



**UNIVERZITET U ZENICI**  
**POLITEHNIČKI FAKULTET U ZENICI**  
**SOFTVERSKO INŽENJERSTVO**  
Napredne tehnike programiranja



**DIPLOMSKI RAD**  
**RAZVOJ WEB APLIKACIJE ZA NARUČIVANJE HRANE**

**Mentor:**

van. prof. dr. Nevzudin Buzadžija

**Student:**

Adis Hodžić

## **BIBLIOGRAFSKA KARTICA RADA**

**NAUČNO PODRUČJE RADA:** Tehničke nauke

**NAUČNO POLJE RADA:** Softversko inženjerstvo

**NAUČNA GRANA RADA:** Napredne tehnike programiranja

**USTANOVA U KOJOJ JE IZRAĐEN RAD:** Univerzitet u Zenici, Politehnički fakultet

**MENTOR RADA:** Van. prof. dr. Nevzudin Buzadžija

**DATUM ODBRANE RADA:**

**ČLANOVI KOMISIJE ZA ODBRANU:**

Prof. dr. soc. Samir Lemeš

V.prof.dr. Sabahudin Jašarević

## **IZJAVA**

Izjavljujem da sam diplomski rad pod naslovom „Razvoj web aplikacije za naručivanje hrane“ korištenjem React.js, Node.js i MongoDB alata izradio samostalno pod mentorstvom Van. prof. dr. Nevzudina Buzadije. U radu sam koristio literaturu koja je navedena na kraju diplomskog rada. Tuđe spoznaje, zaključke, stavove, teorije i zakonitosti koje sam izravno ili parafrizirajući naveo u diplomskom radu na uobičajen, standardan način sam povezao sa fusnotama i korištenim bibliografskim jedinicama.

## SAŽETAK

Web aplikacije su ključni dio digitalnog svijeta i imaju široku primjenu u svakodnevnom životu. S obzirom na brz tehnološki napredak, web aplikacije se stalno razvijaju i prilagođavaju kako bi odgovarale potrebama korisnika. Ovaj projekat je zapravo web aplikacija za naručivanje hrane, koja je kreirana u React.js i Node.js tehnologijama, korištenjem svih raspoloživih biblioteka. Za glavno razvojno okruženje odabran je Visual Studio Code, a za bazu podataka korištena je MongoDB baza u okviru MongoDB Compass. Za manipulaciju API-jima, korišten je alat Postman, sa svojim širokim mogućnostima.

**Ključne riječi:** narudžba, proizvod, korisnik, web aplikacija, React.js, Node.js, MongoDB, VS Code

## ABSTRACT

Web applications are a crucial part of the digital world and have broad applications in everyday life. With the rapid technological advancements, web applications are constantly evolving and adapting to meet user needs. This project is actually a food ordering web application created using React.js and Node.js technologies, utilizing all available libraries. Visual Studio Code was selected as the primary development environment, and MongoDB Compass was used for the database, employing a MongoDB database. Postman was used for API manipulation, leveraging its extensive capabilities.

**Keywords:** order, product, user, web application, React.js, Node.js, MongoDB, VS Code

# **SADRŽAJ**

<b>1. UVOD</b>	<b>5</b>
<b>2. POSTOJEĆE APLIKACIJE ZA DOSTAVU HRANE</b>	<b>6</b>
2.1. Korpa.ba	6
2.2. Glovo.com	7
<b>3. NEDOSTACI APLIKACIJA ZA DOSTAVU HRANE</b>	<b>9</b>
<b>4. PRIJEDLOG POBOLJŠANJA APLIKACIJA</b>	<b>9</b>
<b>5. SAVREMENI PRISTUP IZRADA APLIKACIJA</b>	<b>10</b>
<b>6. RAZVOJNO OKRUŽENJE</b>	<b>11</b>
6.1. Visual Studio Code	11
6.2. MongoDB Compass	14
6.3. Postman	15
<b>7. SOFTWARE ZA RAZVOJ WEB APLIKACIJE</b>	<b>16</b>
7.1. ReactJS	17
7.2. NodeJS	18
7.3. MongoDB	20
<b>8. IDEJNI PRIJEDLOG APLIKACIJE</b>	<b>21</b>
8.1. Analiza zahtjeva	21
8.2. Analiza i dizajn aplikacije	22
8.3. Use Case Dijagram	22
8.4. Klasni dijagram	23
<b>9. IZRADA APLIKACIJE - FRONTEND I BACKEND</b>	<b>24</b>
9.1. Login i Register page	25
9.2. Search Results i Filter page	28
9.3. Product Details page	30
<b>10. ADMIN DASHBOARD</b>	<b>32</b>
<b>11. TESTIRANJE APLIKACIJE</b>	<b>38</b>
<b>12. ODRŽIVOST APLIKACIJE</b>	<b>39</b>
<b>13. ZAKLJUČAK</b>	<b>40</b>
<b>14. LITERATURA</b>	<b>41</b>

## 1. UVOD

Zbog digitalne transformacije i rasta online usluga, tehnološki napredak promijenio je način na koji konzumiramo hranu. Aplikacije za dostavu hrane postale su integralni dio naše svakodnevice, pružajući korisnicima mogućnost da jednostavno i brzo naruče hranu iz omiljenih restorana, bilo da se nalaze kod kuće, na poslu ili u pokretu. Ovaj rad istražuje i analizira savremeni pristup izrade aplikacije za dostavu hrane, fokusirajući se na implementaciju tehnologije poput React-a i Node-a.

U ovom kontekstu, ključni aspekti istraživanja obuhvataju analizu postojećih aplikacija za restorane, identifikaciju nedostataka koje takve aplikacije mogu imati te prijedlogu poboljšanja kako bi se unaprijedilo korisničko iskustvo i efikasnost iste platforme. Pored toga, rad će ponuditi idejni prijedlog aplikacije za dostavu hrane, uz korištenje dijagrama Use Case i klasnog dijagrama kako bi se vizualizirao proces razvoja aplikacija.

Kroz detaljnu analizu procesa izrade aplikacija, fokusirajući se na korake implementacije tehnoloških rješenja poput React-a za frontend i Node-a za backend razvoj, ovaj rad će pružiti uvid u ključne faze izrade aplikacija za dostavu hrane. Na kraju, zaključak će sumirati glavne analize istraživanja i ponuditi smjernice za buduća istraživanja i razvoj u ovoj oblasti.

Kroz kombinaciju teorijskog okvira, analize postojećih rješenja i praktične implementacije, ovaj rad ima za cilj doprinijeti razumijevanju procesa razvoja aplikacije dostave hrane i njihovog potencijala za poboljšanje korisničkih iskustava u oblasti online narudžbe hrane.

## 2. POSTOJEĆE APLIKACIJE ZA DOSTAVU HRANE

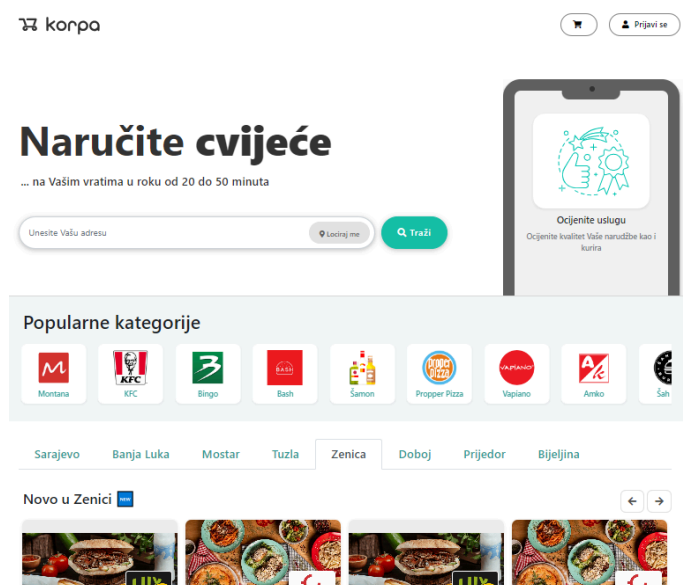
Postoje različite aplikacije za dostavu hrane širom svijeta, uključujući i različite regione. Korpa.ba je popularna platforma za dostavu hrane u Bosni i Hercegovini, dok je Glovo globalna platforma koja se koristi u više zemalja.

Pored Korpa.ba i Glovo-a, postoje i druge poznate aplikacije za dostavu hrane kao što su Uber Eats, Deliveroo, DoorDash, Just Eat, Grubhub, i mnoge druge. Ove aplikacije često nude raznovrsne restorane i opcije za dostavu hrane korisnicima putem mobilnih aplikacija ili web stranica.

Svaka od ovih aplikacija obično ima svoje karakteristike, poput raspona restorana koje podržavaju, geografske dostupnosti, tarifa za dostavu i druge usluge koje nude. Korisnici biraju aplikaciju na osnovu preferencija, dostupnosti restorana ili posebnih ponuda koje nude.

### 2.1. Korpa.ba

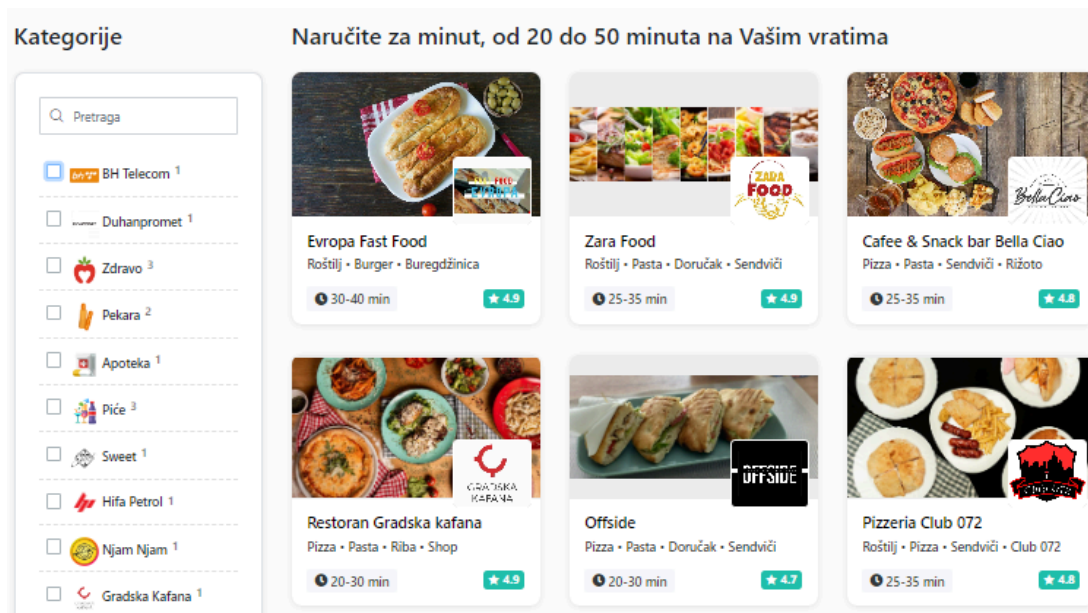
Korpa.ba je popularna platforma za dostavu hrane koja posluje u Bosni i Hercegovini. Ova platforma omogućava korisnicima da naruče hranu iz različitih restorana i prodavnica hrane putem mobilne aplikacije ili web stranice. Korpa.ba obično ima širok izbor restorana i raznovrsnu hranu, od lokalnih specijaliteta do internacionalnih kuhinja.



Slika 1. Izgled web aplikacije Korpa.ba<sup>1</sup>

<sup>1</sup> <https://korpa.ba/>

Korisnici mogu pregledavati menije restorana, birati željenu hranu, dodavati je u korpu i zatim izvršiti narudžbu putem aplikacije. Osim hrane, Korpa.ba često nudi i opcije za dostavu različitih vrsta pića, slatkiša ili namirnica iz prodavnica i drugih artikala.



Slika 2. Kategorije i ponuda (Korpa.ba)<sup>2</sup>

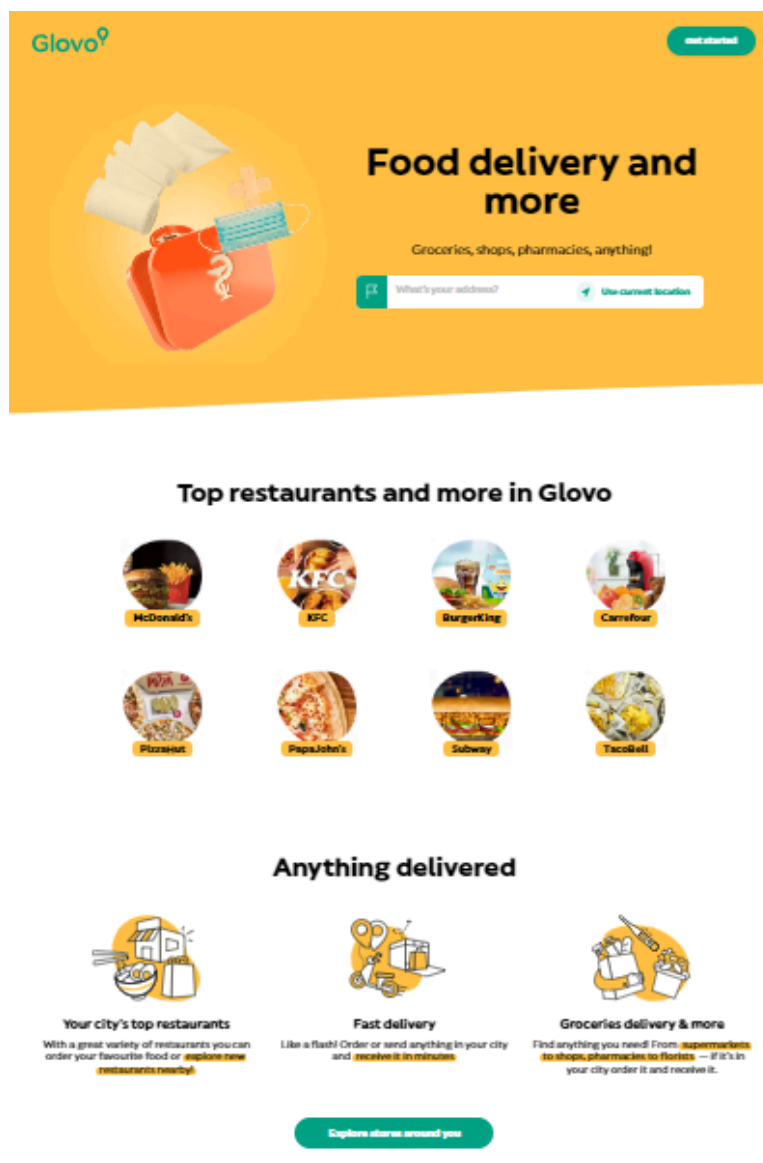
Ova platforma obično pruža informacije o vremenu dostave, cijenama, kao i ocjenama i recenzijama restorana kako bi korisnicima olakšala odabir. Korpa.ba je postala popularan izbor među ljudima koji žele da jednostavno naruče hranu iz različitih restorana i uživaju u dostavi u udobnosti svog doma.

## 2.2. Glovo.com

Glovo je globalna platforma za dostavu hrane i različitih proizvoda koja je dostupna u više zemalja širom svijeta. Osim hrane, Glovo omogućava korisnicima da naruče i dostave različite proizvode kao što su namirnice, lijekovi, kozmetika, kućni aparati i još mnogo toga.

<sup>2</sup> <https://korpa.ba/kategorije/>





Slika 3. Glovo.com<sup>3</sup>

Kroz Glovo aplikaciju, korisnici mogu pretraživati širok spektar restorana, prodavnica i usluga u svojoj blizini, birati proizvode ili hranu koje žele naručiti, te zatim zatražiti dostavu na željenu adresu. Glovo se ističe svojom raznovrsnošću ponude i brzom dostavom.

Osim toga, Glovo često nudi mogućnost praćenja narudžbe u realnom vremenu, tako da korisnici mogu pratiti status svoje dostave od trenutka naručivanja do isporuke. Ova aplikacija je postala popularna zbog svoje praktičnosti i široke ponude proizvoda koji se mogu naručiti i dostaviti na kućnu adresu.

<sup>3</sup> <https://glovoapp.com/>

### 3. NEDOSTACI APLIKACIJA ZA DOSTAVU HRANE

Iako su Korpa i Glovo popularne aplikacije za dostavu hrane i proizvoda, postoji nekoliko potencijalnih nedostataka koje korisnici mogu primijetiti:

- ❖ **Visoke provizije:** Obe platforme često naplaćuju visoke provizije restoranima i prodavačima, što može rezultirati većim cijenama za korisnike ili ograničenim izborom restorana koji su spremni sarađivati s platformom,
- ❖ **Vrijeme isporuke:** Ponekad se može dogoditi da vrijeme isporuke bude duže nego što je očekivano, posebno u vrijeme povećane potražnje ili tokom prometnih perioda,
- ❖ **Kvalitet hrane:** Iako ove platforme nastoje osigurati kvalitet hrane, ponekad se može dogoditi da hrana stigne manje svježa ili da se tokom transporta dogodi neko oštećenje,
- ❖ **Geografska ograničenja:** U nekim slučajevima, ove aplikacije mogu imati ograničenja u pogledu dostupnosti određenih restorana ili prodavnica u određenim područjima,
- ❖ **Troškovi dostave:** Ponekad troškovi dostave mogu biti visoki, posebno za manje narudžbe, što može povećati konačnu cijenu narudžbe,
- ❖ **Ograničenja menija:** Neki restorani ili prodavnice na ovim platformama mogu imati ograničen meni u poređenju s onim što nude direktno u svojim lokalima,

Svi ovi nedostaci mogu varirati u zavisnosti od lokacije, specifičnih restorana ili prodavnica, kao i trenutnog opterećenja i uslova rada aplikacija.

### 4. PRIJEDLOG POBOLJŠANJA APLIKACIJA

Istraživanje i poboljšanje iskustva korisnika unutar platforme za dostavu hrane predstavlja ključni aspekt suvremenog poslovanja. U svrhu optimizacije korisničkog iskustva, potrebno je implementirati niz inovacija i poboljšanja:

- ❖ **Personalizirane preporuke:** Razvijanje algoritama personaliziranih preporuka za jela ili restorane temeljenih na korisničkim preferencijama, prethodnim narudžbama te ocjenama,

- ❖ **Recenzije i ocjene dostave:** Pružiti korisnicima mogućnost ocjenjivanja ne samo hrane već i iskustva dostave, uključujući brzinu, stanje hrane prilikom isporuke i ljubaznost dostavljača,
- ❖ **Praćenje uživo dostave:** Implementacija opcije praćenja u stvarnom vremenu kako bi korisnici mogli pratiti lokaciju svoje narudžbe i predviđeno vrijeme dostave,
- ❖ **Program lojalnosti:** Razvijanje programa lojalnosti za redovite korisnike s povlasticama poput popusta, besplatne dostave ili ekskluzivnih ponuda,
- ❖ **Unaprijeđena korisnička podrška:** Unapređenje sistema podrške kako bi brzo i efikasno rješavali korisničke upite i probleme vezane uz narudžbe,
- ❖ **Ekološki održive opcije:** Promicanje praksi koje podržavaju očuvanje okoliša, uključujući ekološki prihvatljivu ambalažu i mogućnosti recikliranja,
- ❖ **Više mogućnosti plaćanja:** Proširenje opcija plaćanja kako bi se uključile različite metode plaćanja za veću fleksibilnost korisnika,
- ❖ **Suradnja sa lokalnim restoranima:** Poticanje suradnje s lokalnim restoranima kako bi se obogatila ponuda hrane i podržala lokalna zajednica,

Ova poboljšanja su ključna za prilagodbu platforme za dostavu hrane suvremenim potrebama korisnika, pružajući izvanredno iskustvo i potičući konkurentnost na tržištu. Uvažavajući ove aspekte, diplomski rad istražuje i analizira utjecaj implementacije ovih inovacija na kvalitetu usluge i zadovoljstvo korisnika unutar industrije dostave hrane.

## 5. SAVREMENI PRISTUP IZRADE APLIKACIJA

Savremeni pristup izrade aplikacija karakteriše se kombinacijom tehnologija, arhitektura i praksi koje omogućavaju brži razvoj, bolju skalabilnost, modularnost i visoku interaktivnost aplikacija.

Elementi savremenog pristupa su:

- ❖ **Microservices arhitektura:** Razdvajanje aplikacija na manje, nezavisne servise (mikroservise) omogućava fleksibilnost, lakše održavanje i skaliranje, omogućujući timovima da rade nezavisno i operiraju brže,
- ❖ **Cloud tehnologije:** Korištenje cloud platformi omogućava prilagođavanje resursa prema potrebama aplikacije. Cloud tehnologije pružaju i bolju dostupnost, sigurnost i globalnu dostupnost aplikacija,

- ❖ **Serverless tehnologija:** Ova tehnologija omogućava programerima da grade i pokreću aplikacije bez potrebe za upravljanjem infrastrukturom servera. To dovodi do smanjenja operativnih troškova, fleksibilnosti i veće efikasnosti u razvoju,
- ❖ **Frontend framework:** Korištenje modernih frontend framework-a kao što su React.js, Angular ili Vue.js omogućava brži razvoj interaktivnih korisničkih interfejsa, smanjuje ponovno korištenje koda i olakšava upravljanje stanjem aplikacija,
- ❖ **API-first pristup:** Fokus na razvoju API-ja kao osnovne tačke za komunikaciju između različitih servisa ili slojeva aplikacija omogućava bolju fleksibilnost i ponovno korištenje funkcionalnosti,
- ❖ **DevOps prakse:** Integracija razvoja i operacija (DevOps) omogućava automatizaciju procesa razvoja, testiranja, isporuke i održavanja aplikacija. Ovo dovodi do bržeg razvoja, bolje upravljanje promjenama i veće stabilnosti aplikacija,<sup>4</sup>

## 6. RAZVOJNO OKRUŽENJE

Razvojno okruženje je ključno za efikasan razvoj aplikacija, omogućavajući produktivnost, suradnju, testiranje i isporuku softvera. Odabir odgovarajućih alata i tehnologija prema zahtjevima projekta i preferencijama tima poboljšava produktivnost i kvalitet aplikacija.

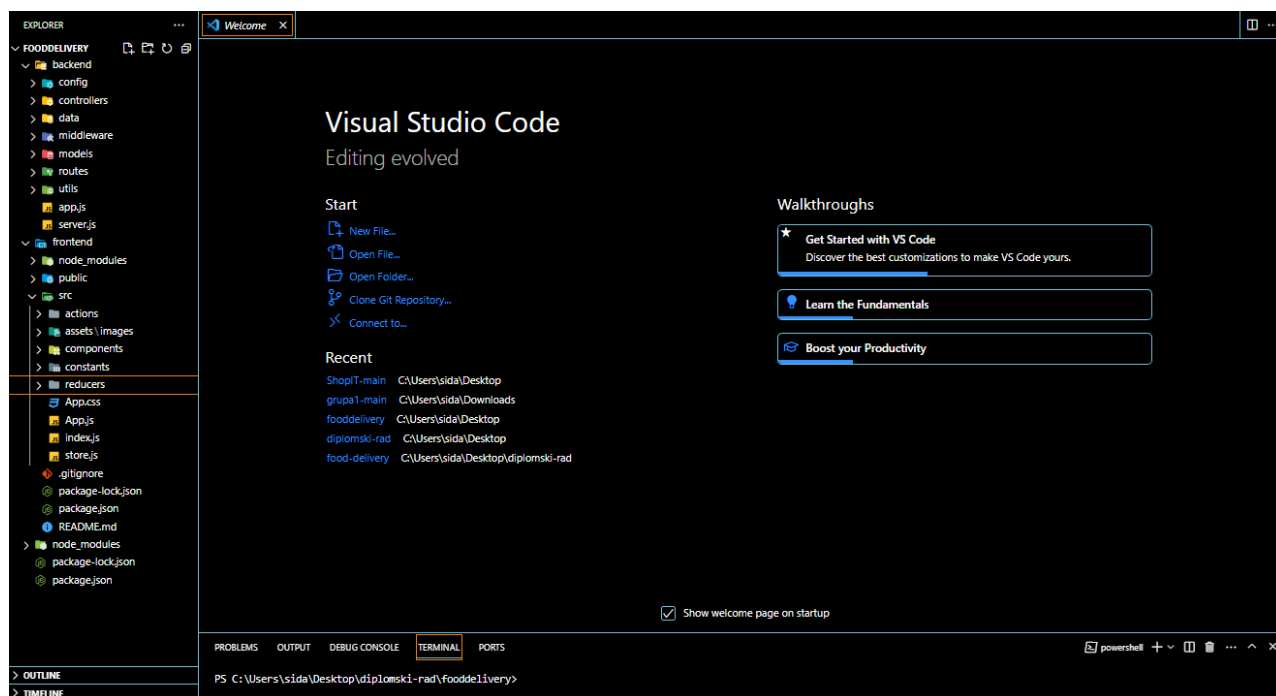
### 6.1. Visual Studio Code

**Visual Studio Code (VS Code)** je besplatan, popularan i svojevrsan tekstualni editor razvijen od strane Microsoft-a. Ovaj editor je namijenjen programerima i razvijateljima softvera, ali se koristi i u drugim oblastima zbog svoje fleksibilnosti i proširivosti.<sup>5</sup>

---

<sup>4</sup><https://successive.cloud/what-you-need-to-know-about-modern-application-development/#:~:text=Conclusion,development%20principles%20and%20best%20practices.>

<sup>5</sup> [https://en.wikipedia.org/wiki/Visual\\_Studio\\_Code](https://en.wikipedia.org/wiki/Visual_Studio_Code)



Slika 4. Visual Studio Code

Neke karakteristika i mogućnosti Visual Studio Code-a:

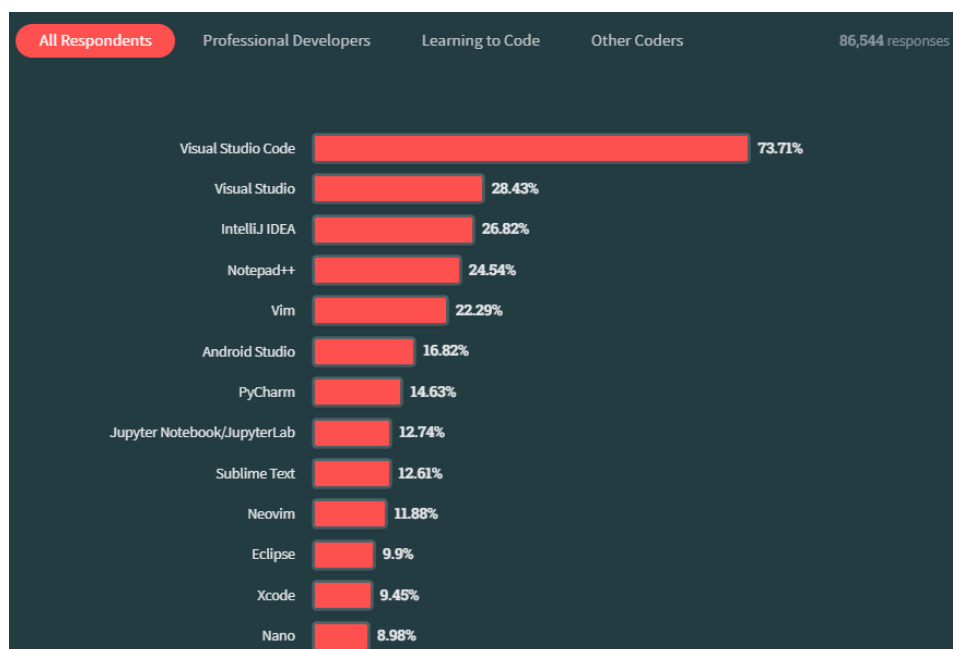
- ❖ **Prilagodljivost:** VS Code je visoko prilagodljiv, omogućavajući korisnicima da prilagode izgled, funkcionalnosti i podešavanja prema svojim potrebama. Također podržava različite teme i proširenja,
- ❖ **Integracija s jezicima i alatima:** Podržava razne programske jezike i alate za razvoj softvera kao što su JavaScript, TypeScript, Python, PHP, HTML, CSS, te omogućava integraciju sa sistemima za kontrolu verzija kao što su Git i GitHub,
- ❖ **Integrirani debager:** Omogućava korisnicima da jednostavno debuguju svoj kod putem integriranog debagera za različite jezike, što olakšava pronalaženje i ispravljanje grešaka,
- ❖ **Ekstenzije i dodaci:** Velika zajednica korisnika razvila je širok spektar ekstenzija koje dodaju nove funkcionalnosti i olakšavaju razvoj aplikacija.

➤ Neki od ekstenzija koje sam koristio za izradu ovog projekta uključuju:

- **Material Icon Theme:** Ova ekstenzija dodaje ikone inspirisane Material Design-om za vizuelno razlikovanje datoteka i foldera u projektu, čineći strukturu projekta vizualno privlačnijom,

- **Auto Rename Tag:** Omogućava automatsko preimenovanje parnih HTML/XML tagova odjednom, poboljšavajući konzistentnost u kodu,
- **Bootstrap 4, Font Awesome 4, Font Awesome 5 & Material Design:** Ove ekstenzije pružaju podršku za Bootstrap 4, Font Awesome 4, Font Awesome 5 i Material Design u Visual Studio Code-u, olakšavajući rad s popularnim bibliotekama i ikonama,<sup>6</sup>
- ❖ **Lakoća korištenja:** Ima intuitivan korisnički interfejs i nudi mnoge korisne funkcije kao što su automatsko dovršavanje koda, razne prečice i alati za olakšavanje razvoja,
- ❖ **Cross-platform podrška:** VS Code je dostupan na različitim platformama (Windows, macOS, Linux), omogućavajući korisnicima da koriste isti alat na različitim operativnim sistemima.

Visual Studio Code je postao izuzetno popularan zbog svoje jednostavnosti, moćnih alata i zajednice koja razvija raznovrsne proširenja, čineći ga omiljenim alatom među programerima i razvijateljima širom svijeta. Preliminarni rezultati pokazuju da je 78% početnika u programiranju izabralo VS Code kao svoj preferirani alat za razvoj softvera, dok je kod profesionalaca ova stopa bila 74%.<sup>7</sup>



Slika 5. Anketa stack overflow<sup>8</sup>

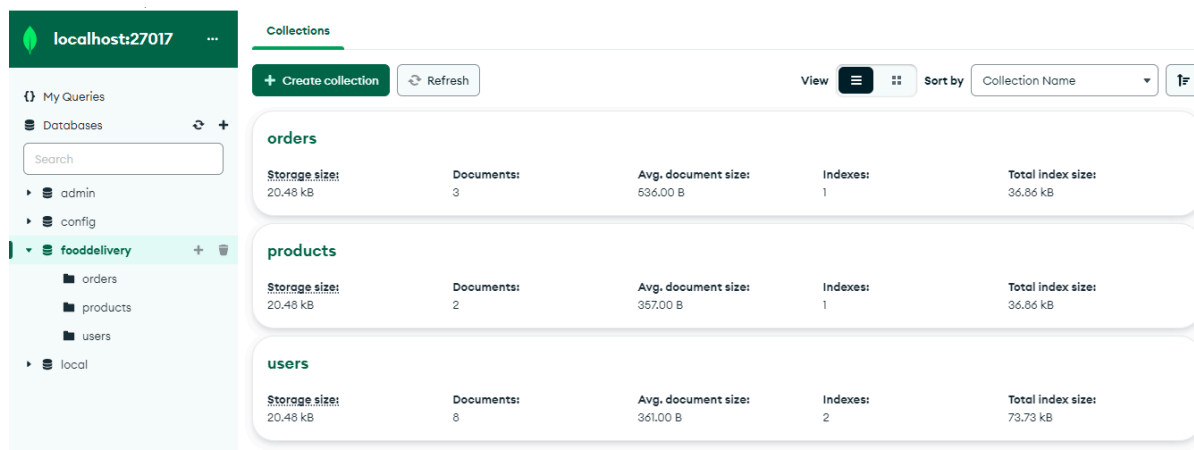
<sup>6</sup> <https://code.visualstudio.com/docs/editor/extension-marketplace>

<sup>7</sup> <https://survey.stackoverflow.co/2023/#most-popular-technologies-new-collab-tools>

<sup>8</sup> <https://survey.stackoverflow.co/2023/#most-popular-technologies-new-collab-tools>

## 6.2. MongoDB Compass

**MongoDB Compass** je vizuelni alat koji pruža grafički interfejs za upravljanje i istraživanje MongoDB baza podataka. Predstavlja koristan alat za developere, administratorske timove i analitičare koji rade s MongoDB-om.<sup>9</sup>



Slika 6. MongoDB Compass

Nekoliko ključnih karakteristika:

- ❖ **Grafički interfejs:** MongoDB Compass pruža intuitivan, vizuelni način za interakciju sa MongoDB bazama podataka, što olakšava kreiranje, pregled, uređivanje i brisanje podataka,
- ❖ **Query Builder:** Omogućava izradu i izvršavanje MongoDB upita bez potrebe za pisanjem kompleksnog JSON sintaksa. Query Builder olakšava kreiranje i testiranje upita,
- ❖ **Vizualizacija podataka:** Pruža mogućnost vizualizacije strukture podataka, omogućavajući korisnicima da bolje razumiju shemu i odnose među podacima u kolekcijama,
- ❖ **Indeksiranje:** Compass omogućava kreiranje, uređivanje i brisanje indeksa kako bi se poboljšala efikasnost upita,
- ❖ **Analiza performansi:** Pruža alate za analizu performansi baze podataka, kao što su spori upiti i analiza indeksa, što pomaže u identifikaciji i optimizaciji performansi,

<sup>9</sup> <https://www.mongodb.com/docs/compass/current/>

- ❖ **Geoprostorni upiti:** Specifično za geoprostorne podatke, Compass nudi alate za rad s lokacijskim podacima i izvođenje geoprostornih upita,

MongoDB Compass je koristan alat za različite faze razvoja aplikacija, omogućavajući korisnicima da lakše istražuju, upravljaju i analiziraju podatke u MongoDB bazama na vizualno intuitivan način. Ovaj alat olakšava rad s MongoDB-om bez potrebe za dubokim znanjem komandne linije ili specifičnosti JSON sintakse za upite.

### 6.3. Postman

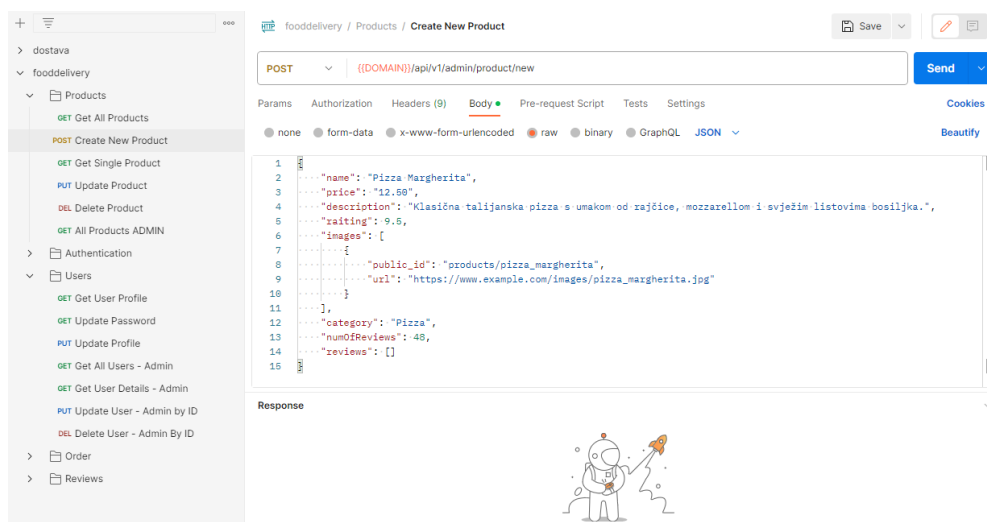
**Postman** je alat za testiranje i razvoj API-ja (Application Programming Interface). Ovaj alat omogućava programerima da kreiraju, testiraju i dokumentuju API-je na lak i efikasan način.<sup>10</sup> Nekoliko ključnih aspekata vezanih za Postman:

- ❖ **Kreiranje zahtjeva:** Postman omogućava korisnicima da kreiraju različite vrste HTTP zahtjeva poput GET, POST, PUT, DELETE i druge, kako bi komunicirali sa API-jem,
- ❖ **Organizacija kolekcija:** Korisnici mogu organizovati svoje zahtjeve u kolekcije, što olakšava grupisanje sličnih zahtjeva ili testova u jednoj lokaciji,
- ❖ **Testiranje API-ja:** Postman pruža mogućnost testiranja API-ja kroz automatsko izvršavanje testova koji provjeravaju odgovor API-ja. To omogućava provjeru ispravnosti odgovora na zahtjeve,
- ❖ **Environments:** Omogućava definiranje različitih okruženja (environments) poput dev, test, production, što olakšava upravljanje varijablama okoline i njihovim vrijednostima prilikom testiranja različitih okruženja,
- ❖ **Generisanje dokumentacije:** Postman olakšava generisanje dokumentacije API-ja, uključujući detaljne informacije o zahtjevima, odgovorima i primjerima korištenja,
- ❖ **Kolektivni rad:** Omogućava timovima da zajednički rade na testiranju i razvoju API-ja putem dijeljenja kolekcija, okruženja i skripti za testiranje.

---

<sup>10</sup> <https://www.postman.com/company/about-postman/>





Slika 7. Izgled POSTMAN okruženja

Postman je izuzetno koristan alat za razvijanje i testiranje API-ja, olakšavajući rad programerima i timovima koji rade na integraciji, razvoju ili testiranju različitih servisa putem API-ja. Svojom intuitivnom interfejsom i bogatim setom funkcija, Postman je postao ključni alat u alatkama za razvoj softvera.

## 7. SOFTWARE ZA RAZVOJ WEB APLIKACIJE

Razvojno okruženje, ili Dev Environment, obuhvata alate, tehnologije i postavke za razvoj softverskih aplikacija:

- ❖ **IDE/tekstualni editori:** Poput Visual Studio Code-a, IntelliJ IDEA, Sublime Text-a ili Atom-a za pisanje, uređivanje i testiranje koda,
- ❖ **Programski jezici i framework-ovi:** Za web aplikacije: JavaScript s React, Angular ili Vue.js; za backend: Node.js, Python, PHP, C# s odgovarajućim framework-om,
- ❖ **Verzionisanje koda:** Korištenje sistema kao što je Git i platformi poput GitHub-a ili GitLab-a za praćenje promjena, suradnju i upravljanje kodom,
- ❖ **Testni okviri i alati za automatizaciju:** Jest, Jasmine, Selenium, Cypress za testiranje funkcionalnosti, performansi i sigurnosti aplikacija,
- ❖ **Baza podataka:** Alati poput MongoDB Compass, MySQL Workbench ili pgAdmin za upravljanje bazama podataka i analizu podataka,
- ❖ **Kontejnerizacija i orkestracija:** Docker za kontejnerizaciju i Kubernetes za orkestraciju kontejnera za efikasno upravljanje aplikacijama i resursima,

- ❖ **CI/CD alati:** Jenkins, CircleCI, GitLab CI/CD, GitHub Actions za automatizaciju procesa razvoja, testiranja i isporuke softvera.

Razvojno okruženje je ključno za efikasan razvoj aplikacija, omogućavajući produktivnost, suradnju, testiranje i isporuku softvera. Odabir odgovarajućih alata i tehnologija prema zahtjevima projekta i preferencijama tima poboljšava produktivnost i kvalitet aplikacija.

## 7.1. ReactJS

**React.js** je popularna JavaScript biblioteka koju je razvio Facebook, a koristi se za izgradnju korisničkih interfejsa (UI) u web aplikacijama.<sup>11</sup>

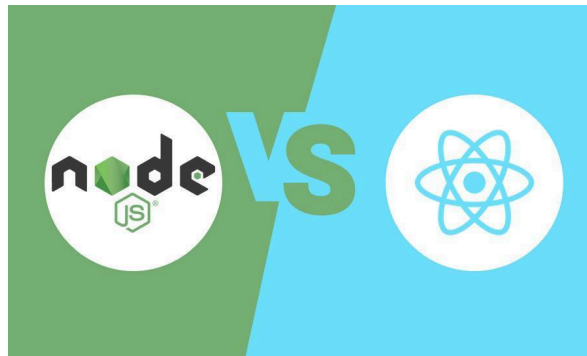
Karakteristike React-a su:

- ❖ **Komponentni pristup:** React se temelji na konceptu komponenti, omogućavajući programerima da grade korisnički interfejs kao skup ponovno upotrebljivih komponenti. Ovo olakšava organizaciju koda i omogućava modularnost,
- ❖ **Virtualni DOM:** React koristi virtualni DOM, efikasan način predstavljanja stvarnog DOM-a u memoriji. Ovo omogućava React-u da efikasno upravlja ažuriranjem DOM-a, čime se postiže bolja performansa,
- ❖ **JSX (JavaScript XML):** JSX je proširenje JavaScript-a koje omogućava pisanje HTML-sličnog koda unutar JavaScript-a. To olakšava pisanje komponenti u React-u,
- ❖ **Jednosmjerno vezivanje podataka (One-way data binding):** React promovira jednosmjerno vezivanje podataka, što znači da se promjene u komponenti automatski odražavaju u UI-u, ali ne i obrnuto. Ovo olakšava praćenje podataka i održavanje stanja aplikacije,
- ❖ **Velika zajednica i ekosistem:** React ima veliku zajednicu programera i obilje dostupnih biblioteka (npr. Redux za upravljanje stanjem, React Router za rutiranje itd.), alata i resursa koji olakšavaju razvoj aplikacija,
- ❖ **Reaktivnost:** React omogućava reaktivne promjene u UI-u. Kada se podaci mijenjaju, React samostalno odlučuje kako će ažurirati odgovarajuće dijelove interfejsa, što čini razvoj i održavanje aplikacija bržim i jednostavnijim,

---

<sup>11</sup><https://blog.hubspot.com/website/react-js#:~:text=js%3F-,React,.,reusable%20piece%20of%20HTML%20code>

React se često koristi u kombinaciji sa savremenim alatima i tehnologijama za razvoj web aplikacija. Njegova popularnost proizlazi iz efikasnosti, modularnosti i jednostavnosti u kreiranju korisničkog interfejsa.



Slika 8. ReactJS and NodeJS<sup>12</sup>

## 7.2. NodeJS

Node.js je open-source okruženje za izvršavanje JavaScript koda na serverskoj strani. Ono omogućava programerima da pišu JavaScript kod koji se izvršava izvan web preglednika, što znači da se JavaScript može koristiti za izradu serverskih aplikacija, API-ja i ostalih aplikacija koje se izvršavaju na serverima.<sup>13</sup>

Jedna od ključnih karakteristika koja proširuje upotrebu Node.js-a je njegov asinhroni i događajno vođen model. Ova karakteristika omogućava Node.js-u da obradi više zahtjeva istovremeno bez blokiranja izvršavanja drugih operacija. To čini Node.js pogodnim za razvoj brzih i skalabilnih serverskih aplikacija koje zahtijevaju efikasnost u rukovanju više istovremenih zahtjeva.

---

<sup>12</sup> <https://www.reactjsindia.com/blog/node-js-vs-reactjs-comparison-which-to-choose-for-your-js-project/>

<sup>13</sup> <https://nodejs.org/en/about>

```

1  const app = require('./app');
2  const connectDatabase = require('./config/database')
3
4  const dotenv = require('dotenv');
5  const cloudinary = require('cloudinary').v2
6
7  // Handle Uncaught exception
8  process.on('uncaughtException', err => {
9    console.log(`ERROR: ${err.stack}`);
10   console.log('Shutting down server due to uncaught exception');
11   process.exit(1)
12 })
13
14 // Setting up config file
15 dotenv.config({ path: 'backend/config/config.env' })
16
17 // connecting to db
18 connectDatabase();
19
20 cloudinary.config({
21   cloud_name: process.env.CLOUDINARY_CLOUD_NAME,
22   api_key: process.env.CLOUDINARY_API_KEY,
23   api_secret: process.env.CLOUDINARY_API_SECRET
24 });
25
26 const server = app.listen(process.env.PORT, () => {
27   console.log(`Server started on PORT: ${process.env.PORT} in ${process.env.NODE_ENV} mode.`)
28 })
29
30 // Handle Unhandled Promise rejections
31 process.on('unhandledRejection', err => {
32   console.log(`ERROR: ${err.stack}`);
33   console.log('Shutting down the server due to Unhandled Promise rejection')
34   server.close(() => {
35     process.exit(1)
36   })
37 })
38

```

Slika 9. Node.js

Karakteristika Node.js-a:

- ❖ **Jednostavnost u korištenju JavaScripta:** Node.js omogućava programerima da koriste JavaScript i na serveru, što olakšava razvoj punog stacka aplikacija koristeći isti jezik i koncepte na frontendu i backendu,
- ❖ **Asinhroni i događajno vođen rad:** Node.js je izgrađen na asinhronom, događajno vođenom modelu, što omogućava da se više zahtjeva obrađuje istovremeno bez blokiranja izvršavanja drugih operacija. Ovo može poboljšati performanse aplikacija,
- ❖ **Modularnost:** Node.js ima veliku biblioteku modula i paketa putem npm (Node Package Manager), koji olakšava integraciju dodatnih funkcionalnosti u aplikacije,
- ❖ **Brzina izvršavanja:** Node.js koristi V8 JavaScript engine, razvijen od strane Google-a, što dovodi do brzog izvršavanja JavaScript koda,
- ❖ **Kreiranje API-ja i serverskih aplikacija:** Node.js je popularan za razvoj API-ja i serverskih aplikacija zbog svoje sposobnosti obrade velikog broja istovremenih zahtjeva,

- ❖ **Streaming podataka:** Node.js podržava streaming podataka, što znači da se podaci mogu obrađivati dok se još uvijek prenose, što može značajno poboljšati performanse pri radu s velikim količinama podataka,

Node.js je postao popularan alat za razvoj serverske strane aplikacija, posebno za aplikacije visokih performansi, real-time aplikacije, API-je i mikroservise. Njegova fleksibilnost, brzina izvršavanja i aktivna zajednica programera doprinose širokoj upotrebi Node.js-a u raznim aplikacijskim scenarijima.

### 7.3. MongoDB

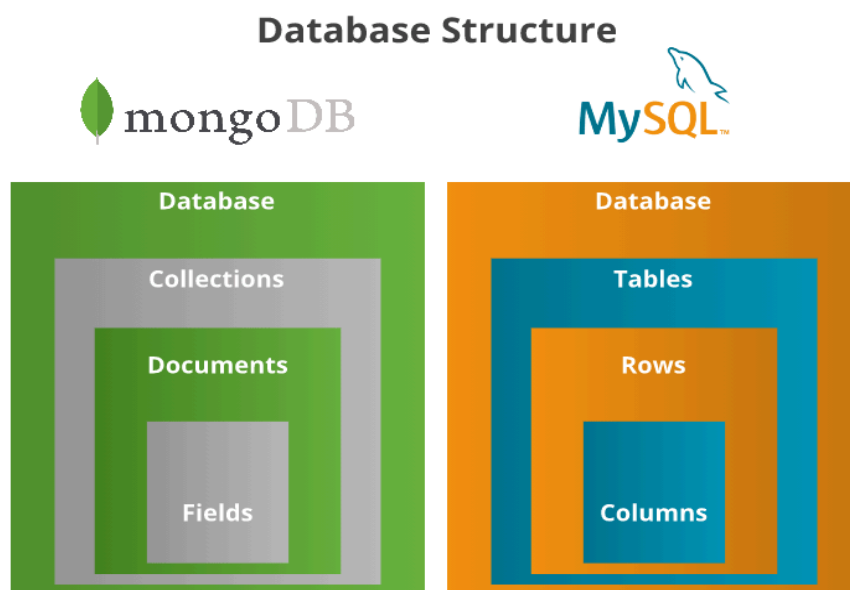
**MongoDB** predstavlja revolucionarni pristup u svijetu baza podataka. Umjesto tradicionalnog relacijskog modela, koji koristi tabele i stroge šeme podataka, MongoDB koristi fleksibilne dokumente, slične JSON-u, nazvane BSON, za organizaciju i skladištenje informacija.

Ovaj NoSQL pristup omogućava MongoDB-u da se prilagodi različitim tipovima podataka bez potrebe za unaprijed definisanom strukturom. Dokumenti u MongoDB-u mogu sadržavati različite vrste podataka, uključujući tekst, brojeve, nizove, datume i složene objekte, čineći ih fleksibilnim i pogodnim za aplikacije koje se razvijaju u promjenjivim okolinama.

Osim toga, MongoDB podržava horizontalno skaliranje, što znači da se kapacitet baze podataka može povećavati dodavanjem novih servera. To omogućava MongoDB-u da raste i prilagođava se potrebama aplikacija koje se šire, bez gubitka performansi ili dostupnosti.

MongoDB je također poznat po brzim upitima i indeksiranju podataka, pružajući efikasnost u pretraživanju i obradi informacija, što ga čini popularnim izborom za aplikacije koje zahtijevaju visoke performanse i velike količine podataka.

Zahvaljujući svojoj fleksibilnosti, skalabilnosti i sposobnosti da se prilagodi promjenama u zahtjevima aplikacija, MongoDB je postao ključni igrač u svijetu modernih aplikacija, uključujući web aplikacije, analitiku podataka, e-trgovinu i mnoge druge.



Slika 10. MongoDB vs Relational<sup>14</sup>

## 8. IDEJNI PRIJEDLOG APLIKACIJE

Naruči.ba je jednostavna i intuitivna mobilna aplikacija koja korisnicima omogućuje naručivanje hrane i dostavu direktno na njihovu adresu. Cilj je pružiti korisnicima praktičnost, širinu izbora i pouzdanu dostavu hrane.

### 8.1. Analiza zahtjeva

Prva faza razvoja aplikacije je analiza korisničkih zahtjeva. Ova faza podrazumijeva ispitivanje tržišta gdje je glavni cilj bio istražiti review postojećih rješenja od strane krajnjeg korisnika sa kritikama, prednostima i nedostacima. Izvor za ove podatke koji će odrediti dalji tok ovog projekta su stranice poput Glovo, Korpa i sl., na osnovu kojih se napravila lista korisničkih zahtjeva koji su odredili funkcionalnosti same aplikacije:

❖ Za admina:

- Upravljanje proizvodima(što uključuje CRUD – Create, Read, Update, Delete operacije),
- Upravljanje narudžbama(što uključuje CRUD operacije),
- Upravljanje korisnicima(što uključuje CRUD operacije),

❖ Za korisnika:

---

<sup>14</sup> <https://www.linkedin.com/pulse/relational-database-vs-mongodb-shaharyar-saleem>

- Pregled dostupnih proizvoda sa opcijom pretrage i filtriranja,
- Narudžba,
- Ažuriranje profila,
- Recenzija,

## 8.2. Analiza i dizajn aplikacije

U fazi analize i dizajna sistema, svi objekti i akteri sistema se prikazuju grafički, zajedno sa svim funkcionalnostima. U ovoj fazi se zapravo vrši projekcija sistema, tj. prikaz međusobno povezanih objekata u sistemu na način na koji bi trebali funkcionirati u realnom sistemu. Sistemi se mogu predstaviti u formi dijagrama, kojih ima mnogo. Za izradu našeg projekta, dijagrami korišteni u ovoj fazi su Use case dijagram (dijagram slučaja korištenja) i klasni dijagram.

## 8.3. Use Case Dijagram

Use case dijagram je vrsta dijagrama u analizi softverskih sistema koji se koristi za opisivanje interakcija između sistema (ili aplikacija) i njenih korisnika (tzv. "aktera") kako bi se identificirali slučajevi upotrebe ili funkcionalnosti sistema.<sup>15</sup>

Elementi Use Case dijagrama:

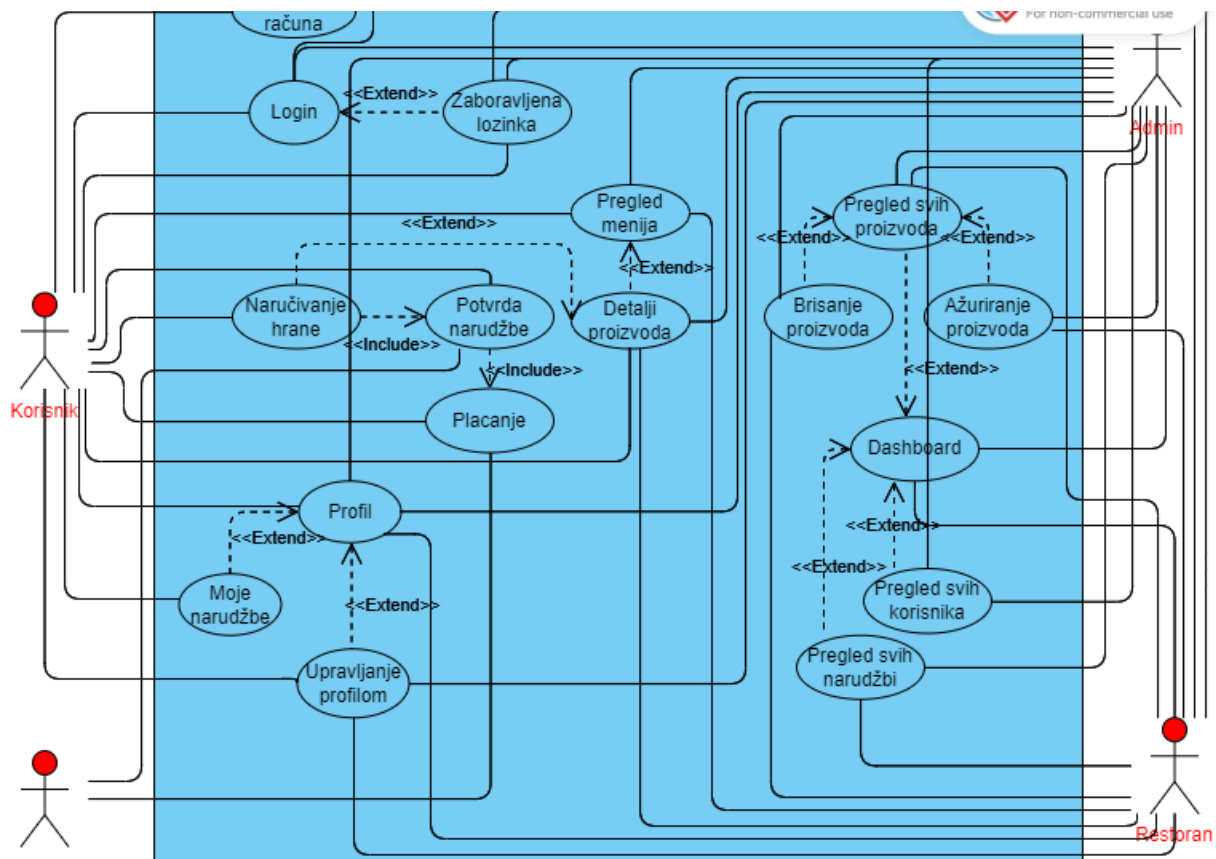
- ❖ Aktor:
  - Predstavlja bilo kojeg entiteta koji interagira s aplikacijom,
  - Može biti korisnik, vanjski sistem, ili drugi entitet koji ima ulogu u interakciji s aplikacijom,
- ❖ Use Case (Slučaj upotrebe):
  - Odnosi se na konkretne funkcionalnosti ili akcije koje sistem pruža akterima,
  - Opisuje ponašanje sistema u odgovoru na specifičan zahtjev korisnika,
- ❖ Linije asocijacije:
  - Veze između aktera i slučajeva upotrebe koje ilustriraju kako akteri koriste funkcionalnosti sistema.<sup>16</sup>

<sup>15</sup><https://www.usability.gov/how-to-and-tools/methods/use-cases.html#:~:text=Elements%20of%20a%20Use%20Case&text=Actor%20%E2%80%93%20anyone%20or%20anything%20that,system%20to%20achieve%20a%20goal>

<sup>16</sup><https://www.usability.gov/how-to-and-tools/methods/use-cases.html#:~:text=Elements%20of%20a%20Use%20Case&text=Actor%20%E2%80%93%20anyone%20or%20anything%20that,system%20to%20achieve%20a%20goal>

Use Case je osnovna komponenta ovog dijagrama i uglavnom predstavlja funkcionalnost. Obično se nalazi unutar granica sistema. Akteri se sa slučajevima povezuju raznim vrstama veza, a najčešće korištene veze su asocijacija, uključivost, te proširenje. Veza asocijacije se uglavnom povezuju aktera sa slučajevima, dok se slučajevi uglavnom povezuju vezom uključenosti ili vezom proširenja.

Za kreiranje use case dijagrama korišteno je razvojno okruženje Visual Paradigm Online<sup>17</sup>



Slika 11. Use Case dijagram

## 8.4. Klasni dijagram

Klasni dijagram je vizualna reprezentacija strukture klasa u softverskom sistemu. Uključuje klase s atributima i metodama te pokazuje odnose među klasama. Ovo olakšava planiranje arhitekture softvera i vizualizaciju načina na koji klase međusobno komuniciraju.<sup>18</sup>

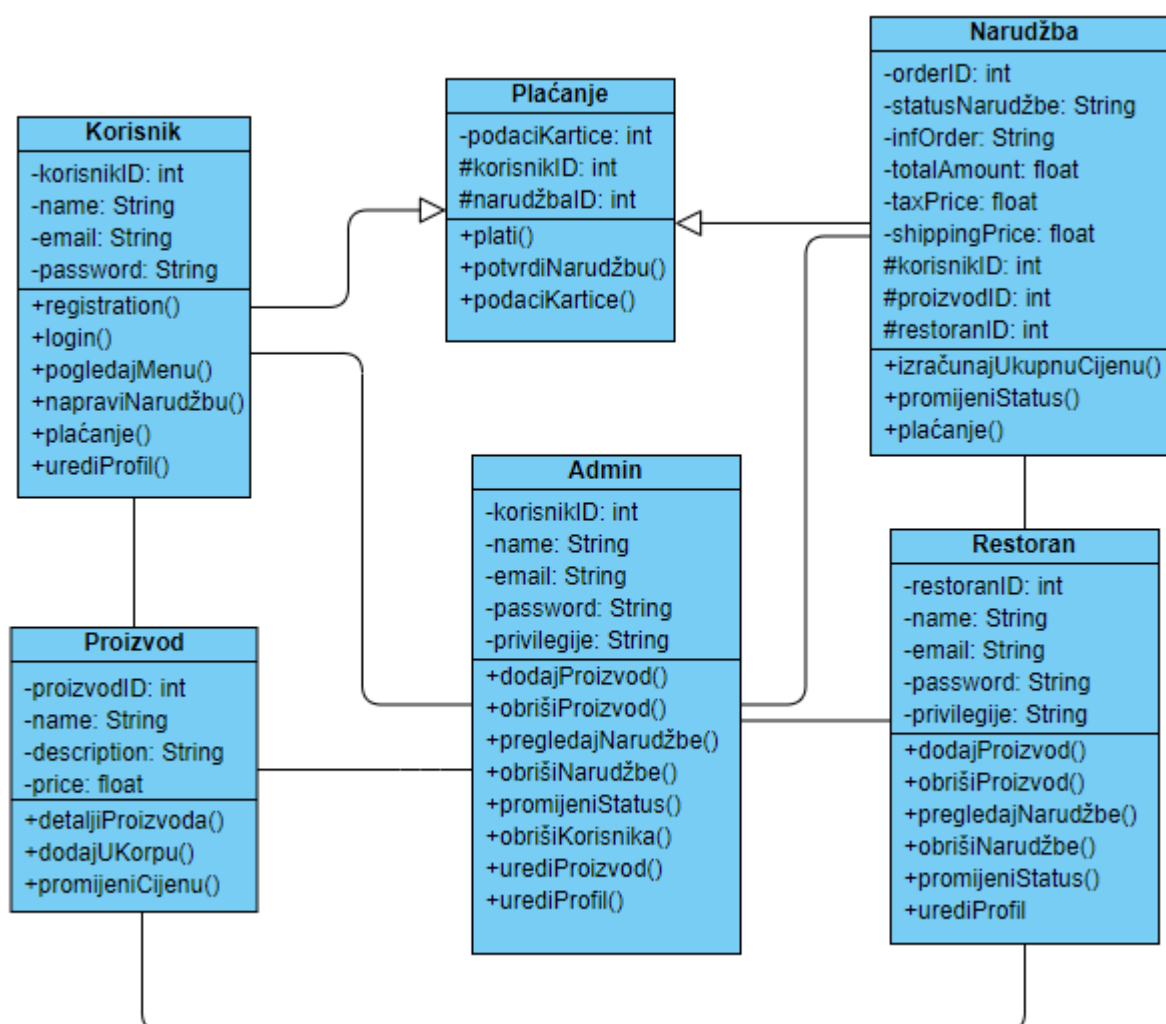
<sup>17</sup> <https://online.visual-paradigm.com/drive/#diagramlist:proj=0&diagram=list>

<sup>18</sup> <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/>



Elementi klasnog dijagrama obuhvaćaju:

- **Ime klase**
  - Ime klase se pojavljuje u prvom dijelu,
- **Atributi klase**
  - Atributi,
  - Tip atributa,
- **Metode klase**
  - Operacije su prikazane u trećem dijelu. To su usluge koje klasa pruža,
  - Povratni tip metode,
  - Povratni tip parametara metode,
  - Operacije.<sup>19</sup>



Slika 12. Klasni dijagram

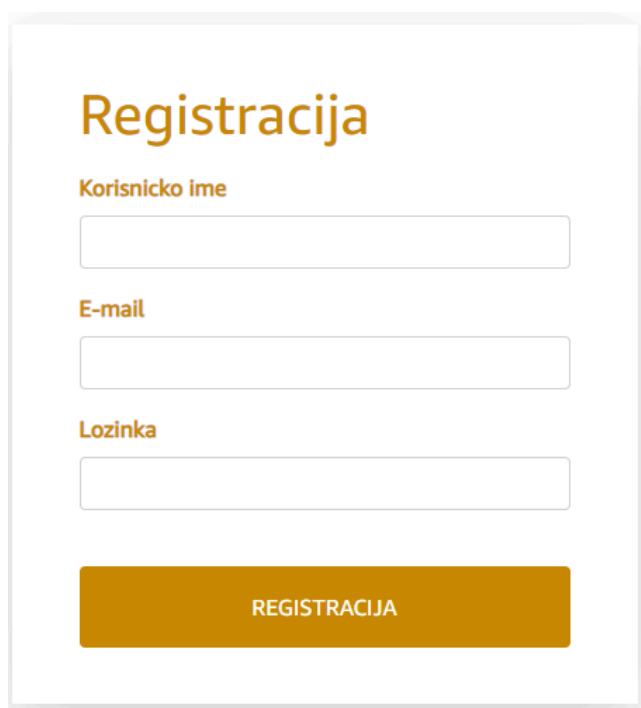
<sup>19</sup> <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/>

## 9. IZRADA APLIKACIJE - FRONTEND I BACKEND

Kao što je prethodno već objašnjeno, za frontend je korišten React.js koji je kombinacija HTML i JavaScript-a sa mnogim korisnim dodacima. U nastavku će biti objašnjen frontend i backend pojedinih segmenata same aplikacije. Kreirani dijagrami i analiza zahtjeva će biti osnova za razvijanje ovog projekta.

### 9.1. Login i Register page

Na samom početku kreiran je Login page za korisnika koji je prethodno registrovan, te Register page za korisnika koji se želi registrirati. Prilikom same registracije, potrebno je unijeti ime, korisničko ime, email i lozinku. Što se tiče frontenda, u pitanju je klasična manipulacija input elementima koji se šalju sa Register forme direktno u bazu iz koje se povlače samo korisničko ime i lozinka prilikom prijave.



The image shows a registration form with the title "Registracija" in orange. Below the title are three input fields with labels "Korisnicko ime", "E-mail", and "Lozinka" in orange. At the bottom of the form is a large orange button with the text "REGISTRACIJA" in white.

Slika 13. Forma za registraciju

```
// Handling users roles
exports.authorizeRoles = (...roles) => {
  return (req, res, next) => {
    if (!roles.includes(req.user.role)) {
      return next(
        new ErrorHandler(`Role (${req.user.role}) is not
allowed to access this resource`, 403))
    }
  }
}
```

```
}  
  next()  
}  
}
```

Funkcija `authorizeRoles` je middleware funkcija koja provjerava da li uloga (role) trenutno prijavljenog korisnika (dostupna u `req.user.role`) odgovara barem jednoj od uloga navedenih u parametrima funkcije.

Kada se ova middleware funkcija koristi u aplikaciji, npr. u rutama Express.js-a, omogućava da se određene rute ili resursi zaštite od pristupa korisnika koji nemaju odgovarajuće uloge. Ako korisnik nema odgovarajuću ulogu, funkcija će vratiti status kod 403 (forbidden) i odgovarajuću poruku.

Ukoliko korisnik unese pogrešan email ili lozinku, sistem prikazuje error koji kaže da je unesen pogrešan email ili lozinka. Nakon uspješne registracije i prijave, korisniku će se prikazati početna stranica (Home page) aplikacije.

## Prijava

E-mail

Lozinka

[Zaboravili ste lozinku?](#)

PRIJAVA

[Novi ste korisnik?](#)

Slika 14. Forma za prijavu

Što se tiče same lozinke, prilikom registracije ona se kodirana sprema u MongoDB bazu koja je prethodno povezana sa našim frontendom. Za ovu potrebu korištena je `bcrypt` Node.js

biblioteka kojom se vrlo jednostavno štite korisnikovi podaci i koja omogućava da se korisničke lozinke ne spremaju u bazu u izvornom obliku. Na osnovu sljedećeg dijela koda objasniti ćemo heširanje same lozinke.

```
// Encrypting password before saving user
userSchema.pre('save', async function (next) {
  if (!this.isModified('password')) {
    next()
  }
  this.password = await bcrypt.hash(this.password, 10)
})
```

Ovaj dio koda je dio Mongoose šeme (userSchema) u Node.js okruženju, a koristi se za spremanje korisnika u bazu podataka. Koristi ".pre" sredstvo koje se aktivira prije spremanja korisničkog objekta. Funkcija unutar ovog koda izvršava se prije nego što se korisnik spremi u bazu podataka prije akcije spremanja.

Ovaj kod osigurava enkripciju lozinke korisnika prije nego što se spremi u bazu podataka, što povećava sigurnost pohranjivanja osjetljivih informacija.

```
{
  _id: ObjectId('653ff3b90815e174a65b64af'),
  name: "Adis",
  email: "ahodzic1@gmail.com",
  password: "$2a$10$dNJAHDci4vEZ95vsBudo/Oz2DRdGts1r2TSdx.hnV84p7UzIst9oC",
  avatar: Object {
    public_id: "samples/pha01gzssfzggqoxaovd4",
    url: "https://res.cloudinary.com/dmui8zfee/image/upload/v1700243392/samples/...",
    role: "admin",
    createdAt: 2023-10-30T18:19:37.779+00:00,
    __v: 0,
    resetPasswordToken: "1698938070783"
  }
}
```

Slika 15. Korisnik u MongoDB

Istovremeno, u Postman programu kreiramo post API-je i definišemo rute za login i register kako bismo provjerili da sve ispravno funkcioniše. Kada je u pitanju autorizacija, objasniti ćemo tri najbitnije funkcije koje su suština cijelog procesa.

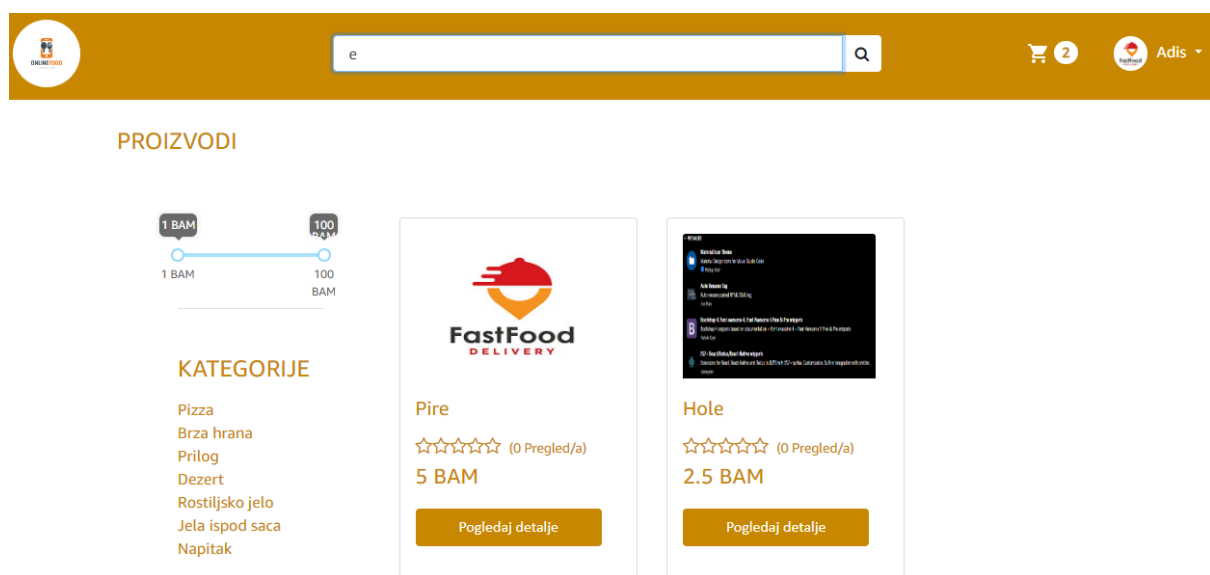
```
userSchema.methods.getJwtToken = function () {
  return jwt.sign({ id: this._id }, process.env.JWT_SECRET, {
    expiresIn: process.env.JWT_EXPIRES_TIME
  });
}
```

Ovaj komad koda dio je Mongoose sheme (userSchema) u Node.js okruženju i predstavlja definiciju metode za generiranje JWT tokena za korisnika.

Ova metoda `getJwtToken()` omogućava generiranje JWT tokena za trenutnog korisnika na temelju njegovog ID-a i definiranih vremenskih ograničenja, koristeći tajni ključ za potpisivanje tokena.

## 9.2. Search Results i Filter page

Kada se radi o pronalaženju pravih proizvoda u meniju, pretraživanje i filtriranje proizvoda postaje ključno za efikasno i korisno iskustvo kupovine. Ove funkcionalnosti su ključne kako bi kupci brzo pronašli željeni proizvod među raznolikim opcijama koje su im dostupne.



Slika 16. Filter and Search page

Mogućnost pretraživanja proizvoda omogućuje korisnicima da unesu ključne riječi ili slova povezane s proizvodom koji traže. Ovaj alat omogućuje korisnicima da brzo pronađu proizvode koje traže, filtrirajući rezultate prema relevantnosti ključnih riječi ili slova koje su unijeli.

```

search() {
    const keyword = this.queryStr.keyword ? {
        name: {
            $regex: this.queryStr.keyword,
            $options: 'i'
        }
    } : {}
    this.query = this.query.find({ ...keyword });
    return this;
}

```

U okviru istraživanja za diplomski rad, istražuje se operacija pretraživanja koja koristi polje “name” u bazi podataka kako bi omogućila korisnicima pretraživanje podataka putem regularnih izraza. Ovaj pristup omogućuje prilagodbu pretraživanja koristeći ključne riječi unesene od strane korisnika.

Filtriranje proizvoda omogućuje korisnicima da sužavaju rezultate pretrage prema određenim kriterijima ili parametrima. U ovom projektu filtriranje se vrši prema kategorijama proizvoda i cijeni koji olakšavaju pronalazak željenog proizvoda u skladu s korisnikovim željama.

```

filter() {
    const queryCopy = { ...this.queryStr }
    //Removing fields from the query
    const removeFields = ['keyword', 'limit', 'page']
    removeFields.forEach(el => delete queryCopy[el]);
    // Advance filter for price, rating
    let queryStr = JSON.stringify(queryCopy);
    queryStr = queryStr.replace(/\b(gt|gte|lt|lte)\b/g, match =>
`$${match}`);
    this.query = this.query.find(JSON.parse(queryStr));
    return this;
}

```

U okviru istraživanja za diplomski rad, istražuje se operacija filtriranja podataka koja omogućuje personalizirano pretraživanje u bazi podataka. Ovaj pristup filtriranju podataka omogućuje prilagodbu pretrage prema različitim parametrima, što korisnicima omogućuje personalizirano pretraživanje prilagođeno njihovim potrebama i preferencijama.

Ova implementacija omogućuje dodavanje različitih tipova pretrage i operacija usporedbe u URL-ove ili druge upite, omogućavajući napredne upite nad podacima. Ako želite proširiti funkcionalnost na druge vrste pretraga ili operacija usporedbe, možete dodati odgovarajuće logike u funkciju filter().

### 9.3. Product Details page

Stranica Product details pruža korisniku informacije o proizvodu koji istražuje. Ova pruža detaljan uvid u informacije proizvoda. Na ovoj stranici, korisnik dobija detaljne informacije kao što su naziv proizvoda, opis, cijenu. Također, dobija i sliku proizvoda prije nego što ga kupi. Dodatno, korisnici imaju mogućnost da dodaju svoju recenziju proizvoda ili da dodaju proizvod u korpu.



Ćevapi

Proizvod #655e505f7128eadaeffa8511

☆☆☆☆☆ (0 Pregleda)

8.5 BAM

- 1 + Dodaj u korpu

Opis:

Ćevapi ili ćevapčići, su bosanskohercegovački nacionalni specijalitet sa roštilja i najpoznatiji bosanskohercegovački brand u inostranstvu. Riječ je o malim valjušcima od mljevenog mesa, obično teletine, sa dodatkom bijelog i crvenog luka, te različitih začina. Prosječna dužina ćevapa je oko 5 cm. Ćevapi se prže na roštilju, a tradicionalno se služe u lepinjama ili somunima natopljenim sosom od prženja, uz dodatak sitno sjeckanog crvenog luka. Ćevapi se mogu služiti i uz rižu, pržene krompiriće (pomfrit), salatu, ajvar, no to je uglavnom praksa u inostranstvu.

Pošaljite svoju recenziju

Slika 17: Product Details page

```
// Get single product details => /api/v1/product/:id
exports.getSingleProduct = catchAsyncErrors(async (req, res, next) => {
  const product = await Product.findById(req.params.id);
  if (!product) {
    return next(new ErrorHandler('Proizvod nije pronadjen', 404))
  }
})
```

```
res.status(200).json({  
  success: true,  
  product  
}) })
```

Ova funkcija, koja je integrirana u serversku logiku, omogućava dohvaćanje detalja pojedinačnog proizvoda putem API endpointa `/api/v1/product/:id`.

Ova funkcija je ključna jer omogućava klijentskoj strani da putem API endpointa dobije detaljne informacije o pojedinom proizvodu temeljem ID-a proslijeđenog u zahtjevu. Ako je proizvod pronađen, vraćaju se detaljni podaci; u protivnom, klijent dobiva obavijest da proizvod nije pronađen. Ovaj proces omogućava bolje korisničko iskustvo i olakšava pristup informacijama o proizvodima.

```
const addToCart = () => {  
  
  dispatch(addItemToCart(match.params.id, quantity))  
  
  alert.success('Proizvod dodan u korpu')  
  
}  
  
const increaseQty = () => {  
  
  const count = document.querySelector('.count')  
  
  if (count.valueAsNumber <= 0) return  
  
  const qty = count.valueAsNumber + 1;  
  
  setQuantity(qty)  
  
}  
  
const decreaseQty = () => {  
  
  const count = document.querySelector('.count')  
  
  if (count.valueAsNumber <= 1) return  
  
  const qty = count.valueAsNumber - 1;
```



```
setQuantity(qty)
```

```
}
```

`addToCart` funkcija je ključna u interakciji korisnika s košaricom proizvoda. Kada se korisnik odluči za dodavanje proizvoda u košaricu, ova funkcija se aktivira. Ona poziva Redux akciju `addItemToCart` koja zahtijeva ID proizvoda (`match.params.id`) i određenu količinu proizvoda (`quantity`). Također, ova funkcija obavještava korisnika o uspješnom dodavanju proizvoda u košaricu, pružajući korisniku povratnu informaciju o izvršenoj radnji.

Funkcija `increaseQty` pruža mogućnost povećanja količine proizvoda za jedan pritiskom na odgovarajuće dugme za povećanje. Nakon provjere trenutne vrijednosti količine, ova funkcija povećava količinu proizvoda za jedan te ažurira trenutno stanje količine proizvoda.

Funkcija `decreaseQty` omogućuje smanjenje količine proizvoda za jedan pritiskom na odgovarajuće dugme za smanjenje. Provjerava trenutnu vrijednost količine i, pod uvjetom da je veća od 1, smanjuje količinu proizvoda za jedan. Nakon toga, ažurira trenutno stanje količine proizvoda.

Ove funkcije su bitne za interaktivnost korisnika s košaricom proizvoda, omogućujući im jednostavno upravljanje količinom proizvoda unutar košarice te pružajući povratne informacije o uspješnim akcijama kako bi poboljšale korisničko iskustvo.

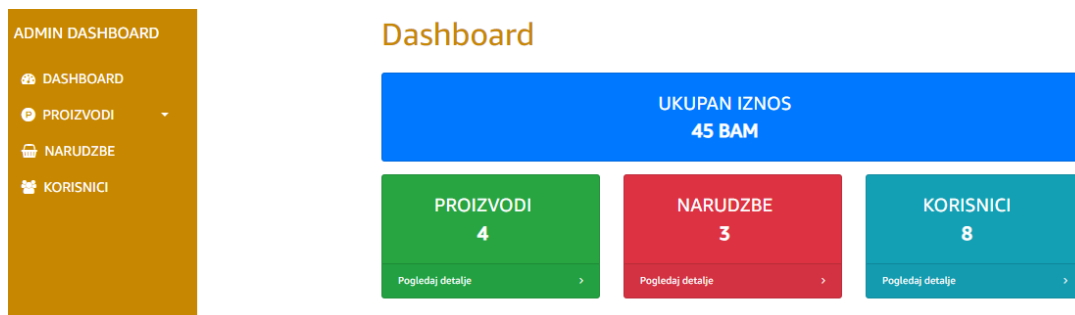
## 10. ADMIN DASHBOARD

Ovaj sistem omogućava administratoru privilegije potrebne za upravljanje različitim segmentima ove aplikacije, prilagođavanje usluga korisnicima i optimizaciju procesa. Kroz ovaj panel, administrator ima mogućnost upravljanja raznovrsnim funkcijama koje obuhvataju proizvode, narudžbe i korisnike. Cilj ovog sistema je omogućiti efikasno upravljanje, istovremeno pružajući fleksibilnost i prilagođavanje promjenjivim zahtjevima.

Na početnoj stranici admin panela nalaze se dvije ključne komponente. Prva je sidebar koja omogućuje administratoru navigaciju kroz sve mogućnosti koje su mu na raspolaganju. Administratorske mogućnosti obuhvataju kreiranje novih proizvoda, pregled svih proizvoda, uređivanje i brisanje postojećih proizvoda, pregled narudžbi svih korisnika, kao i pregled

informacija o korisnicima. Druga komponenta su informacije o broju korisnika, narudžbi i proizvoda

Ovaj raspored omogućava administratoru lakoću pristupa ključnim funkcijama, olakšavajući upravljanje sistemom i brz pristup potrebnim informacijama i akcijama.



Slika 17. Admin Dashboard

Kada je u pitanju stranica users, klikom na istu prikazuje se lista svih korisnika, uključujući njihov ID, korisničko ime, ime, e-mail, rola atribut i dio Actions koji nudi opciju da pogledamo detalje korisnika iz baze.

## Svi korisnici

Show entries

10

Search

KORISNICKI ID	IME	EMAIL	ROLA	AKCIJA
653ff3b90815e174a65b64af	Adis	ahodzic1@gmail.com	admin	
6542a5cee0666ecb78aa4073	Midheta	midheta@gmail.com	user	
6542a6bee0666ecb78aa4096	Merjema	merjema@gmail.com	user	
6556039489f854327a879095	Hana	hana1@gmail.com	user	
KORISNICKI ID	IME	EMAIL	ROLA	AKCIJA

Showing 1 to 4 of 4 entries

Previous 1 Next

Slika 18. All Users (Admin)

```
// Get all user => /api/v1/admin/users
exports.allUsers = catchAsyncError(async (req, res, next) => {
  const users = await User.find();

  res.status(200).json({
    success: true,
    users
  })
})
```

Funkcija dohvaća sve korisnike iz baze podataka koristeći metod `User.find()`. Nakon što su korisnici dohvaćeni, šalje odgovor s statusom 200 i JSON objektom koji sadrži listu korisnika u svojstvu `users`, označavajući uspješnost operacije dohvaćanja. Ovaj endpoint je namijenjen administratorima kako bi dobili pregled svih korisnika registriranih u sistemu.

## Sve narudzbe

Show entries

10

Search

ID NARUDZBE	BROJ PROIZVODA	IZNOS	STATUS	AKCIJA
654698976b42aea413c2168b	1		Delivered	
654977a747627ab53c79d1bb	1		Processing	
654f93a2e300a8582faf6b32	1		Processing	
ID NARUDZBE	BROJ PROIZVODA	IZNOS	STATUS	AKCIJA

Showing 1 to 3 of 3 entries

Previous 1 Next

Slika 19. All Orders (Admin)





Stranica products sadrži listu svih proizvoda iz baze, njihov ID, naziv, cijena i akciju. Ova lista ima dvije akcije koje vam nude mogućnost da admin uredi ili da obriše proizvod. Klikom na uredi vodi na novu stranicu, a dok klikom na obriši obriše proizvod iz baze podataka.

## Svi proizvodi

Show entries

10

Search

No	ID	IME	PRICE	AKCIJA
1	6557a76782fcecadafcfa951	Pire	5 BAM	 
2	655e505f7128eadaeffa8511	Čevapi	8.5 BAM	 
No	ID	IME	PRICE	AKCIJA

Showing 1 to 2 of 2 entries

Previous 1 Next

Slika 20. All Products (Admin)

```
// Get all products(Admin) => /api/v1/admin/products
exports.getAdminProducts = catchAsyncErrors(async (req, res, next) => {
  const products = await Product.find()

  res.status(200).json({
    success: true,
    products
  })
})
```

Ova funkcija dohvaća sve proizvode iz baze podataka korištenjem metode `Product.find()`. Nakon što su proizvodi dohvaćeni, šalje odgovor s statusom 200 i JSON objektom koji sadrži listu proizvoda u svojstvu `products`, označavajući uspješnost operacije dohvaćanja. Ovaj endpoint namijenjen je administratorima kako bi dobili pregled svih proizvoda dostupnih u sistemu.

U sidebar-u imamo padajući meni koji ima dvije opcije a jedna od njih je dodavanje novog proizvoda. Klikom na “Novi” prelazimo na novu stranicu gdje dobijamo formu za unos proizvoda. Admin unosi naziv, cijenu, opis, kategoriju i sliku proizvoda.

**NOVI PROIZVOD**

**NAZIV**

**CIJENA**

 **BAM**

**OPIS**

**KATEGORIJA**

Pizza ▼

**SLIKA**

Izaberi sliku Browse

**KREIRAJ**

Slika 21. Create New Product (Admin)

```
// Create new product => /api/v1/admin/product/new
exports.newProduct = catchAsyncErrors(async (req, res, next) => {
  let images = []
  if (typeof req.body.images === 'string') {
    images.push(req.body.images)
  } else {
    images = req.body.images
  }
}
```

```

let imagesLinks = []
for (let i = 0; i < images.length; i++) {
  const result = await cloudinary.v2.uploader.upload(images[i], {
    folder: 'products'
  })
  imagesLinks.push({
    public_id: result.public_id,
    url: result.secure_url
  })
}

req.body.images = imagesLinks

req.body.user = req.user.id;

const product = await Product.create(req.body);

res.status(201).json({
  success: true,
  product
})
})

```

Ova funkcija prima zahtjev za kreiranje novog proizvoda. Prvo, provjerava se tip slika koje su poslone s zahtjevom te se pripremaju za obradu. Zatim se slike učitavaju na Cloudinary servis jedna po jedna. Nakon uspješnog učitavanja, dobijaju se linkovi ka tim slikama i čuvaju se kao niz objekata koji sadrže javne ID-ove i URL-ove slika.

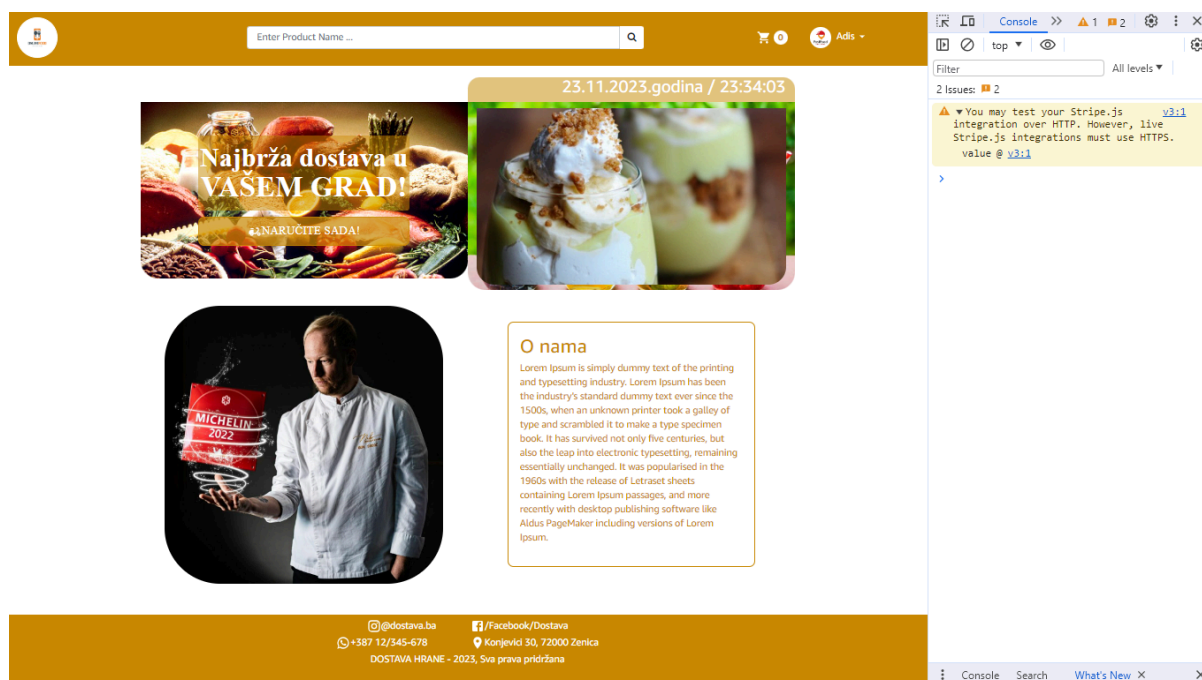
Zatim se informacije o slikama ažuriraju u zahtjevu (`req.body.images`). Korisnik koji je kreirao proizvod postavlja se na osnovu ID-a trenutno prijavljenog korisnika (`req.user.id`). Konačno, koristeći model `Product`, kreira se novi unos u bazi podataka sa informacijama o novom proizvodu koje su poslone u zahtjevu.

Nakon uspješnog kreiranja proizvoda, šalje se odgovor s statusom 201 i JSON objektom koji sadrži informacije o novokreiranom proizvodu označene sa `product`. Ovaj endpoint je namijenjen administratorima za dodavanje novih proizvoda u sistem.

## 11. TESTIRANJE APLIKACIJE

Testiranje aplikacija je ključan korak u razvoju softvera koji osigurava kvalitetu, pouzdanost i funkcionalnost aplikacija prije nego što se isporuče korisnicima. Testiranje je neophodno jer pomaže u identifikaciji grešaka, i nedostataka u softveru, čime se smanjuje rizik od lošeg korisničkog iskustva i potencijalnih problema koji mogu dovesti do gubitka podataka ili sigurnosnih propusta.

Nakon dodavanja svake od funkcionalnosti pojedinačno, urađeno je testiranje funkcionalnosti. Postoje mnoga ograničenja, posebno kada je u pitanju unos podataka, te se vodilo računa da se sva ograničenja testiraju u fazi parcijalnog testiranja.



Slika 22. Testiranje funkcionalnosti aplikacije

Konačno, nakon kodiranja cijelog sistema, odnosno nakon završetka kodiranja i testiranja koda za svaki segment sistema, izvršeno je završno testiranje, gdje je testiran sistem u cjelini i sve funkcionalnosti aplikacije, te ako je pronađena greška, urađene su odgovarajuće ispravke u kodu.

## 12. ODRŽIVOST APLIKACIJE

Aplikacija za naručivanje hrane “Naruči.ba” kreirana u svrhu ovog projekta ima veliki potencijal za upotrebu u stvarnom svijetu kao startup aplikacija. Naravno, uvijek ima prostora za određene popravke, ali i prostora za nadogradnje kad su u pitanju već postojeće ili nove funkcionalnosti, ali i dizajn same aplikacije.

Sa sigurnosnog stanovišta, nadogradnja koja bi bila ključna je funkcionalnost validacije prilikom login-a na aplikaciju. Korisnik u sistemu, unutar baze podataka, ima informacije o svojoj email adresi, a može se nadograditi na način da se unosi i broj telefona. Prilikom kreiranja profila (registracije) korisnik bi mogao imati opciju odabira validacije putem SMS poruke na uneseni broj telefona, na način da dobije poruku o nedavnoj prijavi na sistem, dok bi se notifikacija putem e-mail adrese podrazumijevala. To sve podrazumijeva kreiranje i konfiguraciju servisa za notifikacije putem e-maila i SMS poruka.

Jedna od mogućnosti je i implementacija funkcije "Two-factor authentication" (koja bi bila opcionalna i implementirana u zavisnosti od saglasnosti korisnika), pri čemu prijava na aplikaciju ne bi bila moguća bez potvrde korisnika u vidu otvaranja linka koji bi bio poslan na e-mail korisnika prilikom prijavljivanja.

Funkcionalnosti Admin Dashboard-a ostavljaju prostora za mnogo dalje nadogradnje i poboljšanja. Da bi uopšte ovaj sistem funkcionisao i aplikacija postala startup, ključni su restorani, pa bi se trebalo naći rješenje kako zapravo koristiti sve te restorane u okviru aplikacije i koji bi bio najbolji put kojim bi trebalo ići a da se ispune svi potrebni uvjeti. Uz to, trebao bi se uvesti i sistem plaćanja koji sa sobom nosi još mnoge sigurnosne uslove, ali i tehničke.

Prioritet svih ovih ideja za dalji razvoj bi bio osigurati adekvatan hosting samoj web aplikaciji, te poraditi na izradi plana za realizaciju korištenja narudžbi u okviru sistema, kao i osigurati korisniku funkciju uklanjanja narudžbi od strane korisnika.



### 13. ZAKLJUČAK

U cilju olakšanja procesa naručivanja hrane te unaprjeđenja iskustva korisnika, razvijena je aplikacija za naručivanje hrane. Ova aplikacija, zasnovana na tehnologijama React.js i Node.js, ima za cilj pružiti intuitivno korisničko iskustvo, omogućujući korisnicima jednostavan pristup meniju, brzo dodavanje proizvoda u košaricu i jednostavan proces naručivanja.

Razvoj aplikacije susreo se s određenim tehničkim izazovima, uključujući sigurnost, performanse i skalabilnost sistema. Ti izazovi su prevladani kroz pažljivo planiranje i implementaciju, osiguravajući stabilnost i pouzdanost aplikacije.

Korisničko iskustvo je ključno za uspjeh ove aplikacije. Intuitivno sučelje, brzi procesi naručivanja i povratne informacije korisnicima igraju važnu ulogu u osiguravanju pozitivnog iskustva.

Budući razvoj aplikacije može uključivati integraciju novih značajki poput praćenja narudžbi u stvarnom vremenu, unaprjeđenje korisničkog sučelja ili dodavanje algoritama preporuka hrane.

U cijelosti, aplikacija za naručivanje hrane nije samo tehnološko rješenje već i iskustvo koje pojednostavljuje svakodnevni život korisnicima i pruža potencijal za unaprjeđenje poslovanja u sektoru hrane i uslužne industrije. Njena budućnost leži u kontinuiranom unaprjeđenju kako bi se zadovoljile potrebe korisnika te unaprijedio kvalitet usluge i iskustva.

## 14. LITERATURA

### Linkovi

1. <https://successive.cloud/what-you-need-to-know-about-modern-application-development/#:~:text=Conclusion,development%20principles%20and%20best%20practices> (dostupno 24.11.2023)
2. <https://code.visualstudio.com/docs/editor/extension-marketplace> (dostupno 24.11.2023)
3. <https://survey.stackoverflow.co/2023/#most-popular-technologies-new-collab-tools> (dostupno 24.11.2023)
4. <https://www.mongodb.com/docs/compass/current/> (dostupno 24.11.2023)
5. <https://www.postman.com/company/about-postman/> (dostupno 24.11.2023)
6. <https://blog.hubspot.com/website/react-js#:~:text=js%3F-,React.,reusable%20piece%20of%20HTML%20code> (dostupno 24.11.2023)
7. <https://www.reactjsindia.com/blog/node-js-vs-reactjs-comparison-which-to-choose-for-your-js-project/> (dostupno 24.11.2023)
8. <https://nodejs.org/en/about> (dostupno 24.11.2023)
9. <https://www.usability.gov/how-to-and-tools/methods/use-cases.html#:~:text=Elements%20of%20a%20Use%20Case&text=Actor%20%E2%80%93%20anyone%20or%20anything%20that,system%20to%20achieve%20a%20goal> (dostupno 24.11.2023)
10. <https://www.usability.gov/how-to-and-tools/methods/use-cases.html#:~:text=Elements%20of%20a%20Use%20Case&text=Actor%20%E2%80%93%20anyone%20or%20anything%20that,system%20to%20achieve%20a%20goal> (dostupno 24.11.2023)
11. <https://online.visual-paradigm.com/drive/#diagramlist:proj=0&diagram=list> (dostupno 24.11.2023)
12. <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/> (dostupno 24.11.2023)

### Knjige

1. FullStack React The Complete Guide to ReactJS and Friends - Anthony Accomazzo, Nate Murray, Ari Lerner, Clay Allsopp, David Gutman, and Tyler McGinnis
2. Beginning Node.js, Express & MongoDB Development - Greg Lim
3. JavaScript and JQuery: Interactive Front-End Web Development - Jon Duckett