Poslužiteljska web aplikacija za centralizirano naručivanje hrane u Osijeku

Katić, David

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

Preddiplomski stručni studij

POSLUŽITELJSKA WEB APLIKACIJA ZA CENTRALIZIRANO NARUČIVANJE HRANE U OSIJEKU

Završni rad

David Katić

Osijek, 2018.

SADRŽAJ

1. UVOD

Ubrzani tempo današnjeg svijeta zahtijeva načine povećanja produktivnosti u poslovanju što ujedno donosi i uštedu vremena. U vidu olakšanog poslovanja ovaj rad se bavi izradom Web aplikacije koja će olakšati zaprimanje narudžbi unutar restorana.

Glavni je fokus aplikacije olakšati i organizirati zaprimanje narudžbi restoranima na području grada Osijeka.

U drugom se poglavlju govori o razvojnoj okolini i tehnologijama koje su potrebne pri izradi Web aplikacije. Od toga su tehnologije: HTML, CSS, JAVASCRIPT, jQuery, Google Firebase, Node.js i Visual Studio Code.

Treće poglavlje prikazuje izradu Web aplikacije po koracima. Prolazi se čitavim procesom izrade aplikacije, od osnovnog dizajna do konačne verzije s pripadajućim slikama i dijelovima koda.

U četvrtom poglavlju se objašnjava rad aplikacije.

Zaključak cjelokupnog rada nalazi se u petom poglavlju koje je ujedno i posljednje u ovom završnom radu.

1.1. Zadatak rada

Zadatak je ovog rada izraditi funkcionalnosti za prihvaćanje narudžbi na poslužitelju i prosljeđivanje prema ugostiteljskim obrtima te modelirati bazu podataka koja može podržati rad takve Web aplikacije. Nadalje, zadatak je objasniti način rada centraliziranog naručivanja hrane u nekom gradu i povezati Web aplikaciju s Android klijentskom aplikacijom, zatim testirati i opisati rad poslužiteljske aplikacije.

2. RAZVOJNA OKOLINA I TEHNOLOGIJE ZA WEB APLIKACIJE

Prije samog početka pisanja programskog koda, potrebno je upoznati se s tehnologijama koje će se koristiti za ovaj zadatak kako bi se lakše razumjela problematika zadatka te rješenje istoga.

2.1. HTML

HTML je jezik za opis strukture web stranica. HTML daje autorima sredstva za:

- Objavljivanje mrežnih dokumenata s naslovima, tekstom, tablicama, popisima, fotografijama itd.
- Preuzimanje online informacije putem hipertekstualnih veza klikom na gumb.
- Oblikovanje obrazaca za: obavljanje transakcija udaljenim putem, korištenje u traženju informacija, pri rezerviranju, naručivanju proizvoda itd.
- Uključiti tablice, video isječke, zvučne isječke i druge programe izravno u njihove Web stranice.

Pomoću HTML-a, autori opisuju strukturu stranice korištenjem oznaka (engl. *tags*). Elementi jezika označavaju dijelove sadržaja kao što su: " paragraf ", "lista", "tablica" i tako dalje [1].

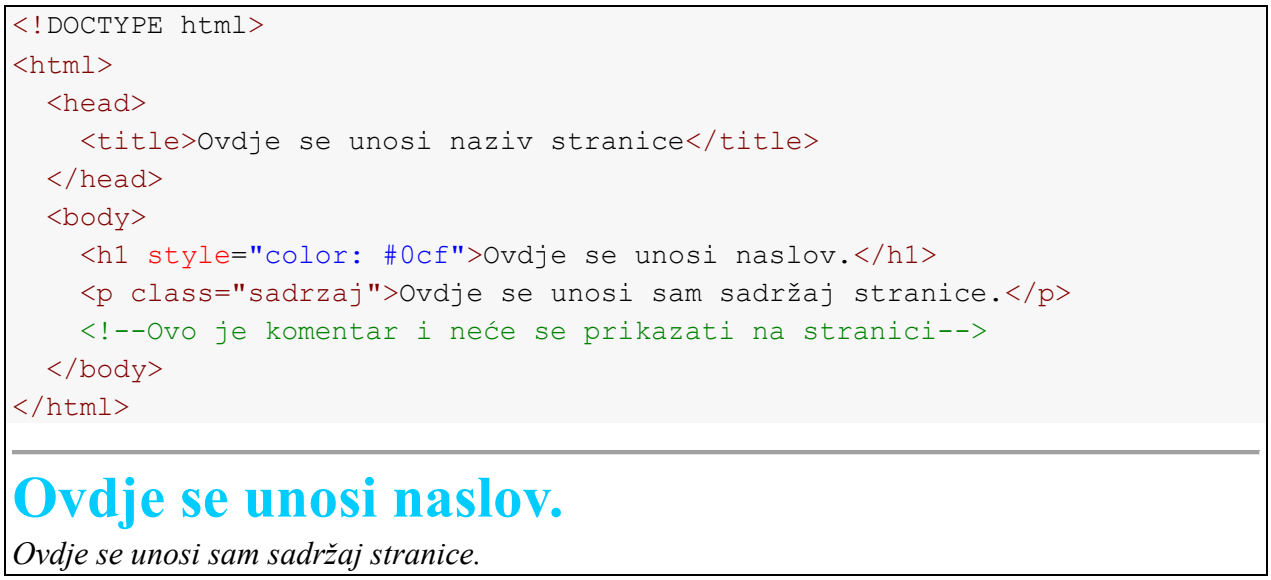
Prvu verziju HTML-a opisao je Tim Berners-Lee krajem 1991. godine. Tijekom prvih pet godina (1990.-1995.), HTML je prošao kroz niz izmjena i dobio brojna proširenja, uglavnom prvo u CERN-u, a potom u IETF.

HTML oznake mogu se razumjeti kao naredbe web pregledniku koje određuju kako će se prikazati sadržaj web stranice. (Npr.: "Ovo je odlomak", "Ovo je slika".) Svaka HTML oznaka navodi se unutar znakova < > , tj. izlomljenih zagrada [2].

Svaka HTML oznaka dolazi u paru, odnosno, sastoji se od početne i završne HTML oznake. Primjerice, oznaka <html> označava početak HTML dokumenta, a oznaka </html> označava završetak HTML dokumenta. Može se uočiti da završna oznaka ispred svojeg naziva ima desnu kosu crtu / [2].

HTML (engl. *HyperText Markup Language*) i CSS (engl. *Cascading Style Sheets*) dvije su osnovne tehnologije za izgradnju Web stranica. HTML pruža strukturu stranice, dok CSS pruža vizualni i zvučni izgled za različite uređaje. Uz grafiku i skripte, HTML i CSS osnova su za izgradnju Web stranica i Web aplikacija [1].

```html
<!DOCTYPE html>
<html>
  <head>
    <title>Ovdje se unosi naziv stranice</title>
  </head>
  <body>
    <h1 style="color: #0cf">Ovdje se unosi naslov.</h1>
    <p class="sadrzaj">Ovdje se unosi sam sadržaj stranice.</p>
    <!--Ovo je komentar i neće se prikazati na stranici-->
  </body>
</html>
```



Isječak programskog kôda 2.1. *Gore je HTML kôd, a dolje rezultat koji će se prikazati na ekranu.*

2.2. CSS

CSS (engl. *Cascading Style Sheets*) je jezik za opis prezentacije Web stranica, uključujući boje, izgled i fontove. Omogućuje prilagodbu prikaza za različite vrste uređaja, poput velikih i malih zaslona ili pisača. CSS neovisan je o HTML-u i može se koristiti s bilo kojim *Markup* jezikom koji se temelji na XML-u.

Razdvajanje HTML-a iz CSS-a olakšava održavanje web stranica, dijeljenje stilskih listova na različitim stranicama i prilagođavanje stranica u različitim okruženjima. To se naziva razdvajanjem strukture (sadržaja) od prezentacije [3].

Oznaka klase na početnoj oznaci stavke ("<p>") može se, između ostalog, koristiti za dodavanje stila. Na primjer, za istaknuti tekst svih stavaka s klasom *"sadrzaj"* (Isječak programskog kôda 2.2.), može se pisati u CSS-u:

```css
p.sadrzaj { font-style: italic; }
```

Isječak programskog kôda 2.2. *Prikaz vezanja klase na paragraf.*

Postavljanjem tog pravila u zasebnu datoteku, stil se može dijeliti s bilo kojim brojem HTML dokumenata.

Selektori su uzorci koji se podudaraju s elementima u stablu i kao takvi čine jednu od nekoliko tehnologija koje se mogu koristiti za odabir čvorova u XML dokumentu. Selektori su optimizirani za upotrebu s HTML-om i XML-om, a oblikovani su tako da budu upotrebljivi u kodu koji je važan za performanse [4].

Selektori mogu biti definirani samim elementom:

```
h1 {
  padding-top: 50px;
  padding-bottom: 25px;
  }
```

Isječak programskog kôda 2.3. *Prikaz označavanja CSS selektora samim elementom.*

ili pomoću klasa/identifikatora:

```
table id="showbooking" class="table table-striped table-bordered"
style="width:100%"


#showbooking tbody tr:hover{
  cursor: pointer;
  background-color: #ccc;
}
```

Isječak programskog kôda 2.3.1. *Prikaz označavanja CSS selektora pomoću klasa/identifikatora.*

```
@media print {
    body { font-size: 20px }
} @
@media screen {
    body { font-size: 23px }
}
```

Isječak programskog kôda 2.4. *Selektori upita (engl. query).*

Gornji primjer će na ispisanom papiru prikazati tekst veličine 20px, dok će se na ekranu prikazivati text veličine 23px.

2.3. JavaScript

U samim začecima interneta, ranih 90-tih godina, Web stranice nisu bile dinamičke. Serveri su tada bili znatno slabiji nego danas, kao što je i brzina veze s internetom bila ograničena na sporu komunikaciju preko telefonske linije [5].

No, 1995. godine tvrtka Netscape Communications zapošljava Brendan Eicha s ciljem ugrađivanja programskog jezika Scheme u njihov Web preglednik Netscape Navigator. Prije nego što je mogao započeti s radom, Netscape Communications surađivao je sa Sun Microsystems-om kako bi u Netscape Navigator uključio Sun-ov više statički programski jezik Java. Sve to kako bi se natjecao s Microsoftom za usvajanje internetskih tehnologija i platformi. Netscape Communications tada je odlučio da skriptni jezik koji žele stvoriti nadopunjuje Java i treba imati sličnu sintaksu, što je isključilo prihvaćanje drugih jezika kao što su: Perl, Python, TCL ili Scheme. Kako bi obranio ideju JavaScript-a protiv konkurentskih prijedloga, tvrtka je trebala prototip koji je Eich napisao u 10 dana, u svibnju 1995. [6].

Iako je razvijen pod imenom Mocha, jezik je službeno nazvan LiveScript kada je prvi put pušten u beta izdanju Netscape Navigator-a 2.0 u rujnu 1995.. Međutim, implementacijom u Netscape Navigator 2.0 beta 3 u prosincu 1995., preimenovan je u JavaScript. Konačan izbor imena izazvao je zbunjenost, dajući dojam da je JavaScript proizašao iz Java programskog jezika, a izbor je obilježen kao marketinški trik Netscape-a kako bi dao JavaScript-u pečat onoga što je tada bio, novi, vrući Web programski jezik [6].

Pomoću JavaScript-a, drastično se rasteretilo opterećenje servera. Npr. umjesto da server obrađuje sve informacije Web stranice, dio posla prebaci na klijenta te prepusti da on obradi podatke i prikaže ih u pregledniku [5]:

```
var a = 5;
var b = 7;
var r = a + b;
alert("Zbroj je: "+r)
```

Isječak programskog kôda 2.5. *Zapis varijable u JS i prikaz rezultata unutar 'skočnog prozora' (engl. pop-up window).*

JavaScript također posjeduje i strukturno programiranje kao što su *if* operacija te petlje: *while, do while, switch* i *for*.

```
function alternate(id){
//manipuliranje redovima
      if(i % 2 == 0){
        rows[i].className = "even";
      }else{
        rows[i].className = "odd";
      }
}
```

Isječak programskog kôda 2.6. *Pisanje jednostavne funkcije u JS.*

Gornji primjer koda pregledat će sve elemente (redove) unutar baze podataka te će svakom neparnom obojati pozadinu u sivu boju.

2.4. jQuery

jQuery je knjižnica napisana u JavaScript-u s više platformi dizajniranih tako da pojednostavljuju *„skriptiranje“* HTML-a na strani klijenta. jQuery besplatan je open-source softver koji koristi permisivnu MIT licencu. Web analiza ukazuje da je ona najšire korištena JavaScript knjižnica [7].

Sintaksa jQuery-a dizajnirana je kako bi se lakše kretala kroz dokument, stvorila animacije, obrađivala događaje i razvijala Ajax aplikacije. jQuery također pruža mogućnost razvojnim programerima za stvaranje dodataka na vrhu JavaScript knjižnice, što im omogućuje stvaranje apstrakcija za nisku razinu interakcije i animacije, napredne efekte i visoku razinu tematiziranih dodataka (engl. *widgets*). Modularni pristup knjižnici, jQuery omogućuje stvaranje snažnih dinamičkih Web stranica i Web aplikacija [7].

Ako se želi odabrati element s identifikatorom ‘*identifikator*’ i klasom '*klasa*', JavaScript-om bi se to postiglo s:

```
document.getElementById('identifikator');
document.getElementByClassName('klasa');
```

Isječak programskog kôda 2.7. *Odabiranje elemenata po klasi i identifikatoru uz pomoć JS.*

Isti zadatak se znatno lakše postiže uz pomoć jQuery-a:

```
$('#identifikator');
$('.klasa');
```

Isječak programskog kôda 2.8. *Odabiranje elemenata po klasi i identifikatoru uz pomoć jQuery-a.*

2.5. Google Firebase

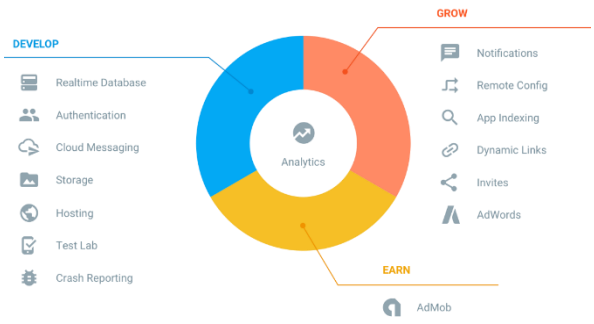
Firebase je platforma za razvoj mobilnih i Web aplikacija koju je razvila tvrtka Firebase, Inc. 2011. godine.

Firebase razvijen je od strane Envolve-a, startup-a koji su prethodno pokrenuli James Tamplin i Andrew Lee 2011. godine. Envolve je programerima pružio API koji omogućuje integraciju funkcije za chat na njihove Web stranice. Nakon objavljivanja chat usluge, Tamplin i Lee utvrdili su da je usluga korištena za prijenos podataka koji nisu bili chat poruke. Razvojni programeri upotrebljavali su Envolve za sinkronizaciju aplikacijskih podataka u stvarnom vremenu. Nakon toga, Tamplin i Lee odlučili su odvojiti sustav za chat i arhitekturu u stvarnom vremenu koja je to omogućila, te su osnovali Firebase kao zasebnu tvrtku u travnju 2012. godine [8].

U listopadu 2014. Firebase je preuzeo Google. Nakon preuzimanja, Firebase je rastao unutar Google-a i proširio svoje usluge kako bi postao jedinstvena platforma za mobilne razvojne programere. Firebase baza sada je integrirana s raznim Google-ovim uslugama kako bi ponudila širi opseg proizvoda za razvojne programere [8].

Neki od servisa koje platforma nudi su:

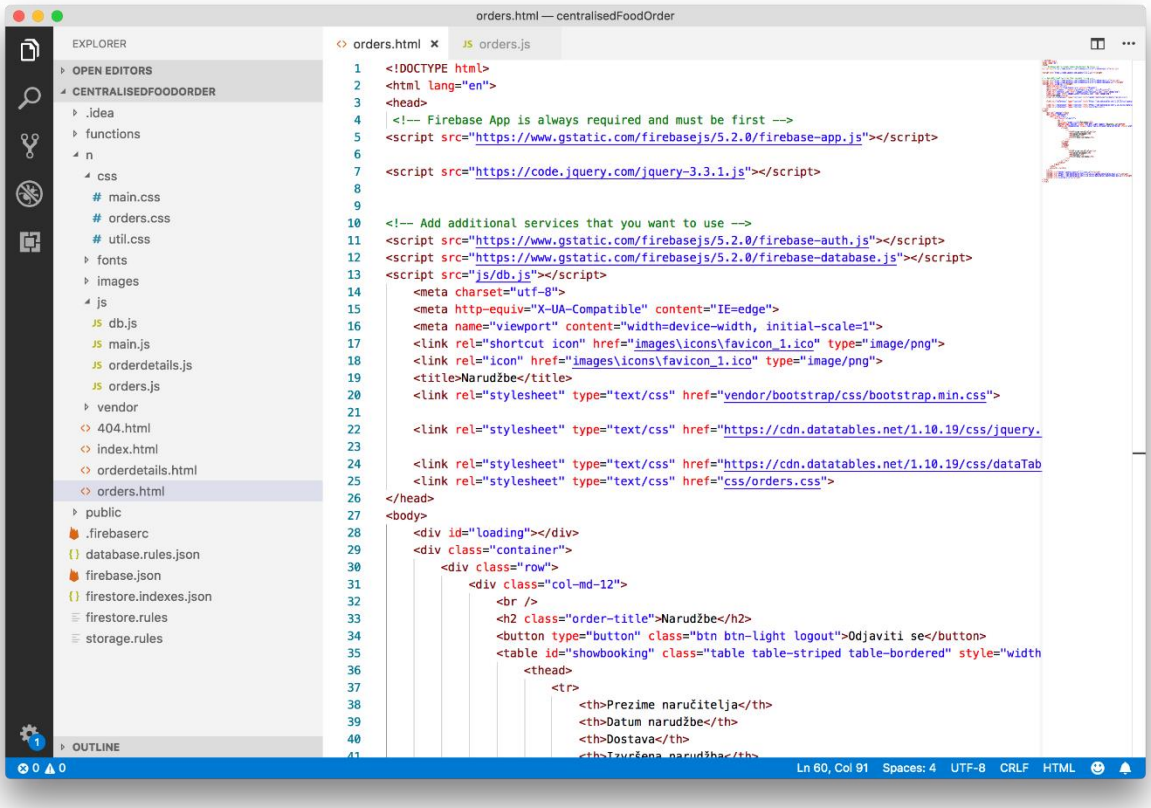
- Analitika (engl. *Analytics*)
- Poruke u oblacima (engl. *Cloud Messaging*)
- Autentifikacija (engl. *Authentication*)
- Baza podataka u stvarnom vremenu (engl. *Real time Database*)
- Skladištenje (engl. *Storage*)
- Usluge poslužitelja (engl. *Hosting*)
- Obavijesti (engl. *Notifications*)



Sl. 2.1. *Servisi Google Firebase platforme.*

2.6. Visual Studio Code

Visual Studio Code program je (engl. *editor*) za pisanje programskog koda, a razvijen je od strane Microsoft-a za sustave: Windows, Linux i MacOS. VS Code ima podršku za uklanjanje pogrešaka, ugrađenu kontrolu za Git, refaktoriranje koda i još mnoge druge značajke. Besplatno je i open-source. VS Code podržava različite programske i skriptne jezike. Neki od podržanih jezika su: C, C++, C#, HTML, CSS, Java, JavaScript, PHP, Python i mnogi drugi [9]. Slika 2.2. prikazuje izgled programa VS Code.



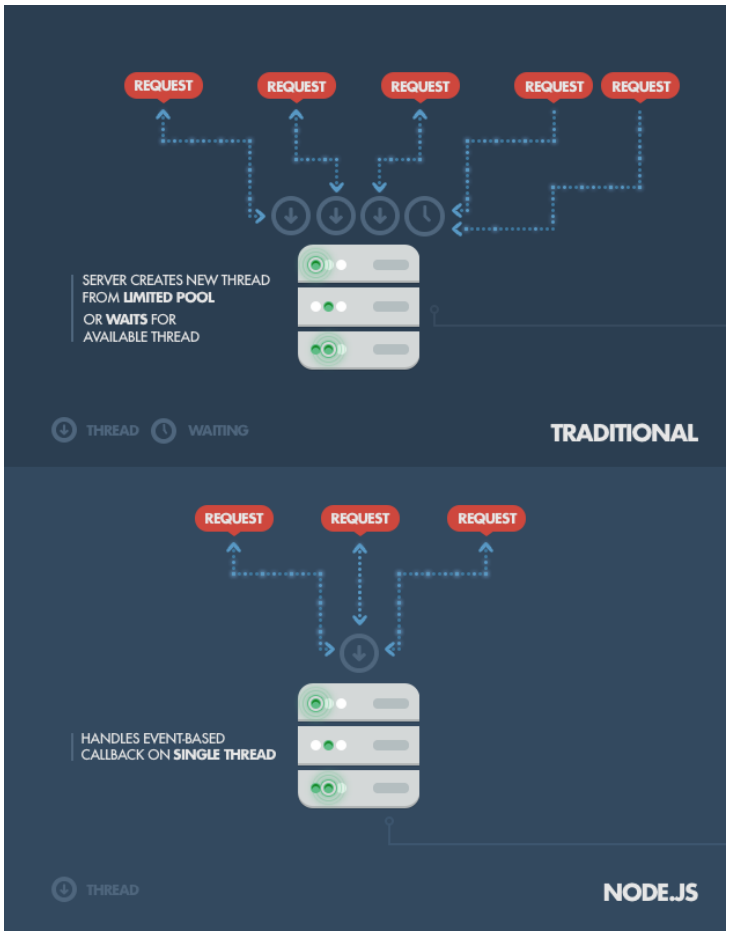
Sl. 2.2. *Snimka ekrana programa Visual Studio Code.*

2.7. Node.js

Node.js razvio je Ryan Dahl zajedno sa svojim kolegama 2009. godine, otprilike trinaest godina nakon uvođenja prvog JavaScript okruženja na poslužitelju, Netscape LiveWire Pro Web. Prva verzija podržavala je samo Linux i Mac OS. Njegov razvoj i održavanje vodio je Dahl [10].

Node.js je open-source, višeplatformsko (engl. cross-platform) JavaScript okruženje koje izvršava JavaScript kod izvan preglednika. Iako je *.js* proširenje za JavaScript, naziv "Node.js" ne odnosi se na određenu datoteku u tom kontekstu, nego je samo naziv proizvoda. Node.js ima arhitekturu upravljanu događajima sposobnom za asinkroni I/O. Ovi izbori imaju za cilj optimizirati propusnost i skalabilnost u Web aplikacijama s mnogim ulazno/izlaznim operacijama, kao i za Web aplikacije u stvarnom vremenu, npr. programi za komunikaciju u stvarnom vremenu [10].

Neki od korisnika Node.js softvera su: GoDaddy, Groupon, IBM, LinkedIn, Microsoft, Netflix, PayPal, Tuenti, Voxer, Walmart, Yahoo! i slični.



Sl. 2.3. *Node.js način rada* [11].

3. RAZVOJ APLIKACIJE

U ovom se poglavlju iznosi analiza problematike zadatka, struktura zadatka te proces izrade aplikacije od postavljanja razvojnog okruženja na početku, pa do pisanja koda na kraju.

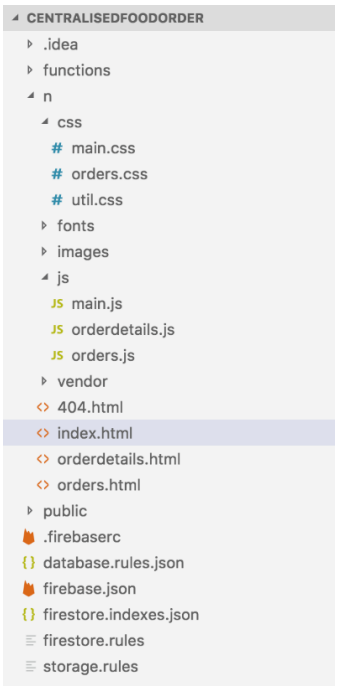
3.1. Analiza problematike

Zadatak je napraviti Web aplikaciju koja će restoranima omogućiti uvid u narudžbe koje su korisnici poslali putem Android klijentske aplikacije za centralno naručivanje hrane na području jednoga grada.

3.2. Struktura projekta

Rješenje problematike ovog završnog rada ostvareno je JavaScrip programskim jezikom. Korišten je Visual Studio Code kao razvojno okruženje za Web aplikaciju. Projekt se sastoji od sljedećih elemenata:

- *n/* : sadrži index.html te ostale .html datoteke
- *css/* : sadrži kodove za stiliziranje
- *fonts/* : sadrži fontove
- *images/* : sadrži ikonu aplikacije i loading.gif
- *js/* : sadrži kodove u JavaScript-u
- *vendor/* : sadrži knjižnicu
- *public/* : sadrži datoteku 404.html



Sl. 3.1. *Sadržaj projekta.*

3.3. Izrada aplikacije

Kako bi pisanje koda moglo biti započeto, prvo se mora postaviti razvojno okruženje na računalu. Program korišten za pisanja koda zove se *Visual Studio Code* kojeg smo ranije opisali i može se preuzeti na stranici: https://code.visualstudio.com/download/.

Nakon što je VS Code preuzet i instaliran potrebno je preuzeti i instalirati *Node.js* sa stranice https://nodejs.org/en/download/.

Nakon što je sve gore od navedenog napravljeno, prije pisanja programskog koda potrebno je instalirati Firebase CLI pomoću npm (engl. *Node Package Manager*) pokretanjem sljedeće naredbe:

```
npm install -g firebase-tools
```

Isječak programskog kôda 3.1. *Naredba za instalaciju Firebase CLI.*

Nakon instalacije Firebase CLI, potrebno je prijaviti se pomoću Google računa:

```
firebase login
```

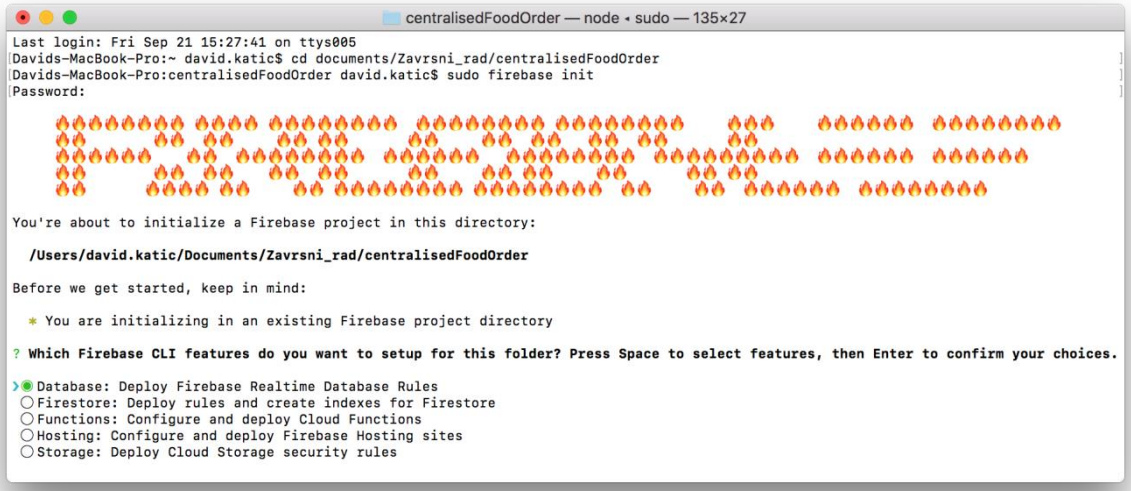
Isječak programskog kôda 3.2. *Naredba za prijavu na Google račun.*

Ova naredba povezuje računalo s Firebase konzolom i omogućuje pristup projektu i bazi podataka.

Kako bi novi Firebase projekt bio inicijaliziran, pokrenuta je sljedeća naredba unutar direktorija Web aplikacije (Sl.3.2.):

```
firebase init
```

Isječak programskog kôda 3.3. *Naredba za inicijalizaciju Firebase projekta.*



Sl. 3.2. *Izgled naredbenog retka s upisanom naredbom.*

Nakon postavljanja razvojnog okruženja, napisan je kod Web aplikacije. Prije svega bilo je potrebno inicijalizirati Firebase bazu podataka unutar aplikacije, što je napravljeno pomoću sljedećeg koda:

```
var config = {
  apiKey: "AIzaSyCqnBvwcMuKHoq7nnGHf_PrfS_jR8yQhjU",
  authDomain: "centralised-food-order-1.firebaseapp.com",
  databaseURL: "https://centralised-food-order-1.firebaseio.com",
  projectId: "centralised-food-order-1",
  storageBucket: "centralised-food-order-1.appspot.com",
  messagingSenderId: "909546652929"
};
firebase.initializeApp(config);
```

Isječak programskog kôda 3.4. *Inicijalizacija Firebase baze podataka unutar aplikacije.*

Kod za inicijalizaciju baze podataka sačuvan je u posebnoj datoteci *db.js* koja se kasnije poziva unutar ostalih datoteka pri vrhu pomoću:

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

Isječak programskog kôda 3.5. *Poziv na inicijalizaciju Firebase baze podataka unutar ostalih datoteka.*

Primjer koda za prijavu u aplikaciju vidljiv je u isječku programskog kôda 3.6. gdje osoba zadužena za zaprimanje narudžbi za određeni restoran treba unijeti e-mail i lozinku koju je dobio od administratora sustava:

```
<form class="login100-form validate-form">
	<span class="login100-form-title p-b-26">
	Dobrodošli
	</span>
<span class="login100-form-title p-b-48">
	<i class="zmdi zmdi-accounts"></i>
</span>
<div class="wrap-input100 validate-input" data-validate = "Važeća e-adresa
je: ime_restorana@test.com">
	<input id="loginEmail" class="input100" type="text" name="email"
onkeypress="return runScript(event)">
	<span class="focus-input100" data-placeholder="Email"></span>
</div>
<div class="wrap-input100 validate-input" data-validate="Upišite svoju
lozinku">
	<span class="btn-show-pass">
<i class="zmdi zmdi-eye"></i>
	</span>
	<input id="loginPassword" class="input100" type="password"
name="pass" onkeypress="return runScript(event)">
	<span class="focus-input100" data-placeholder="Lozinka"></span>
</div>
<div class="container-login100-form-btn">
	<div class="wrap-login100-form-btn">
	<div class="login100-form-bgbtn"></div>
	<div id='login' class="login100-form-submit">Prijaviti se</div>
	</div>
</div>
</form>
```

Isječak programskog kôda 3.6. *Forma prijave restorana u sustav putem e-mail-a i lozinke.*

Nakon što je poslan upit, vrijednosti iz <*input*> elementa, e-mail i lozinka šalju se u datoteku *main.js* gdje se upitom uspoređuju s podacima unutar baze podataka. Ako se e-mail i lozinka podudaraju s jednim od zapisa u bazi podataka, omogućava se pristup te se osoba zadužena za zaprimanje narudžbi unutar prijavljenog restorana šalje na popis narudžbi.

```
var input = $('.validate-input .input100');

$('#login').on('click', function(){
      var check = true;

for(var i=0; i<input.length; i++) {
      if(validate(input[i]) == false){
      showValidate(input[i]);
      check=false;
      }
}
if ( check ) {
      $('#loading').show();
      var  email=$('#loginEmail').val();
      var password=$('#loginPassword').val();
      firebase.auth().signInWithEmailAndPassword(email, password)
      .then(function(authData) {
      var userId = firebase.auth().currentUser.uid;
      localStorage.setItem( 'userId', userId );
      $('#loading').hide();
      window.location.href = "/orders.html";
})
.catch(function(error) {
      // Handle Errors here.
      $('#loading').hide();
      var delayTime = 500;
      setTimeout(function() {
      var errorCode = error.code;
      var errorMessage = error.message;
      if (errorCode === 'auth/wrong-password') {
      alert('Wrong password.');
      } else {
      alert(errorMessage);
      }
      console.log(error);
      }, delayTime);
      });
      }
      else
      return check;
});
```

Isječak programskog kôda 3.7. *Provjera zapisa traženog računa te greške prilikom prijave u sustav.*

Primjer koda za kreiranje tablice *narudžbi* vidljiv je u isječku programskog kôda 3.8. gdje se nalaze sljedeći podatci:

- Prezime naručitelja – Prezime korisnika (naručitelja).
- Datum narudžbe – Datum i vrijeme kad je narudžba zaprimljena.
- Dostava – Pokazuje hoće li se narudžba dostavljati.
- Izvršena narudžba – Pokazuje je li narudžbu izvršili ili ne.

```
<table id="showbooking" class="table table-striped table-bordered"
style="width:100%">
    <thead>
        <tr>
            <th>Prezime naručitelja</th>
            <th>Datum narudžbe</th>
            <th>Dostava</th>
            <th>Izvršena narudžba</th>
        </tr>
    </thead>
    <tbody>
    </tbody>
    <tfoot>
        <tr>
            <th>Prezime naručitelja</th>
            <th>Datum narudžbe</th>
            <th>Dostava</th>
            <th>Izvršena narudžba</th>
        </tr>
    </tfoot>
</table>
```

Isječak programskog kôda 3.8. *Izgled tablice narudžbi.*

Kod za kreiranje tablice *detalji narudžbi* vidljiv je u isječku programskog kôda 3.9. gdje su sljedeći podatci.

U slučaju kada se narudžba dostavlja vidljiva je dodatna tablica naziva *Adresa dostave* sa podatcima: *prezime, ulica, kućni broj, grad, kat, broj stana i broj telefona*.

Tablica *Naručene stavke* koja se prikazuje neovisno o tome dali se narudžba dostavlja ili ne sadrži podatke: *vrsta jela, naziv jela, sastojci, količina i cijena.*

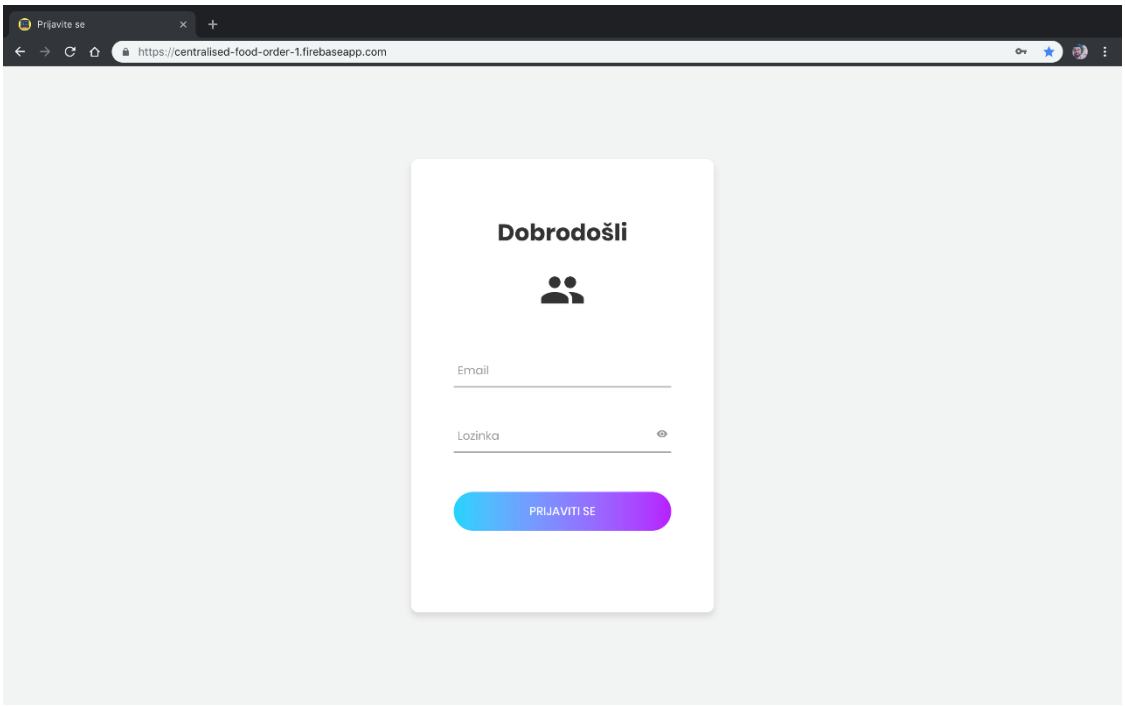
```
<div id="section-to-print" class="address">
    <br />
    <h2>Adresa dostave</h2>
    <table id="address" class="table table-striped table-bordered"
style="width:100%">
        <thead>
            <tr>
                <th>Prezime</th>
                <th>Ulica</th>
                <th>Kućni broj</th>
                <th>Grad</th>
                <th>Kat</th>
                <th>Broj stana</th>
                <th>Broj telefona</th>
            </tr>
        </thead>
        <tbody>
        </tbody>
    </table>
</div>
<div id="section-to-print" class="orderItems">
    <br />
    <h2>Naručene stavke</h2>
    <table id="orderItems" class="table table-striped table-bordered"
style="width:100%">
        <thead>
            <tr>
                <th>Vrsta jela</th>
                <th>Naziv jela</th>
                <th>Sastojci</th>
                <th>Količina</th>
                <th>Cijena</th>
            </tr>
        </thead>
        <tbody>
        </tbody>
    </table>
</div>
```

Isječak programskog kôda 3.9. *Izgled tablice detalji narudžbe.*

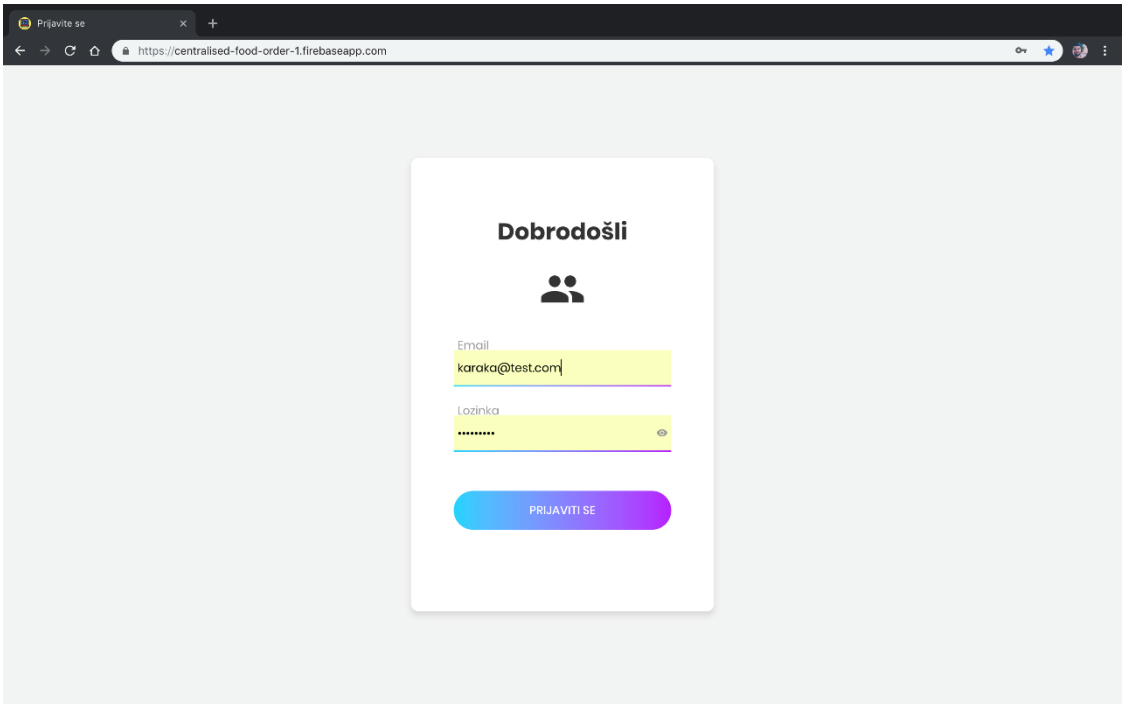
4. KORIŠTENJE APLIKACIJE

Kako bi restoran mogao koristiti Web aplikaciju, od administratora sustava treba prethodno zatražiti upis svog restorana na servis za centralizirano naručivanje hrane. Jednom kad se upiše, odgovorna osoba unutar restorana dobiva potrebne podatke za prijavu na poslužiteljsku aplikaciju.

Otvaranjem aplikacije prikazuje se početni zaslon za prijavu u sustav koji se sastoji od tri dijela: polje za upis e-mail adrese, polje za upis lozinke te gumb za prijavu u sustav. Slika 4.1. prikazuje izgled početnog zaslona.

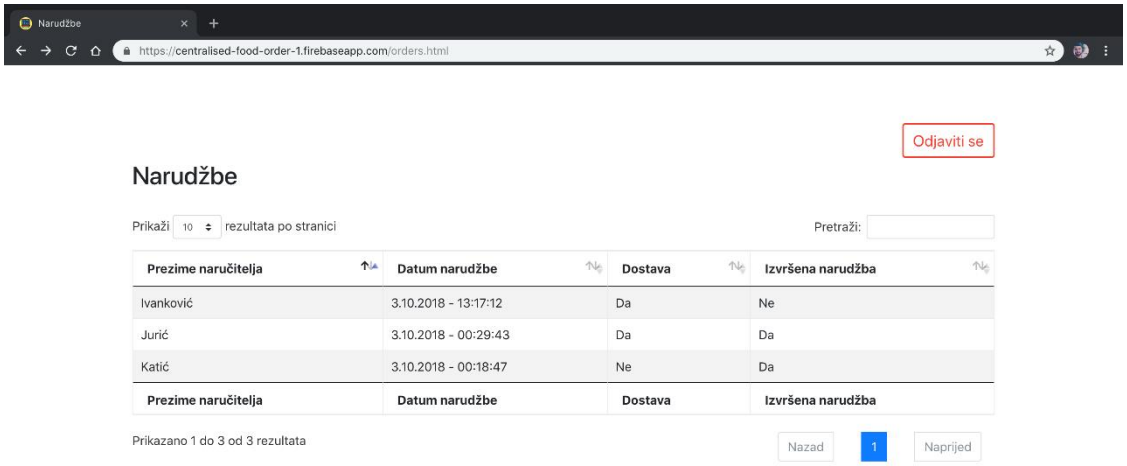


Sl. 4.1. *Izgled početnog zaslona za prijavu u sustav.*



Sl. 4.2. *Izgled početnog zaslona za prijavu u sustav s upisanim potrebnim podatcima.*

Nakon što je korisnik upisao potrebne podatke na za to predviđena mjesta, klikom na gumb „Prijaviti se“ pokreće se provjera upisanih podataka prikazana u isječku programskog kôda 3.4. Ako su upisani podatci ispravni, korisnika se šalje na zaslon s ispisom narudžbi. Slika 4.3. prikazuje zaslon s ispisom narudžbi.



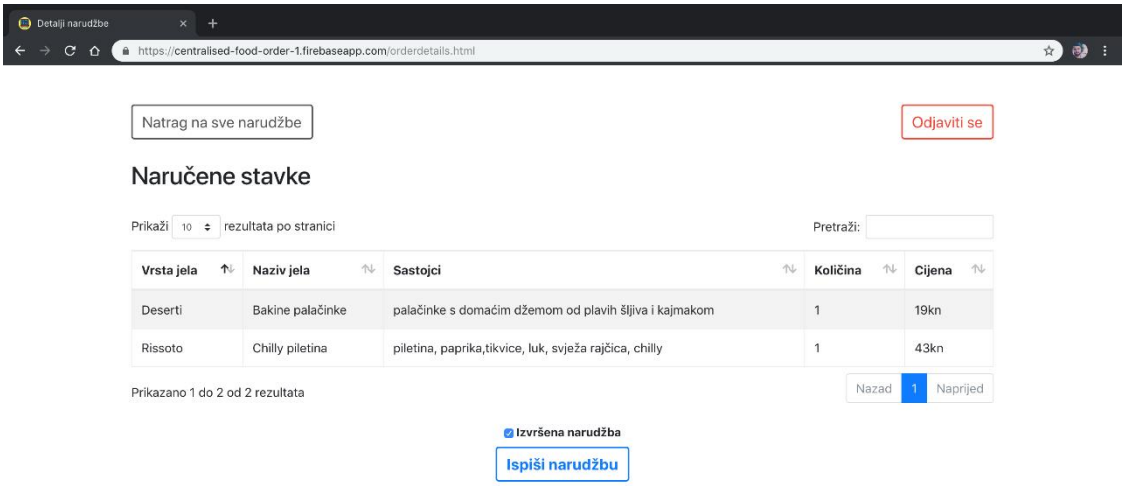
Sl. 4.3. *Izgled zaslona s ispisom narudžbi.*

Nakon uspješne prijave korisnik vidi sve narudžbe za restoran s čijim se podatcima prijavio. Svaka narudžba sadrži tri podatka: prezime naručitelja, datum narudžbe te podatak dostavlja li se narudžba ili će ju naručitelj sam pokupiti u restoranu.

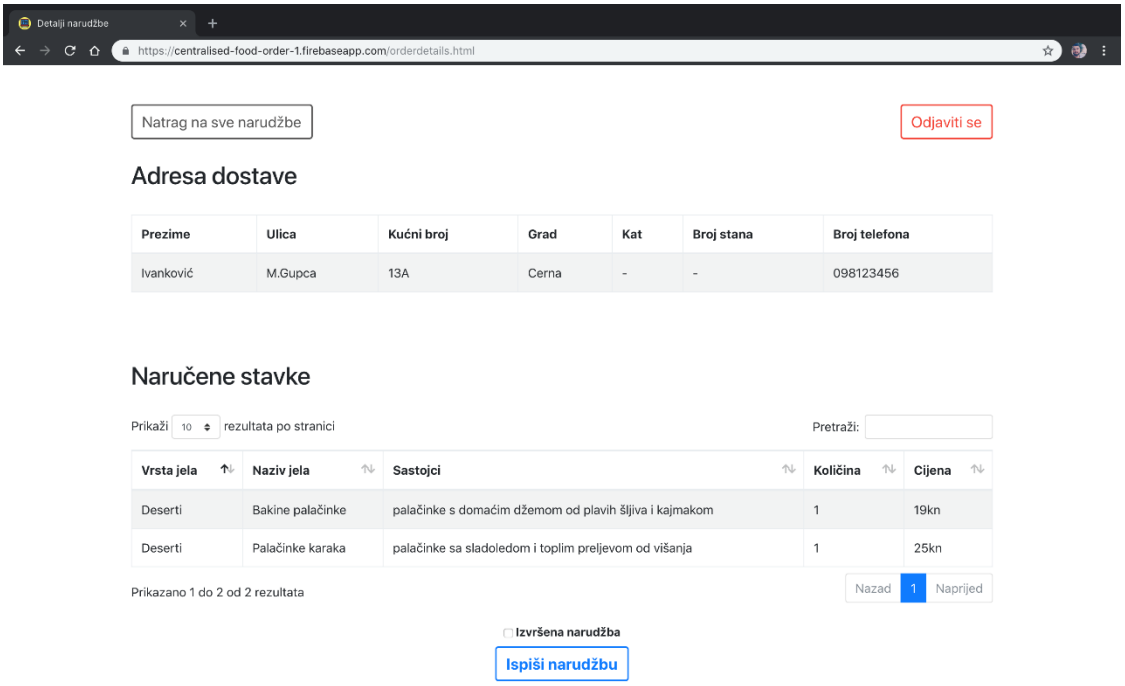
Na ovom zaslonu postoji i mogućnost odabira prikaza količine narudžbi po stranici te polje za pretraživanje ako tražimo točno određeni podatak iz narudžbe.

Klikom na pojedinu narudžbu otvara se novi zaslon s detaljima narudžbe. Zaslon *detalji narudžbe* sadržava sljedeće podatke: adresa dostave (prezime, ulica, kućni broj, grad, kat, broj stana i broj telefona) u slučaju da se narudžba dostavlja na naručiteljevu adresu (Sl.4.5.), te stavke koje su naručene (vrsta jela, naziv jela, sastojci, količina i cijena).

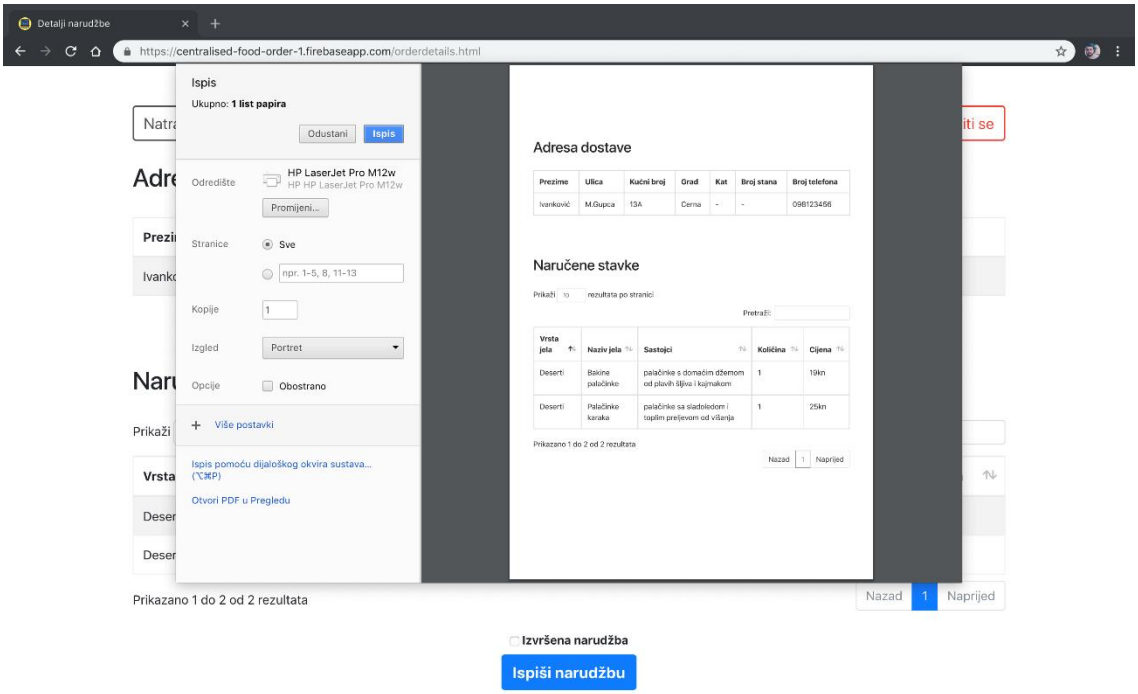
Također i na ovom zaslonu postoji mogućnost pretraživanja za točno određeni podatak kao i broj prikaza stavki po stranici. Ovaj zaslon dodatno ima gumb za ispis detalja narudžbe (Sl.4.6.).



Sl. 4.4. *Izgled zaslona s detaljima narudžbe bez dostave.*



Sl. 4.5. *Izgled zaslona s detaljima narudžbe s dostavom.*



Sl. 4.6. *Izgled dijaloškog okvira za ispis narudžbe.*

5. ZAKLJUČAK

U sklopu ovog završnog rada prikazan je proces izrade i način rada usluge za centralizirano naručivanje hrane na primjeru grada Osijeka. Naime, rad se veže na temu rada „*Android klijentska aplikacija za centralizirano naručivanje hrane u Osijeku"*[12] te je ovaj rad kompletirao jednu cjelinu.

Kao što je već spomenuto u samom zadatku rada, cilj je bio napraviti Web aplikaciju za prihvaćanje narudžbi na poslužitelju, koje su poslane od strane Android klijentske aplikacije. Kao rezultat toga, ugostiteljski objekti imaju jednostavnu i preglednu aplikaciju s kojom mogu pregledavati narudžbe poslane od strane njihovih klijenata. Implementiran je pretraživač kao i broj prikaza narudžbi po stranici, a sve kako bi se korisnik aplikacije (ugostiteljski objekt) lakše snalazio.

Za brz odaziv aplikacije zaslužan je jednostavan kod te činjenica da je većina odrađena JavaScript-om i Firebase bazom podataka koja radi u realnom vremenu. Jedina prepreka koja bi mogla usporiti rad aplikacije velika je količina podataka, tj. s vremenom će količina narudžbi unutar baze podataka porasti kako aplikaciju bude koristilo sve više korisnika. U tom slučaju, mogla bi se napraviti provjera narudžbi starijih od određenog vremenskog roka koje bi automatski bile obrisane iz baze podataka.

Aplikacija je napravljena tako da ima još prostora za nadogradnju i implementiranje novih elemenata. U narednim nadogradnjama aplikacije bit će implementiran *chat* sustav, tj. sustav razmjenjivanja poruka između korisnika i ugostiteljskog objekta, a sve radi povećavanja kvalitete usluge koju restorani pružaju.

LITERATURA

[1] W3C – HTML & CSS, What is HTML?,
URL: https://www.w3.org/standards/webdesign/htmlcss
(pristupljeno: 23.9.2018.)

[2] Oblak znanja – HTML oznake,
URL: http://www.oblakznanja.com/2013/04/html-dokument-i-osnove-html-jezika
(pristupljeno: 23.9.2018.)

[3] W3C – HTML & CSS, What is CSS?,
URL: https://www.w3.org/standards/webdesign/htmlcss
(pristupljeno: 23.9.2018.)

[4] W3C – Selector Level 3, Abstract,
URL: https://www.w3.org/TR/selectors-3
(pristupljeno: 23.9.2018.)

[5] Matijašević, D. (2016) WEB APLIKACIJA ZA REZERVACIJU PACIJENATA. Diplomski rad. Osijek: Fakultet elektrotehnike, računarstva i informacijskih tehnologija.

[6] Wikipedia – JavaScript, History - Beginnings at Netscape,
URL: https://en.wikipedia.org/wiki/JavaScript#History
(pristupljeno: 23.9.2018.)

[7] Wikipedia – jQuery,
URL: https://en.wikipedia.org/wiki/JQuery
(pristupljeno: 23.9.2018.)

[8] Wikipedia – Firebase, History,
URL: https://en.wikipedia.org/wiki/Firebase
(pristupljeno: 23.9.2018.)

[9] Wikipedia - Visual Studio Code, Features,
URL: https://en.wikipedia.org/wiki/Visual_Studio_Code
(pristupljeno: 23.9.2018.)

[10] Wikipedia – Node.js, History,
URL: https://en.wikipedia.org/wiki/Node.js
(pristupljeno: 23.9.2018.)

[11] Medium – Node.js, How Does It Work?,
URL: https://medium.com/the-node-js-collection/why-the-hell-would-you-use-node-js-4b053b94ab8e
(pristupljeno: 23.9.2018.)

[12] Katić, V. (2016) ANDROID KLIJENTSKA APLIKACIJA ZA CENTRALIZIRANO NARUČIVANJE HRANE U OSIJEKU. Završni rad. Osijek: Fakultet elektrotehnike, računarstva i informacijskih tehnologija.

SAŽETAK

Naslov: Poslužiteljska web aplikacija za centralizirano naručivanje hrane u Osijeku

U ovome radu prikazana je izrada Web aplikacije po koracima koja omogućuje restoranima prijavu u poslužiteljski sustav za centralizirano naručivanje hrane na primjeru jednoga grada. Opisane su odabrane tehnologije i alati koji su korišteni tijekom realizacije projekta. Opisana je izrada aplikacije i sam tijek izrade. Aplikacija je napravljena tako da bude jednostavna za korištenje kao i za kasnije dorade i implementacije novih elemenata. Prijavom putem korisničkog računa restoran ima uvid u narudžbe svojih klijenata te može pregledati detalje narudžbe. Unutar detalja narudžbe restoran ima sve potrebne informacije o korisniku i samoj narudžbi kako bi istu mogao izvršiti. Na kraju je ukratko opisan način korištenja aplikacije. Za razvoj aplikacije, korišten je JavaScript programski jezik te Google Firebase baza podataka.

Ključne riječi: baza podataka, javascript, naručivanje hrane, web aplikacija, web tehnologije

ABSTRACT

Title: Web server application for centralized ordering food in Osijek

This paper presents a step-by-step development of a Web application that allows restaurants access in the centralized ordering system server of a particular city. Selected technologies and tools used in project implementation are described. Application development and its development cycle are also described. The application was designed to be simple for as well as for later implementation of new elements. Signing in with the user account restaurant has overview of its client's orders and can view details of the order. In order details restaurant has all the necessary information about the user and the order itself. In the end, how to use the application is described. To develop this application, JavaScript program language and Google Firebase database was used.

Keywords: databases, javascript, ordering food, web application, web technology

ŽIVOTOPIS

David Katić rođen je 20. lipnja 1993. godine u Vinkovcima. 2000. godine započinje svoje školovanje u Osnovnoj školi Matije Antun Reljković u Cerni. Nakon završetka osnovne škole, 2008. godine upisuje Tehničku školu Ruđera Boškovića u Vinkovcima, smjer Elektrotehničar. Položenom državnom maturom, 2012. godine upisuje Fakultet elektrotehnike, računarstva i informacijskih tehnologija u Osijeku, preddiplomski stručni studij elektrotehnike, smjer informatika. Aktivan je u više udruga za promicanje kulture i očuvanje tradicije. Trenutno studira i paralelno radi u vlastitom obrtu na poziciji Web programera.

_____

(potpis studenta)

PRILOZI

Svi prilozi se nalaze na CD-u u datoteci PRILOZI.

Kompletan kôd aplikacije se može pronaći u datoteci PRILOZI / PROGRAMSKI KOD.