Izrada web aplikacija primjenom Angular 8 i Lumen razvojnog okvira

Brezovec, Miriam-Valentina

Undergraduate thesis / Završni rad

2019

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University North / Sveučilište Sjever**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:122:492165

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-20**

Repository / Repozitorij:

University North Digital Repository

zir.nsk.hr

dabar
DIGITALNI AKADEMSKI ARHIVI I REPOZITORIJI

Sveučilište Sjever

Završni rad br. 654/MM/2019

Izrada web aplikacija primjenom Angular 8 i Lumen razvojnog okvira

Miriam Brezovec, 3076/601

Varaždin, rujan 2019. godine

Sveučilište
Sjever

Odjel za Multimediju, oblikovanje i primjenu

Završni rad br. 654/MM/2015

Izrada web aplikacija primjenom Angular 8 i Lumen razvojnog okvira

Student

Miriam Brezovec, 3076/601

Mentor

Mr.sc. Vladimir Stanisavljević

Varaždin, rujan 2019. godine

Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3, HR–42000 Varaždin

Prijava završnog rada

Definiranje teme završnog rada i povjerenstva

ODJEL Odjel za multimediju

STUDIJ preddiplomski stručni studij Multimedija, oblikovanje i primjena

PRISTUPNIK Brezovec Miriam-Valentina | MATIČNI BROJ 3076/601

DATUM 24.09.2019. | KOLEGIJ Programski alati 2

NASLOV RADA Izrada web aplikacija primjenom Angular 8 i Lumen razvojnog okvira

NASLOV RADA NA ENGL. JEZIKU Developing web appliaction with Angular 8 and Lumen framework

MENTOR mr.sc. Vladimir Stanisavljević | ZVANJE Viši predavač

ČLANOVI POVJERENSTVA

1. doc.dr.sc. Andrija Bernik, pred. - predsjednik
2. doc.dr.sc. Dean Valdec - član
3. mr.sc. Vladimir Stanisavljević, v.pred. - mentor
4. mr.sc. Matija Mikac, v. predavač - rezervni član
5.

Zadatak završnog rada

BROJ 654/MM/2019

OPIS

Da bi se poboljšala strukturiranost aplikacija napisanih u PHP-u poželjno je koristiti neku napredniju programsku metodologiju poput MVC (od engl. Model-View-Controler) oblikovnih obrazaca. U PHP-u podršku za MVC može se ugraditi primjenom razvojnih okvira poput Laravel-a proširenog Lumen dodacima. Dodatne mogućnosti mogu se ostvariti na klijentskoj strani pomoću Angular razvojnog okvira.

U ovom radu potrebno je:
* opisati osnove xAMP-a i ulogu programskog jezika PHP-a u njemu,
* detaljno obraditi MVC programsku arhitekturu podržanu kroz Lumen i Angular (8) razvojne okvire,
* osmisliti i oblikovati web aplikaciju koja koristi Lumen za neku stvarnu primjenu s bazom podataka,
* istražiti dodatne web tehnologije kojima bi se mogle ostvariti dodatna poboljšanja stranica.

Detaljno opisati sve korištene tehnologije i korake u radu potrebne da bi se ostvarilo traženo te detaljno opisati stečena iskustva i postignute rezultate.

ZADATAK URUČEN 24.10.2019. | POTPIS MENTORA

REPUBLIKA HRVATSKA 5 KOPRIVNICA SVEUČILIŠTE SJEVER

SVEUČILIŠTE SJEVER

Predgovor

Ovom prilikom željela bih se zahvaliti svim kolegama s posla na podršci i iskustvima koja su doprinijela mom osobnom razvoju kao web developeru i strpljenju tijekom razvoja web aplikacije korištene za ovaj završni rad. Posebne zahvale našim web i java timovima na njihovoj općenitoj pomoći i vodstvu!

Na kraju, želim se zahvaliti profesoru Vladimiru Stanisavljeviću za prihvaćanje i potporu u ovom projektu.

Prologue

On this occasion, I would like to thank all of my colleagues for their support and experience that have contributed to my personal growth as a web developer and patience during the development of the web applications used for this thesis. Special thanks to our web and java teams for their overall help and guidance!

Finally, I would like to thank Professor Vladimir Stanisavljevic for his thesis acceptance and support in this project.

Sažetak

Web aplikacije su jedno od najpopularnijih programskih rješenja današnjice. U radu je opisan proces izrade jedne takve aplikacije za web, opisane tehnologije korištene za postizanje istog i objašnjen proces same izrade. U svrhu slijeđenja novijih trendova razvoja web aplikacija, korišteni su noviji razvojni okviri Lumen i Angular 8. Web aplikacija je izrađena pomoću okruženja Visual Studio Code-a.

Završni rad se sastoji od dva dijela. U prvom, teoretskom dijelu, objašnjena su pomagala i jezici korišteni u stvaranju web aplikacije, a u praktičnom dijelu prikazan je i opisan put razvoja željene web aplikacije. Na završetku se može vidjeti konačan izgled aplikacije.

Ključne riječi: web aplikacija, razvojni okvir, Lumen, Angular, PHP, HTML, SCSS, TypeScript, XAMPP, komponente, moduli, servisi

Summary

Web applications are one of the most popular software solutions of today. The goal of this paper is to create one such web application, to describe the technologies used to achieve it, and to describe the process of making it. For the purpose of following the latest web application development trends, I will use the newer development frameworks Lumen and Angular 8. The web application was created using Visual Studiao Code.

The final paper consists of two parts. The first, theoretical part explains the tools and programming languages used to create the web application, and the practical part shows and describes the path of development of the desired web application. Before the conslusiom, the final appearance of the application can be seen.

Keywords: web application, framework, Lumen, Angular, PHP, HTML, SCSS, TypeScript, XAMPP, components, modules, services

Popis korištenih kratica

HTML HyperText Markup Language
Prezentacijski jezik za stvaranje hipertekstualnih dokumenata

CSS Cascading Style Sheets
Kaskadni stilski jezik za opis prezentacije HTML dokumenata

SASS Syntactically Awesome Style Sheets
CSS Preprocessor

SCSS Sassy Cascading Style Sheets
CSS Preprocessor

JS Javascript
Skriptni programski jezik

TS Typescript
Skriptni programski jezik

API Application Programming Interface
Sučelje za programiranje aplikacija

HTTP Hypertext Transfer Protocol
Protokol za prijenos informacija na Web-u

PHP Hypertext Preprocessor
Skriptni jezik sa strane servera

JSON Javascript Object Notation
Javascript objekt

NPM Node package manager
Menadžer paketa za Node.js

SQL Structured Query Language
Strukturirani upitni jezik

XAMPP XAMPP -Cross-Patform, Apache, MariaDB/MySQL, PHP, Pearl
Multi-platforma korisna za lokalno (na vlastitom računalu) testiranje i razvoj web stranica i aplikacija

IDE Integrated development environment
Integrirano razvojno okruženje

Sadržaj

1. Uvod 1
2. Programiranje, kodiranje i alati za razvoj web aplikacije 2
   2.1. Programski i skriptni jezici i razvojna okruženja 2
   2.2. Angular 8 3
      2.2.1. HTML 4
      2.2.2. CSS/SASS/SCSS 5
      2.2.3. JS/TS 8
   2.3. Node.js 9
      2.3.1. NPM 10
   2.4. XAMPP 11
   2.5. Lumen/Laravel 11
      2.5.1. PHP 12
   2.6. Baza podataka - SQL 12
      2.6.1. MySQL 13
3. Izrada aplikacije 14
   3.1. Planiranje i skiciranje 14
   3.2. Postavljanje aplikacije 15
      3.2.1. Instalacija alata 16
      3.2.2. Priprema baze podataka 17
   3.3. Izrada jednostavnog back-enda 18
      3.3.1. Izrada novog projekta u Lumenu 18
      3.3.2. Povezivanje Lumena i baze podataka 19
      3.3.3. Slanje podataka prema front-endu 21
   3.4. Izrada projekta u Angularu 21
      3.4.1. Stvaranje novog projekta 21
      3.4.2. Komponente 22
      3.4.3. Moduli 22
      3.4.4. Servisi 23
      3.4.5. Povezivanje front-end aplikacije s Lumen 23
4. Konačan izgled aplikacije 25
5. Zaključak 26
6. Literatura 28

1. Uvod

U današnje doba, tehnologija se nalazi u svakom aspektu našeg života. Od osnovnih potreba do onih kompliciranijih, koristimo tehnologiju kako bismo si olakšali obavljanje svakodnevnih poslova. Neki od takvih pomagala su aplikacije na našim uređajima (mobiteli, tableti, računala itd.), na primjer aplikacije za računanje, praćenje aktivnosti i obaveza, pa i kompliciranije aplikacije kao socijalne platforme i slično.

Cilj ovog rada je izraditi jednu takvu aplikaciju za web, opisati tehnologije korištene za postizanje istog i opisati proces same izrade. Web aplikacija će biti korištena u svrhu praćenja sakupljenih kredita kupaca u trgovini. Razvoj web aplikacija i okuženja u kojem se razvijaju ubrzano raste. U svrhu slijeđenja novijih trendova razvoja web aplikacija, za razvoj ove aplikacije koriste se moderni razvojni okviri Lumen i Angular 8, te novije inačice programskih i skriptnih jezika poput TypeScript-a i PHP 7+ .

Aplikacija koristi programsku metodologiju MVC (Model-Prikaz-Kontroler, eng. Model-View-Controller) obrasca kako bi se poboljšala strukturiranost aplikacije u PHP-u pomoću Lumena. Za razvoj korisničkog prikaza i dodatnih funkcija, primjenjuje se Angular 8.

Završni rad sastoji se od teoretskog i praktičnog dijela. U prvom, teoretskom dijelu, objašnjena su pomagala i jezici korišteni u stvaranju web aplikacije, kao što su PHP, HTML, SCSS, TS i naravno, Angular i Lumen. U praktičnom dijelu prikazan je i opisan put razvoja željene web aplikacije. Prikazuje se kako osmisliti i implementirati jednostavnu bazu podataka koja je spojena s Lumenom. Kreira se nova Lumen aplikacija i vuće podatke iz baze, te ih šalje prema Angularu. Izrađuje se novi projekt u Angularu, te se kreira prikaz za preglednike. Na kraju se Angular i Lumen povezuju, te se u prikazu koriste podaci iz baze podataka. Za neke od tih funkcija, koriste se Node.js biblioteke pomoću NPM-a.

2. Programiranje, kodiranje i alati za razvoj web aplikacije

Programiranje[1] je primjena logike za olakšavanje određenih računalnih operacija i funkcionalnosti. Javlja se na jednom ili više jezika koji se razlikuju po primjeni, domeni i programskom modelu. To je metoda[2] pisanja kôda na tzv. „višoj razini“ (kôda razumljivog ljudima). Kako bi računalo taj kôd razumjelo na strojnom jeziku, većina programskih jezika danas koristi kompajlere (eng. compilers), što znači da je izvor (napisan kao skripta) sastavljen u računalni kôd. Pakira se na način da računalo taj kôd može izvršiti, primjerice, kao izvršne datoteke. Danas njima obično rukuje operativni sustav računala, poput Windows EXE datoteka (ili čak i operativni sustav na pametnom telefonu ili tabletu).

Iako često možemo čuti „programiranje“ i „kodiranje“ u istom smislu, no to nije sasvim točno. [3] Na najosnovnijoj razini, programiranje je šira disciplina, dok je kodiranje uža. Kodiranje uključuje pisanje mnogih redaka koda u svrhu stvaranja softverskog programa. Programiranje uključuje ne samo kodiranje, već i druge zadatke, poput analize i implementacije algoritama, razumijevanja struktura podataka, rješavanja problema i slično. Jednostavno rečeno, svi programeri su koderi, no svi koderi nisu programeri.

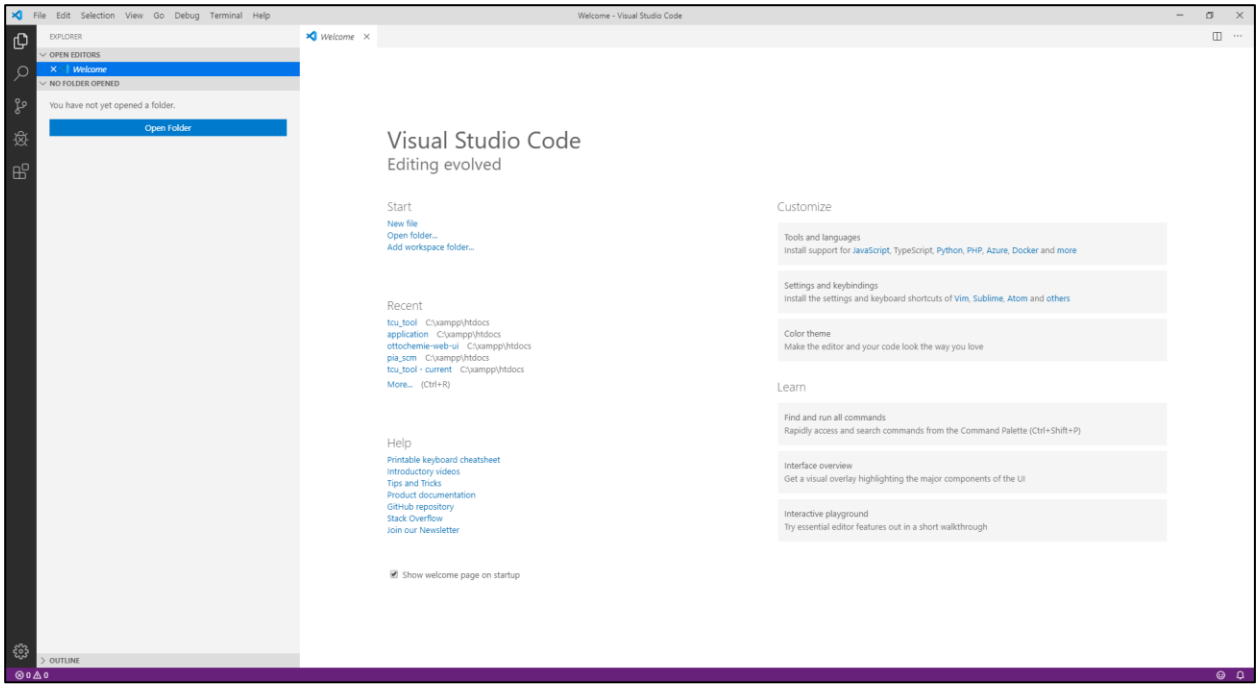
2.1. Programski i skriptni jezici i razvojna okruženja

Programski jezik[4] je skup naredbi, uputa i sintakse koje se koriste za stvaranje softverskog programa. Jezici koje programeri koriste za pisanje koda nazivaju se „jezicima visoke razine“. Taj se kôd može kompilirati u „jezik niske razine“ ili „jezik strojeva“, koji računalo pomoću hardvera može izravno prepoznati. Neki primjeri takvih jezika su BASIC, C, C++, Java, R ili Pascal.

Skriptni jezik[5] je programski jezik dizajniran za integraciju i komunikaciju s drugim programskim jezicima. Razlika između skriptnog jezika i jezika koji se koristi za pisanje cijelih aplikacija je da se programski jezik obično sastavlja (kompajlira) prije nego što mu je omogućeno pokretanje, dok se skriptni jezik tumači iz izvornog koda ili tzv. bytecode-a. Neki od najčešće korištenih skriptnih jezika su JavaScript, PHP, Perl, Python, Ruby, ASP i Tcl.

Integrirano razvojno okruženje (eng. IDE) [6] je softverski paket koji objedinjuje osnovne alate potrebne za pisanje i testiranje softvera. Programeri koriste brojne alate tijekom stvaranja, izgradnje i testiranja softverskog koda. Alati za razvoj često uključuju uređivače teksta, biblioteke koda, prevoditelje i testne platforme. IDE okuplja mnoge od tih alata kao jedinstveni okvir, aplikaciju ili uslugu. Integrirani skup alata dizajniran je za pojednostavljenje razvoja softvera, te se također koristi i za prepoznavanje i minimiziranje pogrešaka u kodu. Neki od popularnijih IDE-a ove godine su Visual Studio Code, PhpStorm, WebStorm, Atom i NetBeans.

Za razvoj web aplikacije u Angularu i Lumenu koriste se PHP i TypeScript skriptni jezici. Kao razvojno okruženje, odabran je Visual Studio Code. Koristi se jer je besplatan i ima mogućnost proširenja pomoću ekstenzija, tako da se lako može prilagoditi potrebama projekta.



Slika 2.1 Prikaz IDE-a Visual Studio Code

2.2. Angular 8

Angular[7] je platforma i razvojni okvir za izgradnju klijentskih aplikacija u HTML-u i TypeScript-u. Napisan je u TypeScript. Uključuje osnovne i opcionalne funkcionalnosti kao skup TS knjižnica koje se implementiraju u aplikacije.

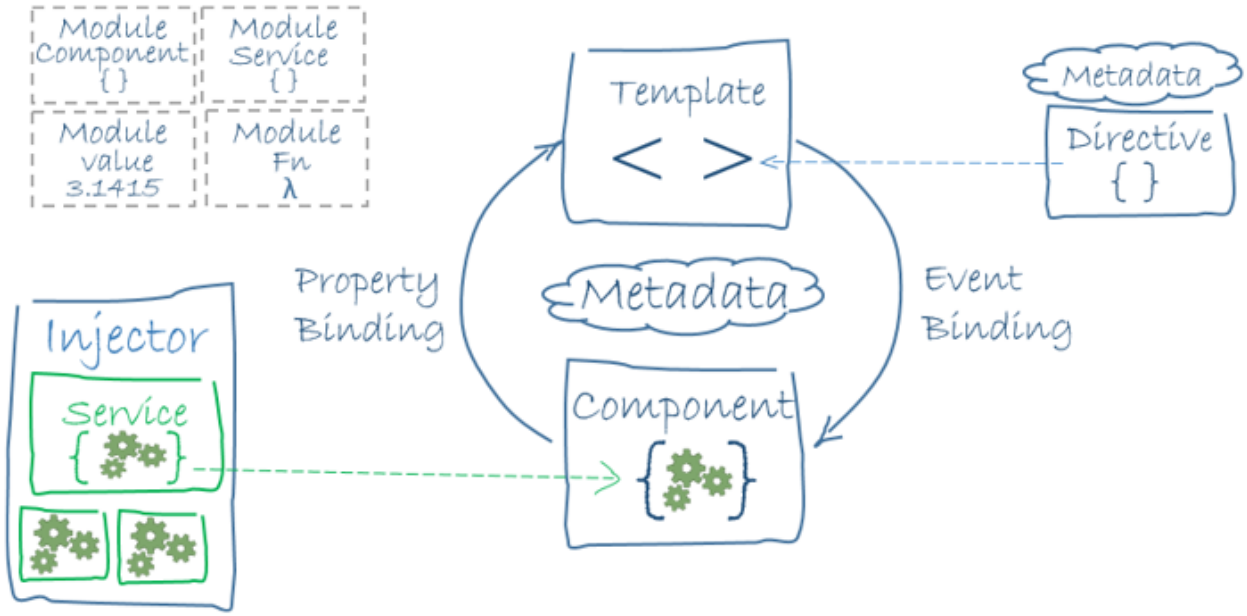
Osnovni blokovi za gradnju aplikacija su moduli (eng. *NgModules*) koji pružaju kontekst kompilacije komponenti. *NgModules* sakupljaju povezani kôd u funkcionalne skupove. Aplikacija je u Angularu definirana skupom modula. Mora sadržavati barem korijenski modul koji omogućuje *boostraping* (kompiliranje osnovnih komponenti koje omogućuje pokretanje aplikacije), a obično ima mnogo više modula.

Komponente definiraju prikaze u pregledniku. Prikazi su skupovi zaslonskih elemenata koje Angular može odabrati i mijenjati u skladu s programskom logikom i podacima.

Komponente koriste servise/usluge koje pružaju specifičnu funkcionalnost koja nije izravno povezana s prikazom. Davatelji usluga mogu se „ubrizgati“ (eng. *inject*) u komponente kao ovisnosti, čineći kôd modularnim, višekratno upotrebljivim i učinkovitim.

Komponente i usluge su „klase“ (eng. *classes*) s markerima koji označavaju njihov tip i nude meta podatke koji upućuju Angular kako promijeniti HTML prije prikaza na zaslon.

Meta podatci za klase komponenata pridružuju te klase s povezanim predlošcima koji definiraju prikaz. Predložak kombinira uobičajeni HTML s direktivama i njihovim oznakama. To omogućuje Angularu da mijenja HTML prije prikaza na zaslonu.



2.2 Službeni dijagram relacije osnovnih blokova za gradnju u Angularu 8 [7]

2.2.1. HTML

HTML[8] (Hypertext Markup Language) je tekstualni pristup koji opisuje kako se strukturira sadržaj sadržan u HTML datoteci. Takvo označavanje govori web pregledniku kako prikazivati tekst, slike i druge oblike multimedije na web stranici.

HTML je službena preporuka World Wide Web Consortium-a (W3C) i uglavnom ga se pridržavaju svi glavni web-preglednici, uključujući i desktop i mobilne web-preglednike. HTML5 je najnovija inačica specifikacije.

HTML oznake govore web pregledniku kako prikazati sadržaj web stranice. Svaka oznaka koristi se u paru, npr. <html> označava početak dokumenta dok </html> označava kraj.

```
<html>
     <head>
          <title>Ovo je HTML stranice</title>
     </head>
     <body>
          <p>Ovo je jedan odlomak teksta.</p>
     </body>
</html>
```

2.3 Primjer kôda vrlo jednostavne HTML stranice

U ovom primjeru koristi se html oznaka za početak i završetak dokumenta. Dokument se sastoji od naslova (prikaz u kartici preglednika) i tijela koje sadrži paragraf s tekstom.

2.2.2. CSS/SASS/SCSS

CSS[9] stoji za „Cascading Style Sheet“ ili kaskadne stilske listove. CSS se koristi za oblikovanje izgleda web stranica. Mogu se koristiti za definiranje stilova teksta, veličine tablica i drugih aspekata web stranica koje su se prije mogle definirati samo u HTML-u stranice.

CSS pomaže web programerima da stvore ujednačen izgled na nekoliko stranica web stranice. Umjesto definiranja stila svake tablice i svakog bloka teksta na HTML stranici, uobičajeni stilovi moraju biti definirani u samo jednom CSS dokumentu. Nakon što je stil definiran, može ga koristiti bilo koja HTML stranica koja upućuje na tu CSS datoteku. Osim toga, CSS olakšava promjenu stilova na nekoliko stranica odjednom. Na primjer, web developer želi povećati zadanu veličinu teksta sa 10pt na 12pt za pedeset stranica web stranice. Ako te sve stranice koriste istu CSS datoteku, veličinu teksta treba promijeniti samo toj datoteci.

HTML kôd

```
<html>
  <head>
    <link rel="stylesheet"
          href="css.css">
    <title>Ovo je HTML
           stranica</title>
  </head>
  <body>
    <div>
      <h1>Ovo je naslov</h1>
      <p>Ovo je jedan odlomak
         teksta.</p>
    </div>
  </body>
</html>
```

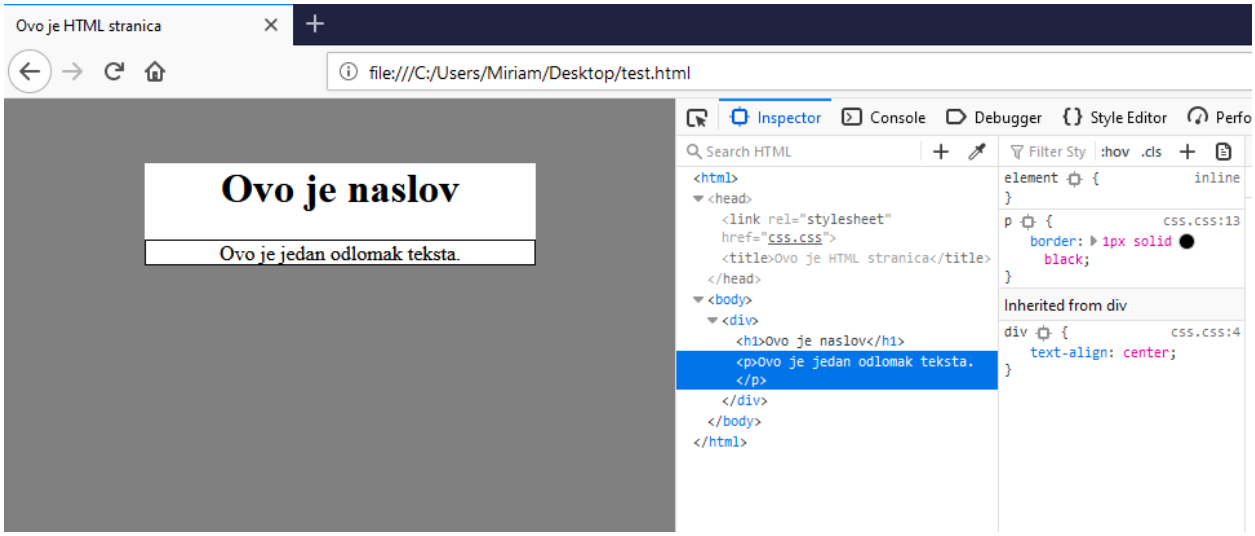
CSS kôd

```
html, body {
     background-color: gray;
}

div {
     background-color: white;
     margin: 50px auto;
     text-align: center;
     width: 60%;
}

title {
     color: blue;
}
```

```
p {
    border:1px solid black;
}
```

2.4 Primjer korištenja css-a s html-om

U ovom primjeru koristi se html kod pomoću kojeg se css dokument povezuje s html-om, te pomoću css-a oblikuje prikaz tog istog html-a u pregledniku.



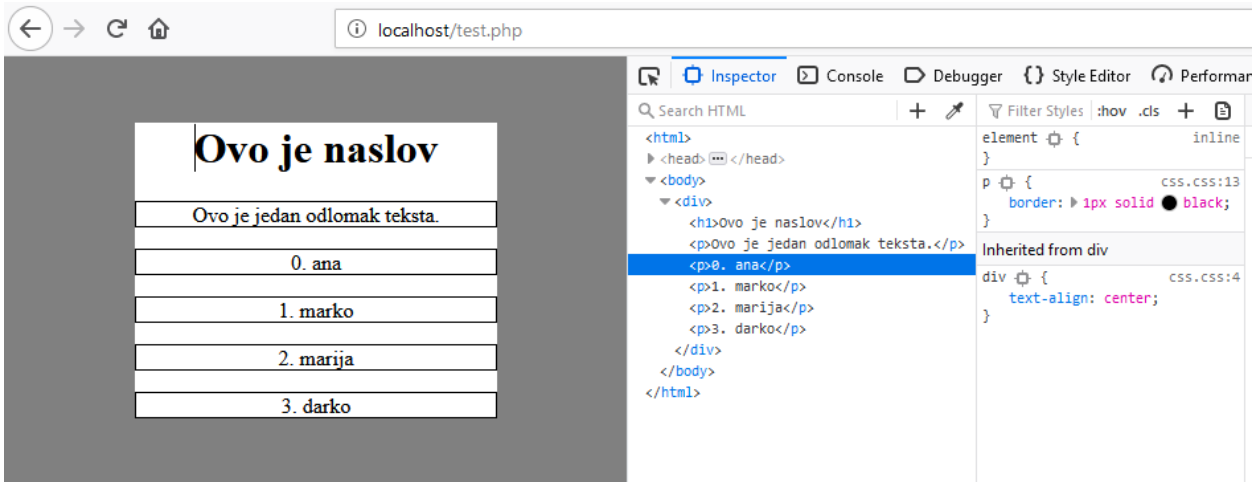
2.5 Primjer korištenja css-a s html-om – zaslonski prikaz

Na slici se može vidjeti prikaz gore navedenog koda u pregledniku. S lijeve strane je prikaz html dokumenta, dok s desne vidimo kako preglednik taj kôd interpretira pomoću ugrađene inspektorske funkcije, najčešće dostupne pritiskom na F12.

Vrijedno je spomenuti da istu stvar možemo postići i PHP kôdom. Jedna od razlika je u tome što html-om ne možemo postići ništa više osim prikaza pomoću oznaka, dok u php-u možemo koristiti funkcije ili metode. Druga važna razlika je da html dokumente možemo direktno vidjeti u pregledniku, dok je za php stranice nužan poslužitelj. U sljedećem primjeru koristi se html predložak gore opisane jednostavne stranice, te se prikazuje ranije navedena PHP FOR petlju na stvarnom primjeru. Ovaj primjer također koristi istu CSS datoteku. Osim toga, kako bi se stranica mogla prikazati u pregledniku, koristi se XAMPP-ov Apache lokalni poslužitelj.

```
<?php

  $cssFile = "css.css";

  echo "<html>
```

```
            <head>
              <link rel='stylesheet' href='" . $cssFile . "'>
                  <title>Ovo je HTML stranica</title>
            </head>
            <body>
              <div>
              <h1>Ovo je naslov</h1>
              <p>Ovo je jedan odlomak teksta.</p>";

              $studenti = array("ana", "marko", "marija", "darko");

              for ($i = 0; $i < count($studenti); $i++) {
                echo "<p>". $i . ". " . $studenti[$i] . "\n\n</p>";
              }

    echo     "</div>
            </body>
          </html>";

?>
```



2.6 Primjer korištenja php-a, html-a i css-a zajedno

SASS[10] (Syntactically Awesome Style Sheets) je skriptni jezik pretprocesora koji se interpretira ili sastavlja u Cascading Style Sheets (CSS). SassScript je sâm skriptni jezik. Sass se sastoji od dvije sintakse.

Izvorna sintaksa, nazvana „razvedena sintaksa“ (eng. the indented syntax), koristi sintaksu sličnu Haml-u. Haml (HTML Abstraction Markup Language) je sistem za predloške koji je dizajniran za izbjegavanje pisanja koda u jednoj liniji u web dokument kako bi HTML kod bio čišći i čitkiji. SASS koristi uvlačenje za odvajanje blokova kodova i znakove nove linije za odvajanje pravila.

Novija sintaksa, „SCSS“ (Sassy CSS), koristi formatiranje blokova poput onog u CSS-u. Koristi zagrade za označavanje blokova kodova i zareza za odvajanje crta unutar bloka. SASS i SCSS datoteke tradicionalno dobivaju ekstenzije .sass i .scss.

```
/* SCSS */
$blue: #3bbfce;
$margin: 16px;

.content-navigation {
  border-color: $blue;
  color: darken($blue, 9%);
}

.border {
  padding:  $margin  /  2;  margin:
$margin / 2; border-color: $blue;
}
```

```
/* SASS */
$blue: #3bbfce
$margin: 16px

.content-navigation
  border-color: $blue
  color: darken($blue, 9%)

.border
  padding: $margin / 2
  margin: $margin / 2
  border-color: $blue
```

2.7 Razlika između SASS i SCSS sintakse [11]

U web aplikaciji se koristi SCSS. Uz to, koristi se i Bootstrapov grid sustav. To je knjižnica HTML / CSS komponenti koje omogućuju lako strukturiranje web mjesta i postavljanje sadržaja web stranice na željene lokacije.

2.2.3. JS/TS

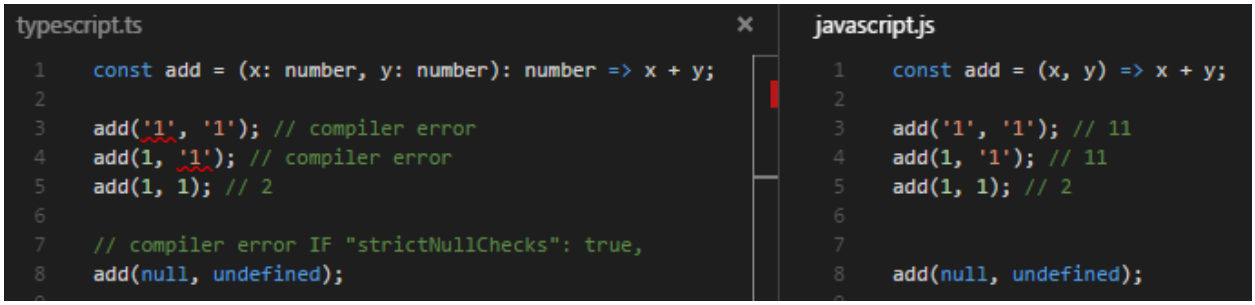
JavaScript[12] (JS) je programski jezik koji se obično koristi u web razvoju. Razvijen od strane Netscape-a, služi kao sredstvo za dodavanje dinamičkih i interaktivnih elemenata na web stranice. Dok na JS utječe Java, sintaksa je sličnija C-u i temelji se na ECMAScript-u, skriptnom jeziku kojeg je razvio Sun Microsystems.

JavaScript je skriptni jezik na strani klijenta, što znači da se izvorni kôd obrađuje u web pregledniku, a ne na web poslužitelju. To znači da se JavaScript funkcije mogu pokrenuti nakon učitavanja web stranice bez komunikacije s poslužiteljem. Na primjer, JavaScript funkcija može provjeriti web obrazac prije nego što se pošalje kako bi provjerila jesu li sva potrebna polja ispunjena. JavaScript kôd može proizvesti poruku o pogrešci prije nego što se bilo kakve informacije stvarno prenesu na poslužitelj.

Kao i skriptni jezici na strani poslužitelja, npr. PHP i ASP, JavaScript kôd se može umetnuti bilo gdje unutar HTML-a web stranice. Međutim, u HTML-u se od strane poslužitelja prikazuje samo izlazni kôd, dok JavaScript kôd ostaje u potpunosti vidljiv u izvoru web stranice. JS se može pozivati i iz zasebne .JS datoteke koja se također može vidjeti u pregledniku.

TypeScript[13] je programski jezik otvorenog kôda koji je razvio i održavao Microsoft. To je strog sintaktički nadskup (eng. *superset*) JavaScripta koji JS jeziku dodaje tipove.

TS je dizajniran za razvoj velikih aplikacija i kompilira se u JavaScript. Kako je TS nadskup JavaScript-a, postojeći JS programi su, također, važeći TS programi. TS se može koristiti za razvoj JS aplikacija za izvršavanje na strani klijenta i poslužitelja (Node.js).

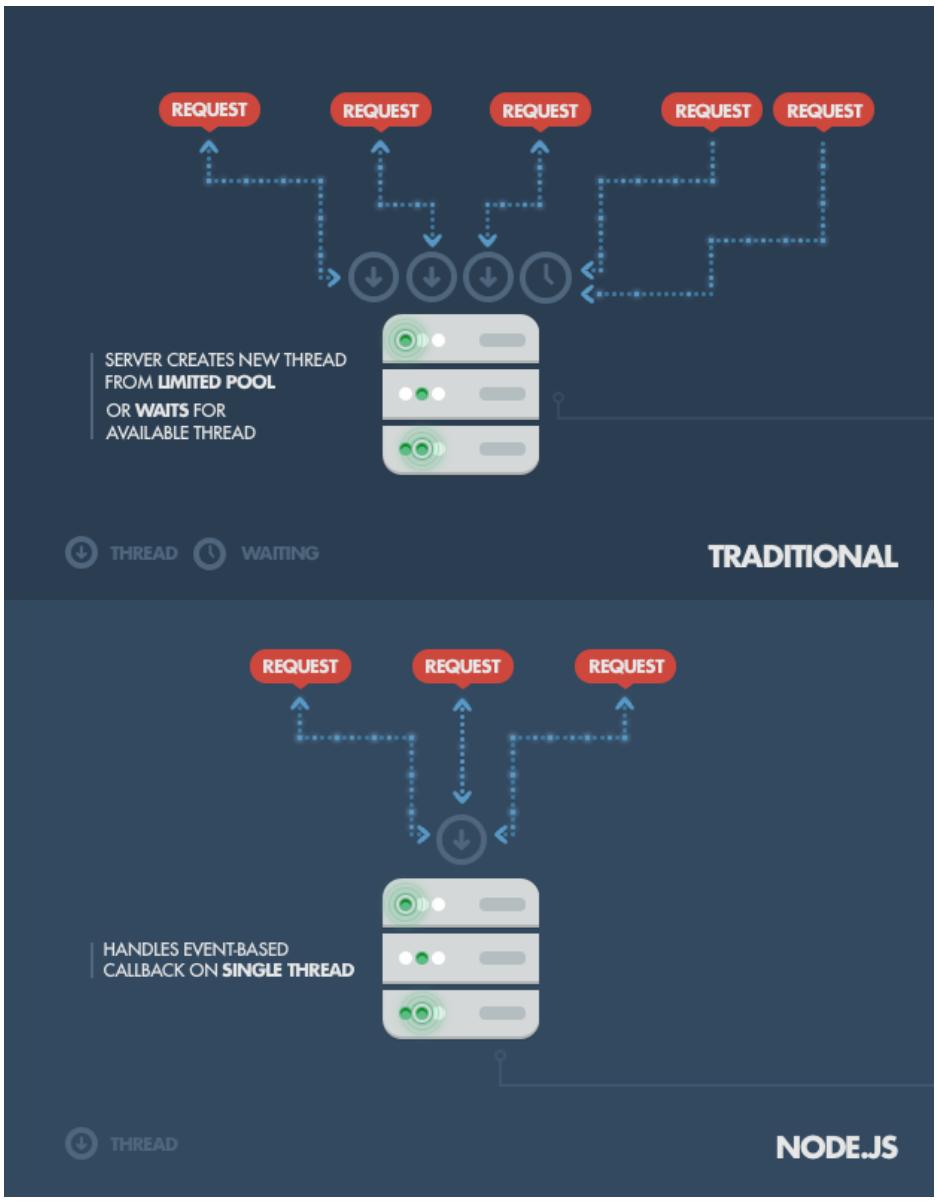


2.8 Razlika između JS i TS kôda [14]

Kako bi se lakše uhvatile greške prilikom sastavljanja u JS, TS može koristiti definiranje tipova varijabla[14], dok JS kôd ne prepoznaje krivu upotrebu tipova i vrijednosti varijabli.

2.3. Node.js

Node.js[15] je open-source, cross-platform, JavaScript run-time okruženje koje izvršava JavaScript kôd izvan preglednika. Node.js omogućava programerima da koriste JavaScript za pisanje alata za naredbene retke (command line tools) i za skriptiranje na strani poslužitelja[16] - pokretanje skripti na strani poslužitelja za izradu dinamičnog sadržaja web stranice prije nego što se stranica pošalje u web-preglednik korisnika. Node.js se najbolje koristi u web-aplikacijama u stvarnom vremenu (real-time web-apps) koristeći tzv. „push“ (gurajuću) tehnologiju preko websockets-a. Za razliku od starijih alata, Node.js omogućuje ostvarivanje web aplikacije s dvosmjernom komunikacijom u stvarnom vremenu, pri kojoj i klijent i poslužitelj mogu započeti komunikaciju. U usporedbi s tradicionalnim tehnikama posluživanja web mjesta gdje svaka veza (zahtjev) proizvodi novu nit koristeći RAM sustava i na kraju time smanjuje količinu dostupne RAM memorije, Node.js djeluje na jednoj niti, koristeći ne-blokirajuće I/O pozive (non-blocking I/O calls), što omogućuje podržavanje nekoliko tisuća istovremenih veza održanih u petlji događaja.



Slika 2.9 Razlika između tradicionalnih i node.js zahtjeva [16]

2.3.1. NPM

Govoreći o Node.js, važno je i spomenuti ugrađenu podršku za upravljanje paketima pomoću NPM-a (eng. Node Package manager)[16], alata koji dolazi sa svakom Node.js instalacijom. Ideja NPM modula jest skup javno dostupnih komponenata za višekratnu upotrebu, dostupnih jednostavnom instalacijom putem internetskog repozitorija. Potpuni popis modula može se naći na NPM web stranici ili im se može pristupiti pomoću NPM CLI alata koji se automatski instalira s Node.js. Sustav modula je otvorenog tipa i svatko može objaviti vlastiti modul koji će biti naveden u NPM repozitoriju. Neki od najpopularnijih modula su express.js, Moment.js, lodash, Async, Request, Grunt i Debug.

```
Microsoft Windows [Version 10.0.18362.418]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\xampp\htdocs\faks-projekt>npm list -g --depth 0
C:\Users\Miriam\AppData\Roaming\npm
+-- @angular/cli@8.3.6
+-- grunt-cli@1.3.2
`-- sass@1.23.0-module.beta.1


C:\xampp\htdocs\faks-projekt>
```

Slika 2.10 Prikaz lokalno instaliranih npm paketa pomoću npm naredbe

2.4. XAMPP

XAMPP[17] je softver otvorenog kôda razvijen od strane Apache-a i suradnika. Paket XAMPP sadrži Apache distribucije za Apache server, MariaDB, PHP i Perl. To je ustvari lokalni poslužitelj na lokalnom računalu. Koristi se za testiranje i razvoj web stranica i aplikacija prije nego što se te stranice postave (upload) na udaljeni web poslužitelj. Ikona XAMPP-a je „X“ koja označava funkcionalnost na više različitih platforma (eng. *Cross*-platform), (A) Apache server, (M) MariaDB, (P) PHP i (P) Perl. Cross-platforma obično znači da se može pokretati na bilo kojem računalu s bilo kojim operativnim sustavom.

2.5. Lumen/Laravel

Laravel[18] je jedan od najpopularnijih PHP okvira koji se koristi širom svijeta za izradu web aplikacija u rasponu od malih do velikih projekata. To je izbor profesionalnih programera zbog svojih performansi, značajki i skalabilnosti. Laravel slijedi MVC (Model View Controller) strukturu koja olakšava učenje i brzo prototipiranje web aplikacija jer pruža ugrađene značajke poput provjere autentičnosti, pošte, usmjeravanja, sesije itd.

MVC[19] je model dizajna aplikacije koji se sastoji od tri međusobno povezana dijela. To uključuje model (podatke), prikaz (korisničko sučelje) i kontroler (proces koji upravlja ulazom). Taj model se često koristi za razvoj modernih korisničkih sučelja.

Ukratko, Lumen[20] je lakša verzija Laravela. Dok je Laravel kompletan framework s raznoraznim značajkama, Lumen je više usmjeren na mikroservise - male, slabo spojene komponente koje obično podržavaju i poboljšavaju temeljni projekt. Mikroservisi su odvojene komponente s ograničenim kontekstima, stoga imaju dobro međusobno definirana sučelja.

Lumen je „mikro-framework“[21], što znači da je riječ o maloj, bržoj, mršavijoj verziji cjelovitog Laravel framework-a za web aplikacije. Koristi se za projekte i komponente koji koriste pogodnosti i snage Laravela, no žrtvuju konfigurabilnost i fleksibilnost u svrhu povećanja brzine. Lumen također ima mogućnost proširenja funkcionalnosti prema Laravelu.

2.5.1. PHP

PHP[22] je skriptni jezik i tumač koji je slobodno dostupan i koristi se prvenstveno na Linux web poslužiteljima. PHP, izvorno izveden iz „Personal Home Page Tools“, sada predstavlja PHP: Hypertext Preprocessor, što PHP FAQ opisuje kao „rekurzivnu kraticu“.

PHP se izvršava na poslužitelju, dok se usporediva alternativa, JavaScript, izvršava na klijentu. PHP je alternativa Microsoftovoj tehnologiji Active Server Page (ASP). Kao i kod ASP-a, PHP skripta je ugrađena u web stranicu zajedno s HTML-om. Prije nego što se stranica pošalje korisniku koji ju je zatražio, web poslužitelj poziva PHP da protumači i izvrši operacije na koje se poziva u PHP skripti.

```
// PHP kôd

$studenti = array("ana", "marko", "marija", "darko");

for ($i = 0; $i < count($studenti); $i++) {
  echo $studenti[$i];
}
```

2.11 Primjer PHP FOR petlje

U ovom primjeru se koristi FOR petlja kako bi se ispisale vrijednosti niza studenata na zaslon.

2.6. Baza podataka - SQL

Baza podataka[23] je sistematska kolekcija podataka. Baze podataka podržavaju pohranu i manipulaciju podacima, to jest, olakšavaju upravljanje podacima. Sustav upravljanja bazama podataka (DBMS, eng. Database Management System) je zbirka programa koja svojim korisnicima omogućuje pristup bazi podataka, manipuliranje podacima, izvještavanje i pregled podataka. Također, pomaže u kontroli pristupa samoj bazi.

- **Tipovi sustava za upravljanje bazama podataka su:**[24]

  Hijerarhijske baze podataka
  Mrežne baze podataka
  Relacijske baze podataka
  Objektno orijentirane baze podataka
  Grafičke baze podataka
  Baze podataka ER modela
  NoSQL baze podataka

Strukturirani upitni jezik (SQL, eng. Structured Query Language)[25] standardni je računalni jezik za upravljanje relacijskim bazama podataka i manipulaciju podacima. SQL se koristi za pretraživanje, umetanje, ažuriranje i izmjenu podataka. Većina relacijskih baza podataka podržava SQL, što je prednost za administratore baza podataka (DBA, eng. Database Administrator), jer su ti jezici često potrebni za podršku baza podataka na nekoliko različitih platformi.

2.6.1. MySQL

MySQL[26] je sustav za upravljanje relacijskim bazama podataka temeljen na strukturiranom upitnom jeziku (SQL). Radi na gotovo svim platformama, uključujući Linux, UNIX i Windows. Iako se može koristiti u širokom rasponu aplikacija, najčešće je povezan s web aplikacijama i internetskim objavljivanjem.

MySQL se temelji na modelu klijent-poslužitelj. Jezgra MySQL-a je MySQL poslužitelj koji obrađuje sve upute (ili naredbe) baze podataka. Taj poslužitelj je dostupan kao zaseban program za korištenje u umreženom okruženju klijent-poslužitelj, te kao knjižnica koja se može ugraditi (ili povezati) u zasebne aplikacije.

Bazom podataka može se upravljati pomoću jednostavnih naredba poput *CREATE*, *SELECT*, *UPDATE* i *DELETE*, i mnogih drugih.

```
SELECT * FROM `studenti` WHERE `studentski_broj` = 1;
```

2.12. Primjer SQL upita

U ovom primjeru tražimo prikaz reda iz tablice „studenti“ gdje je kolona „studentski_broj“ jednaka broju 1.

3. Izrada aplikacije

Izrada aplikacije sastoji se od nekoliko različitih dijelova, uključujući planiranje, instalaciju, stvarnu izradu, i testiranje. U sljedećim dijelovima detaljnije su opisani koraci razvoja aplikacije.

3.1. Planiranje i skiciranje

Jedna od najvažnijih stvari prilikom izrade aplikacije jest planiranje. Dobar plan i strukturirana izrada često olakšavaju rješavanje problema koji se mogu javiti tijekom izrade.

Kako bi produkt bio što sličniji željama klijenta, prvo se radi sastanak (eng. *Kick-off*) za početak projekta. To je proces koji se sastoji od dogovaranja temeljnih funkcija aplikacije, njenog generalnog izgleda, te eventualno detalja vezanih uz sam projekt. Obzirom da je ova aplikacija relativno mala opsegom, taj proces je vrlo jednostavan.

Tijekom sastanka određene su slijedeće točke koje odgovaranju na pitanja:

Što je primarna funkcija aplikacije?

Primarna funkcija je prikaz svih kredita korisnika, te spremanja novih kredita

Kako će cijela aplikacija funkcionirati?

Prikaz svih kredita u pregledniku kao redova s korisničkim imenom, trenutnim kreditom i opcijom za spremanje istih

Koji jezici i razvojni okviri će biti korišteni za izradu aplikacije?

MySQL kao baza podataka, Lumen (PHP) kao back-end, Angular 8 (TS, HTML, SCSS) kao front-end

Koji su primarni elementi aplikacije?

Početna stranica za redirekciju na originalnu web stranicu i stranicu za prikaz kredita

Stranica za prikaz kredita

Originalna web stranica (nevezana uz aplikaciju – vidi dodatne informacije)

Kako će se primati i spremati podatci?

Shema za pristup podacima:

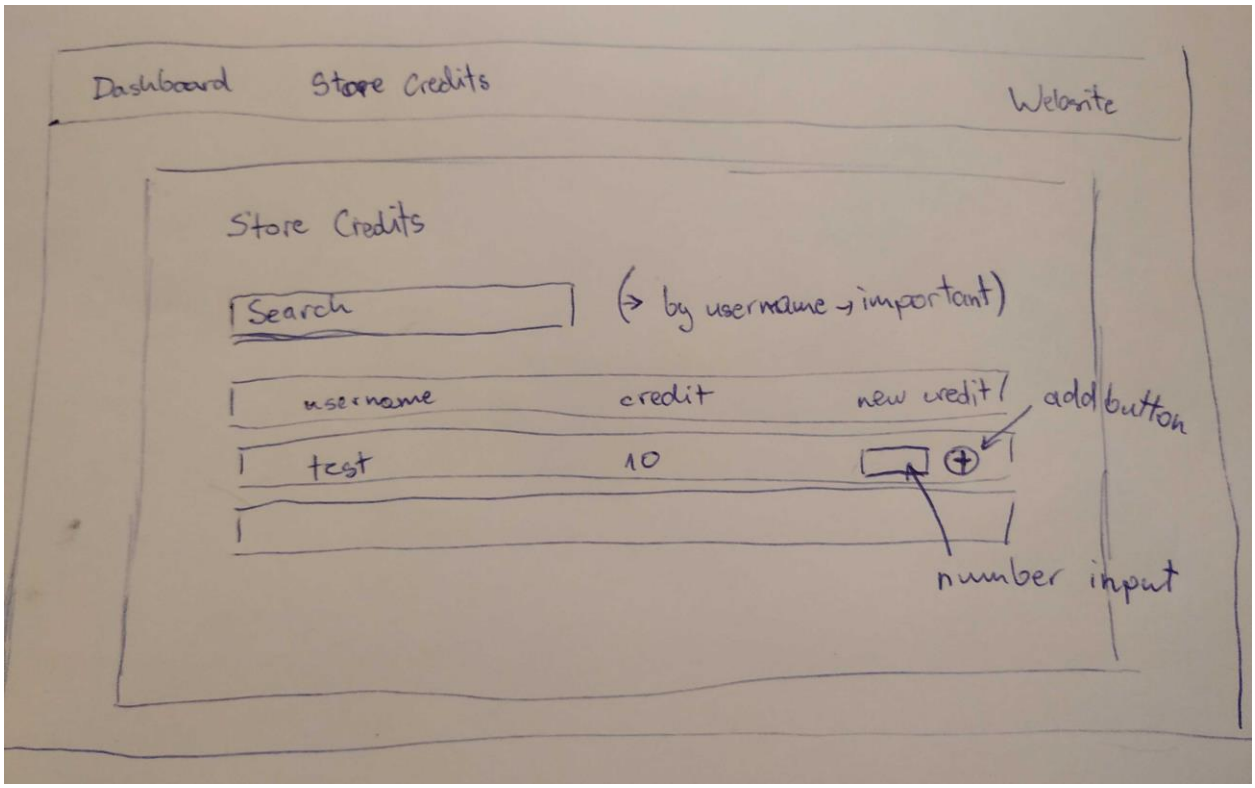
Angular 8 http get request → Lumen raw DB upit → MSQL → Lumen → Angular8

Shema za spremanje podataka:

Angular 8 http post request → Lumen raw DB upit → MSQL → Lumen response →Angular 8 response

Postoji li skica ili ideja za dizajn?

Skica prikaza korisničkih kredita:



3.13 Skica prikaza korisničkih kredita

Dodatne informacije?

Dizajn same aplikacije je direktno vezan za dizajn postojeće web stranice klijenta, adresa: www.trading-cards-united.at → Izraditi prikaz kredita prema stranici

Postojeća web stranica treba biti ukomponirana u sam projekt → Lumen

Obzirom da je većina odgovora na ta pitanja objašnjena u prvom dijelu rada, neki su odgovori skraćeni kako bi se pažnja posvetila samoj izradi.

3.2. Postavljanje aplikacije

U ovom odjeljku opisani su procesi instalacije svih potrebnih i nekih dodatnih alata za razvoj ove aplikacije.

3.2.1. Instalacija alata

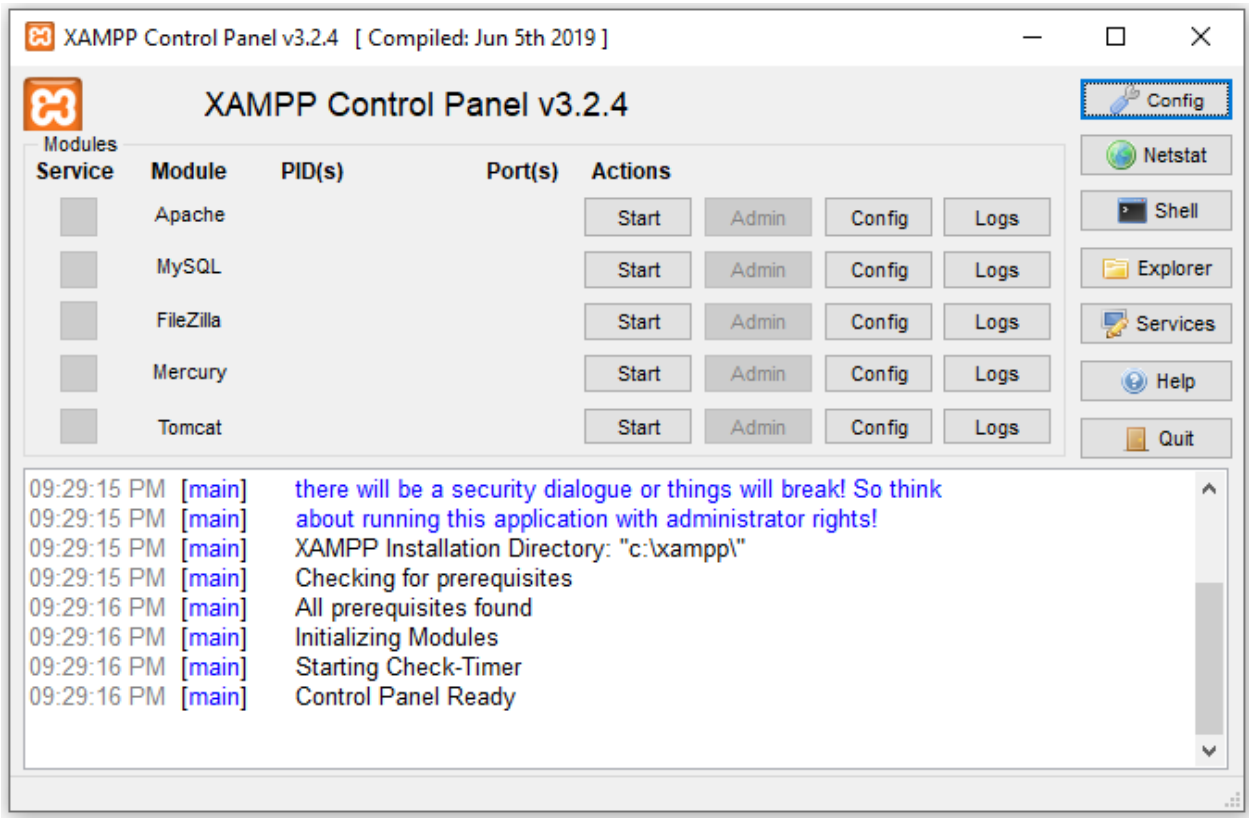
Prilikom instalacije sljedećih alata dobro je slijediti navedeni redoslijed kako bi se izbjegle moguće greške ili koruptirane datoteke.

Instalacija back-enda:

XAMPP (PHP, Apache, MySQL)

Na računalima s Windows operativnim sustavom, prvo moramo instalirati XAMPP kako bismo imali pristup virtualnom Apache serveru i MSQL bazi podataka. Uz to, XAMPP dolazi s PHP-om koji bi se u drugom slučaju također morao zasebno instalirati na računalo. Kao i većina uobičajenih Windows aplikacija, XAMPP dolazi kao izvršna datoteka koja instalaciju čini vrlo jednostavnom. Nakon instalacije, iznimno je važno staviti php.exe u Windows putanju (eng. *path*) u varijable okruženja (eng. *Enviromental Variables*). Pod pretpostavkom da je XAMPP instaliran na "C:\xampp\", putanja bi trebala izgledati ovako: "C:\xampp\php\".

Jedan od čestih problema na Windows računalima javlja se zbog okupiranosti port:80. Windows ponekad taj port koristi za servise. Kako bi se problem riješio, taj servis treba isključiti ili promijeniti postavke servisa tako da se port:80 oslobodi.



Slika 2.14 Prikaz XAMPP-a

Composer

Nakon instalacije XAMPP-a (točnije, PHP-a), može se instalirati Composer koji je potreban kako bi se mogao instalirati Lumen. Poput XAMPP-a, Composer dolazi kao izvršna datoteka za Windows operativni sustav. Iako postoji više načina instalacije, na ovaj način dobiva se najnovija verzija i automatsko dodavanje u Windows *path*.

Lumen

Kada je instalacija Composer-a završena, mogu se upotrijebiti jednostavne naredbe kako bi se instalirao Lumen. Za instalaciju Lumena, u prozoru naredbenog retka valja upisati naredbu:

```
composer global require "laravel/lumen-installer"
```

3.15 Instalacija - Lumen

Instalacija front-enda:

Node.js

Node.js dolazi kao izvršna datoteka, što čini instalaciju vrlo jednostavnom. Prilikom instalacije također se dobiva pristup npm-u. Važno je naglasiti da node.js i npm, kao i Composer, moraju biti uvršteni u Windows *path*. Primjer npm putanje: "C:\Users\Miriam\AppData\Roaming\npm".

Angular 8

Kako se Composer koristi za instalaciju Lumen-a, tako se Angular može instalirati pomoću npm-a. Za instalaciju se koristi sljedeća naredba, gdje -g označuje globalnu instalaciju:

```
npm install -g @angular/cli
```

3.16 Instalacija - Angular 8

Nakon globalne instalacije, postoji mogućnost potrebe za instalacijom lokalne instalacije (direktno vezano za projekt). To se postiže uklanjanjem "-g" iz gore navedene naredbe.

3.2.2. Priprema baze podataka

Prije početka razvoja same aplikacije, dobro je imati plan baze podataka, kako bi se kasnije izbjegli problemi u slučaju proširenja ili mijenjanja iste.

U ovoj aplikaciji, baza podataka se sastoji od 2 tablice podataka. Tablica korisnika sadržava korisnički identifikacijski broj (id), korisničko ime, stvarno ime i prezime korisnika, korisničku email adresu, te datume stvaranja i zadnje promjene. Tablica kredita sadržava identifikacijski broj kredita, korisnički identifikacijski broj (id), kredit (vrijednost kao broj), korisničku email adresu, te datume stvaranja i zadnje promjene.

3.3. Izrada jednostavnog back-enda

U ovom odjeljku opisana je izrada osnovne Lumen aplikacije i kako povezati tu aplikaciju s bazom podataka.

3.3.1. Izrada novog projekta u Lumenu

Nakon instalacije, dobiven je pristup Lumen CLI-u, što znači da se za izradu novog projekta mogu koristiti Lumen komande u prozoru naredbenog retka. Naredba za novi projekt:

```
lumen new ime
```

3.17 Izrada nove aplikacije u Lumen-u

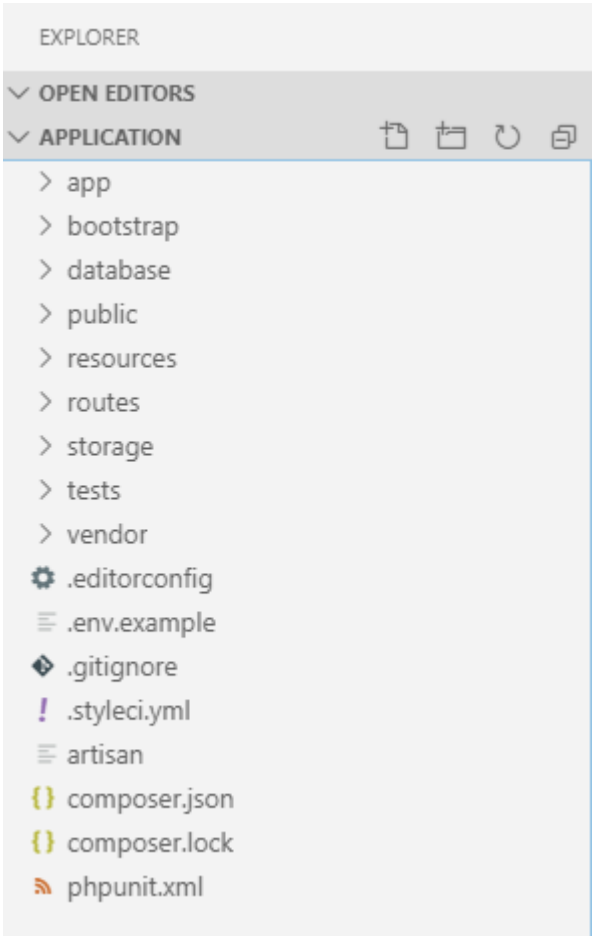
pri čemu je „ime“ naziv aplikacije. Valja napomenuti da se isti rezultat može postići pomoću Composer-a, no ta je metoda mnogo sporija nego preko Lumen-CLI. Primjer:

```
composer create-project --prefer-dist laravel/lumen ime
```

3.18 Izrada nove aplikacije u Lumen-u pomoću Composer-a

Izrada je gotova kada se u terminalu vide sljedeće poruke:
„Generating optimized autoload files“
„Application ready! Build something amazing.“

Kad se novo izrađena aplikacija otvori u željenom okruženju, može se vidjeti osnovna struktura projekta:



3.19 Prikaz strukture novog Lumen projekta

3.3.2. Povezivanje Lumena i baze podataka

Kako bi mogli pristupiti bazi podataka, ista se mora prvo i stvoriti. Uz bazu podataka, u sam projekt treba uključiti i .env datoteku koja sadrži lokaciju baze podataka i podatke potrebne za pristup istoj. Primjer .env datoteke:

```
APP_ENV=local
APP_DEBUG=true
APP_KEY=
APP_LOCALE=de
APP_DEBUG_SQL=true

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=baza_podataka
DB_USERNAME=root
DB_PASSWORD=
CACHE_DRIVER=array
SESSION_DRIVER=array
QUEUE_DRIVER=sync
```

3.20 Primjer .env datoteke

Za kreiranje baze podataka, Lumen koristi Laravelove tzv. migracijske klase[27]. Klasa migracije sadrži dvije metode: gore (eng. *up*) i dolje (eng. *down*). Metoda *up* koristi se za dodavanje novih tablica, stupaca ili indeksa u bazu podataka, dok metoda *down* dolje treba poništiti operacije izvedene metodom *up*. Jedan primjer takve migracije je klasa korisnika za aplikaciju:

```
<?php
use Illuminate\Support\Facades\Schema;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Database\Migrations\Migration;

class Users extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
      Schema::create('users', function (Blueprint $table) {
        $table->bigIncrements('id');
        $table->string('username');
        $table->string('first_name');
        $table->string('last_name');
        $table->string('email');
        $table->timestamps();
      });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
      Schema::drop('users');
    }
}
```

3.21 Migracijska klasa korisnika

U Lumenu se za slanje i primanje upita koriste rute i kontroleri kako bi se postigla komunikacija s bazom podataka (i nadalje, s front-endom).

Rute se uključuju u web.php datoteku, dok se kontroleri stavljaju u Controllers.php datoteku. Kako bi se dobili zatraženi podaci neke rute, ta ruta se mora povezati s kontrolerom, koji će tada te podatke uzeti iz baze podataka.

3.3.3. Slanje podataka prema front-endu

Za slanje podataka koriste se ranije spomenute rute i kontroleri. Kako bi se neki zahtjev obradio, taj zahtjev treba prvo i primiti. U web.php se dodaje određeni kod kako bi se Lumenu dalo do znanja da treba izvršiti narednu funkciju, koja u ovoj aplikaciji preko kontrolera vraća zatražene podatke iz baze podataka.

```
$router->get('storecredit', 'Controller@getStoreCreditList');
```

3.22 Ruta za dobivanje podataka vezanih za kredit u trgovini

U kontroler se dodaje sljedeća funkciju kako bi se vratili podaci iz baze podataka kao odgovor na zahtjev.

```
  function getStoreCreditList() {
    $credits = DB::select( DB::raw("SELECT u.username, c.credit,
c.deleted_at FROM `credits` c join users as u on c.userid = u.id") );
    return json_encode($credits);
  }
```

3.23 Kontroler za DB upit – Krediti

3.4. Izrada projekta u Angularu

U ovom odjeljku opisana je izrada osnovne Angular 8 aplikacije i kako povezati tu aplikaciju s Lumen razvojnim okvirom.

3.4.1. Stvaranje novog projekta

Nakon instalacije Angular CLI, dobiva se pristup Angularovim komandnim naredbama pomoću kojih se izrađuje, ne samo osnovna aplikacija, nego i dodatne komponente i servise. Za izradu nove aplikacije koristi se sljedeća naredba:

```
ng new my-app
```

3.24 Izrada nove Angular aplikacije pomoću Angular CLI

pri čemu je „ime“ naziv aplikacije. Angular CLI nudi i druge mogućnosti poput „ng generate“, koji stvara nove komponente, servise i sl., „ng serve“, koji poslužuje aplikaciju u lokalnom

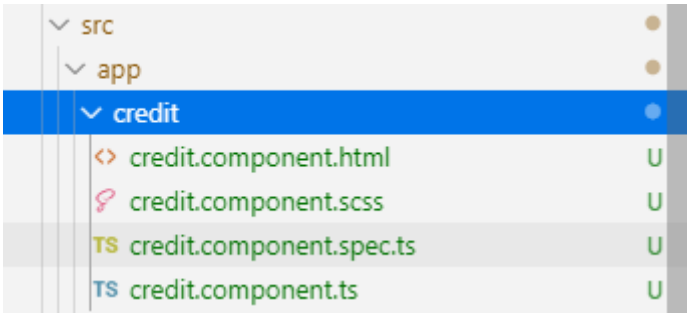
okruženju, „ng build“, koji gradi aplikaciju, i mnoge druge. „ng build“ sa oznakom --prod se koristi za gradnju aplikacije za produkciju.

3.4.2. Komponente

Kako je ranije napomenuto, za izradu komponenata koristi se „ng generate“ naredba s oznakom „component ime“, gdje je „ime“ naziv komponente. Tom naredbom automatski se stvara komponenta koja sadrži 3 (opcionalno 4) datoteke: html, ts i scss. U tim datotekama se piše kôd vezan samo uz tu komponentu.

```
ng generate component credit
```

3.35 Izrada novih komponenata u Angularu 8



3.36 Prikaz nove komponente „credit“ u Angularu

3.4.3. Moduli

NgModuli[28] konfiguriraju injektore i kompajler, te pomažu u organiziranju povezanih stvari. To su datoteke koje definiraju sve dijelove programa.

NgModule uzima objekt meta podataka koji opisuje kako sastaviti predložak komponenata i kako stvoriti injektor za vrijeme izvođenja. Identificira vlastite komponente, direktive i cijevi (eng. *pipes*) modula, čineći neke od njih javnim putem svojstva tako da ih vanjske komponente mogu koristiti. @NgModule također može dodati davatelje usluga injektorima ovisnosti. Slijedi prikaz osnovnog modula koji svaka aplikacija mora koristiti i bez kojeg ne može funkcionirati:

```
// imports
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';

import { AppComponent } from './app.component';

// @NgModule decorator with its metadata
```

```
@NgModule({
  declarations: [AppComponent],
  imports: [BrowserModule],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule {}
```

3.37 Prikaz osnovnog (App) modula – Angular 8

3.4.4. Servisi

Servisi[29] su široka kategorija koja obuhvaća bilo koju vrijednost, funkciju ili značajku koju aplikacija treba. Servis je obično klasa s uskom, dobro definiranom svrhom.

Komponenta može delegirati određene zadatke servisima, kao što je dohvaćanje podataka s poslužitelja, provjeravanje korisničkog unosa ili prijava izravno na konzolu. Servisi su na raspolaganje bilo kojoj komponenti. U ovoj aplikaciji, servisi služe kao pomagači za http pozive prema Lumenu (poslužitelju).

```
import { Injectable } from '@angular/core';

@Injectable({ providedIn: 'root' })
export class StoreCreditService {
    }
```

3.38 Primjer praznog servisa – Angular 8

3.4.5. Povezivanje front-end aplikacije s Lumen-om

Pripremanje Angulara za primanje podataka svodi se na servise i pripremu prikaza podataka. Http zahtjev odvija se pomoću servisa (npr. *storecredit.service.ts*), koji se poziva u komponenti koja prikazuje i manipulira tim podacima. U ovoj aplikaciji, taj servis se koristi za dobivanje liste kredita iz baze podataka.

```
getStoreCreditList() {
        return this.http.get<StoreCredit[]>(
            this.url + 'storecredit',
            this.httpOptions
        );
    }
```

3.39 Prikaz http get zahtjeva - Angular 8

Za povezivanje Angulara s Lumen back-endom, prvo treba provjeriti šalje li Lumen točne podatke na željenu url putanju. Kako je ranije navedeno, u datoteci web.php određena je url putanja preko koje se obrađuje zahtjev front-enda. U tom primjeru „storecredit“ je tražena putanja. Ako sve radi kako treba s strane Lumena, prije no što su te aplikacije povezane, na toj stranici bi trebali vidjeti jedino JSON objekt koji sadrži podatke iz tablice „credits“ baze podataka.

```
{ username: Korisničko Ime, credit: 50, deleted_at: NULL }
```

3.40 Primjer povrata željenog JSON objekta

U komponenti “StoreCredits”, tijekom inicijalizacije, poziva se servis za primanje liste kredita iz baze podataka. Ako je „this.url + storecredit“ isti url na koji Lumen šalje odgovor, komponenta će imati pristup JSON objeku koji sadrži tražene podatke.

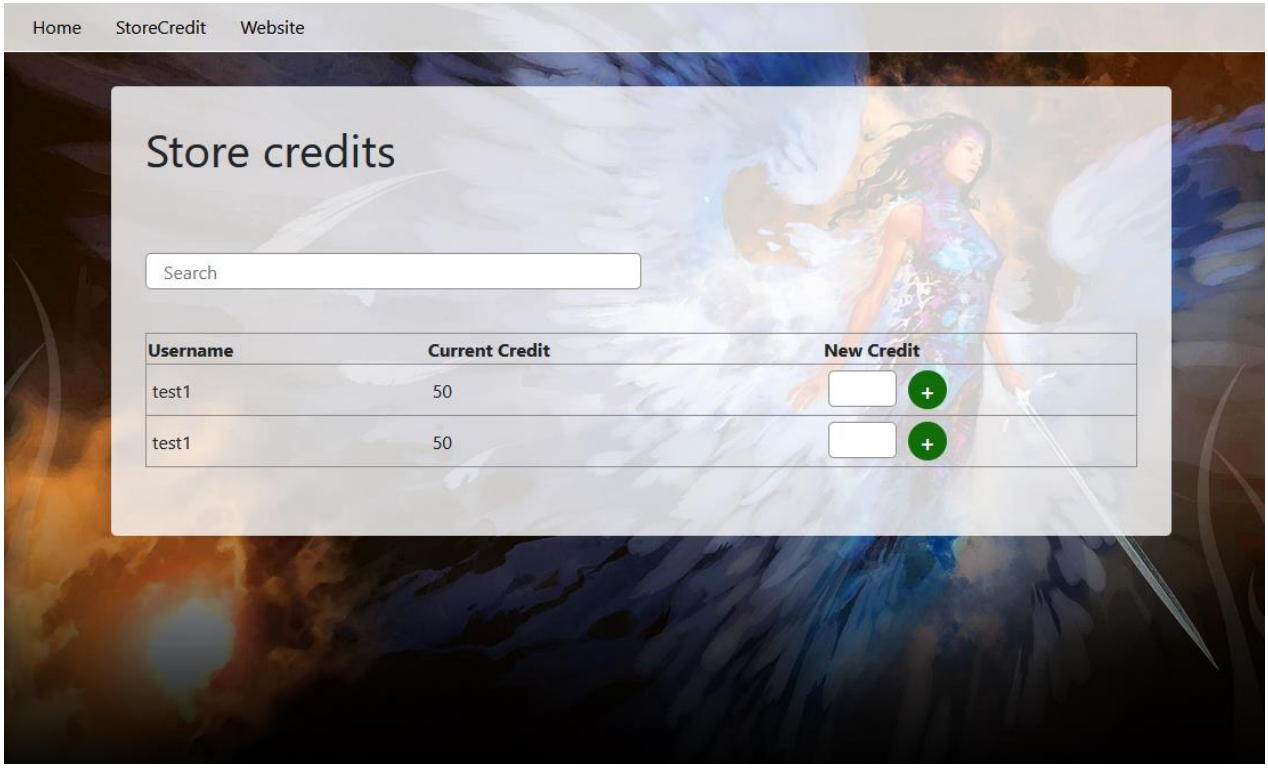
Tijekom objavljivanja aplikacije, važno je promijeniti osnovni url Angular aplikacije kako bi se mogla koristiti kroz lumen bez pokretanja. U tsconfig.json datoteci možemo odrediti željeni direktorij za spremanje izgrađene aplikacije. Lumen mora imati pristup tom direktoriju. Naposljetku, prije finalne izgradnje aplikacije, nužno je promijeniti bazni url aplikacije, na direktorij gdje će se izgrađena aplikacija na kraju i nalaziti.

```
<!-- Use this to serve the app -->
<base href="/" />

<!-- Use this for building! -->
<base href="http://localhost:3000/app/" />
```

3.41 Primjer osnovne url putanje - Angular 8

4. Konačan izgled aplikacije

Ovo je finaliziran prikaz liste kredita po korisnicima.



4.42 Konačni izgled aplikacije

5. Zaključak

Jedna od češćih dilema s kojom su programeri suočeni je koju bi platformu trebali koristiti za izradu web aplikacija. Odabir pravog razvojnog okvira je najčešće presudan dio razvojnog procesa.

Ovaj rad prikazuje postupak izgradnje jedne web aplikacije uz pomoć Angular 8 i Lumen razvojnih okvira. Programski i skriptni jezici, te HTML i CSS osnova su razvoja web aplikacija.

Angular je besplatan razvojni okvir izgrađen na JavaScriptu / TypeScriptu te je dobro poznat po svojoj fleksibilnosti. Minimizira vrijeme pisanja kôda, koristi MVC strukturu za razvoj i nudi mogućnost pisanja vlastitih testova. Lumen je također besplatan razvojni mikro-okvir koji je dizajniran za izgradnju brzih mikro usluga i API-a. Koristi biblioteke Laravela te ima mogućnost proširenja ako temeljne funkcije u nekom trenutku više nisu dovoljne za web aplikaciju (npr. nova verzija ili proširenje funkcionalnosti).

Kroz izradu web aplikacije korišten je XAMPP kao lokalni poslužitelj, Lumen kao PHP back-end i MySQL baza podataka. Za razvoj front-end aplikacije, koristio se Angular 8, pri čemu su korištene Node.js biblioteke pomoću npm paket menadžera. Na posljetku, Angular 8 je pomoću servisa povezan s Lumenom kako bi se dobio pristup podatcima iz baze podataka.

Razlozi odabira upravo tih razvojnih okvira su jednostavna instalacija i implementacija, te činjenica da su oba vrlo dobro održavana i stalno ažurirana. Iako se web aplikacija može razviti i samo u Lumenu, postoje neke situacije u kojima je bolje koristiti JS/TS, to jest u ovom slučaju Angular, kao npr. korištenje AJAX-a za provjeru neke vrijednosti u bazi podataka, ili čak provjeru unosa prije samo slanja na server, kako bi se izbjeglo ponovno učitavanje stranice.

U Varaždinu, 06.11.2019. Brezovec

Datum Potpis

Sveučilište
Sjever

SVEUČILIŠTE
SJEVER

IZJAVA O AUTORSTVU
I
SUGLASNOST ZA JAVNU OBJAVU

Završni/diplomski rad isključivo je autorsko djelo studenta koji je isti izradio te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, izvora s interneta, i drugih izvora) bez navođenja izvora i autora navedenih radova. Svi dijelovi tuđih radova moraju biti pravilno navedeni i citirani. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom, odnosno nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, Miriam-Valentina Brezovec *(ime i prezime)* pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/~~diplomskog~~ *(obrisati nepotrebno)* rada pod naslovom Izrada web aplikacija primjenom Angular8 i Lumen razvojnog okvira *(upisati naslov)* te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:
(upisati ime i prezime)

Miriam Brezovec

(vlastoručni potpis)

Sukladno Zakonu o znanstvenoj djelatnost i visokom obrazovanju završne/diplomske radove sveučilišta su dužna trajno objaviti na javnoj internetskoj bazi sveučilišne knjižnice u sastavu sveučilišta te kopirati u javnu internetsku bazu završnih/diplomskih radova Nacionalne i sveučilišne knjižnice. Završni radovi istovrsnih umjetničkih studija koji se realiziraju kroz umjetnička ostvarenja objavljuju se na odgovarajući način.

Ja, Miriam Valentina Brezovec *(ime i prezime)* neopozivo izjavljujem da sam suglasan/na s javnom objavom završnog/~~diplomskog~~ *(obrisati nepotrebno)* rada pod naslovom Izrada web aplikacija primjenom Angular8 i Lumen razvojnog okvira *(upisati naslov)* čiji sam autor/ica.

Student/ica:
(upisati ime i prezime)

Miriam Brezovec

(vlastoručni potpis)

6. Literatura

Sve poveznice/linkovi su bili dostupni tijekom rujna i listopada 2019.

[1] https://www.techopedia.com/definition/13128/programming

[2] https://yearofcodes.tumblr.com/what-is-coding

[3] https://info.jkcp.com/blog/coding-vs-programming

[4] https://techterms.com/definition/programming_language

[5] https://www.techopedia.com/definition/3873/scripting-language

[6] https://searchsoftwarequality.techtarget.com/definition/integrated-development-environment

[7] https://angular.io/guide/architecture

[8] https://www.theserverside.com/definition/HTML-Hypertext-Markup-Language

[9] https://techterms.com/definition/css

[10] https://en.wikipedia.org/wiki/Sass_(stylesheet_language)

[11] https://responsivedesign.is/articles/difference-between-sass-and-scss/

[12] https://techterms.com/definition/javascript

[13] https://en.wikipedia.org/wiki/TypeScript

[14] https://medium.com/front-end-weekly/typescript-vs-javascript-a3c0beb8b6d9

[15] https://en.wikipedia.org/wiki/Node.js

[16] https://www.toptal.com/nodejs/why-the-hell-would-i-use-node-js

[17] https://www.wpblogx.com/what-is-xampp/

[18] https://www.larashout.com/what-is-laravel-and-why-you-should-learn-it

[19] https://techterms.com/definition/mvc

[20] https://mattstauffer.com/blog/introducing-lumen-from-laravel/

[21] https://medium.com/@9series.solution/lumen-a-magnificently-fast-micro-service-php-framework-part-i-25f77a480023

[22] https://whatis.techtarget.com/definition/PHP-Hypertext-Preprocessor

[23] https://www.guru99.com/introduction-to-database-sql.html

[24] https://www.c-sharpcorner.com/UploadFile/65fc13/types-of-database-management-systems/

[25] https://www.techopedia.com/definition/1245/structured-query-language-sql

[26] https://searchoracle.techtarget.com/definition/MySQL

[27] https://laravel.com/docs/6.x/migrations

[28] https://angular.io/guide/ngmodules

[29] https://angular.io/guide/architecture-services

Popis slika i primjera kodova

2.2 Službeni dijagram relacije osnovnih blokova za gradnju u Angularu 8 [7] 4
2.3 Primjer kôda vrlo jednostavne HTML stranice 5
2.4 Primjer korištenja css-a s html-om 6
2.5 Primjer korištenja css-a s html-om – zaslonski prikaz.....6
2.6 Primjer korištenja php-a, html-a i css-a zajedno.....7
2.7 Razlika između SASS i SCSS sintakse [11].....8
2.8 Razlika između JS i TS kôda [14].....9
Slika 2.9 Razlika između tradicionalnih i node.js zahtjeva[16].....10
Slika 2.10 Prikaz lokalno instaliranih npm paketa pomoću npm naredbe.....11
2.11 Primjer PHP FOR petlje.....12
2.12. Primjer SQL upita.....13
3.13 Skica prikaza korisničkih kredita.....15
Slika 2.14 Prikaz XAMPP-a 16
3.15 Instalacija - Lumen 17
3.16 Instalacija - Angular 8.....17
3.17 Izrada nove aplikacije u Lumen-u.....18
3.18 Izrada nove aplikacije u Lumen-u pomoću Composer-a 18
3.19 Prikaz strukture novog Lumen projekta.....19
3.20 Primjer .env datoteke 19
3.21 Migracijska klasa korisnika 20
3.22 Ruta za dobivanje podataka vezanih u kredit u trgovini.....21
3.23 Kontroler za DB upit – Krediti 21
3.24 Izrada nove Angular aplikacije pomoću Angular CLI.....21
3.35 Izrada novih komponenata u Angularu 8.....22
3.36 Prikaz nove komponente „credit“ u Angularu 22
3.37 Prikaz osnovnog (App) modula – Angular 8 23
3.38 Primjer praznog servisa – Angular 8 23
3.39 Prikaz http get zahtjeva - Angular 8 23
3.40 Primjer povrata željenog JSON objekta.....24
3.41 Primjer osnovne url putanje - Angular 8.....24
4.42 Konačni izgled aplikacije.....25

Prilozi

Aplikacija se nalazi na glavnom računalu u K-29.