



UNIVERZITET U BIHAĆU
TEHNIČKI FAKULTET
ODSJEK: ELEKTROTEHNIKA
SMJER: RAČUNARSTVO I INFORMATIKA

Napredne tehnike internet programiranja

PROJEKTNI ZADATAK
TEMA: E-COMMERCE SECOND HAND

Profesor: Van. prof. dr. Jasna Hamzabegović
Asistent: MA Zinaid Kapić

Student:
Azra Bajrektarević, 1223

Bihać, april 2025. godine

Sažetak

U savremenom razvoju web aplikacija sve više se koristi pristup CSR (eng. *Client-Side Rendering*), koji omogućava dinamično i responzivno korisničko iskustvo. Za razliku od tradicionalnog server-side pristupa, CSR omogućava da se sadržaj renderuje direktno na korisnikovom uređaju, čime se smanjuje opterećenje na serveru i omogućava brža interakcija u stvarnom vremenu. U skladu sa ovim pristupom, razvijena je e-commerce aplikacija za online prodaju polovne robe. Frontend aplikacije razvijen je korištenjem React biblioteke, što omogućava brzo učitavanje stranica i interaktivno korisničko iskustvo, dok je backend razvijen pomoću Node.js i Express frameworka, koji pružaju sigurnu i skalabilnu osnovu za upravljanje proizvodima, korisnicima i narudžbama. Za pohranu podataka korištena je MongoDB baza podataka, koja omogućava fleksibilnu strukturu dokumenata i efikasno upravljanje podacima. Ključne funkcionalnosti aplikacije uključuju registraciju i prijavu korisnika, pregled i filtriranje proizvoda, dodavanje proizvoda u korpu, upravljanje narudžbama i responzivni dizajn koji omogućava korištenje aplikacije na različitim uređajima. Ova aplikacija predstavlja savremeno rješenje koje objedinjuje funkcionalnost i jednostavnost upotrebe, kako za krajnje korisnike, tako i za administratore i prodavače.

Ključne riječi: CSR, Node.js, Express, React, MongoDB

Abstract

In modern web application development, the Client-Side Rendering (CSR) approach is increasingly being used to provide users with a dynamic and responsive experience. Unlike the traditional server-side approach, CSR allows content to be rendered directly on the user's device, reducing server load and enabling faster real-time interaction. In line with this approach, an e-commerce application for the online sale of second-hand goods was developed. The frontend of the application was developed using the React library, enabling fast page loading and interactive user experience, while the backend was built with Node.js and the Express framework, providing a secure and scalable foundation for managing products, users, and orders. MongoDB was used as the database for data storage, allowing flexible document structure and efficient data management. Key features of the application include user registration and login, product browsing and filtering, adding items to the cart, order management, and a responsive design that allows usage across various devices. This application represents a modern solution that combines functionality and ease of use, both for end-users and for administrators and sellers.

Keywords: CSR, Node.js, Express, React, MongoDB

Sadržaj

1	Uvod	1
2	Pregled trenutnog stanja	2
2.1	Opis problema	2
2.2	Slični primjeri	2
3	Modeliranje aplikacije	4
3.1	Opis aplikacije	4
3.2	Strukturalni UML dijagrami	4
3.2.1	Klasni dijagram	5
3.3	Bihevijoralni UML dijagrami	6
3.3.1	Dijagrami slučajeva korištenja	6
3.3.2	Sekvencijalni dijagram	8
4	Implementacija	10
4.1	Tehnologija izrade aplikacije	10
4.2	Node JS i Express JS	10
4.3	MongoDB	10
4.4	React	11
4.5	MERN arhitektura	11
4.6	ODM	11
4.7	REST Api	12
5	Analiza rada aplikacije	13
5.1	Opis slučajeva korištenja za korisnika	13
5.2	Opis slučajeva za admina	19
5.3	Testiranje API-a	22
6	Zaključak	24

Popis slika

1	Klasni dijagram	6
2	Dijagram slučajeva korištenja	7
3	Sekvencijalni dijagram za korisnika	8
4	Sekvencijalni dijagram za admina	9
5	MERN arhitektura	11
6	Prijava na sistem	13
7	Prijava na sistem netačna	14
8	Početna stranica	14
9	Prikaz stranice za pregled proizvoda	15
10	Prikaz korpe	17
11	Prikaz narudžbe	18
12	Prijava na sistem za admina	19
13	Početna stranica nakon prijave na sistem	20
14	Stranica za dodavanje proizvoda	21
15	Stranica brisanje proizvoda	22
16	Zahtjev za dodavanje proizvoda	22
17	Zahtjev za listu proizvoda	23

1 Uvod

Projekt web aplikacije Used Up razvijen je s ciljem da unaprijedi iskustvo online kupovine polovnih proizvoda, s fokusom na jednostavnost korištenja, sigurnost i moderni dizajn. Korištenjem JavaScripta kao osnovnog jezika, te tehnologija poput Node.js i Express za serversku stranu aplikacije, omogućena je brza i skalabilna obrada korisničkih zahtjeva. Frontend aplikacije razvijen je pomoću React biblioteke, koja pruža snažne mogućnosti za izgradnju responzivnih i interaktivnih korisničkih sučelja. Za pohranu podataka korišten je MongoDB, zbog svoje fleksibilnosti i mogućnosti rada sa strukturiranim i nestrukturiranim podacima. Platforma Used Up omogućava korisnicima da pretražuju i filtriraju proizvode, pregledaju detalje artikala, dodaju proizvode u košaricu, kreiraju narudžbe i upravljaju svojim korisničkim profilom. Kupovina se može vršiti prema različitim kriterijima poput kategorije, veličine i cijene, čime se osigurava efikasno i prilagođeno korisničko iskustvo. Client-Side Rendering (CSR) pristup omogućio je brzo učitavanje stranica i fluidnu interakciju, bez potrebe za stalnim osvježavanjem stranice ili čekanjem na odgovor servera. Administratorska strana aplikacije omogućava potpunu kontrolu nad upravljanjem proizvodima, korisnicima, kategorijama i narudžbama. Primijenjene su sigurnosne mjere poput email verifikacije prilikom registracije, te autentifikacije korisnika pomoću tokena, kako bi se osigurala sigurnost aplikacije i zaštita korisničkih podataka. Prvi dio rada fokusira se na analizu tržišta i definisanje problema koji se pokušava riješiti kroz aplikaciju Used Up. Također su predstavljeni slični primjeri i njihova rješenja, zajedno s analizom njihovih prednosti i nedostataka. Nakon toga slijedi detaljan opis funkcionalnosti aplikacije, struktura baze podataka i relacije između ključnih entiteta, podržani odgovarajućim dijagramima koji dodatno pojašnjavaju arhitekturu sistema. Drugi dio rada prikazuje tehnički aspekt implementacije sistema s naglaskom na korištenje MERN stacka (MongoDB, Express, React, Node.js), objašnjavajući prednosti kao što su modularnost, brzina obrade zahtjeva i jednostavno upravljanje podacima. Završni dio dokumentuje testiranje aplikacije kroz korisničke i administratorske scenarije, uključujući prijavu, rad s proizvodima, kupovinu, te testiranje API-ja pomoću alata kao što je Postman. Aplikacija Used Up pokazuje kako moderne web tehnologije mogu biti iskorištene za izgradnju funkcionalne, sigurne i vizuelno privlačne e-commerce platforme za prodaju polovnih stvari.

2 Pregled trenutnog stanja

U savremenom digitalnom dobu, online trgovina predstavlja ključni segment maloprodaje. E-commerce platforme omogućavaju brzu, jednostavnu i dostupnu kupovinu iz udobnosti doma, čime mijenjaju navike potrošača. Kada je riječ o polovnoj robi, sve više korisnika traži održivije načine kupovine, želeći uštedjeti novac, ali i doprinijeti smanjenju tekstilnog otpada. Ipak, iako interes za second-hand proizvodima raste, mnoge manje trgovine ili pojedinci nemaju tehničke kapacitete da razviju vlastite e-commerce platforme koje bi zadovoljile moderne standarde korisničkog iskustva. Ovo poglavlje istražuje trenutne izazove u second-hand e-commerce sektoru, analizira slične platforme i upoređuje njihove prednosti i nedostatke s ciljem identifikacije prostora za napredak i inovaciju kroz projekat Used Up.

2.1 Opis problema

Tržište online trgovine polovnom robom suočava se s brojnim izazovima. Iako su mnoge velike platforme razvile sisteme koji korisnicima nude jednostavnu pretragu, sortiranje i filtriranje proizvoda, manjim prodavačima – naročito onima koji prodaju unikatne polovne artikle – često nedostaje infrastruktura za kvalitetnu prezentaciju i prodaju robe. Fizičke second-hand radnje ograničene su lokacijom i radnim vremenom, dok su online alternative nerijetko tehnički zastarjele, nepregledne ili neprilagođene mobilnim uređajima. Kupci se susreću s problemima kao što su slaba organizacija kategorija, nedovoljno informacija o stanju artikla, komplikovan proces kupovine ili nedostatak korisničke podrške. Postoji izražena potreba za modernim, sigurnim i jednostavnim rješenjem koje bi povežalo prodavače i kupce na efikasan način, kroz aplikaciju koja omogućava lako dodavanje artikala, jednostavno filtriranje i personalizovano korisničko iskustvo – upravo ono što Used Up nastoji ostvariti.

2.2 Slični primjeri

Na tržištu već postoje poznate platforme za prodaju odjeće i second-hand robe, kao što su Vinted, Depop, eBay i ThredUp. Ove platforme nude širok izbor proizvoda, moderni dizajn i funkcionalnosti poput recenzija, preporuka i naprednih filtera. Prednosti:

- Profesionalno dizajnirane platforme s fokusom na korisničko iskustvo.
- Mogućnost komunikacije između kupca i prodavača.
- Mobilne aplikacije i responzivni dizajn za sve uređaje.

Nedostaci:

- Visoki troškovi oglašavanja ili provizije po prodaji.
- Kompleksnost platforme može biti prepreka za nove korisnike.

- Zasićenost tržišta velikim brojem proizvoda, što otežava vidljivost manjih prodavača.
- Neki korisnici prijavljuju lošu korisničku podršku i probleme pri povratu proizvoda.

3 Modeliranje aplikacije

Za uspješnu izradu aplikacije važno je prethodno napraviti detaljno planiranje. To podrazumijeva jasno razumijevanje ciljeva aplikacije, potreba korisnika, problema koje rješava te načina na koji će funkcionalnosti biti implementirane. Prije same implementacije, koristi se faza modeliranja sistema putem UML dijagrama kako bi se dobio jasan uvid u arhitekturu i tokove unutar aplikacije. Ovi dijagrami omogućavaju bolju organizaciju rada, posebno u timskim projektima, jer članovima tima olakšavaju razumijevanje logike aplikacije i međusobnih veza, bez potrebe da poznaju tehnologije u kojima je aplikacija razvijena. Tokom planiranja aplikacije Used Up, izrađeni su različiti UML dijagrami – kako strukturni tako i dinamički – koji omogućavaju bolju vizualizaciju ključnih funkcionalnosti sistema, odnosa između entiteta te interakcija korisnika sa aplikacijom. Ovakav pristup doprinosi kvalitetnijem razvoju i smanjuje mogućnost grešaka.

3.1 Opis aplikacije

Used Up je web aplikacija za online prodaju polovne robe. Korisnicima omogućava da na jednostavan način postavljaju proizvode koje žele prodati, pregledavaju dostupne artikle, koriste filtere za pretragu (kategorija, cijena, stanje proizvoda, naziv), te da izvrše kupovinu na siguran način. Korisnik može svaki proizvod dodati u korpu i započeti proces narudžbe. Za kreiranje narudžbe, potrebno je da korisnik bude registrovan i da potvrdi svoj email. Svaka narudžba ima svoj status koji korisnik može pratiti (npr. na čekanju, poslano, isporučeno). Pored kupaca, sistem podržava i ulogu prodavača koji mogu samostalno dodavati proizvode uz slike i opis. Administrator ima mogućnost upravljanja korisnicima, pregledom i izmjenom narudžbi, dodavanjem novih kategorija i kontrolom sadržaja. Cilj aplikacije je da pruži jednostavno i sigurno okruženje za prodaju i kupovinu polovnih stvari, omogućavajući bolju vidljivost lokalnim prodavačima i doprinos održivoj potrošnji.

3.2 Strukturalni UML dijagrami

Strukturalni UML (Unified Modelling Language) dijagrami prikazuju statičke aspekte sistema, uključujući klase, objekte, komponente i odnose među njima. Ovi dijagrami daju jasan pregled arhitekture aplikacije Used Up, olakšavajući razumijevanje načina na koji su elementi sistema organizovani. Korišteni su sljedeći strukturalni UML dijagrami:

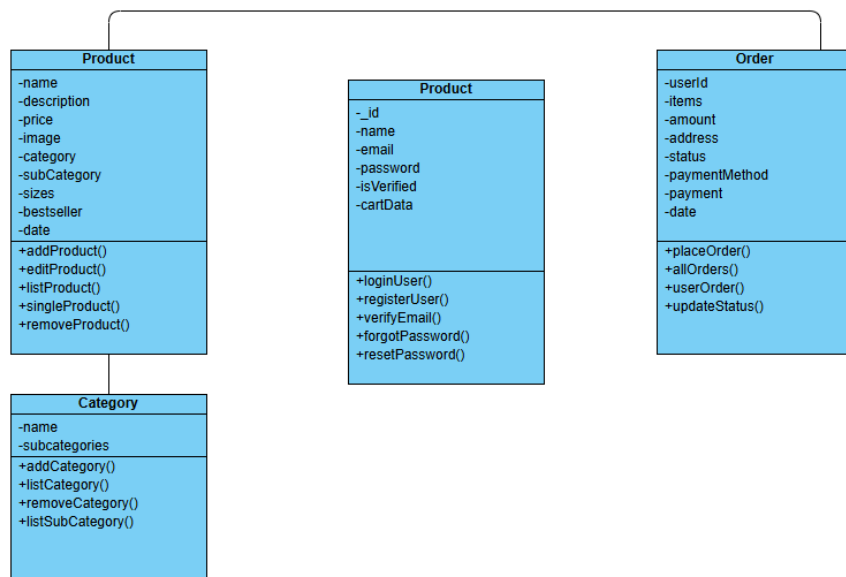
- Dijagram klasa (Class Diagram) – prikazuje entitete kao što su Korisnik, Proizvod, Narudžba, Kategorija itd., te njihove međusobne veze (nasljeđivanje, relacije).
- Dijagram komponenti (Component Diagram) – daje pregled glavnih logičkih modula sistema kao što su modul za korisnike, modul za proizvode, modul za narudžbe itd.

- Dijagram objekata (Object Diagram) – prikazuje konkretne instance klasa u određenom trenutku.
- Dijagram složene strukture (Composite Structure Diagram) – koristi se za prikaz unutrašnje strukture pojedinih klasa koje se sastoje od više dijelova.
- Dijagram razmještaja (Deployment Diagram) – opisuje fizičko razmještanje komponenti na serverskoj infrastrukturi (npr. frontend na klijentskoj strani, backend na serveru, baza podataka).
- Profilni dijagram (Profile Diagram) – omogućava proširenje UML modela i prilagođavanje specifičnim zahtjevima projekta.

3.2.1 Klasni dijagram

Najčešće korišteni UML dijagram u objektno-orijentisanim softverskim sistemima je dijagram klasa, koji predstavlja osnovu modeliranja jer prikazuje statičku strukturu sistema, uključujući njegove klase, attribute, metode i međusobne odnose između klasa. Ovaj dijagram je od posebne važnosti u fazi planiranja jer pruža jasan pregled entiteta koji čine osnovu aplikacije i načina na koji su povezani, čime se olakšava implementacija baze podataka i logike na serverskoj strani, jer daje uvid u sve potrebne informacije i odnose. U okviru aplikacije Used Up, klasni dijagram obuhvata osnovne entitete koji čine sistem, uključujući korisnike sa svojim podacima, proizvode sa svim relevantnim informacijama, kategorije kojima pripadaju proizvodi, narudžbe koje korisnici kreiraju, stavke narudžbi koje čine sadržaj svake narudžbe, te poruke ili komentare koji omogućavaju komunikaciju između korisnika. Dijagram jasno prikazuje odnose između klasa, poput toga da jedan korisnik može imati više proizvoda, svaki proizvod pripada jednoj kategoriji, jedna narudžba može sadržavati više stavki, dok je svaka stavka narudžbe vezana za tačno jedan proizvod.

Klasni dijagram aplikacije prikazan je na slici 1.



Slika 1: Klasni dijagram

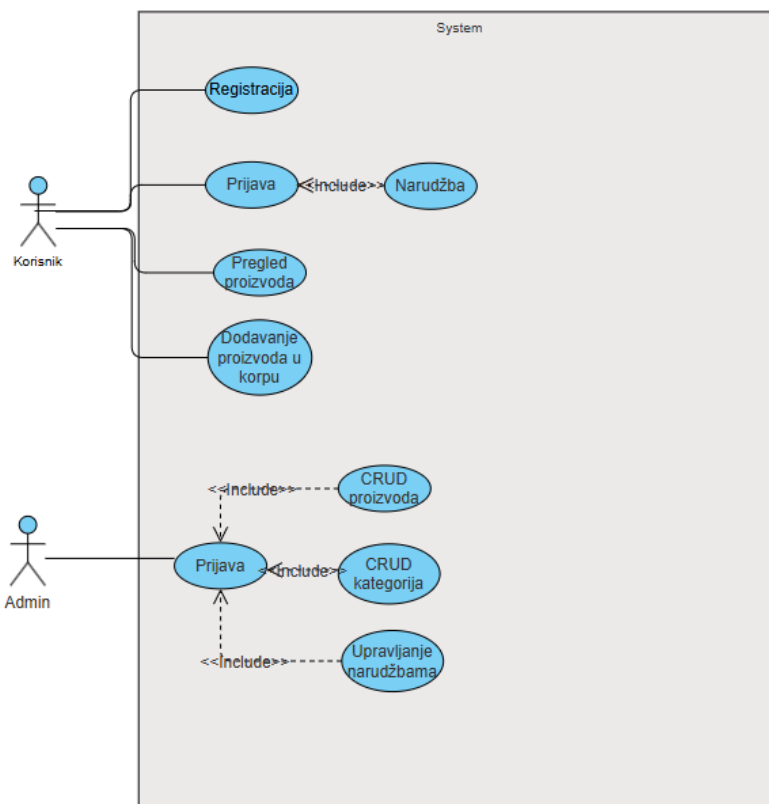
3.3 Bihevijoralni UML dijagrami

Bihevijoralni UML dijagrami predstavljaju vizualne prikaze dinamičkih aspekata sistema, pri čemu ilustruju način na koji objekti međusobno komuniciraju i ponašaju se tokom vremena kao odgovor na određene događaje. Ovi dijagrami pomažu u razumijevanju ponašanja sistema kroz različite funkcionalnosti i interakcije. U okviru razvoja aplikacije, među najznačajnijim bihevijoralnim dijagramima nalaze se dijagram aktivnosti, dijagram stanja, dijagram interakcije i dijagram slučajeva upotrebe. U nastavku su detaljnije opisani dijagram slučajeva korištenja, kao i sekvencijalni dijagram koji pripada dijagramima interakcije.

3.3.1 Dijagrami slučajeva korištenja

Dijagrami slučajeva korištenja (engl. *Use Case Diagram*) koriste se za prikazivanje funkcionalnosti sistema ili njegovih pojedinačnih dijelova. Široko se koriste za ilustraciju funkcionalnih zahtjeva sistema i njegove interakcije s vanjskim akterima. Ovaj dijagram pruža pregled različitih scenarija u kojima se sistem može koristiti, omogućavajući sagledavanje osnovnih funkcionalnosti bez ulaska u tehničke detalje implementacije.

Prikazani dijagram slučajeja korištenja razvijene aplikacije prikazan je na slici 2.

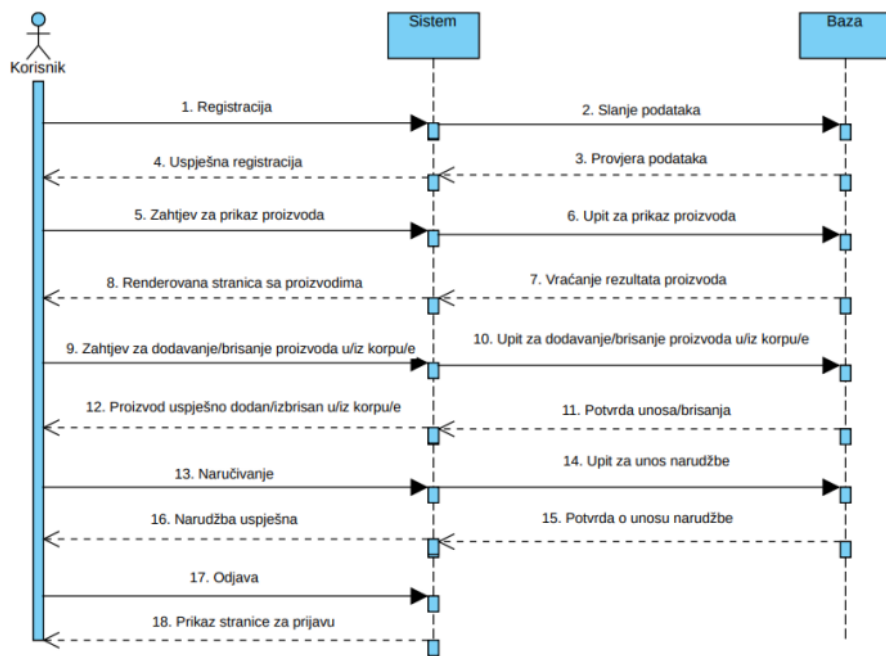


Slika 2: Dijagram slučajeja korištenja

Kao što se može vidjeti iz priloženog Use-Case dijagrama, u sistemu postoje dvije vrste korisnika sa širokim spektrom mogućnosti – korisnik i administrator. Ukoliko korisnik već posjeduje korisnički račun, može se prijaviti na sistem, dok u suprotnom prvo mora proći kroz proces registracije, a zatim se prijaviti. Prijavom korisnik dobiva pristup opciji narudžbe, dok su pregled proizvoda i dodavanje proizvoda u korpu omogućeni i neregistrovanim korisnicima. S druge strane, administrator nakon prijave dobiva pristup opcijama za upravljanje proizvodima i kategorijama putem CRUD operacija, kao i mogućnost upravljanja svim narudžbama.

3.3.2 Sekvencijalni dijagram

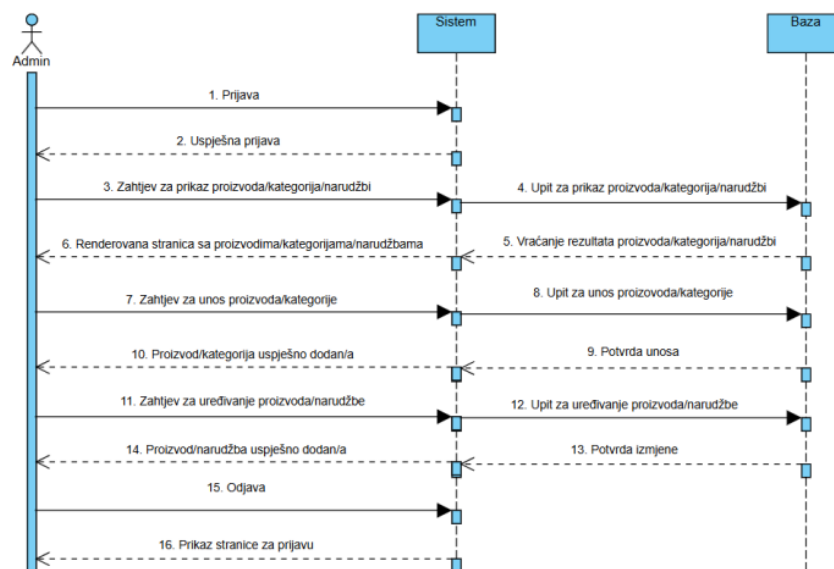
Sekvencijalni dijagram predstavlja jednostavan način prikaza interakcije između objekata u tačno određenom redoslijedu kojim se te interakcije odvijaju. Često se koristi i naziv dijagram događaja ili scenarij događaja, jer detaljno opisuje kojim tokom se odvijaju komunikacije među komponentama sistema. Sekvencijalni dijagram koji prikazuje interakcije korisnika sa sistemom prikazan je na slici 3.



Slika 3: Sekvencijalni dijagram za korisnika

Ovi dijagrami omogućavaju jasno razumijevanje kako i kojim redoslijedom objekti u sistemu funkcionišu, te su posebno korisni za programere i poslovne analitičare prilikom dokumentovanja i analiziranja zahtjeva sistema.

Sekvencijalni dijagram za admina prikazan je na slici 4.



Slika 4: Sekvencijalni dijagram za admina

Kao što se može vidjeti na slikama 3. i 4., prikazani su sekvencijalni dijagrami koji ilustruju vremenski tok radnji za admina i korisnika, odnosno ponašanje sistema tokom njihove interakcije. Dijagrami prikazuju redoslijed međusobne komunikacije između korisnika ili admina, samog sistema i baze podataka, čime se jasno vizualizuje način na koji se zahtjevi obrađuju i odgovori generišu unutar aplikacije.

4 Implementacija

Implementacija platforme Used Up koristi PHP (Laravel) za backend i HTML, CSS, JavaScript za frontend. Baza podataka je MySQL. Korisnici mogu pregledavati proizvode, dodavati ih u korpu i obaviti kupovinu, dok admin upravlja proizvodima, kategorijama i narudžbama. Fokus je na jednostavnoj navigaciji, sigurnosti plaćanja i responzivnosti.

4.1 Tehnologija izrade aplikacije

Aplikacija je razvijena u JavaScript programskom jeziku koristeći Node.js runtime okruženje i Express.js framework za upravljanje backend logikom. Express.js omogućava brzu izradu RESTful API-ja, olakšavajući rute i middleware za obradu korisničkih zahtjeva. Za frontend je korišten React, koji omogućava dinamičko renderiranje i interaktivnost na klijentskoj strani putem komponentnog pristupa. Baza podataka je MongoDB, dokumentni sistem koji je korišten za pohranu podataka u JSON formatu, pružajući fleksibilnost i skalabilnost aplikacije. Za pohranu i upravljanje korisničkim slikama, integriran je Cloudinary, usluga za spremanje i manipulaciju medijskim datotekama u oblaku. Za testiranje i slanje emailova korišten je Mailtrap, alat za sigurno i jednostavno testiranje slanja emailova tijekom razvoja aplikacije.

4.2 Node JS i Express JS

Node.js je open-source runtime okruženje koje omogućava izvršavanje JavaScript koda na serveru, temeljen na V8 JavaScript engine-u (istom koji koristi Google Chrome). Cilj Node.js-a je omogućiti JavaScript programerima da koriste isti jezik za razvoj aplikacija i na klijentskoj i na serverskoj strani. Zbog svoje sposobnosti neblokirajućeg, asinkronog izvršavanja koda, Node.js je idealan za razvoj visokop performansnih i skalabilnih aplikacija. Express.js je lagani i fleksibilni framework koji se koristi za izradu web aplikacija i RESTful API-ja u Node.js okruženju. Pruža dodatne apstrakcije koje olakšavaju razvoj složenih aplikacija, koristeći moćne mogućnosti koje nudi Node.js.

4.3 MongoDB

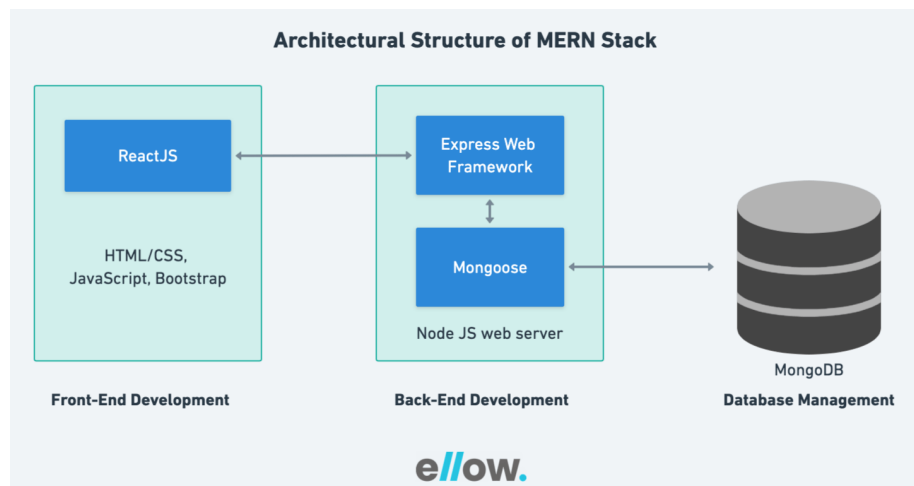
MongoDB je NoSQL baza podataka koja pohranjuje podatke u obliku dokumenata koristeći BSON (Binary JSON) format, čime omogućava visoku fleksibilnost u modeliranju podataka. Za razliku od tradicionalnih relacijskih baza podataka, MongoDB ne zahtijeva striktnu shemu, što olakšava upravljanje nestrukturiranim i polustrukturiranim podacima. Baza je dizajnirana za visoku skalabilnost i performanse, podržavajući horizontalno skaliranje putem shardinga, što omogućava distribuciju podataka na više servera. MongoDB je idealna za aplikacije koje zahtijevaju brzo rukovanje velikim količinama podataka, često u realnom vremenu, te za scenarije u kojima je potrebna fleksibilnost u pohrani i pristupu podacima.

4.4 React

React je open-source JavaScript biblioteka koja se koristi za izradu korisničkih sučelja, osobito za razvoj dinamičnih i interaktivnih web aplikacija. Razvijen od strane Facebooka, React omogućava razvoj aplikacija temeljenih na komponentama, što omogućava ponovnu upotrebu koda i bolju organizaciju aplikacije. React koristi virtualni DOM (Document Object Model) kako bi optimizirao renderiranje, čime omogućava brže ažuriranje sučelja kada se podaci promijene. Kada dođe do promjene stanja aplikacije, React prvo ažurira virtualni DOM, zatim uspoređuje promjene sa stvarnim DOM-om i ažurira samo dijelove stranice koji su se promijenili, što poboljšava performanse aplikacije.

4.5 MERN arhitektura

MERN stack se sastoji od četiri tehnologije: MongoDB, Express.js, React.js i Node.js, koje zajedno omogućavaju razvoj modernih web aplikacija. React.js se koristi za izradu korisničkog sučelja i slanje HTTP zahtjeva prema Express.js backendu. Express.js obraduje te zahtjeve i koristi Mongoose za interakciju s MongoDB bazom podataka. Node.js pokreće Express.js aplikaciju i upravlja asinkronim I/O operacijama, čime omogućava visoku skalabilnost. Ovaj stack omogućava razvoj cijele aplikacije koristeći samo JavaScript, što pojednostavljuje razvoj i poboljšava efikasnost u izradi interaktivnih i skalabilnih web aplikacija.



Slika 5: MERN arhitektura

4.6 ODM

ODM (engl. *Object-Document Mapper*) je alat koji omogućava mapiranje objekata u aplikacijskom kodu na dokumente u NoSQL bazama podataka, poput

MongoDB. Slično kao ORM (Object-Relational Mapper) koji se koristi za relacijske baze podataka, ODM omogućava aplikacijama da komuniciraju s bazom podataka koristeći objektno-orijentirani pristup, bez potrebe za pisanjem sirovih SQL upita. Glavna funkcionalnost ODM-a je omogućiti aplikaciji da koristi objekte u svom kodu, dok se podaci pohranjuju u formatu koji koristi NoSQL baza podataka (kao što su JSON-like dokumenti u MongoDB). ODM obavlja konverziju između objektnog modela aplikacije i dokumentnog modela baze podataka, što pojednostavljuje rad s podacima. Jedan od najpoznatijih ODM-a za MongoDB je Mongoose, koji omogućava definiranje modela, validaciju podataka, kao i izvođenje operacija poput dohvaćanja, umetanja, ažuriranja i brisanja podataka u MongoDB bazi koristeći JavaScript objekte. Korištenjem ODM-a, developerski timovi mogu održavati čisti objektno-orijentirani kod, dok istovremeno koriste prednosti fleksibilnosti i skalabilnosti NoSQL baza podataka.

4.7 REST Api

REST API (Representational State Transfer Application Programming Interface) je stil arhitekture za izgradnju web servisa koji omogućava komunikaciju između klijenta i servera koristeći standardne HTTP metode. REST se temelji na principima jednostavnosti, skalabilnosti i stateless komunikacije, pri čemu svaki resurs sistema, poput korisnika, proizvoda ili narudžbi, ima jedinstveni URI (Uniform Resource Identifier). Klijent koristi standardne metode poput GET, POST, PUT i DELETE za pristup, kreiranje, ažuriranje ili brisanje resursa. Podaci između klijenta i servera obično se razmjenjuju u formatima poput JSON-a ili XML-a, što olakšava njihovo razumijevanje i integraciju s različitim platformama. REST API podržava modularan dizajn, omogućujući fleksibilnost i ponovnu upotrebu koda, čime se postiže visoka interoperabilnost među različitim aplikacijama i uređajima. Ovaj pristup je široko prihvaćen zbog svoje jednostavnosti i mogućnosti skaliranja, što ga čini standardnim izborom za izgradnju modernih web i mobilnih aplikacija.

5 Analiza rada aplikacije

Analiza rada web aplikacije realizovana je kroz detaljan pregled pojedinačnih slučajeva korištenja, pri čemu su identifikovani akteri, opisani tokovi procesa i navedene povratne informacije ili akcije, koje mogu imati pozitivan ili negativan ishod.

5.1 Opis slučajeva korištenja za korisnika

Slučaj korištenja 1: Prijava na sistem

- Naziv SK: Prijava na sistem,
- Akteri SK: Korisnik,
- Učesnici SK: Akter i sistem,
- Preduslov: Postoji registrovan korisnik na sistem sa datim korisničkim računom,
- Osnovni scenarij SK:
 - Akter unosi svoju e-mail adresu i lozinku,
 - Sistem provjerava da li ima korisnik i provjerava šifru,
 - Ukoliko su podaci tačni sistem otvara početnu stranicu.

Prikaz stranice za logiranje/registraciju. 6.

UsedUp
sustainable style

HOME COLLECTION ABOUT CONTACT

Q ⓘ 🔔

Login

admin@usedup.com

[Forgot your password?](#) [Create account](#)

Login

UsedUp
sustainable style

Welcome to UsedUp! Our mission is to offer high-quality second-hand clothing, footwear, and accessories. We believe in sustainability and reuse, helping to reduce waste and protect the environment. Every piece has its own story and is now waiting for a new owner.

COMPANY

[Home](#)
[About Us](#)
[Delivery](#)
[Privacy Policy](#)

GET IN TOUCH

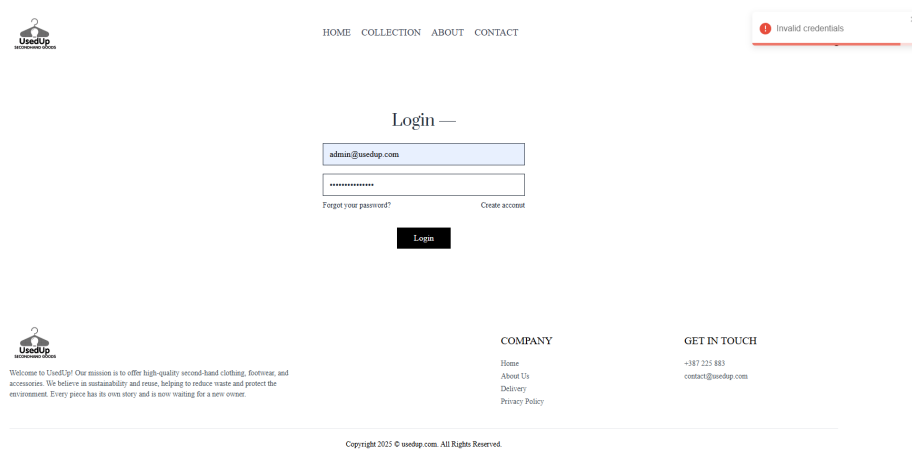
+387 225 883
contact@usedup.com

Copyright 2025 © usedup.com. All Rights Reserved.

Slika 6: Prijava na sistem

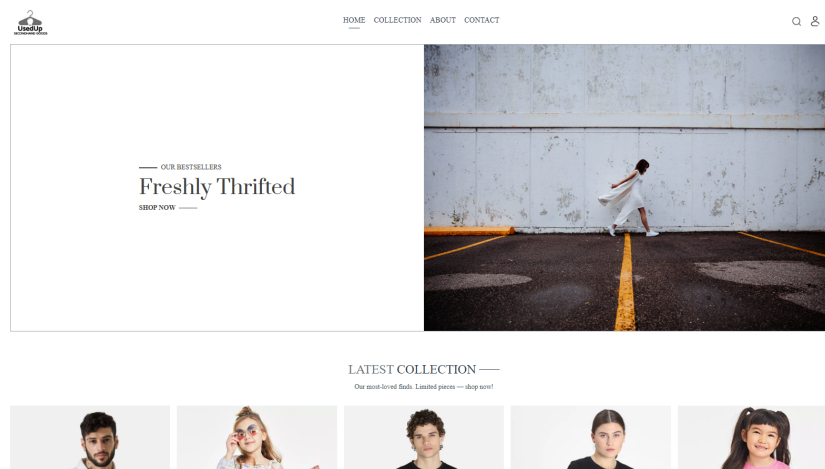
- Sporedni scenarij SK:

- Podaci za prijavu nisu ispravni,
- Ispisuje poruka prikazana na slici 7.



Slika 7: Prijava na sistem netačna

Prikaz početne stranice u slučaju uspješne prijave na sistem je predstavljen na slici 8.



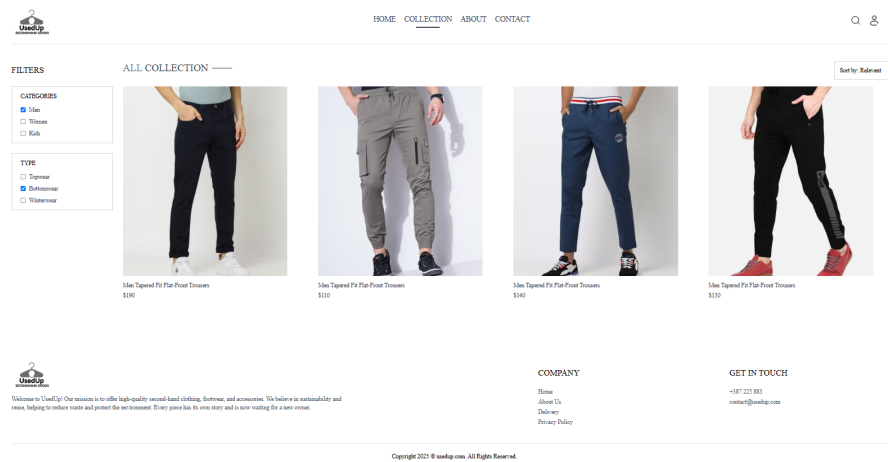
Slika 8: Početna stranica

Slučaj korištenja 2: Prikaz proizvoda

- Naziv SK: Prikaz proizvoda,
- Akteri SK: Korisnik,

- Učesnici SK: Akter i sistem,
- Preduslov: Akter je prijavljen,
- Osnovni scenarij SK:
 - Akter na navigacijskoj traci bira opciju collection,
 - Sistem ga vodi na stranicu za prikaz proizvoda

Prikaz stranice za prikaz svih proizvoda prikazan je na slici 9.



Slika 9: Prikaz stranice za pregled proizvoda

- Sporedni scenarij SK:
 - Sesija je istekla i korisnik se vodi ponovno na stranicu za prijavu.

Slučaj korištenja 3: Prikaz odabranog proizvoda

- Naziv SK: Prikaz odabranog proizvoda,
- Akteri SK: Korisnik,
- Učesnici SK: Akter i sistem,
- Preduslov: Akter je prijavljen,
- Osnovni scenarij SK:
 - Akter pregledom liste proizvoda klikom otvara na određeni proizvod,
 - Sistem prikazuje stranicu za prikaz određenog proizvoda

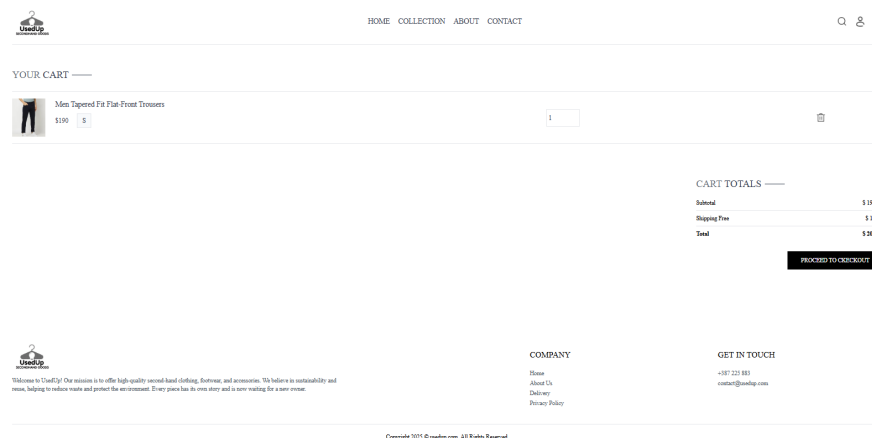
Slučaj korištenja 4: Dodavanje proizvoda u korpu

- Naziv SK: Dodavanje korisnika,
- Akteri SK: Korisnik,
- Učesnici SK: Akter i sistem,
- Preduslov: Akter je prijavljen,
- Osnovni scenarij SK:
 - Akter pregledom liste proizvoda dodaje određeni proizvod u korpu za kupovinu,
 - Sistem provjerava da je korisnik prijavljen (sesija nije istekla) i dodaje proizvod u korpu
- Sporedni scenarij SK:
 - Sesija je istekla i korisnik se vodi ponovno na stranicu za prijavu.

Slučaj korištenja 5: Prikaz korpe

- Naziv SK: Prikaz korpe,
- Akteri SK: Korisnik,
- Učesnici SK: Akter i sistem,
- Preduslov: Akter je prijavljen,
- Osnovni scenarij SK:
 - Akter nakon odabranih proizvoda klikom na ikonu korpe otvara svoju listu odabranih proizvoda,
 - Sistem prikazuje stranicu za prikaz sadržaja korpe

Prikaz korpe sa odabranim proizvodima prikazana je na slici 11.



Slika 10: Prikaz korpe

- Sporedni scenarij SK:
 - Sesija je istekla i korisnik se vodi ponovno na stranicu za prijavu.

Slučaj korištenja 6: Prikaz stranice za naručivanje

- Naziv SK: Prikaz stranice za naručivanje,
- Akteri SK: Korisnik,
- Učesnici SK: Akter i sistem,
- Preduslov: Akter je prijavljen,
- Osnovni scenarij SK:
 - Akter dodavanje proizvoda u korpu bira opciju proceed to checkout,
 - Sistem ga vodi na stranicu za naručivanje.

Prikaz sistema za naručivanje prikazan je na slici 12.

UsedUp

HOME COLLECTION ABOUT CONTACT

DELIVERY INFORMATION

First name Last name

Email address

Street

City State

Zipcode Country

Phone

CART TOTALS

Subtotal	\$ 190.00
Shipping Free	\$ 10.00
Total	\$ 200.00

PAYMENT METHOD

☐ PayPal ☒ UniCredit ☐ CASH ON DELIVERY

PLACE ORDER

Slika 11: Prikaz narudžbe

- Sporedni scenarij SK:
 - Sesija je istekla i korisnik se vodi ponovno na stranicu za prijavu.

Slučaj korištenja 7: Naručivanje

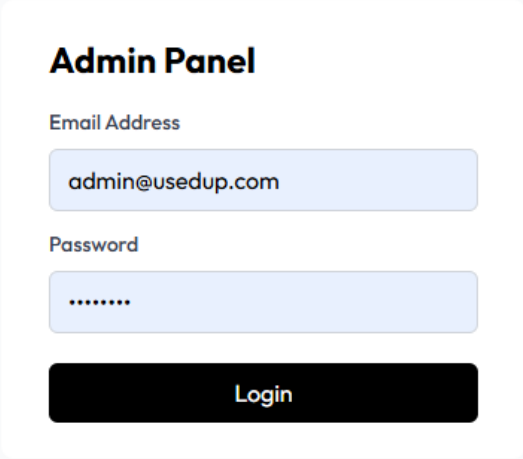
- Naziv SK: Naručivanje,
- Akteri SK: Korisnik,
- Učesnici SK: Akter i sistem,
- Preduslov: Akter je prijavljen,
- Osnovni scenarij SK:
 - Akter na stranici naručivanje bira opciju place order
 - Sistem provjerava da je korisnik prijavljen (sesija nije istekla) i vodi ga na stranicu za naručivanje
- Sporedni scenarij SK:
 - Sesija je istekla i korisnik se vodi ponovno na stranicu za prijavu.

5.2 Opis slučaja za admina

Slučaj korištenja 1: Prijava na sistem

- Naziv SK: Prijava na sistem,
- Akteri SK: Admin,
- Učesnici SK: Admin i sistem,
- Preduslov: podaci za admina su uneseni u .env fajlu,
 - Akter unosi svoju e-mail adresu i lozinku,
 - Sistem provjerava da li su unešeni podaci tačni,
 - Ukoliko su podaci tačni sistem otvara početnu stranicu.

Prikaz stranice za prijavu admina. 12.

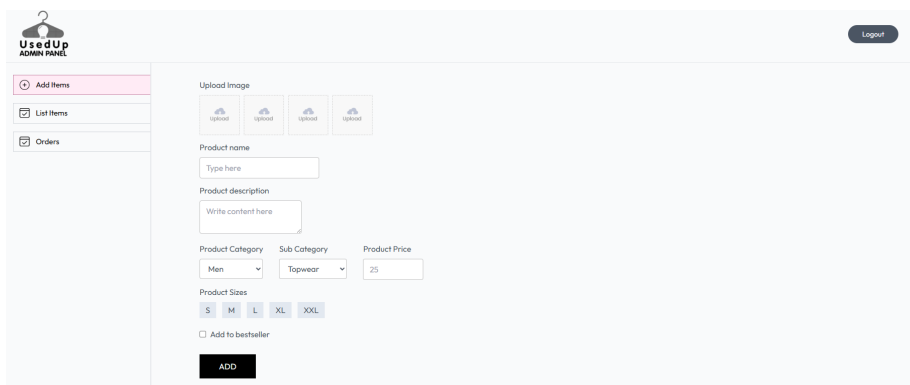


Slika 12: Prijava na sistem za admina

- Sporedni scenarij SK:

- Podaci za prijavu nisu ispravni,

Prikaz početne stranice u slučaju uspješne prijave na sistem je predstavljen na slici 14.



The screenshot displays the 'UsedUp ADMIN PANEL' interface. On the left is a sidebar with navigation links: 'Add Items' (highlighted in pink), 'List Items', and 'Orders'. The main content area is titled 'Add Items' and contains the following fields and controls:

- Upload Image:** Four 'upload' buttons arranged in a row.
- Product name:** A text input field with the placeholder 'Type here'.
- Product description:** A text area with the placeholder 'Write content here'.
- Product Category:** A dropdown menu currently showing 'Men'.
- Sub Category:** A dropdown menu currently showing 'Topwear'.
- Product Price:** A text input field containing the value '25'.
- Product Sizes:** A row of buttons labeled 'S', 'M', 'L', 'XL', and 'XXL'.
- Add to bestseller:** An unchecked checkbox.
- ADD:** A black button at the bottom of the form.

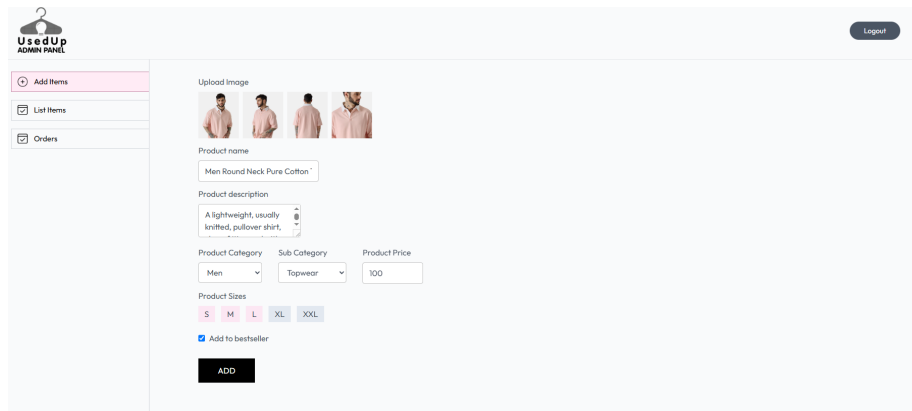
A 'Logout' button is located in the top right corner of the admin panel.

Slika 13: Početna stranica nakon prijave na sistem

Slučaj korištenja 2: Dodavanje proizvoda

- Naziv SK: Dodavanje proizvoda,
- Akteri SK: Admin,
- Učesnici SK: Admin i sistem,
- Preduslov: Admin je prijavljen,
- Osnovni scenarij SK:
 - Akter na navigacijskoj traci bira opciju add items,
 - Sistem ga vodi na stranicu za dodavanje proizvoda.

Prikaz stranice za dodavanje proizvoda predstavljen je na slici 15.



The screenshot shows the 'UsedUp ADMIN PANEL' interface for adding a new product. On the left is a sidebar with navigation links: 'Add Items' (highlighted), 'List Items', and 'Orders'. The main content area contains the following fields and controls:

- Upload Image:** Four small thumbnail images of a light-colored shirt.
- Product name:** A text input field containing 'Men Round Neck Pure Cotton'.
- Product description:** A text area containing 'A lightweight, usually knitted, pullover shirt'.
- Product Category:** A dropdown menu with 'Men' selected.
- Sub Category:** A dropdown menu with 'Topwear' selected.
- Product Price:** A text input field containing '100'.
- Product Sizes:** A row of buttons for 'S', 'M', 'L', 'XL', and 'XXL'. The 'S' button is highlighted in pink.
- Checkboxes:** A checked checkbox labeled 'Add to bestseller'.
- Buttons:** A black 'ADD' button at the bottom.

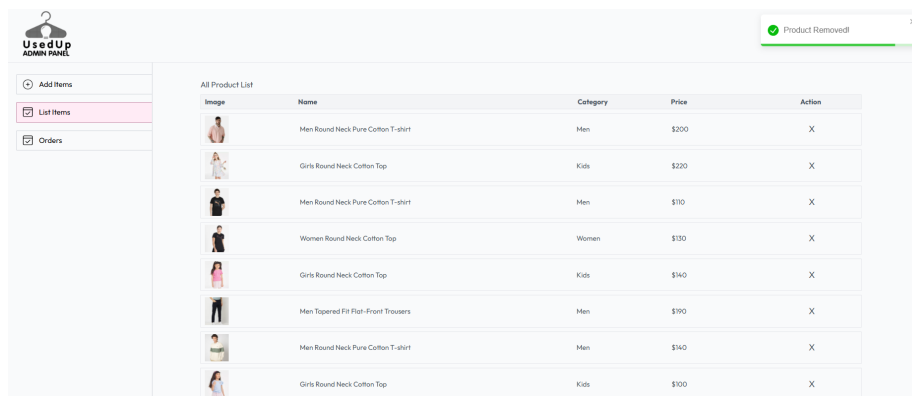
Slika 14: Stranica za dodavanje proizvoda

- Sporedni scenarij SK:
 - Sesija je istekla i korisnik se vodi ponovno na stranicu za prijavu.

Slučaj korištenja 3: Brisanje proizvoda

- Naziv SK: Brisanje proizvoda,
- Akteri SK: Admin,
- Učesnici SK: Admin i sistem,
- Preduslov: Akter je prijavljen,
- Osnovni scenarij SK:
 - Akter na stranici za prikaz proizvoda bira opciju za brisanja,
 - Sistem prikazuje poruku za uspješno brisanje.

Prikaz stranice za brisanje proizvoda prikazan je na slici 15.

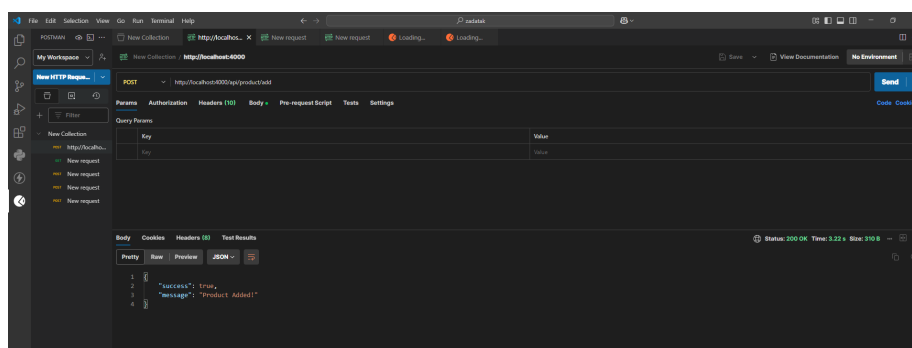


Slika 15: Stranica brisanje proizvoda

- Sporedni scenarij SK:
 - Sesija je istekla i korisnik se vodi ponovno na stranicu za prijavu.

5.3 Testiranje API-a

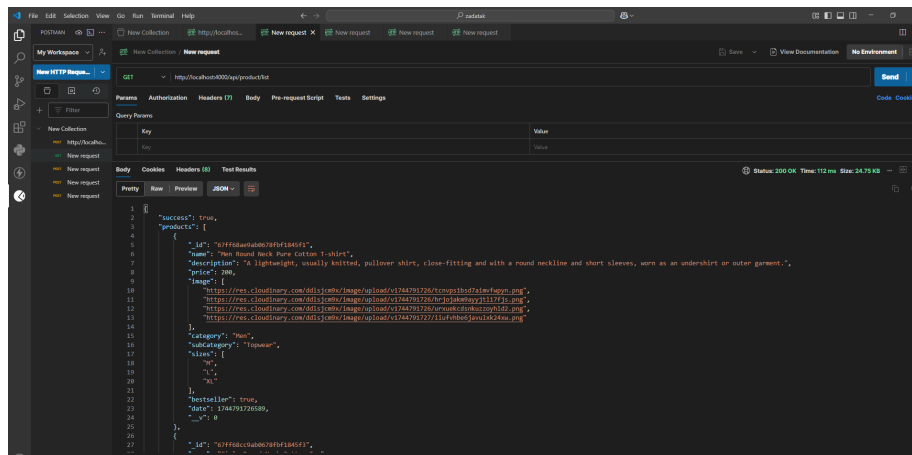
U ovom dijelu prikazan je proces testiranja API-ja, s fokusom na rute vezane za proizvode. Za testiranje je korišten alat Postman, kojim su slani HTTP zahtjevi prema serveru. Na sljedećim slikama prikazani su neki od zahjeva. Prikaz zahtjeva za dodavanje proizvoda prikazan je na slici 16.



Slika 16: Zahtjev za dodavanje proizvoda

Za dodavanje novog proizvoda korišten je POST zahtjev u Postman-u. Za ovaj zahtjev, u tijelo (body) unose se odgovarajući podaci proizvoda. Nakon slanja zahtjeva, proizvod je uspješno dodan u bazu podataka.

Prikaz zahtjeva za listu proizvoda prikazan je na slici 17.



Slika 17: Zahtjev za listu proizvoda

Za prikaz liste proizvoda korišten je GET zahtjev u Postman-u. Ovaj zahtjev omogućava da se dobiju svi proizvodi koji su trenutno pohranjeni u bazi podataka. Nakon slanja zahtjeva, odgovor u formi JSON-a sadrži sve potrebne informacije o proizvodima, čime se omogućava njihov pregled na stranici.

6 Zaključak

Razvoj aplikacije Used Up, e-commerce platforme koja omogućava korisnicima pregled i kupovinu polovne robe, a administratorima postavljanje proizvoda i upravljanje narudžbama, predstavlja duboko razumijevanje modernih web tehnologija i njihovih mogućnosti u kreiranju funkcionalnih online trgovina. Ovaj projekt temelji se na korištenju tehnologija poput Node.js, Express, MongoDB i React, čime se osigurava efikasna organizacija podataka i poslovnih procesa unutar sistema. Korištenje MongoDB kao baze podataka omogućava fleksibilnost u radu s dokumentima i strukturama podataka, dok Mongoose ODM pojednostavljuje rad s bazama podataka, omogućujući definisanje modela podataka i obavljanje složenih operacija putem jednostavnih funkcija. Ovaj pristup olakšava manipulaciju podacima i smanjuje mogućnost grešaka u pristupu bazi podataka. MERN arhitektura primijenjena u ovom projektu omogućava jasnu podjelu odgovornosti među korištenim tehnologijama i razumijevanje interakcije između njih. React.js je korišten za izradu korisničkog sučelja, omogućavajući dinamičko prikazivanje proizvoda i interakciju s korisnicima, dok Express.js backend obrađuje zahtjeve i koristi Mongoose za interakciju s MongoDB bazom podataka. Node.js upravlja asinkronim I/O operacijama, omogućujući visoku skalabilnost i efikasno upravljanje velikim brojem korisničkih interakcija. U procesu razvoja, korišteni su dijagrami kao što su UML dijagrami za vizualizaciju strukture aplikacije i odnosa između komponenti. Dijagrami slučajeva korištenja omogućili su detaljan pregled interakcija između korisnika i sistema, dok su sekvencijalni dijagrami prikazali tok podataka unutar aplikacije. Testiranje API-ja i baze podataka bilo je ključno za osiguranje funkcionalnosti aplikacije, dok je korištenje MongoDB Atlas za upravljanje bazom podataka omogućilo stabilnost, skalabilnost i lakše testiranje novih funkcionalnosti. Zaključak koji se može donijeti iz ovog projekta jest da moderne web tehnologije poput Node.js, Express, MongoDB i React značajno olakšavaju razvoj fleksibilnih, skalabilnih i efikasnih e-commerce platformi. Node.js omogućava visoke performanse server-side aplikacija, dok Express pojednostavljuje kreiranje RESTful API-ja. React pruža dinamično i interaktivno korisničko sučelje, a MongoDB kao NoSQL baza omogućava efikasno upravljanje podacima i brzo skaliranje aplikacije. Arhitektura temeljena na MVC omogućava modularnost i brze promjene u kodu aplikacije. Na temelju iskustava stečenih u ovom projektu, može se zaključiti da su moderne tehnologije poput Node.js, MongoDB i React ključne za optimizaciju razvoja e-commerce platformi, kao što je *Used Up*. Ove tehnologije smanjuju potrebu za složenim SQL upitima i ponavljanjem koda, omogućavaju brži razvoj, lakše održavanje i bržu implementaciju novih funkcionalnosti, čime se stvara učinkovita i održiva online trgovina za prodaju polovne robe.

Literatura

- [1] MongoDB, Dostupno na linku: <https://www.mongodb.com/resources/languages/mern-stack/>
- [2] DOT Academy , Dostupno na linku: <https://www.dctacademy.com/blog/what-is-object-document-mapper-odm/>
- [3] GeeksforGeeks, Dostupno na linku: <https://www.geeksforgeeks.org/unified-modeling-language-uml-introduction/>
- [4] DEV Community, Dostupno na linku: <https://dev.to/kingsley/mern-stack-project-structure-best-practices-2adk/>