Upotreba novih alata i procesa pri izradi web stranice

Džambo, Ivica

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Sveučilišni studij

UPOTREBA NOVIH ALATA I PROCESA PRI IZRADI WEB STRANICE

Završni rad

Ivica Džambo

Osijek, 2016.

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom sveučilišnom studiju

Osijek, 07.09.2016.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada

Ime i prezime studenta:	Ivica Džambo
Studij, smjer:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R3585, 31.07.2013.
OIB studenta:	51000805082
Mentor:	Doc.dr.sc. Mirko Kohler
Sumentor:	
Naslov završnog rada:	Upotreba novih alata i procesa pri izradi web stranice
Znanstvena grana rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Predložena ocjena završnog rada:	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 Postignuti rezultati u odnosu na složenost zadatka: 2 Jasnoća pismenog izražavanja: 2 Razina samostalnosti: 3
Datum prijedloga ocjene mentora:	07.09.2016.
Datum potvrde ocjene Odbora:	12.09.2016.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

IZJAVA O ORIGINALNOSTI RADA

Osijek, 13.09.2016.

Ime i prezime studenta:	Ivica Džambo
Studij:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R3585, 31.07.2013.
Ephorus podudaranje [%]:	0

Ovom izjavom izjavljujem da je rad pod nazivom**: Upotreba novih alata i procesa pri izradi web stranice**

izrađen pod vodstvom mentora Doc.dr.sc. Mirko Kohler

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

Sadržaj

1. UVOD

Razvoj današnjih tehnologija doveo je do toga da u današnje vrijeme imamo puno programskih jezika i alata kojima možemo razvijati vlastite aplikacije i web stranice. Isključivo taj napredak tehnologija u većini slučajeva programere prisiljava da proučavaju nove vrste baš tih tehnologija, što znači da su spremni cijeli život učiti i pripremati se za 'nešto' novo što dolazi. S obzirom da je veliki problem ovog posla vrijeme, pomoću učenja novih tehnologija i raznih alata, ušteda vremena i skraćivanje koda ima veliku prednost jer je to ono što je zbilja potrebno kako bi se projekti završavali prije vremena koje je predviđeno za izradu. Upravo glavni cilj završnog rada je prikazati upotrebu novih alata i procesa pri izradama raznih aplikacija i web stranica. Također je bitna usporedba između klasičnog HTML [1] i CSS [2] koda te koda novih alata i procesa, prikazati prednosti i nedostatke te vremensku uštedu koja igra veliku ulogu.

Budući da se radi o web tehnologiji, korišteni su adekvatni programski jezici i alati koji su neophodni za izradu istog. Pri oblikovanju samog izgleda stranice u cilju krajnjeg prikaza korisniku korišteni su jezici Jade/Pug i SASS, dok alati koji su potrebni za implementiranje tih datoteka i jezika korišteni su Node.js i gulp.js. Alat za pisanje koda koji je korišten je Sublime 3 Text Editor koji se može preuziti na [3].

U drugom poglavlju se radi o problematici i ideji stvaranja web stranice pomoću novih alata. Treće poglavlje govori o tehnologijama koje su korištene pri izradi stranice, te o novim alatima i procesima koji nam pomažu pri izradi, pomoću kojih štedimo vrijeme i dobivamo jednostavnost. Četvrto poglavlje donosi detaljan opis korištenja alata koji prevode nove jezike, te opisuju detaljan postupak instalacije te korištenja istih. Također su opisane razne mogućnosti koje su moguće s novim alatima. Posljednje, peto poglavlje je poglavlje analize i prikaza web stranice.

1.1. Zadatak završnog rada

Zadatak završnog rada je prikazati uporabu novih alata i procesa, predprocesorskih naredbi pri izradi web stranice. Omogućiti prikaz novih jezika i alata te ih usporediti sa starijim primjerima programskih jezika i alata te pomno opisati i obrazložiti svrhu svakog novog alata.

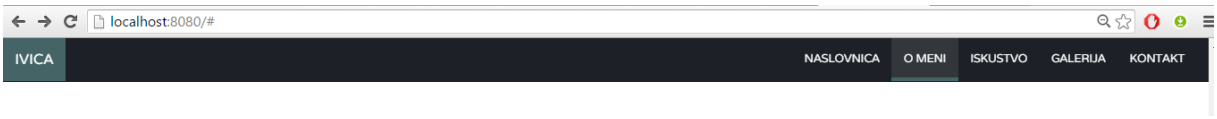
2. PROBLEMATIKA I IDEJA STVARANJA WEB STRANICE

Razvojem i svakodnevnim korištenjem Interneta javila se potreba za nečim boljim i jednostavnijim, što će olakšati posao svima koji se koriste i bave programiranjem. Jako je bitno da razlike između novih i starih alata ili jezika nije velika, točnije rečeno bitno je da je princip i logika svih tih alata ista što je velika prednost. Svaki novi alat ili jezik je tu da pomogne isključivo nama, da nekakav problem riješimo na nama lakši način i da se maksimalno uštedi vrijeme, dok je to uglavnom bila problematika kod starijih i ostalih alata. Također dolaskom novih alata i jezika dolaze i nove mogućnosti u programiranju, što je još jedna pozitivna stvar koja karakterizira iste. Upravo te činjenice odvlače sve korisnike starijih jezika i alata da se privikavaju na nove mogućnosti jer je to ono što se trenutno traži.

Web stranice je osmišljena tako da sadržaj iste bude vlastita privatna stranica, odnosno životopis (portfolio). Portfolio se sastoji od 5 izborničkih traka:

- Naslovnica
- O meni
- Iskustvo
- Galerija
- Kontakt

Po nazivima izborničkih traka se vidi da se radi o klasičnim opisnim stvarima koje bi svaki životopis trebao sadržavati, gdje se radi o raznim opisima, fotografijama u galeriji, fotografijama u traci „Iskustvo“ gdje su uz slike i detaljni opisi svih iskustava. Također tu su i kontakt podaci te razni linkovi i zanimljivi „progress bar-ovi“. Sve trake povezane su funkcijama te možemo bez problema prelaziti jednostavnim klikovima iz jedne u druge.



Sl. 2.1. *Izborničke trake*

3. TEHNOLOGIJE, NOVI ALATI I PROCESI

Web sučelje vlastite stranice kreirano je pomoću više zasebnih tehnologija kao što su Jade/Pug, SCSS/SASS te alat koji prevode baš te jezike Node.js i alat napisan u samom Node-u GULP.

3.1. JADE/PUG

Pug je programski jezik koji u današnjem svijetu programiranja igra veliku i bitnu ulogu. To je predložak koji HTML i JavaScript kod [4] oblikuje u posve drugačiji i skraćeni oblik koda te koji se prevodi preko drugih alata koje ćemo poslije spomenuti.

Jade je stari naziv koji je morao biti promijenjen zbog autorskih prava, te autori upravo toga jezika su se bez problema uspjeli dogovoriti oko novog naziva jer je to zapravo i najmanje bitno. U daljnjem tekstu ćemo koristiti naziv Pug.

Na slici 3.1. možemo vidjeti primjer usporedbe PUG predloška i HTML koda te vidjeti kako je Pug zapravo „čist“ kod iz razloga što ne sadrži dodatne znakove i nepotrebne i zamarajuće stvari.

```
doctype html
html(lang="en")
  head
    title= pageTitle
    script(type='text/javascript').
      if (foo) {
         bar(1 + 5)
      }
  body
    h1 Jade - Node Predložak
    #container.col
      if youAreUsingJade
        p Ti si super!
      else
        p Nabavi ga obavezno!
      p.
        Pug je jednostavan i kratak template
        jezik sa naglaskom na puno performansi
        i jako snažnim značajkama.
```

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Jade</title>
    <script type="text/javascript">
      if (foo) {
         bar(1 + 5)
      }
    </script>
  </head>
  <body>
    <h1>Jade - Node Predložak</h1>
    <div id="container" class="col">
      <p>Ti si super!</p>
      <p>
        Pug je jednostavan i kratak template
        jezik sa naglaskom na puno performansi
        i jako snažnim značajkama.
      </p>
    </div>
  </body>
</html>
```

Sl. 3.1. *Usporedba PUG i HTML*

Pug je korišten za izgled web stranice, točnije rečeno Pug je zamijenio klasični HTML koji se uobičajeno i koristi za izgled stranice. Nedostatak Pug-a za razliku od HTML-a je u tome što je Pug osjetljiv na bilo kakav razmak ili prazninu u kodu dok kod HTML-a to ne stvara nikakve probleme.

Iz razloga što je Pug trenutno novi alat koji je za neke nejasan i nerazumljiv, tipičnim primjerima usporedbe koda se može prikazati najveća prednost istoga, a to su jednostavnost i brzina pisanja koda. Prikazat ćemo usporedbu i izgled izborničke trake koji u sebi sadrži „progress-bar“ koji je u HTML kodu dosta kompliciran i zahtjevan dok je u Pugu moguće odraditi vrlo jednostavno pomoću *for* petlje.

Pug dio koda:

```
- var topics = [{ name: 'Programiranje I i II', likeability: '89' }]
  - topics[1] = { name: 'Osnove elektrotehnike I i II', likeability: '82'}
  - topics[2] = { name: 'Vjerojatnost i statistika', likeability: '60' }
  - topics[3] = { name: 'Matematika I, II i III', likeability: '76' }
  - topics[4] = { name: 'Objektno orijentirano programiranje', likeability: '92' }
//primjer for petlje u Jade template-u.
for topic, i in topics
              .wrapp-progress
                p.pull-left #{topic.name}
                .progress
                  .progress-bar.clearfix(role='progressbar',
                                         aria-valuenow=topic.likeability,
                                         aria-valuemin='0', aria-valuemax='100',
                                         style='width: 0%')
                    span.precent-value
                    i.fa.fa-caret-down
```

Sl. 3.2. *Pug dio koda – Primjer for petlje*

Na slici 3.2. možemo vidjeti jednostavnost Pug jezika gdje se dodavanje novih i sličnih stvari obavlja preko jednostavne *for* petlje, gdje *item* predstavlja broj elemenata u nizu, dok je *„i“* indeks u istom tome nizu. Također možemo vidjeti da se „topic-name“ poziva preko *mixina* [5] koji je objekt koji se poziva preko funkcija koje se nalaze u glavnoj mapi gdje se nalaze i pojedini Bootstrapi [6] uz koje smo si pomogli na vanjskom izgledu stranice..

Nakon što smo vidjeli kako nam izgled Pug jezik, na slici 3.3. možemo pogledati isti tu deklaraciju „progress-bara“, ali prevedenu u HTML jezik. Možemo vidjeti kako je deklaracija kolegija u Pugu deklarirana u nekoliko redova dok je u HTML-u to dosta kompliciranije i bitno je raditi sve posebno dok je u Pugu to moguće odraditi preko jednostavne *for* petlje.

Na slici 3.3. su navedeni primjeri samo tri kolegija iz razloga što deklariranje svih zauzima previše prostora.

Pug kod preveden u HTML:

```
<div class="col-md-6">
        <div class="build progressbar-a">
          <div class="wrapp-progress">
            <p class="pull-left">Programiranje I i II</p>
            <div class="progress">
              <div role="progressbar" aria-valuenow="89" aria-valuemin="0"
aria-valuemax="100" style="width: 0%" class="progress-bar clearfix"><span class="precent-value">
</span><i class="fa fa-caret-down"></i></div>
            </div>
          </div>
          <div class="wrapp-progress">
            <p class="pull-left">Osnove elektrotehnike I i II</p>
            <div class="progress">
              <div role="progressbar" aria-valuenow="82" aria-valuemin="0"
aria-valuemax="100" style="width: 0%" class="progress-bar clearfix">
<span class="precent-value"></span><i class="fa fa-caret-down"></i></div>
            </div>
          </div>
          <div class="wrapp-progress">
            <p class="pull-left">Vjerojatnost i statistika</p>
            <div class="progress">
              <div role="progressbar" aria-valuenow="60" aria-valuemin="0"
aria-valuemax="100" style="width: 0%" class="progress-bar clearfix">
<span class="precent-value"></span><i class="fa fa-caret-down"></i></div>
```

Sl. 3.3. *HTML dio koda – preveden iz Pug dijela*

Također treba napomenuti kako je *for* petlja korištena i u izborničkim trakama *Galerija* i *Iskustva*, baš iz toga razloga što nam je tu najlakše pokazati jednostavnost i skraćenost Pug predloška.

Jednostavnost i značajke Puga se još mogu dokazati kroz razne primjere usporedbe sa raznim programskim jezicima, ali vjerujem da je ovo sasvim dovoljno da se pokaže koliko Pug može biti koristan u budućnosti, te dodatne stvari možete potražiti na službenim stranicama Puga.[7]

3.2. SCSS/SASS

Jednostavnost CSS-a je uvijek bila njegova najpoznatija karakteristika iz razloga što su CSS stilovi samo dugačke liste pravila koje se sastoje od različitih stilova koje se koriste za uređivanje web stranica. Napredak tehnologije je doveo do još jednog velikog problema iz razloga što su današnje aplikacije i web stranice sve složenije te su ti CSS stilovi još veće i duže liste.

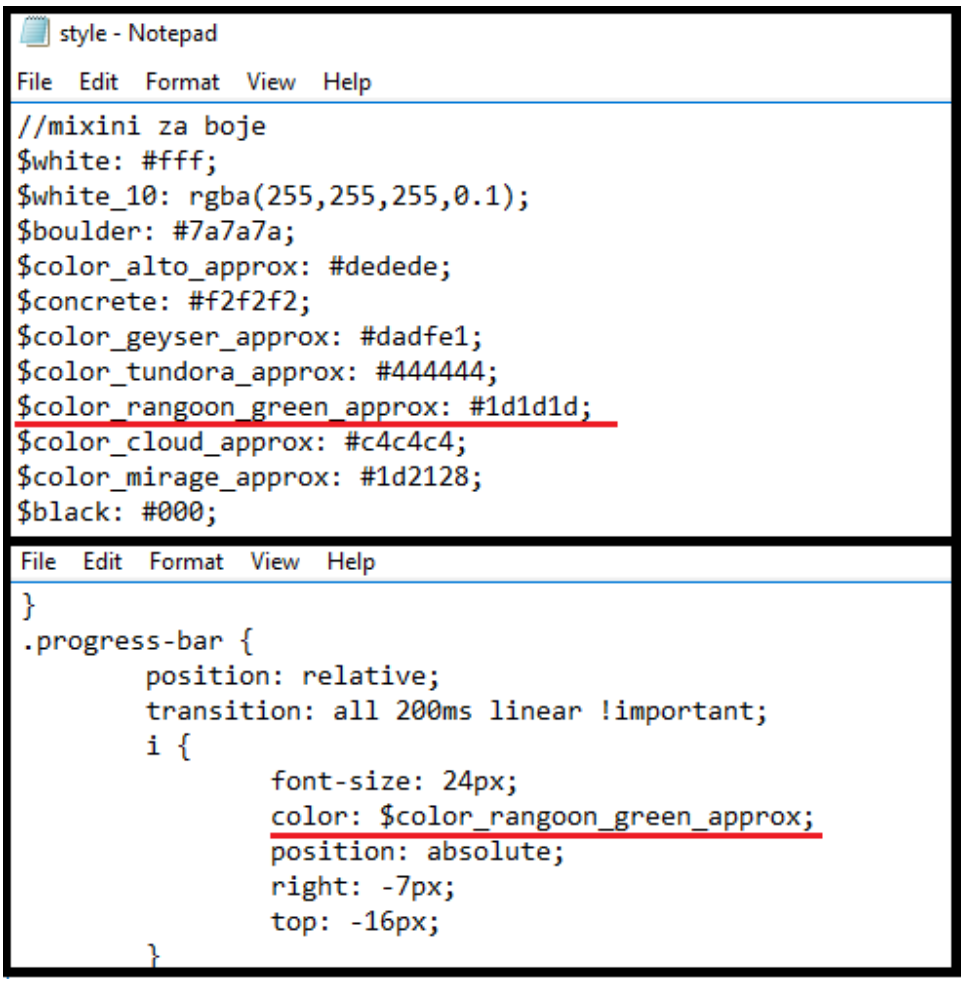
Jedino rješenje je bilo da se kreira nova sintaksa CSS-a čiji je cilj da olakša upravljanje kompliciranim CSS datotekama. Lakše upravljanje baš tim datotekama se postiže upotrebom predprocesora (programa koji se izvršavaju na vašem računalu ili serveru) koji prevodi jezik SCSS u stari klasični CSS koji internet preglednici razumiju.

SASS [8] je predprocesor, točnije rečeno sloj između stilova kojima mi upravljamo i *.css* datoteka koje prosljeđujemo serveru. SASS popunjava rupe u CSS-u kao jeziku, dozvoljavajući vam da pišete kod koji će biti brži, efikasniji i lakši za održavanje. Sve to opisuje i sam naziv SASS koji je skraćenica od „syntactically awesome style sheets“.

3.2.1. SCSS prednost umetaka

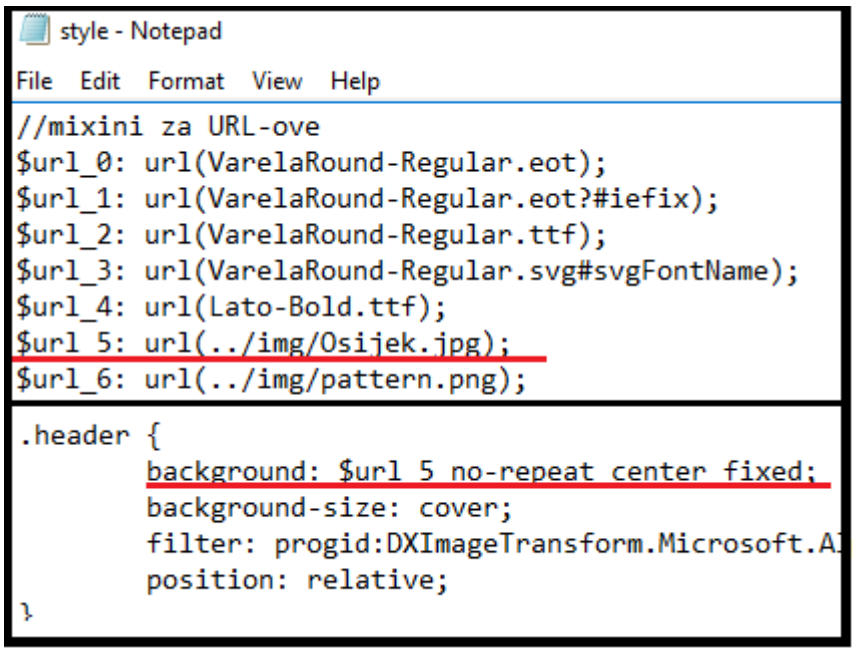
Za razliku od regularnog CSS-a, SASS je prvi skriptni jezik sa izrazima, funkcijama, varijablama, uvjetnom logikom i petljama, što ćemo i pokazati sljedećim primjerima.

Na slici 3.4. možemo vidjeti upravo primjer povezivanja referenci, takozvani *mixin (umetak),* gdje unaprijed preko stringa (znaka) $ deklariramo kao u primjeru, te ih poslije pozivamo preko istog toga znaka, bez potrebe za ponovnim pisanjem. Crvenom linijom (gornji dio slike) označena je određena nijansa boje koja nam je potrebna za uređivanje određenom dijela teksta koja je u određenoj klasi pozvana na ispravan način (donji dio slike). Određene boje i fontovi su preuzeti sa određenih Framework-ova.



Sl. 3.4. *Pozivanje mixina za boju u SCSS predlošku*

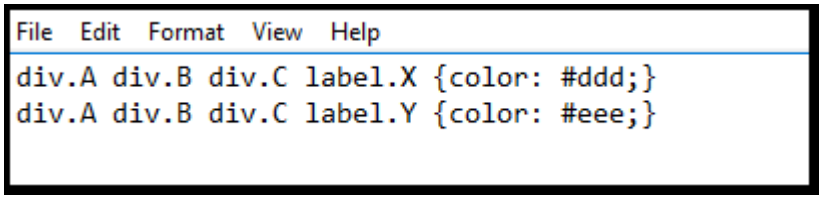
Na slici 3.5. također možemo vidjeti kako se na isti način mogu ranije deklarirati i posebno određene veličine slova te posebni linkovi za koje nije potrebno ništa više od pozivanja sa već spomenutim znakom i istim nazivom varijable (fonta ili url-a).



Sl. 3.4. *Pozivanje mixina za url u SCSS predlošku*

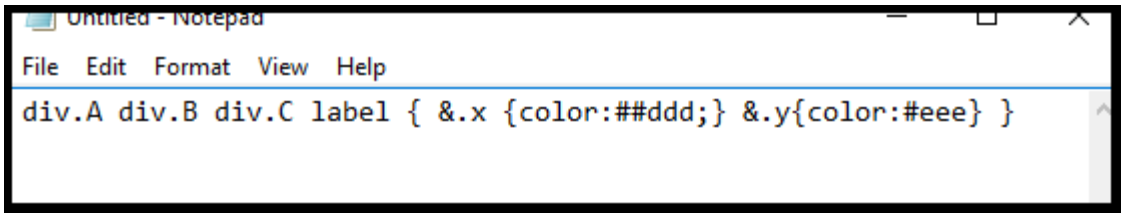
3.2.2. SCSS strukturiranje

Nakon primjera povezivanja referenci i umetaka, velika prednost u SCSS jeziku ima i strukturiranje gdje izrazi izgledaju urednije i povezani. Razlog je upravo taj što je svrha svakog većeg projekta da linije koda budu složene i povezane te bez nepotrebnih linija koje SCSS strukturira. Na slici 3.6. možemo vidjeti primjer linije koda u CSS-u. Kod je izmišljen isključivo za prikaz prednosti SCSS jezika.



Sl. 3.6. *Strukturiranje u CSS-u*

Možemo vidjeti kako se u ovakvom primjeru svaka linija mora posebno pisati jer su nam *label-i* različiti, upravo to je moguće spojiti i pojednostaviti u SCSS-u. Na slici 3.7. možete vidjeti primjer strukturiranja i povezivanja koje nam omogućava SCSS.



Sl. 3.7. *Strukturiranje u SCSS-u*

U prethodnom primjeru su korištene samo dvije varijable *label* koje su služile samo za prikaz, te napominjemo kako u složenijim projektima gdje ovakvih sličnih varijabli koje se mogu povezati ima i na desetke te čak i stotine, razumijem se koliku prednost SCSS jezik.

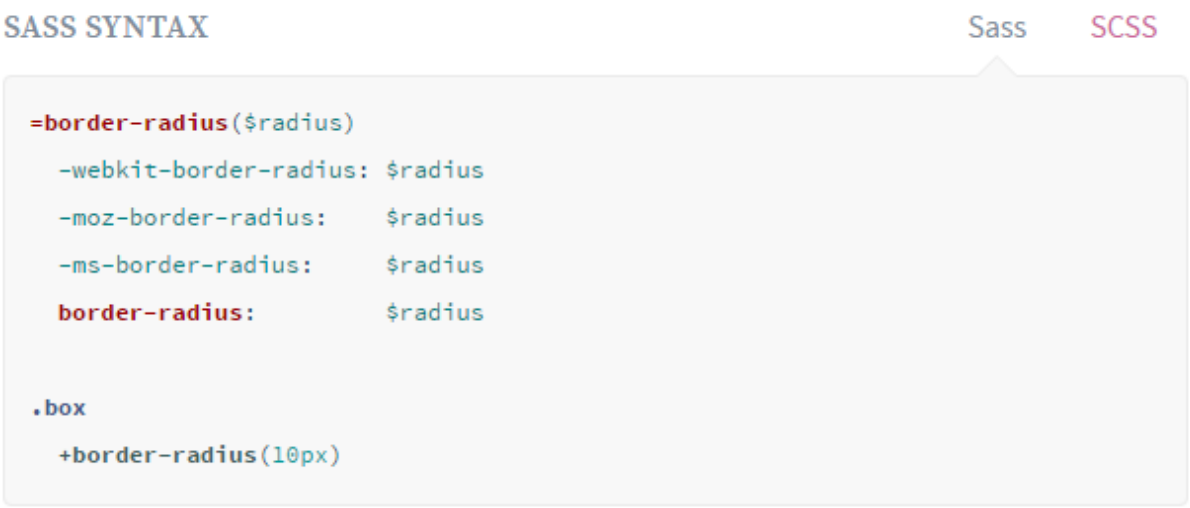
3.2.3. Razlike SASS i SCSS

Nakon pojašnjavanja jezika SASS/SCSS, vrijeme je da ih razdvojimo te pojasnimo da zapravo i nisu ista stvar iako bi se svaka osoba koja se susrela sa obje vrste koda složila s time da je ista stvar. Po svemu sudeći SASS i SCSS odrađuju istu stvar i njihova uloga u novom svijetu programiranja je ista, razlikuju se za nijansu. Tu nijansu upravo čine određeni znakovi koje je najjednostavnije dočarati prikazom usporedbe u kodu.

Na slici 3.8. možete vidjeti SCSS dio koda koji u sebi sadrži dodatne znakove koji su osjetljivi i jako bitni u kodu kada je taj jezik u pitanju.

Sl. 3.8. *SCSS primjer koda*

Dok je SCSS jezik malo više osjetljiv i „teži“ baš iz toga razloga što se mora paziti na znakove kao što su zagrade, dvotočke i ostali znakovi koji povezuju i spajaju klase u kodu, u SASS-u je to dosta jednostavnije u usporedbi na SCSS kod. Pogledajte sliku 3.9. te je usporedite sa prethodnom slikom i primjer SCSS koda te će biti jasnije u čemu je velika prednost SASS jezika.



Sl. 3.9. *SASS primjer koda*

Možemo primijetiti da ovo zapravo i nisu velike razlike između dvije vrste koda, točnije rečeno samo su drugačiji nazivi i male izmjene u kodu. Možemo reći da je SASS malo „ispred“ SCSS jezika iz razloga što je u neku ruku jednostavniji i pregledniji, dok je nekima preglednije ukoliko svoje klase i redove odvajaju zagradama gdje im je jasnije što su radili. Moramo napomenuti kako većini programera SASS predstavlja isto što i SCSS stoga ne želimo praviti razliku između ta dva jezika, nego jednostavno možemo samo pobliže objasniti svrhu i korisnost istih što smo se nadam se postigli.

3.3. JavaScript

Nakon pojašnjavanja bitnijih tehnologija koje su korištene, na red je stigao i JavaScript. Iako često povezivan s programskim jezikom Java, ovdje se priča o potpuno drugoj vrsti tehnologije. JavaScript je najpopularniji skriptni jezik na Internetu kojeg podržavaju svi poznatiji preglednici poput Internet Explorera, Mozzila, Opera itd.

Razvijen je od tvrtke pod nazivom Netscape, a cilj kreiranja JavaScripta je bio dodavanje interaktivnosti HTML stranicama. JavaScript je zapravo interpreter, što zapravo znači da se skripta izvršava naredbu po naredbu bez da je prije toga izvršeno prevođenje (kompajliranje) cijelog programa i kreiranje izvršne datoteke. Baš iz toga razloga nam omogućuje izradu dinamičkih web stranica u kombinaciji s HTML i CSS tehnologijama. Također nam omogućuje kreiranje varijabli za pohranu podataka te pozivanje mnogobrojnih metoda koje stranicu čine zanimljivijom za korisnika. Svi koji imaju bilo kakvog znanja o programiranju, neće imati problema sa učenjem JavaScripta iz razloga što je jako sličan jezicima poput C#, Jave, C++ i drugih.

U ovome radu korišten je JavaScript za dodavanje interaktivnosti i poboljšanju izgleda same web stranice. Korišteno je nekoliko funkcija za animacije, slike te samo „buttone“ koji odrađuju određenu funkciju. Na slici 3.10. možete vidjeti funkciju za „progress bar“ koja u sebi poziva drugu funkciju koja točnije deklarira vrijeme učitavanja te same postotke koje traka učitavanja treba sadržavati.

```
function barScroll(){
        setTimeout(function(){

        $('.progress-bar').each(function() {
                var me = $(this);
                var pe =  $(this).children('.precent-value');
                var perc = me.attr("aria-valuenow");

                var current_perc = 0;

                var progress = setInterval(function() {
                        if (current_perc>=perc) {
                                clearInterval(progress);
                        } else {
                                current_perc +=1;
                                me.css('width', (current_perc)+'%');
                        }

                        pe.text((current_perc)+'%');

                },90);
        });
        }, 300);
}
```

Sl. 3.10. *Funkcija za „progress bar“*

Prema rasporedu web stranice, traka s postotcima se ne nalazi na naslovnoj stranici te je potrebno u JavaSriptu podesiti da se postotci ne učitavaju pokretanjem stranice, već podesiti „okidač“ koji će pokrenuti učitavanje postotaka na trakama. U ovom slučaju nam je taj okidač dugme „Saznajte više“ koje se nalazi na samoj web stranici, ili u slučaju kao na slici 3.11., gdje nam je upravo to dugme nazvano „.btnAbout“. Klikom na dugme se poziva funkcija te se izvršava pokretanje trake s postotcima.

```
$('.precent-value').hide();
//VS: skrivamo sve trake s postotcima dok ne krenemo puniti progress bar.
$('.btnAbout').click(function () {
//VS: Postavljamo pokretanje progess barova po kliku na dugme.
        $('.precent-value').show();
//VS: Prikazujemo trake s postotcima.
        barScroll();
});
```

Sl. 3.11. *Okidač za „progress bar“*

Skriptnim jezikom JavaScript bi zaključili detaljan opis tehnologija koji je korišten u radu te bi se u sljedećem dijelu rada potrudili detaljnije opisati alate pomoću kojih prevodimo nove jezike.

4. NODE.js

Nakon pojašnjavanja jezika i tehnologija koje služe za izgled i princip rada određenih dijelova stranice, na red je došlo pojašnjenje i svega onoga što se nalazi u pozadini svega toga. Točnije rečeno pojasniti ćemo sami princip prevođenja svih tih novih jezika, iz razloga što je računalo potrebno novi jezik prevesti u njemu razumljivi jezik, a za to je u našem slučaju zaslužan alat Node.js [9]. Node uvelike pomaže za izgradnju brzih i interaktivnih stranica te pomaže programerima pri izradi stranica baš zbog svih svojih specifičnosti i prednosti. U nastavku teksta će detaljnije biti pojašnjen sam princip korištenja Node-a kao alata potrebnog za implementiranje stranice uz korištenje novih jezika.

4.1. Što je Node.js

Node.js ili jednostavnije Node, podloga je za izgradnju brzih, modernih i stabilnih aplikacija. Prednost upravo te podloge, ili kako je programeri nazivaju platforme, je u tome da tu platformu čini i ugrađena biblioteka za HTTP i IO.socket komunikaciju što zapravo znači da je Node i web server bez dodatnih serverskih softwarea. Iz toga razloga upravo ta platforma ima velike prednosti jer je pisanje serverskog koda u samoj platformi zapravo i pisanje samo servera.

Node.js, kao što i naziv alata govori, ima nekakvu povezanost s JavaScript jezikom, što zapravo možemo reći da i je jako bitna poveznica između njih. U samoj pozadini Nodea nalazi se Googleov V8 JavaScript engine koji izvodi kod i velik broj osnovnih modula koji su napisani u JavaScriptu. Iako se većina programera ne bi složila s tim da pišu „server-side“ kod u JavaScriptu, može se reći da i nisu prepoznali neke od prednosti baš JavaScripta. Glavni razlog je taj što nudi velik broj biblioteka koje olakšavaju izradu aplikacija. Također je istina da JavaScript kod dopušta i različite „nečistoće“ po pitanju strukture koda, ali ako se koriste određeni principi objektno orijentiranog programiranja i dizajna, kod može biti pregledniji, kvalitetniji i jednostavan za održavanje.

4.2. Kreiranje web servera

Na slici 4.1. možete vidjeti primjer kreiranja web servera pomoću Nodea. Node pruža modul koji se može koristiti za stvaranje HTTP poslužitelja. U primjeru spajanja s poslužiteljem koristimo 8081 port. Prije svega potrebno je kreirati datoteku po nazivom *server.js* te u nju upisati kod kao na slici.

```
var http = require('http');
var fs = require('fs');
var url = require('url');


// Kreiranje servera
http.createServer( function (request, response) {
   // Analiziranje zahtjeva s nazivom datoteke
   var pathname = url.parse(request.url).pathname;
   // Ispis naziva datoteke
   console.log("Request for " + pathname + " received.");
   // Čitanje datoteke zahtjeva
   fs.readFile(pathname.substr(1), function (err, data) {
      if (err) {
         console.log(err);
         response.writeHead(404, {'Content-Type': 'text/html'});
      }else{
         //Stranica pronađena
         response.writeHead(200, {'Content-Type': 'text/html'});
         response.write(data.toString());
      }

      response.end();
   });
}).listen(8081);

// Ispis poruke
console.log('Server running at http://127.0.0.1:8081/');
```

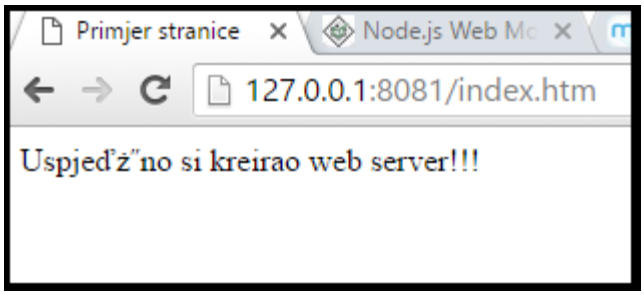
Sl. 4.1. *Sadržaj datoteke „server.js“*

Nakon što se ispiše adresa servera, potrebno je kreirati HTML datoteku u istoj mapi na računalu gdje se nalazi i datoteka *server.js*, te nakon toga pisati u HTML datoteku ovisno o namjeri. Na slici 4.2. pogledajte primjer HTML datoteke.

```
<html>
<head>
<title>Primjer stranice</title>
</head>
<body>
Uspješno si kreirao web server!!!
</body>
</html>
```

Sl. 4.2. *Sadržaj datoteke HTML*

Nakon kreiranja i pisanja HTML datoteke, potrebno ju je spremiti, te pokrenuti Node.js i potražiti mapu u kojoj se nalaze, te u „*command prompt-u*“ Nodea pokrenuti izradu servera funkcijom „*node server.js*“. Nakon toga kreiran je web server na određenoj HTTP adresi sa slike, kojoj se može pristupiti kopirajući ju u određeni web preglednik, kao na slici 4.3.



Sl. 4.3. *Kreirani web server*

Nakon toga kreiramo klijent server pomoću http modula na isti način, uz različit kod .js datoteke te pozivanja naredbe u Nodeu. Nakon kreiranja servera može se pristupiti uređivanju stranice, baš kao u načinu izrade web stranice u završnom radu.

4.3. Značajke Node.js

Uvodnim opisivanjem prednosti Nodea, dodatno ćemo nabrojati i pojasniti najbitnije značajke Nodea zbog čega je postao jako bitan alat u svijetu programera. Iako već spomenuta, velika prednost Nodea je njegova biblioteka koja sadrži različite JavaScript module koji olakšavaju razvoj web aplikacija te zbog nje se Node.js i koristi u velikoj mjeri. Iako nije samo to glavni razlog, nabrojat ćemo glavne značajke koje postavljaju Node.js na prvo mjesto programerskih alata:

1. **Asinkronost**: Svi dijelovi API-a (application programming interface) su asinkroni, što u pravilu znači da nema blokiranja, točnije rečeno jedna radnja ne blokira druge radnje. Također temeljni poslužitelj (server) Nodea nikad ne čeka API za povratak podataka, što znači da informacije do korisnika dolaze puno brže jer nema potrebe za ponovnim učitavanjem čitavih stranica. Svi postojeći API-ji su vrlo elegantni te je kompleksnost koja dolazi do programera zadržana na što nižoj razini.
2. **Velika brzina**: Iz razloga što je izgrađen na već spomenutom Googleovom JavaScript V8 engineu, biblioteka je dovoljno brza u izvršavanju određenog koda.
3. **Jednonitan ali visoko skalabilan**: Iako većina tehnologija za web aplikacije poput JSP, Spring MVC, ASP.NET, HTML, Ajax, jQuery itd. su više-nitne, Node.js koristi jednonitni model s mehanizmom petlje. Mehanizam događanja petlje pomaže poslužitelju (serveru) da odgovori u ne blokiranom načinu i čini server visoko skalabilan, za razliku od ostalih servera koji stvore ograničene probleme za obradu zahtjeva. Visoka skalabilnost servera je sposobnost baš tog servera da s lakoćom obradi rastuću količinu posla, ili potencijal da se dodatno proširi kako bi se smjestio taj rast. Node koristi jednonitni program i isti taj program može pružiti uslugu na mnogo veći broj zahtjeva od tradicionalnog servera poput npr. Apache HTTP Servera.
4. **Nema obrađivanja i raspakiravanja**: Node.js aplikacije nikad ne obrađuju ili raspakiravaju podatke. Ove aplikacije jednostavno izlazne podatke daju u gotovim komadima.
5. **Node Package Manager (NPM)**: NPM [10] koristimo ukoliko želimo izgraditi nešto dosta kompleksnije, tada bez problema u svoj projekt možemo ubaciti „third-party“ module. Ti moduli služe za upravljanje i instaliranje paketa te služe kao standard za određene pakete. NPM zapravo olakšava programerima podijeliti kod koji je stvoren da riješi nekakav problem, tada se taj kod može dodatno nadograđivati i stvarati bolje i

logičnije rješenje u kodu. Upravo takvi kodovi se dodatno pakiraju i nazivaju se paketi koji uvelike pomažu gdje izrada nekakvih web stranica može ovisiti o tim paketima. U svakom trenutku na internetu se mogu potražiti određeni paketi koji mogu pomoći pri izradi vlastite aplikacije.

6. **Čuvanje podataka**: Node.js se pobrinuo i oko čuvanja podataka te se može koristiti i MySQL bazom i sve popularnijom u svijetu programiranja, NoSQL bazom MongoDB.
7. **Upotreba**: S obzirom na sve prednosti koje ovaj alat sadržava, objavljen je i popis poznatih tvrtki koje koriste Node.js, a to su: eBay, General Electric, GoDaddy, Microsoft, PayPal, Uber, Wikipins, Yahoo ! itd.

4.4. GULP

Pojašnjavanjem različitih mogućnosti Nodea, njegovih modula i paketa, došli smo do nama dosta bitnog paketa koji je korišten u radu, Gulp. Radi se o alatu koji nam u velikoj mjeri pomaže pri izradi i razvoju naše web stranice. Pravilnije rečeno sistemski je alat, a svrha mu je automatizacija procesa poput prevođenja jezika ili minimizacije skripti. Često se koristi u vanjskom („front end“) programiranju za poslove poput podizanja web poslužitelja, automatskog osvježavanja stranica kad god je datoteka spremljena, korištenje predprocesora poput SASSa ili LESSa te za kreiranje HTML stranice iz Jade koda kao u našem slučaju.

Za početak moramo imati Node.js, nakon toga kako bi koristili gulp, moramo ga instalirati globalno iz razloga što je tako upotrebljiv iz cijelog sustava, a ne samo iz neke lokalne mape. Instalaciju pokrećemo pozivanjem raznih naredbi u „node.js command promptu“, a prva naredba koju trebamo upotrijebiti je:

npm rm - -global gulp

što znači da prvo uklanjamo neku prije instalaciju gulpa ukoliko je postojala, te nakon toga je potrebno instalirati gulp, ali napomena na to je da se instalira globalno, a direktiva „- -global“ to označava. Zatim:

npm install –global gulp-cli

nakon toga je potrebno je se u komandnom prozoru pozicionirati u mapu u kojoj je kod (primjer C:\Programiranje\Završni rad), te nakon toga pokrenuti sljedeću naredbu:

npm install --save-dev gulp

Gulpfile.js u odabranoj mapi sadrži potrebne gulp procese koji će vršiti prevođenje i predprocesiranje SASS i Jade šablona. Također je još potrebno instalirati i ostale module koje ćemo koristiti u našem radu:

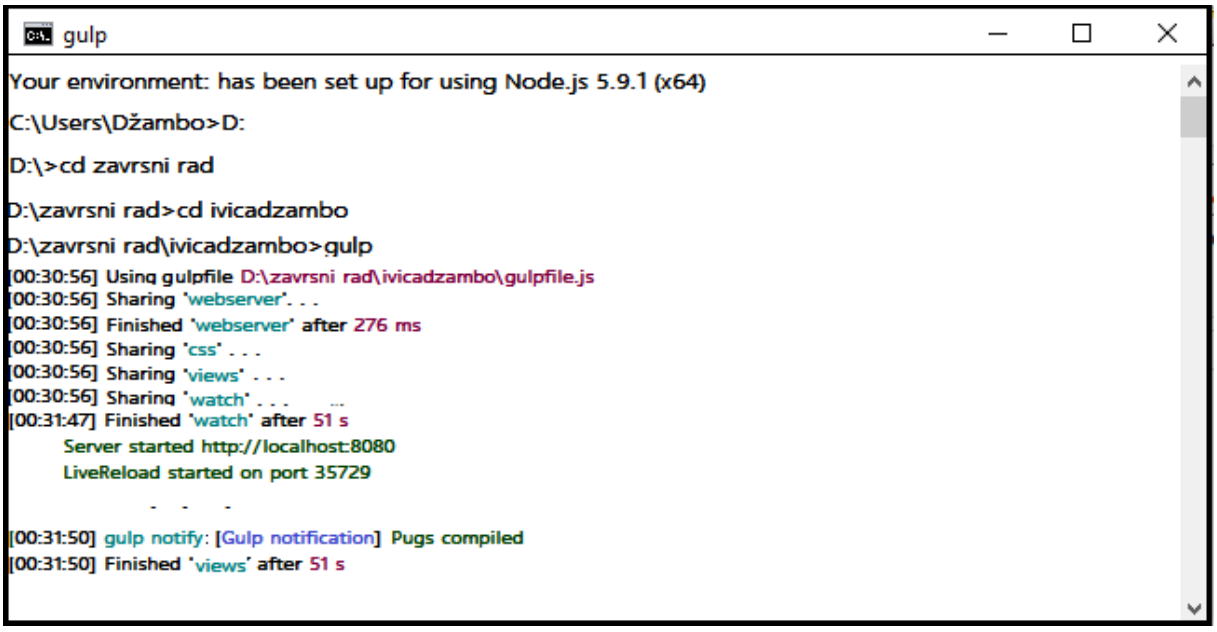
npm install gulp-pug

npm install gulp-notify

npm install gulp-sass

npm install gulp-connect

Svi potrebni moduli su instalirani u mapi koju smo odabrali te je jedino što je potrebno pozvati gulp funkciju, dok ostale Node.js module možete potražiti na [11]. Na slici 4.4. možete vidjeti primjer pozivanja gulpa u primjeru završnog rada.



Sl. 4.4. *Pokretanje gulpa u Node.js command prompt*

Nakon toga u konzoli dobivamo informaciju da je web server pokrenut na portu 8080 na local host-u. Upravo na toj web adresi možemo pokrenuti vlastitu web stranicu.

U završnom radu koristili smo gulp za pojedine stvari, prva stvar je ta da smo povezali gulp i SASS jezik te smo pomoću gulpa taj jezik preveli prvo u SCSS, te nakon toga u nama potrebni CSS. Već spomenuta jednostavnost gulpa i ovdje će biti pokazana, gdje ćemo kroz nekoliko pozivanja funkcija prevesti SASS u nama potrebni CSS. Na slici 4.5. možete vidjeti upravo primjer prevođenja SASS-a u CSS i jednostavnost koju pokušavamo prikazati.

```
var gulp = require('gulp');
var sass = require('gulp-sass');

gulp.task('styles', function() {
    gulp.src('sass/**/*.scss')
        .pipe(sass().on('error', sass.logError))
        .pipe(gulp.dest('./css/'))
});

//Watch task
gulp.task('default',function() {
    gulp.watch('sass/**/*.scss',['styles']);
});
```

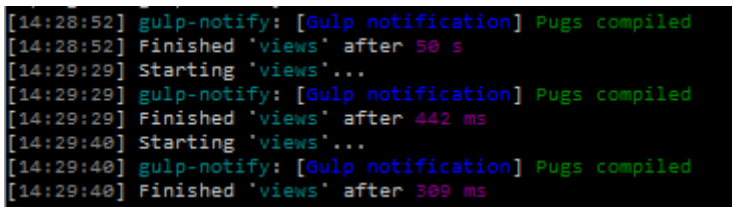
Sl. 4.5. *Primjer prevođenja SASS-a u CSS*

Prvi korak je već spomenut ranije, gdje je potrebno instalirani module i pakete SASS-a u Nodeu, te ga nećemo dodatno objašnjavati. Potrebno je pokretanje gulpa i pozivanje modula, a to smo uradili funkcijom *require()*. Iduća jako bitna stvar je dodavanje *gulp.task()*-a koji definira naše zadatke, a ta metoda prima dva parametra. Prvi parametar je zadatak ili proces, a drugi parametar je povratna funkcija koja pokreće trenutni proces, a u ovom slučaju je to *'styles'*. Dalje smo postavili datoteke koje želimo pogledati, a gledaju se pomoću funkcije *gulp.src()*, te smo unutar nje prošli niz mjesta koje pregledamo u ovisnosti o drugoj datoteci.

Dalje smo koristili metodu *.pipe()* da ne proslijedi ništa od prethodne *src()* metode, nego unutar sebe uvodi SASS module za prevođenje SASS jezika. Funkcija sass prikazuje događanja te ako postoji pogreška prikazuje ju. Jako bitna stvar je da pri raspakiravanju SASS-a ako se i pojavi greška, metoda *.pipe()* će samo ubiti proces, ali će i javiti gdje se dogodila greška, što je vrlo korisno i bitno.

Završni korak se također odrađuje pomoću *.pipe()* metode, a to je da govorimo gulp gdje da postavi naš prevedeni SASS. *.pipe()* metoda tako preuzima podatke iz prethodne *.pipe()* metode, te ponovno uzimamo novu metodu *.dest()* kojoj kao string dodijelimo put do odredišta gdje spremiti naš prevedeni CSS kod. Jednostavnije objašnjeno postavlja izlaz u našoj css mapi, znači uzima nekakvu mapu *style.scss* te ju sprema u istoj toj mapi kao *style.css*. Završna metoda je *.watch()* gdje samo pratimo niz zadaća koje se odrađuju bez ikakvog našeg dodatnog posla.

Upravo isti takav posao je gulp odradio i s Jade jezikom, gdje smo također ranije dodali u Node.js Jade module i pakete te ih na skoro identičan način preveli u nama potrebni HTML jezik. Također treba ponoviti još dodatnu prednost gdje promijenjeni kod se automatski mijenja preko gulpa, znači dovoljno je samo spremanje unutar našeg programa kojim uređujemo kod, te se automatski odvijaju promjene na našoj web stranici. Na slici 4.6. možemo vidjeti primjer prikaza u konzoli nakon sitnih promjena u kodu gdje je promjena odmah vidljiva i na našoj web stranici.



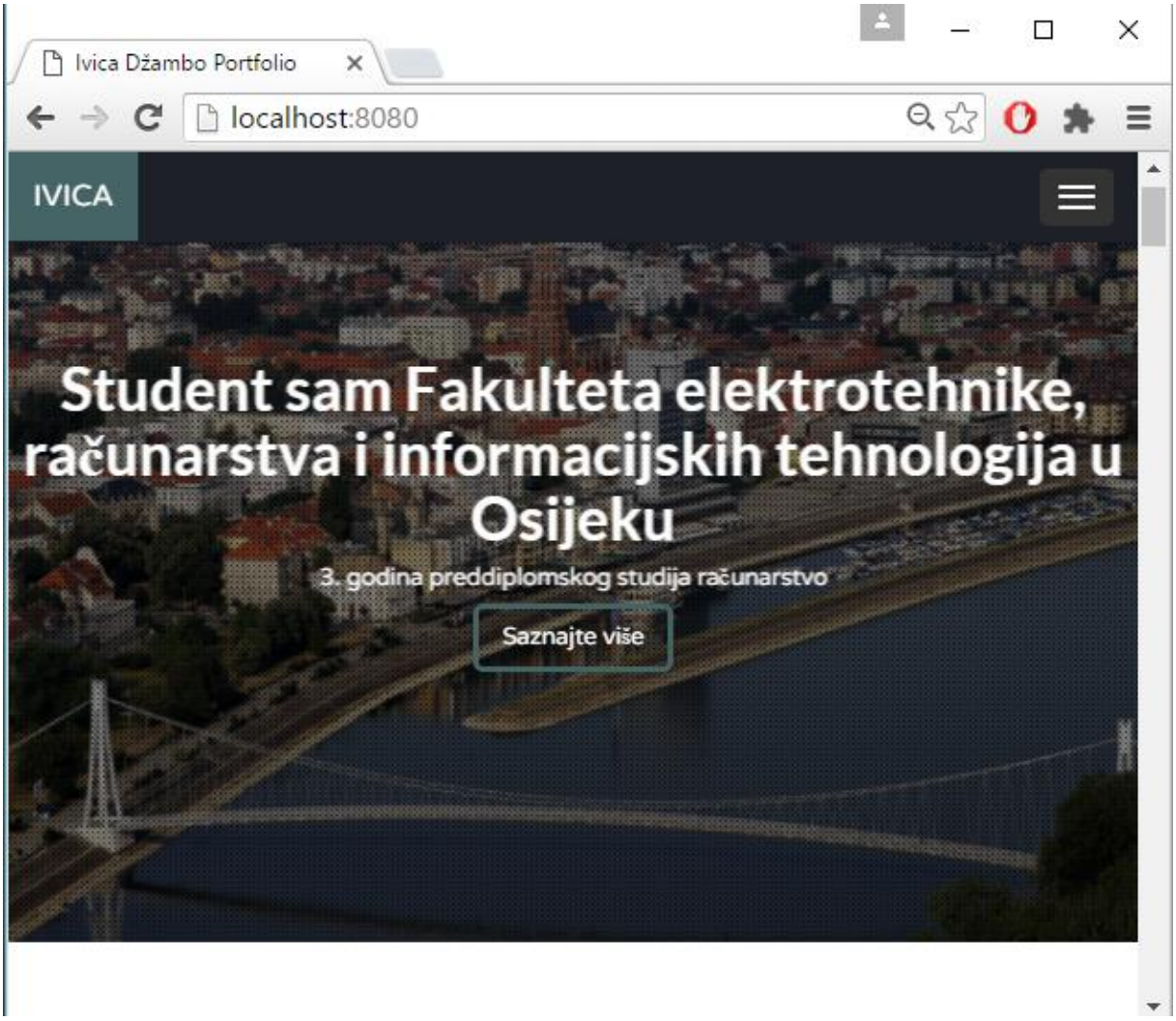
Sl. 4.6. *„Command prompt“ automatsko povezivanje web stranice i gulpa*

Prikazivanjem prednosti Nodea i njegovih modula, u konačnici se može zaključiti da se radi o super jednostavnom i modernom alatu s jako puno mogućnosti koja je potrebna većini programera, isključivo ako se radi o kompliciranijoj web stranici gdje se prednosti Nodea pokazuju kao jako korisne i bitne.

5. ANALIZA WEB STRANICE

Završno poglavlje opisuje mogućnosti vlastite web stranice te prikazuje prednosti i mogućnosti koje su moguće uraditi pomoću novih alata i jezika. Web stranica prikazuje vlastiti Portfolio gdje su vidljivi svi podaci korisnika te njegov privatni i poslovni život.

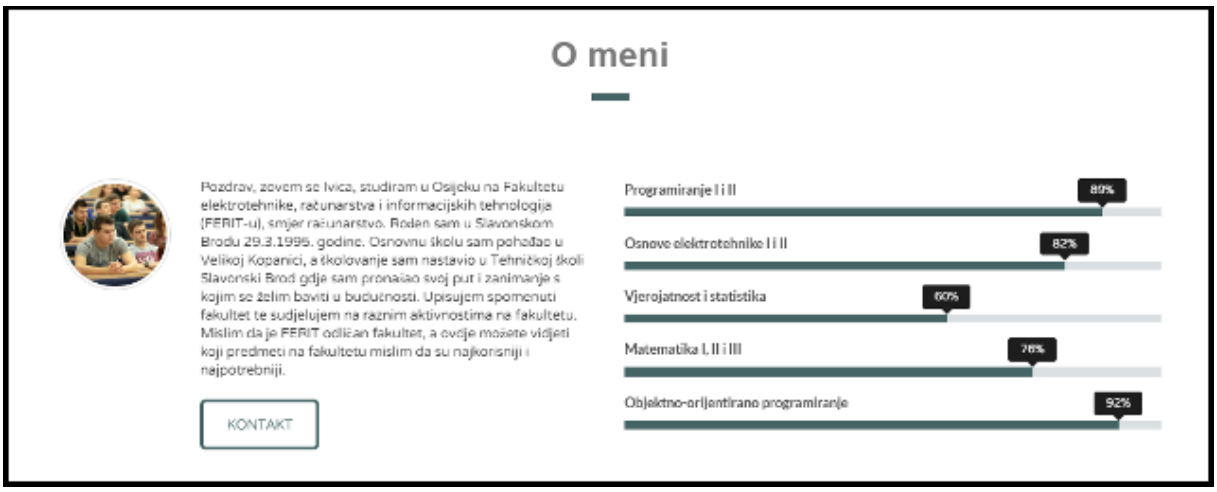
Web stranica je pisana i stvarana kroz već spomenute razne programske jezike i alate. Spominjući bootstrap koji je služio za dizajn aplikacije ne može se izostaviti kako je aplikacija prilagodljiva različitoj veličini ekrana [12] tako slika 5.1 prikazuje prikaz naslovne stranice na manjoj razlučivosti. Vidljivo je kako se navigacijska traka suzila i može se otvoriti pomoću dodatnog gumba pri vrhu stranice.



Sl.5.1. *Prikaz naslovne stranice (responzivan izgled)*

Na naslovnoj stranice postoji dugme „Saznajte više“, koji i nije samo obično dugme već i nekakav oblik već spomenutog okidača. Upravo klikom na to dugme na samoj web stranici

prelazite na novu stranicu stranicu s izborničke trake „O meni“, gdje se pojedinosti o osobi, te dodatna zanimljiva stvar „progress-bar“ koji možete vidjeti na slici 5.2.



Sl.5.2. *Prikaz „progress-bara“ na web stranici*

Već spomenuto dugme „saznajte više“ je također i okidač za punjenje „progress-bara“, što znači da se počinje puniti tek nakon pritiska na to dugme. „Progress-bar“ se puni do određenih vrijednosti koje su ranije određene u kodu. Također je tu još jedno dugme „Kontakt“ koje klikom na njega vodi na završni dio web stranice koji prikazuje određene kontakte o osobi.

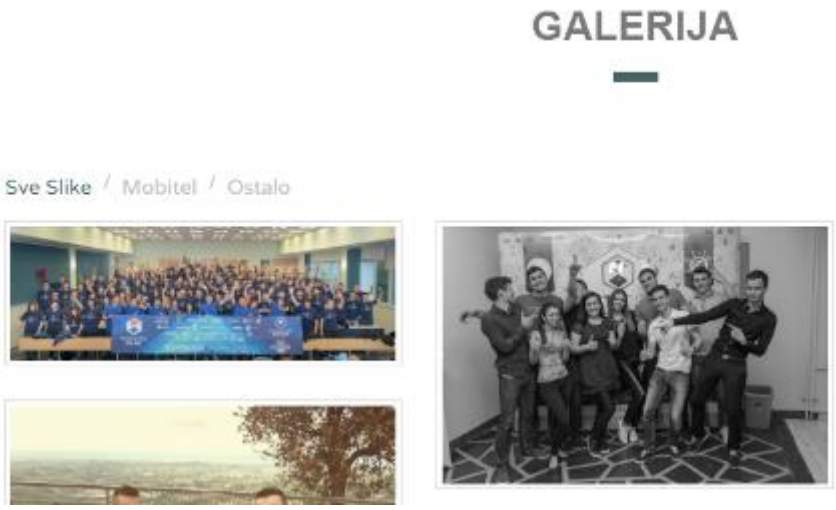
Sljedeće polje stranice je dio „Iskustvo“ gdje se nalaze razne slike posebnih iskustava osobe. Slika prikazuje logo pojedinih projekata ili firmi s kojom je osoba imala veze. Prikaz takvog slike možete vidjeti na slici 5.3.

Sl.5.3. *Prikaz slike iskustva*

Na slici 5.3. možete vidjeti također dva kruga, koja se pojavljuju samo kada pokazivačem dođete prođete preko slike. Desni kružić je poveznica na određenu stranicu koja vas automatski povezuje s tom stranicom koja se tiče upravo toga iskustva, u ovom slučaju kao na slici 5.2. vas povezuje s Facebook profilom Studentskog zbora FERIT-a. Klikom na lijevi kružić otvarate novi prozor gdje je detaljno opisano iskustvo u određenoj firmi. Na slici 5.4. možete vidjeti primjer prozora koji se otvori klikom na lijevi kružić.

Sl.5.4. *Prikaz otvorenog dodatnog prozorčića*

Web stranica također sadržava stranicu „Galerija“, gdje se nalaze određene fotografije, koje je moguće sortirati pod prikaz svih fotografija, mobilnih fotografija te ostalih. Klikom na fotografiju otvara se stvarna veličina fotografije s opisom što je na fotografiji. Također nakon ulaska u fotografiju moguće je listati sve slikom uz klik na strelicu sa strana, ovisno želite li sljedeću ili prethodnu fotografiju. Na slici 5.5. možete vidjeti prikaz galerije.



Sl.5.5. *Prikaz galerije*

Kao završni dio web stranice postavljena je stranica s kontaktima, na koju se moglo automatski prebaciti i klikom na dugme sa stranice „O meni“. U kontakt stranici navedena je adresa osobe, kontakt telefon, e-mail adresa korisnika te link koji vodi na portal STUP FERIT-a [13], gdje se može pronaći profil osobe. Također pri dnu imate linkove koji su povezani s društvenim mrežama Facebook i Twitter, te prikaz godine stvaranja web stranice. Na slici 5.6. je prikazano upravo dio web stranice s kontaktima.

Sl.5.6. *Prikaz dna web stranice, „Kontakt“*

6. ZAKLJUČAK

U završnom radu programski je izrađena web stranica s korištenjem alata koji su već u upotrebi, a u budućnosti njihova primjena će rasti jer su jednostavni i praktični. Glavni cilj je bio usporediti upravo te alate pisanjem koda te dati pregled CSS predprocesora i postprocesora i njihove mogućnosti.

Glavna prednost pri korištenju novih alata te CSS predprocesora i postprocesora je prije svega u brzini i jednostavnosti pisanja koda. Kada se priča o brzini i jednostavnosti, prikazane su usporedbe novih alata i jezika s ostalim klasičnim jezicima, te na temelju tih usporedbi se može zaključiti da se broj linija koda s novim alatima i predprocesorskim naredbama smanjio za tri puta. Iz razloga što je vrijeme izrade aplikacije ovisno upravo o tome broju linija koda te jednostavnosti korištenja alata, može se reći da se vrijeme izrade upravo ovakve aplikacije može smanjiti također u sličnom omjeru, iako taj broj je promjenjiv jer je vrijeme u izradi web stranica relativan pojam. Dodatna prednost je također jednostavnost korištenja alata poput Nodea i njegovih modula, od kojih smo mi koristili gulp. Uz jednostavnost korištenja, tu je prednost što ne morate brinuti o problemima s instaliranjem i dodavanjem dodatnih biblioteka i modula, već ih je sami alat dodao i dao rješenje o kojem vi ne bi trebali razmišljati i pri tome možete iskoristiti vrijeme na nešto drugo.

Ozbiljnim pristupom izrade završnog rada odrađeni su svi zahtjevi koji su opisani zadatkom završnog rada te tako doveli do krajnjeg projekta koji prikazuje korištenje nekih od novih alata te uporabu CSS predprocesora i postprocesora. Dodatno su opisani i neki dodatni alati koji se također koriste. Svima je poznato da je u svijetu programiranja vrijeme novac, te upravo iz toga razloga ne sumnjamo da će upotreba novih alata i predprocesa prijeći u naviku već u skorijoj budućnosti.

LITERATURA

[1] About HTML, http://www.w3schools.com/html/html_intro.asp 23.6.2016.

[2] About CSS, http://www.tutorialspoint.com/css/what_is_css.htm 21.7.2016

The Book of CSS3: Peter Gasston 21.7.2016.

[3] Sublime 3 text editor download, http://www.sublimetext.com/3 19.8.2016.

[4] About JavaScript http://www.w3schools.com/js/ 22.6.2016.

[5] About the mixin dirctive, https://www.sitepoint.com/sass-basics-the-mixin-directive/ 30.8.2016.

[6] Bootstrap framework, http://getbootstrap.com/ 30.8.2016.

[7] About Pug, http://jade-lang.com/ 13.6.2016.

[8] About SASS, http://sass-lang.com/guide 22.6.2016.

[9] Node.js, http://www.tutorialspoint.com/nodejs/index.htm 15.8.2016.

[10] About Node.js NPM, http://www.tutorialspoint.com/nodejs/nodejs_npm.htm 15.8.2016.

[11] Node.js modules, https://nodejs.org/api/modules.html 20.8.2016.

[12] Responsive web design, https://www.smashingmagazine.com/2011/01/guidelines-for-responsive-web-design/ 20.6.2016.

SAŽETAK

U završnom radu programski je izrađena web stranica s korištenjem alata koji će se u budućnosti primjenjivati jer su jednostavniji i praktičniji. Glavni cilj je bio usporediti upravo te alate pisanjem koda te dati pregled CSS predprocesora i postprocesora i njihove mogućnosti. Cilj prikazivanja novih alata i programskih jezika je povećanje korištenja istih, te se smatra da će se u skoroj budućnosti u velikoj mjeri primjenjivati jezici i alati koji nam pomažu što brže i jednostavnije izraditi web stranicu. Sadržaj web stranice je osobna web stranice. Za izradu web stranice korišten je HTML programski jezik kao osnova te uz njega popratne tehnologije koje smo obradili kao što su: Node.js, Node gulp, Bootstrap framework te programski jezici HTML, CSS, Pug, SASS, SCSS.

Ključne riječi: CSS predprocesori i postprocesori, NodeJS, portfolio, Pug

ABSTRACT

THE USE OF NEW TOOLS AND PROCESSES IN DESIGNING A WEBPAGE

In this final work a website was made with the help of tools that will be used in the future because they are simpler and more practical. The main goal was to compare these tools by writing code and to give a review of CSS preprocessors and postprocessors and their possibilities. The goal to show new tools and programming languages is to increase their use and it is considered that in the near future tools and languages that help us make webpages more easily and quickly will be in wider use. The content of the webpage is a personal webpage. To make the webpage HTML programming language was used as a foundation and with it, other technologies like Node.js, Node gulp, Bootstrap framework and programming languages HTML, CSS, Pug, SASS, SCSS.

Keywords: CSS preprocessors and postprocessors, NodeJS, portfolio, Pug.

ŽIVOTOPIS

Ivica Džambo rođen je 29.03.1995. u Slavonskom Brodu. 2009. godine, nakon završene osnovne škole, upisao se u Tehničku školu Slavonski brod te sudjelovao u dva velika projekta škole kao što su stručna praksa „LeonardoDaVinci Programme“ u Njemačkoj, Leipzig te „SB SOLAR“, gdje Tehnička škola postaje prva škola u Hrvatskoj koja proizvodi vlastitu struju i prodaje ju. 2013. godine upisao se na preddiplomski studij računarstva na Elektrotehničkom fakultetu u Osijeku. U ožujku 2015. godine pridružuje se Studentskom Zboru Elektrotehničkog fakulteta. U kolovozu 2016. godine ide na stručnu praksu u Siemens Convergence, programersku firmu gdje je odradio određeni period prakse.

Potpis:

_____

PRILOZI

Na CD-u priloženom uz Završni rad nalaze se:

Dokumenti:

ZavršniRadIvicaDžambo.docx

ZavršniRadIvicaDžambo.pdf

WebStranica.zip