



Slika 9.1 Implementacija kalendara pomoću bootstrap-datepicker biblioteke

9.2 Izračuni

Izračuni se vrše pomoću javascript-a i HTML DOM naredbi.

U obrascu za klasu 1 i 2, potrebno je izračunati $\overline{\delta L\imath}$ i ΔL_c.

Kako bi izračunali $\overline{\delta L\imath}$ klikom na njegovo polje, potrebno je dodati parametar **onfocus="srednja_vrijednost()"** unutar **<input>** oznake koja definira polje za unos $\overline{\delta L\imath}$. Slično vrijedi i za ΔL_c gdje je potrebni parametar **onfocus=" odstupanje_sredine()"**.

Kako bi se sljedeći javascript kod uspješno izvršavao, potrebno ga je smjestiti unutar, ali na dnu **<body>** elementa te pripaziti da sve **id** oznake imaju jednako ime kao i u javascript-u.

```
<script type="text/javascript">
    function srednja_vrijednost() {
        var s1 =parseFloat(document.getElementById('mjerenje1').value);
        var s2 =parseFloat(document.getElementById('mjerenje2').value);
        var s3 =parseFloat(document.getElementById('mjerenje3').value);
        var s4 =parseFloat(document.getElementById('mjerenje4').value);
        var s5 =parseFloat(document.getElementById('mjerenje5').value);
        var sum = s1 + s2 + s3 + s4 + s5;
        var sred= sum/5;
        document.getElementById('srednjeMjerenje').value = sred.toFixed(3);
    }

    function odstupanje_sredine() {
        var s1 =parseFloat(document.getElementById('srednjeMjerenje').value);
```

```
        var s2 =parseFloat(document.getElementById('odstupanjeCert').value);
        var sum = s1 + s2;
        document.getElementById('odstupanjeSredine').value = sum.toFixed(3);
    }
</script>
```

9.3 Dohvaćanje tablice tolerancija DIN 861

Za određivanje te koristimo tablicu dopuštenih tolerancija po DIN 861 standardu, orijentirajući se prema ln i klasi etalona.

Za dohvaćanje vrijednosti iz tablice **dopusteno** koristimo AJAX metodu.

Kao i u prethodnom poglavlju parametar **onfocus="dohvatibazu()"** stavljamo unutar odgovarajuće **<input>** oznake te sljedeći javascript stavljamo unutar **<body>** elementa pri dnu, pazeći da HTML elementi imaju odgovarajuće **id**-ove koji se koriste u javascript-u.

```
function dohvatibazu() {
    var ln =document.getElementById('ln').value;
    var id= <?php echo $nazivna_id; ?>;
        if (window.XMLHttpRequest) {
            // code for IE7+, Firefox, Chrome, Opera, Safari
            xmlhttp = new XMLHttpRequest();
        } else {
            // code for IE6, IE5
            xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
        }
        xmlhttp.onreadystatechange = function() {
            if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
                document.getElementById("te").value =
parseFloat(xmlhttp.responseText).toFixed(3);
            }
        }
xmlhttp.open("GET","config/sredinabaza.php?id="+id+"&ln="+ln+"&c="+Math.rando
m(),true);
        xmlhttp.send();
}
```

Javascript kod AJAX metode zove i dohvaća odaziv iz datoteke **sredinabaza.php** prenoseći u nju **id** i **ln** parametar. Unutar **sredinabaza.php** se pomoću **id**-a prvo iz tablice **nazivna** dohvaća **klasa**, te se pomoću **klasa** i **ln**-a iz tablice **dopusteno** dohvaća te.

sredinabaza.php

```
<?php
$id = $_GET['id'];
$ln = $_GET['ln'];

include 'db_connect.php';

$q = $con->query("SELECT klasa FROM nazivna WHERE id=$id");
$f = $q->fetch();
$klasa=$f["klasa"];

$q = $con->query("SELECT sredina$klasa FROM dopusteno WHERE nazivnaOd<=$ln
AND nazivnaDo>$ln");
$f = $q->fetch();
$izlaz=$f["sredina$klasa"];

echo $izlaz;
?>
```

9.4 Izbor mjeritelja i provjeravača

Izbor mjeritelja i provjeravača riješen je na poseban način. Umjesto korištenja samo **<select>** i **<option>** elementa, koristili smo PHP i PDO za dohvaćanje tablice **user**.

U sljedećem kodu radimo izbor mjeritelja. Koristimo **<select>** element, unutar kojeg su opcije definirane **<option>** elementom, koji se generira koristeći PHP kod temeljem broja redaka u **user** tablici, gdje je **prava="m"**. Za provjeravače radimo istu stvar sa **prava="p"**.

```
<div class="form-group">
  <label class="col-md-4 control-label" for="umjeravao">Umjeravao</label>
  <div class="col-md-3">
    <select id="umjeravao" name="umjeravao" class="form-control">
        <?php
            $query="SELECT id, ime, prezime FROM users  WHERE prava='m'";
            $stmt=$con->prepare($query);
            $stmt->execute();
            $num=$stmt->rowCount();
            if($num>0){
                while ($row=$stmt->fetch(PDO::FETCH_ASSOC)){
                    extract($row);
                    echo "<option value='{$id}'>{$ime} {$prezime}</option>";
                }
            }
            else {
                echo "<option value='0'>Nema podataka</option>";
            }
        ?>
    </select>
  </div>
</div>
```

Time smo postigli lako dodavanje i uklanjanje korisnika jer je izbor definiran samom tablicom **user** koju možemo lako editirati sa **phpMyAdmin**.

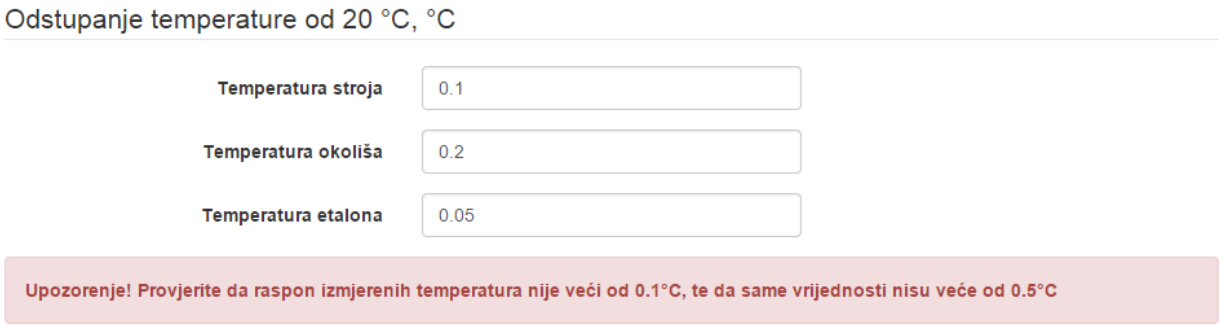
9.5 Upozorenje

Temperatura prilikom mjerenja etalona duljine mora biti strogo kontrolirana. Mjeri se odstupanje od 20°C za stroj, etalon i okoliš. Ako jedna od navedenih odstupanja prelazi 0,5°C ili ako je razlika između međusobno izmjerenih odstupanja veća od 0,1°C, mjerenje je nevažeće. Kako bi na to upozorili korisnika dodali smo ispod samih polja za unos temperatura jedan skriveni ** ** u kojem je ispisano upozorenje koje postaje vidljivo ukoliko razlike temperature pređu 0,1 °C odnosno 0,5 °C.

```
<div id="temperature" class='alert alert-danger' role='alert' style="display:
none">
     <b>Upozorenje! Provjerite da raspon izmjerenih temperatura nije veći od
0.1&deg;C, te da same vrijednosti nisu veće od 0.5&deg;C</b>
    </button>
</div>
```

Slijedi javascript kod koji treba staviti na dno stranice unutar **<body>** elementa. Sama funkcija **temperature()** se aktivira kada se otiđe s polja gdje se unose temperature sljedećim parametrom **onblur="temperature()"** koji se treba smjestiti u odgovarajuće **<input>** oznake, pazeći da svi potrebni elementi imaju iste **id**-ove kao i u javascriptu.

```
<script type="text/javascript">
function temperature(){
        var s1 =parseFloat(document.getElementById('tempStroj').value);
        var s2 =parseFloat(document.getElementById('tempOkolis').value);
        var s3 =parseFloat(document.getElementById('tempEtalon').value);
        if (Math.abs(s1-s2)>0.1 || Math.abs(s1-s3)>0.1 || Math.abs(s2-s3)>0.1
|| Math.abs(s1)>0.5 || Math.abs(s2)>0.5 || Math.abs(s3)>0.5){
            document.getElementById("temperature").style.display = "block";
        }
        else {
            document.getElementById("temperature").style.display = "none";
        }
}
</script>
```



Slika 9.2 Upozorenje uslijed prevelikog odstupanja temperatura

9.6 Zaštita

Prilikom rada s web stranicom, slučajno pritiskanje tipke ENTER pokazalo se problemom, što bi prijevremeno proslijedilo podatke serveru. Javascript-om smo isključili mogućnost takve radnje. Za uspješan rad, slijedeći javascript kod treba dodati unutar **<header>** elementa **indeks.php**-a.

```
<script type="text/javascript">
function stopRKey(evt) {
  var evt = (evt) ? evt : ((event) ? event : null);
  var node = (evt.target) ? evt.target : ((evt.srcElement) ? evt.srcElement :
null);
  if ((evt.keyCode == 13) && (node.type=="text"))  {return false;}
}
    document.onkeypress = stopRKey;
</script>
```

10 KREIRANJE PDF-A

Za kreiranje PDF-a koristimo biblioteke FPDF, FPDI, PHP i PDO.

Napravili smo zasebnu **kreirajpdf.php** datoteku u koju prosljeđujemo **id** retka iz **nazivna** tablice pomoću URL adrese.

Kreiranje PDF-a radimo sljedećim koracima:

- Izradimo PDF predloške pomoću obrazaca prikazanih na slikama 5.1, 5.2 i 5.3.
- Koristeći PDO naredbe dobavljamo sve podatke iz tablice **nazivna** koristeći **GET** metodu i polje **id**. Pomoću **umjeravao** i **provjerio** polja dobavljamo **ime** i **prezime** iz tablice **user**.
- Pozovemo datoteku **fpdf.php** i započinjemo dokument deklaracijom **$pdf =& new FPDI('P', 'pt');** Oznaka **pt** unutar zagrade označava korištenje točaka umjesto milimetara za pozicioniranje što uvelike povećava preciznost.
- Pozovemo datoteku **fpdi.php** i dokument definiramo pomoću predložaka
- Pomoću FPDF naredbi ispisujemo podatke iznad predloška pazeći pritom da koristimo dobre koordinate za svako polje.
- Orijentirajući se po vrsti klase i broju etalona, biramo odgovarajući predložak i učitavamo odgovarajuće podatke iz baze kako bi ih unijeli u predložak.
- Proces počinje od početka sve dok se ne ispišu svi podaci asocirani sa **id** iz **nazivna** tablice.
- Nakon izvršavanja koda generira se PDF dokument

11 ZAKLJUČAK

Web stranica koju smo izgradili mogla je biti izrađena na jako puno različitih načina koristeći različite web alate. Nakon organizacije projekta i izrade konkretne vizije, najveći izazov i najteža odluka je bila izbor web alata koji se slažu sa tom vizijom i mogućnostima koje želimo postići. Tim izborom smo odlučili u što ćemo investirati vrijeme i trud, s rizikom nemogućnosti ostvarivanja određenih funkcija web stranice ukoliko se izbor pokazao lošim. Krajnji izbor nije bila mala selekcija: HTML, CSS, JavaScript, Bootstrap, HTML DOM, PHP, MySQL, PHP PDO, AJAX, FPDF i FPDI.

Prilikom rada sa tolikim brojem alata rastao je i broj potencijalnih problema. Jedna od poteškoća je prenositi varijable iz jednog programskog jezika u drugi kako bi osigurali dostupnost varijabli svim segmentima stranice. Često su nastajali problemi i sa slabo dokumentiranim korištenjem hrvatskih znakova te problemi kompatibilnosti određenih biblioteka.

Kao rezultat svega navedenog imamo modernu interaktivnu web stranicu koja komunicira sa bazom podataka, automatski računa potrebna polja, izvlači podatke iz DIN 861 standarda, te nudi mogućnosti lakog ažuriranja podataka i konačno generiranje PDF certifikata, time optimizirajući posao umjeravanja etalona i čineći život nekolicini mjeritelja lakšim.

Uzevši u obzir mogućnosti web stranica, uvijek se može napraviti više: dodavanje sloja sigurnosti za pristup podacima, pretraživanje podataka, dodavanje komunikacije sa mjernim uređajima za automatski unos podataka itd. Umrežavanjem na takvu web stranicu bi se postiglo jedinstveno rješenje za potrebe svih Laboratorija za mjerenje etalona u Hrvatskoj, sa unificiranom bazom podataka za dobivanje bržih i točnijih rezultata.

LITERATURA

[1] B. Runje, *Istraživanje mjernih nesigurnosti u postupcima umjeravanja etalona duljine*, Zagreb, 2002.

[2] http://www.codecademy.com/ - *HTML&CSS i PHP*

[3] http://www.w3schools.com/

[4] http://getbootstrap.com/

[5] Jen Kramer, *Up and Running with Bootstrap 3*, http://www.lynda.com/

[6] https://www.codeofaninja.com/2011/12/php-and-mysql-crud-tutorial.html

[7] http://php.net/manual/en/

[8] http://www.fpdf.org/

[9] http://www.setasign.com/products/fpdi/manual/

[10] http://eternicode.github.io/bootstrap-datepicker

RESURSI

- XAMPP - https://www.apachefriends.org/index.html
- Bootstrap - http://getbootstrap.com/
- jQuery - http://jquery.com/download/
- respond.js https://github.com/scottjehl/Respond
- bootstrap-datepicker - http://eternicode.github.io/bootstrap-datepicker
- FPDF - http://www.fpdf.org/
- FPDI - http://www.setasign.com/products/fpdi/downloads/