

**ELEKTROTEHNIČKA I PROMETNA ŠKOLA OSIJEK**

ZAVRŠNI RAD

**IZRADA WEB APLIKACIJE**

**Luka Baćani**

Osijek, svibanj 2024.

# **ELEKTROTEHNIČKA I PROMETNA ŠKOLA OSIJEK**

**ZAVRŠNI RAD**

## **IZRADA WEB APLIKACIJE**

Učenik: Luka Baćani  
Razred: 4TR2  
Obrazovni sektor: Elektrotehnika i računalstvo  
Zanimanje: Tehničar za računalstvo  
Mentor: Ivan Marušić

Opis zadatka:

Zadatak ovog rada je izraditi web aplikaciju za iznajmljivanje apartmana. Sastoji se od dvije web-stranice, od kojih je jedna namijenjena za korisnika koji pregledava apartmane sa svrhom iznajmljivanja i jednu za vlasnika apartmana koji treba upravljati svojim imanjima. Potrebno je izraditi bazu podataka za apartmane, goste i rezervacije, ugraditi sve funkcionalnosti, objasniti način rada i programski kod.

## Sadržaj

1. Uvod .....	5
2. Funkcionalnosti sustava .....	6
3. Korištene tehnologije .....	7
3.1. HTML, CSS.....	7
3.2. JavaScript .....	7
3.3. React.....	8
3.4. Firebase .....	11
4. Backend .....	12
4.1. Povezivanje sa Firebase sustavom .....	13
4.2. Dohvaćanje podataka iz baze .....	14
4.3. Brisanje podataka iz baze .....	15
4.4. Dodavanje unosa u bazu podataka .....	16
4.5. Uređivanje spremljenih unosa.....	17
5. Frontend .....	18
5.1. Stranica za upravljanje apartmanima .....	18
5.2. Sustav za rezervaciju .....	22
6. Zaključak .....	<b>Pogreška! Knjižna oznaka nije definirana.</b>
LITERAURA .....	26

## 1. Uvod

Predmet ovog rada je izrada sustava za rezervaciju apartmana korištenjem modernih tehnologija za razvoj web aplikacija. U radu će biti detaljno opisana funkcionalnost web stranice, korištene tehnologije, proces izrade baze podataka i integracija u web stranicu, te stvaranje same stranice.

Plan i organizacija rada uključuju nekoliko ključnih dijelova. Prvi dio rada opisuje sve potrebne funkcionalnosti web aplikacije. Njene konačne ciljeve, potrebe i način korištenja. U ovom dijelu razrađuju se elementi koje web stranica sadrži te njihovu svrhu.

Drugi dio opisuje odabir korištenih tehnologija, među kojima su React i JavaScript za frontend razvoj, te Firebase Firestore za bazu podataka. Ovaj dio također sadrži osnovne informacije o arhitekturi sustava i načinu na koji su ove tehnologije povezane.

Nakon toga slijedi implementacija svih funkcionalnosti sustava. Ovdje će biti objašnjeno kako se dodaju novi apartmani u sustav, kako se upravlja slikama i tekstualnim opisima, te kako se prikazuju apartmani korisnicima. Također implementiran je i sustav za rezervaciju koji omogućuje korisnicima da pregledaju dostupnost apartmana i rezerviraju ih za određene datume.

Nakon integracije svih sustava potrebno je dizajnirati korisničko sučelje i prikazati podatke iz baze podataka korisnicima. Ovdje će biti prikazan izgled aplikacije, interakcije sa web stranicom i proces razvoja nekih ključnih dijelova rada stranice.

Razlog za izbor ove teme leži u osobnom interesu za razvoj web aplikacija i želji za stjecanjem praktičnih znanja iz ovog područja. Ova tema je izabrana i zbog obuhvaćenosti svih područja razvoja na web-u, dizajn, frontend i backend integraciju. Nadam se da će rezultati ovog rada biti korisni ne samo meni, već i svima koji žele naučiti više o izradi kompleksnih web aplikacija.

## 2. Funkcionalnosti sustava

Cilj ovog sustava je pružiti platformu za upravljanje jedinicama apartmana i njihovo rezerviranje putem web sučelja. Sustav je dizajniran s ciljem jednostavnog i modernog korisničkog sučelja za korisnike i upravljačke funkcionalnosti kako bi zadovoljio potrebe i zahtjeve vlasnika apartmana.

Upravljačka aplikacija obuhvaća sve potrebne funkcionalnosti poput dodavanja, uređivanja i brisanja apartmana, pružajući vlasnicima mogućnost da detaljno upravljaju svojim smještajima. Kroz sučelje za upravljanje, administratori imaju uvid u raspoložive apartmane, njihove karakteristike i dostupnost, zaradu, goste i rezervacije.

Korisnički dio sustava omogućuje pregled apartmana i kreiranje rezervacija za odabrani smještaj. Kroz intuitivno sučelje, korisnici mogu pretraživati dostupne apartmane, pregledavati njihove detalje, filtrirati željeni apartman i jednostavno izvršiti rezervaciju unoseći željene datume boravka.

Korisnik sustava za rezervaciju ima slijedeće mogućnosti:

- Korisničko sučelje za pregled apartmana
- Filter za pretraživanje odgovarajućeg smještaja
- Prikaz detalja svakog apartmana
- Rezervaciju apartmana u određenom vremenskom roku

Vlasniku apartmana omogućeno je:

- Pregled svih jedinica
- Uređivanje, dodavanje i brisanje jedinica
- Pregled gostiju i rezervacija
- Izvještaj o zaradi svih apartmana

### 3. Korištene tehnologije

#### 3.1. HTML, CSS

HTML (HyperText Markup Language) je jezik za izradu strukture web stranice. Služi za strukturiranje sadržaja na web stranici pomoću elemenata kao što su naslovi, odlomci, liste, poveznice, gumbi i slike. Svaki element je označen svojom oznakom, npr. `<h1>` za naslove ili `<p>` za odlomke. HTML omogućuje dodavanje i organizaciju sadržaja na stranici, stvarajući osnovnu strukturu koju preglednik prikazuje korisnicima.

CSS (Cascading Style Sheets) je jezik za oblikovanje i izgled web stranica. Služi za definiranje stilova elemenata kreiranih uz HTML, uključujući boje, fontove, razmake i raspored. CSS se može pisati u posebnim datotekama ili u sklopu HTML-a. Služi programerima za pružanje izgleda stranici, definiranje boja, teksta, rasporeda elemenata, ponašanja pojedinih elemenata i sl.

Kombinacija HTML-a i CSS-a omogućuje izradu vizualno privlačnih i funkcionalnih web stranica koje pružaju ugodno korisničko iskustvo.

HTML i CSS pružaju osnovne funkcionalnosti u razvoju web-stranica ali kod većih projekata postaju manje efikasni. Također pružaju ograničenu mogućnost dodavanja dodatnih funkcionalnosti i upravljanja sa podacima. Zbog navedenog razloga u kombinaciji sa HTML-om i CSS-om koristimo programske ili skriptne jezike poput JavaScript i Frontend razvojnih okružja poput React-a.

#### 3.2. JavaScript

JavaScript je skriptni jezik koji se koristi za dodavanje interaktivnosti i dinamičkog sadržaja web stranicama. Dok HTML pruža strukturu, a CSS izgled, JavaScript omogućuje korisnicima da komuniciraju sa stranicom na različite načine.

Primjena JavaScript-a u razvoju web stranica uključuje:

- Manipulacija DOM-om: JavaScript omogućuje dinamičku promjenu HTML sadržaja i CSS stilova na stranici bez ponovnog učitavanja. Na primjer, može se koristiti za prikazivanje ili skrivanje elemenata, promjenu teksta ili stilova te dodavanje novih elemenata u DOM.

- **Obrada događaja:** JavaScript omogućuje reagiranje na korisničke geste poput klika, unosa teksta ili pomicanja miša. Ovo omogućuje izradu interaktivnih elemenata kao što su padajući izbornici, modali i forme koje validiraju unos u stvarnom vremenu. Kroz događaje možemo napraviti stranicu interaktivnom i povećati korisničko iskustvo.
- **Komunikacija s poslužiteljem:** JavaScript može slati i primati podatke s poslužitelja pomoću tehnologija kao što su AJAX i Fetch API, omogućujući dohvaćanje raznih podataka sa interneta i asinkronu komunikaciju sa poslužiteljima bez ponovnog učitavanja stranice. Ovo je ključno za moderne web aplikacije koje zahtijevaju učitavanje podataka u stvarnom vremenu.
- **Animacije i efekti:** JavaScript omogućuje stvaranje kompleksnih animacija i vizualnih efekata koje CSS ne može postići samostalno, čineći korisničko iskustvo bogatijim i atraktivnijim.

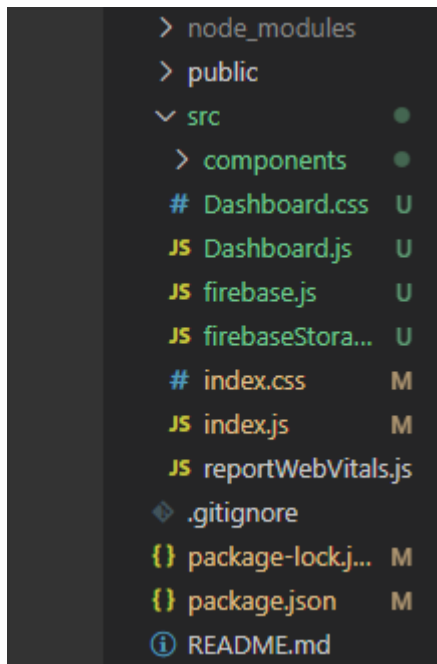
Kombinacija HTML-a, CSS-a i JavaScript-a omogućuje izradu modernih, interaktivnih i dinamičnih web stranica koje pružaju napredno korisničko iskustvo.

### 3.3. React

Frontend razvojna okružja (framework) su alati koji pomažu programerima u izradi interaktivnih i responzivnih web aplikacija. Oni pružaju strukturu i gotova rješenja za uobičajene probleme u razvoju, što ubrzava i olakšava proces izrade i održavanja web stranica.

React je jedan od najpopularnijih frontend razvojnih okružja, kojeg je razvila tvrtka Facebook. React je u osnovi JavaScript biblioteka za izgradnju korisničkih sučelja, fokusirajući se na izradu komponenti.





Slika 1: Struktura projekta

React projekt sastoji se od više različitih paketa koji mu osiguravaju funkcionalnost u mapi `node_modules`.

U mapi `public` nalaze se statične datoteke potrebne za rad aplikacije.

U mapi `src` nalazi se glavni dio koda aplikacije. U datoteci `package.json` prikazani su svi instalirani paketi i postavke aplikacije, `index.js` predstavlja prvu komponentu koja se pokreće, a ostale komponente nalaze se u mapi `components`.

Komponente su osnovni elementi u React-u zbog kojih se on upravo razlikuje od običnog HTML-a. Ukoliko želimo napraviti neki element koji ima jedan izgled ali više različitih podataka, u HTML-u bi morali pisati isti kod za svaki novi element, koji onda imaju svoje različite atribute. Korištenjem komponenata ovaj problem se rješava. Komponente su gotovi, ponovno upotrebljivi dijelovi koda koji definiraju određeni dio korisničkog sučelja sa svojim ponašanjima i podacima. Komponente mogu biti funkcijske ili klase sa svojim vlastitim izgledom, stilom i funkcionalnošću, a svaka komponenta može imati i svoje ulazne podatke nazvane `property` ili `props` (svojstva).

```

6  const Navbar = () => {
7    return (
8      <>
9        <div className='NavLeft'>
10         <div className='LeftNavLinks'>
11           <Link to="/" className='LinkStyle'><h2 style={{color: "white"}}>B</h2></Link>
12           <Link to="/apartmani" className='LinkStyle'><div className='NavLink'>
13             <FontAwesomeIcon className='NavIcon' icon={"house-chimney"}></FontAwesomeIcon>
14             <p>Apartmani</p>
15           </div></Link>
16           <Link to="/rezervacije" className='LinkStyle'><div className='NavLink'>
17             <FontAwesomeIcon className='NavIcon' icon={"book-open"}></FontAwesomeIcon>
18             <p>Rezervacije</p>
19           </div></Link>
20
21           <Link to="/gosti" className='LinkStyle'><div className='NavLink'>
22             <FontAwesomeIcon className='NavIcon' icon={"user"}></FontAwesomeIcon>
23             <p>Gosti</p>
24           </div></Link>
25
26           <Link to="/zarada" className='LinkStyle'><div className='NavLink'>
27             <FontAwesomeIcon className='NavIcon' icon={"chart-line"}></FontAwesomeIcon>
28             <p>Zarada</p>
29           </div></Link>
30         </div>
31       </div>
32
33       <div className='NavTop'>
34
35     </div>
36   </>
37 )
38 }
39
40 export default Navbar
41

```

Slika 2: Komponenta

Komponente se sastoje od deklaracije, naziva, property-a, i return bloka koda koji prikazuje našu stranicu (samo jedan element). Komponenta se naredbom `export` prosljeđuje dalje drugim elementima stranice.

React koristi virtualni DOM (Document Object Model) kako bi optimizirao ažuriranje korisničkog sučelja. Kada se dogodi promjena svojstava, React prvo ažurira virtualni DOM, a zatim uspoređuje virtualni DOM s pravim DOM-om i primjenjuje samo potrebne promjene. Ovaj proces čini React brzim i efikasnim.

Za sintaksu React koristi JSX (JavaScript XML). JSX je sintaksna ekstenzija za JavaScript koja omogućuje pisanje HTML-a unutar JavaScript koda. Ovo znači da je svaka komponenta unutar razvojnog okružja JavaScript kod i svaka datoteka ima nastavak `.js`. JSX omogućuje da u istoj datoteci pišemo kod sličan HTML-u po sintaksi za raspored i stil, i JavaScript kod za funkcionalnost što nam omogućuje laganu integraciju dodatnih mogućnosti.

React također podržava korištenje dodatnih alata i knjižnica kao što su React Router za upravljanje rutama (navigacijom), Redux za upravljanje složenim stanjem aplikacije, Fontawesome za ikonice i brojne druge vanjske knjižice za rad sa web-om.

```
18 const root = ReactDOM.createRoot(document.getElementById('root'));
19 root.render(
20   <Router>
21     <Navbar></Navbar>
22     <Routes>
23       <Route path="/" element={<Dashboard></Dashboard>}></Route>
24       <Route path="/apartmani" element={<Apartments></Apartments>}></Route>
25       <Route path="/apartmani/:id" element={<ApartmentDetails />} />
26       <Route path="/rezervacije" element={<Reservations></Reservations>}></Route>
27       <Route path="/gosti" element={<Gosti></Gosti>}></Route>
28       <Route path="/apartmani/apartman-dodaj" element={<DodajApartman></DodajApartman>}></Route>
29       <Route path="/apartmani/apartman-uredi/:id" element={<UrediApartman></UrediApartman>}></Route>
30       <Route path="/zarada" element={<Zarada></Zarada>}></Route>
31     </Routes>
32   </Router>
33 );
34
```

Slika 3: React Router

Ukratko, React i slična razvojna okružja omogućuju programerima izradu složenih, dinamičnih i responzivnih web aplikacija na učinkovit i održiv način, koristeći modularne komponente i optimizirane procese ažuriranja korisničkog sučelja.

### 3.4. Firebase

Firebase je platforma za razvoj aplikacija koja pruža alate i usluge za izgradnju, testiranje i upravljanje mobilnim, desktop i web aplikacijama. Omogućuje autentikaciju korisnika, korištenje baze podataka sa ažuriranjem u stvarnom vremenu, hosting, analitiku, obavještanje, spremnik podataka i još mnogo toga. Firebase olakšava brzo stvaranje aplikacija uz pomoć cloud infrastrukture, čime eliminira potrebu za održavanjem vlastitih servera i razvoja svog backend sustava. Nudi jednostavnu integraciju s našim aplikacijama i drugim Google-ovim alatima. U projektu korišten je Firebase Firestore i Storage.

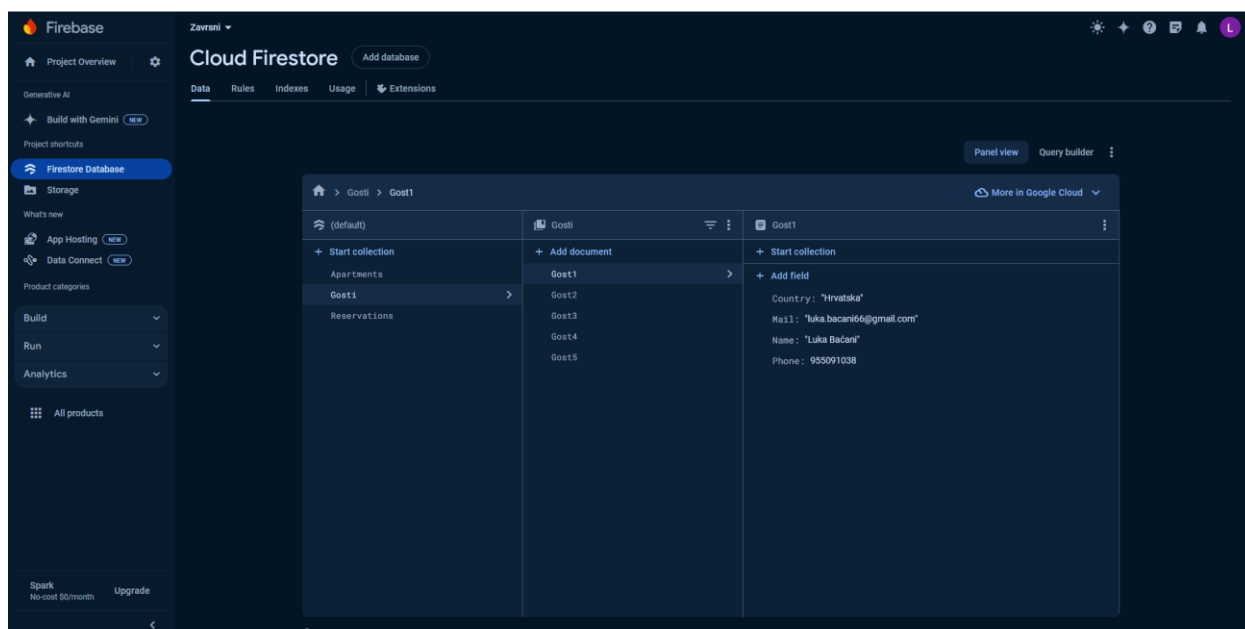
Firestore Database je fleksibilna, skalabilna baza podataka koja omogućuje pohranu i sinkronizaciju podataka u stvarnom vremenu između klijentskih aplikacija i servera. Pruža strukturirane podatke u dokumentima s podrškom za upite, transakcije i indekse. Firestore se koristi za pohranu podataka u mobilnim, desktop i web aplikacijama te omogućuje jednostavno skaliranje aplikacija bez potrebe za upravljanjem infrastrukturom servera i drugih backend rješenja. U ovom radu Firebase sprema podatke o apartmanima, gostima i rezervacijama.

## 4. Backend

U backend dio aplikacije spada interakcija s bazom podataka, čitanje podataka, spremanje podataka i njihovo upravljanje radi ostvarenja željene funkcionalnosti web stranice.

Upravljačka stranica treba imati mogućnosti prikazivanja apartmana, dodavanja apartmana i njihovo uređivanje. Također omogućuje i prikaz ukupne zarade, gostiju i rezervacija za pojedine jedinice.

Svi apartmani spremljeni su u kolekciju „Apartments“ u Firestore bazi podataka.



Slika 4: Firebase Firestore

## 4.1. Povezivanje sa Firebase sustavom

Firebase omogućava jednostavnu integraciju u React razvojno okruženje putem gotove biblioteke koju možemo uključiti u naš projekt. Koristeći funkcije iz ove knjižice možemo se spojiti na našu bazu podataka i upravljati podacima iz nje.

```
1  import { initializeApp } from "firebase/app";
2  import { getFirestore } from "firebase/firestore";
3
4  const firebaseConfig = {
5    apiKey: "AIzaSyCKgW9xEaNM1ZorWMfY1SLNpYCcCU590c8",
6    authDomain: "zavrsni-27830.firebaseio.com",
7    databaseURL: "https://zavrsni-27830-default-rtdb.europe-west1.firebaseio.com",
8    projectId: "zavrsni-27830",
9    storageBucket: "zavrsni-27830.appspot.com",
10   messagingSenderId: "447036788552",
11   appId: "1:447036788552:web:d94e008b7c6ecb5524ed7d",
12   measurementId: "G-Z506Q0ZL4R"
13 };
14
15 const app = initializeApp(firebaseConfig);
16
17 const firestore = getFirestore(app);
18
19 export default firestore;
20
21
22
```

Slika 5 Konfiguracija Firebase sustava u React-u

Datoteka `firebase.js` sadrži osnovnu konfiguraciju projekta i spajanja na Firebase sustav. Konfiguracija se onda inicijalizira (spajanje na sustav), te onda možemo dohvatiti pojedine elemente Firebase sustava, u ovom slučaju Firestore bazu podataka. Datoteku Firestore sada možemo koristiti kada trebamo upravljati našom bazom podataka

## 4.2. Dohvaćanje podataka iz baze

```
10  const [apartments, setApartments] = useState([]);
11  const [loading, setLoading] = useState(true);
12  const navigate = useNavigate();
13
14  useEffect(() => {
15    const fetchApartments = async () => {
16      try {
17        const query = await getDocs(collection(firestore, "Apartments"));
18        const apartmentsList = query.docs.map(doc => ({ id: doc.id, ...doc.data() }));
19        setApartments(apartmentsList);
20        setLoading(false);
21      } catch (e) {
22        console.error("Greška u dohvaćanju dokumenata: ", e);
23      }
24    };
25
26    fetchApartments();
27  }, []);
28
```

Slika 6: Dohvaćanje podataka iz baze

Podatke za apartmane iz baze podataka dohvaćamo preko funkcije `fetchApartments`. Oni će biti spremljeni u state za kasnije korištenje. State u React-u se može objasniti kao varijabla odnosno podatak koji ujedno ima i funkciju za svoje pohranjivanje. Pozivanjem te funkcije njegova vrijednost se mijenja, a tu vrijednost možemo koristiti bilo gdje u našoj komponenti.

Korištenjem naredbe `useEffect` naš kod će se izvršiti čim se stranica učita, te će onda asinkrono (ne utječe na ostatak stranice ako treba duže da se izvrši) dohvatiti podatke iz baze koristeći funkcije iz Firebase biblioteke `getDocs` i `docs`. Dohvaćena lista se sprema u state „`apartments`“.

Ukoliko dođe do greške pri komunikaciji `try`, `catch` blok koda će to uhvatiti i obavijestiti korisnika.

### 4.3. Brisanje podataka iz baze

Stranica za upravljanje sustavom omogućuje korisnicima da obrišu stvorene apartmane ili rezervacije.

Za izvršavanje ove funkcionalnosti potrebno je obrisati unos iz baze podataka. Nakon brisanja potrebno je osvežiti stranicu kako bi se prikazala uređena lista apartmana.

```
1  import './Apartments.css';
2  import { Link } from 'react-router-dom';
3  import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';
4  import { doc, deleteDoc } from 'firebase/firestore';
5  import firestore from '../..//firebase';
6
7  const ApartmentCardMain = ({ Apartment }) => {
8
9      const handleDelete = async () => {
10         try {
11             const apartmentRef = doc(firestore, 'Apartments', Apartment.id);
12             await deleteDoc(apartmentRef);
13             window.location.reload();
14         } catch (error) {
15             console.error('Greška u brisanju apartmana:', error);
16         }
17     };
18
19     return (
20         <div className='ApartmentCard'>
21             <Link className='LinkStyle' to={"/apartmani/" + Apartment.id}>
22                 <h3>{Apartment.Name}</h3>
23             </Link>
24             <div className="Icons">
25                 <Link to={"/apartmani/apartman-uredi/" + Apartment.id}>
26                     <FontAwesomeIcon icon={"pencil-alt"} className="EditIcon" />
27                 </Link>
28                 <FontAwesomeIcon icon={"trash"} onClick={handleDelete} className="DeleteIcon" />
29             </div>
30         </div>
31     );
32 };
33
34 export default ApartmentCardMain;
```

Slika 7: Brisanje unosa iz baze podataka

Asinkrona funkcija `handleDelete` se poziva klikom na gumb. Funkcija dohvaća dokument iz baze podataka preko funkcije `doc` i id-a apartmana. Taj unos se onda briše i ukoliko dođe do greške u brisanju, obavijestiti će korisnika.

#### 4.4. Dodavanje unosa u bazu podataka

Kao vlasnik apartmana omogućeno je dodavanje novih jedinica u web stranicu.

```
9  const DodajApartman = () => {
10    const [name, setName] = useState('');
11    const [city, setCity] = useState('');
12    const [description, setDescription] = useState('');
13    const [price, setPrice] = useState('');
14    const [image, setImage] = useState(null);
15    const [imageUrl, setImageUrl] = useState('');
16
17    const handleSubmit = async (e) => {
18      e.preventDefault();
19
20      try {
21        const storageRef = ref(storage, `images/${image.name}`);
22        await uploadBytes(storageRef, image);
23
24        const url = await getDownloadURL(storageRef);
25
26        await addDoc(collection(firestore, 'Apartments'), {
27          Name: name,
28          Grad: city,
29          Description: description,
30          Price: parseFloat(price),
31          Slika: url
32        });
33
34        alert('Apartman uspješno dodan!');
35        setName('');
36        setCity('');
37        setDescription('');
38        setPrice('');
39        setImage(null);
40        setImageUrl('');
41      } catch (error) {
42        console.error('Greška u dodavanju apartmana:', error);
43      }
44    };
45  }
```

Slika 8 Dodavanje unosa u bazu podataka

Korisnik piše podatke u formu čije se vrijednosti spremaju u state varijable. Slika apartmana dodaje se u Firebase Storage, te onda spremamo njenu poveznicu sa mreže za daljnje korištenje.



## 4.5. Uređivanje spremljenih unosa

```
14  useEffect(() => {
15    const fetchApartment = async () => {
16      try {
17        const docRef = doc(firestore, 'Apartments', id);
18        const docSnap = await getDoc(docRef);
19
20        if (docSnap.exists()) {
21          const data = docSnap.data();
22          setName(data.Name);
23          setCity(data.Grad);
24          setDescription(data.Description);
25          setPrice(data.Price);
26        } else {
27          console.log('No such document!');
28        }
29      } catch (error) {
30        console.error('Error fetching apartment:', error);
31      }
32    };
33
34    fetchApartment();
35  }, [id]);
36
37  const handleSubmit = async (e) => {
38    e.preventDefault();
39
40    try {
41      const apartmentRef = doc(firestore, 'Apartments', id);
42
43      await updateDoc(apartmentRef, {
44        Name: name,
45        Grad: city,
46        Description: description,
47        Price: parseFloat(price)
48      });
49
50      alert('Apartment updated successfully!');
51    } catch (error) {
52      console.error('Error updating apartment:', error);
53    }
54  };
55
```

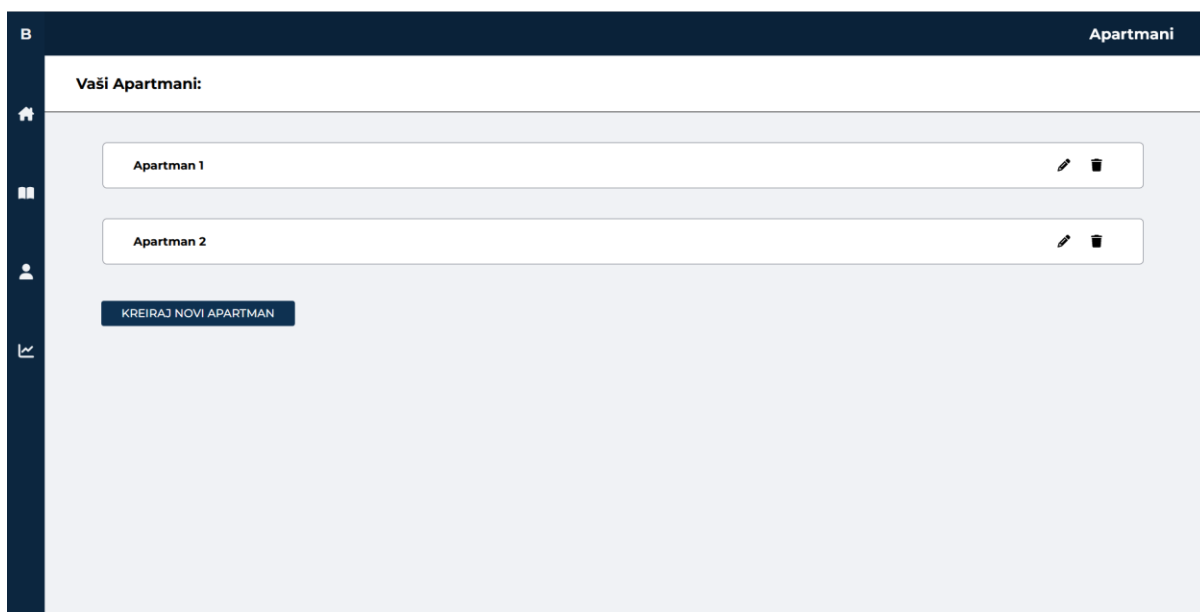
Slika 9 Uređivanje spremljenih apartmana

Preko id-a apartmana dohvaća se odgovarajući unos te upisuje u formu. Promjenom vrijednosti u formi i slanjem forme te promijenjene vrijednosti se spremaju u bazu podataka putem funkcije updateDoc.

## 5. Frontend

Nakon uspješne integracije baze podataka i spremanje potrebnih podataka slijedi dizajn stranice i primjena dostupnih podataka.

### 5.1. Stranica za upravljanje apartmanima



Slika 10 Stranica za upravljanje apartmanima

Sa lijeve strane ekrana nalazi se izbornik sa poveznicama na različite dijelove stranice.

Stranica u sebi ima konfigurirane rute za slijedeće dijelove:

- Početnu stranicu
- Apartmani (kartični prikaz, detaljni prikaz, uređivanje i dodavanje)
- Gosti
- Rezervacije
- Zarada

Navigacija u stranici rađena je uz pomoć React-Routera. Router omogućava navigaciju i prikaz različitih komponenti kroz URL web stranice. Kroz njega definiramo rute koje povezuju URL sa određenim komponentama. Odlaskom na taj URL učitava se određena komponenta.

```
const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <Router>
    <Navbar></Navbar>
    <Routes>
      <Route path="/" element={<Dashboard></Dashboard>}></Route>
      <Route path="/apartmani" element={<Apartments></Apartments>}></Route>
      <Route path="/apartmani/:id" element={<ApartmentDetails />} />
      <Route path="/rezervacije" element={<Reservations></Reservations>}></Route>
      <Route path="/gosti" element={<Gosti></Gosti>}></Route>
      <Route path="/apartmani/apartman-dodaj" element={<DodajApartman></DodajApartman>}></Route>
      <Route path="/apartmani/apartman-uredi/:id" element={<UrediApartman></UrediApartman>}></Route>
      <Route path="/zarada" element={<Zarada></Zarada>}></Route>
    </Routes>
  </Router>
);
```

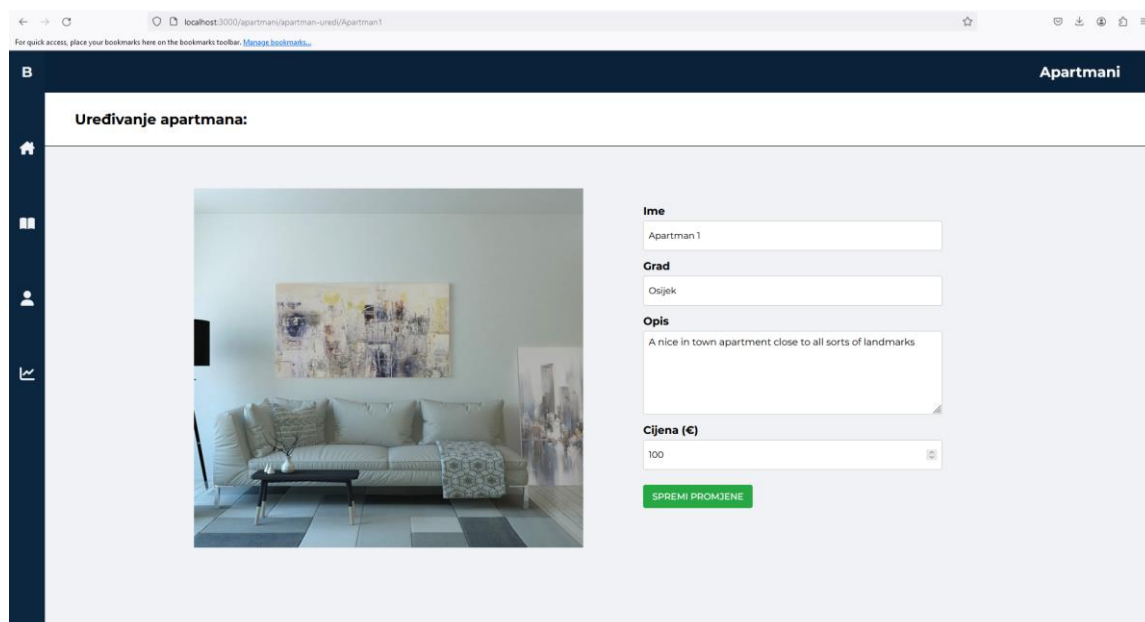
Slika 11 Rute za navigaciju

```
7 const Navbar = () => {
8   const [currentPage, setCurrentPage] = useState('Početna Stranica');
9   const changeState = (page) => {
10     setCurrentPage(page);
11   };
12
13   return (
14     <>
15       <div className="NavLeft">
16         <div className="LeftNavLinks">
17           <Link to="/" className="LinkStyle" onClick={() => changeState("Početna Stranica")}><h2 style={{ color: "#f0f2f5" }}>B</h2></Link>
18           <Link to="/apartmani" className="LinkStyle" onClick={() => changeState("Apartmani")}>
19             <div className="NavLink">
20               <FontAwesomeIcon className="NavIcon" icon="house-chimney"></FontAwesomeIcon>
21               <p>Apartmani</p>
22             </div>
23           </Link>
24           <Link to="/rezervacije" className="LinkStyle" onClick={() => changeState("Rezervacije")}>
25             <div className="NavLink">
26               <FontAwesomeIcon className="NavIcon" icon="book-open"></FontAwesomeIcon>
27               <p>Rezervacije</p>
28             </div>
29           </Link>
30           <Link to="/gosti" className="LinkStyle" onClick={() => changeState("Gosti")}>
31             <div className="NavLink">
32               <FontAwesomeIcon className="NavIcon" icon="user"></FontAwesomeIcon>
33               <p>Gosti</p>
34             </div>
35           </Link>
36           <Link to="/zarada" className="LinkStyle" onClick={() => changeState("Zarada")}>
37             <div className="NavLink">
38               <FontAwesomeIcon className="NavIcon" icon="chart-line"></FontAwesomeIcon>
39               <p>Zarada</p>
40             </div>
41           </Link>
42         </div>
43       </div>
44
45       <div className="NavTop">
46         {currentPage && (
47           <h2 style={{ color: "white" }}>{currentPage}</h2>
48         )}
49       </div>
50     </>
51   );
52 }
```

Slika 12 Navigacijska traka

Korištenjem Link komponenti možemo usmjeriti preglednik na različite URL-ove te tako prikazati različite dijelove stranice.

Poveznice mogu biti i na dinamičke URL-ove, koji će odvesti stranicu na neki URL koji mi postavimo kao varijablu. Ovo je korisno kod dijelova poput uređivanja apartmana.

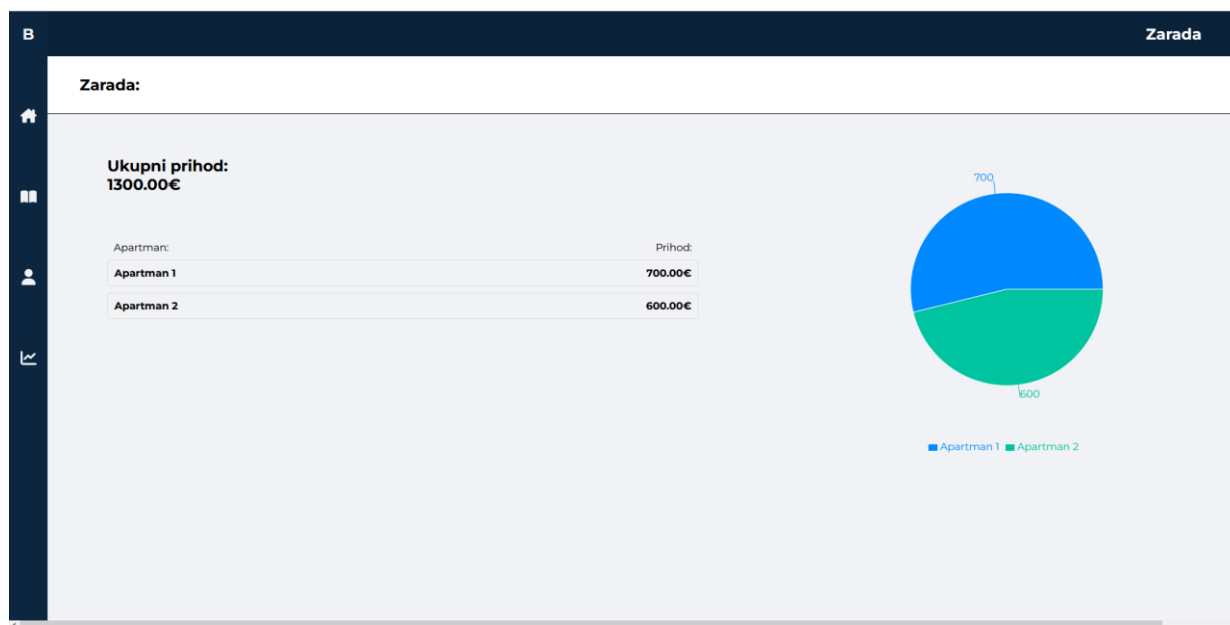


The screenshot shows a web browser window with the address bar displaying 'localhost:3000/apartmani/apartman-uredi/Apartman1'. The page has a dark blue header with the word 'Apartmani' on the right. A sidebar on the left contains several icons. The main content area is titled 'Uređivanje apartmana:' and features a large image of a modern living room with a sofa and a large abstract painting. To the right of the image is a form with the following fields: 'Ime' (containing 'Apartman 1'), 'Grad' (containing 'Osijek'), 'Opis' (containing 'A nice in town apartment close to all sorts of landmarks'), and 'Cijena (€)' (containing '100'). Below the form is a green button labeled 'SPREMI PROMJENE'.

Slika 13 Forma za uređivanje apartmana

Klikom na ikonicu za uređivanje na kartici apartmana otvara se novi prozor koji detaljnije prikazuje njegove podatke. Preko URL-a otvorili smo „`apartmani/apartman-uredi/Apartman1`“. Ova pod-domena otvara prozor za uređivanje, te onda preko id-a koji smo proslijedili u URL dohvaća podatke za odgovarajući apartman. Klikom na gumb spremi promjene, naši zapisi će se spremiti u bazu podataka.

React Router je jedan od primjera vanjskih biblioteka koje nam omogućavaju određene funkcionalnosti. Još jedan primjer takvih korištenih biblioteka je „PieChart“.



Slika 14 Stranica zarade

Na stranici zarade prikazan je ukupan prihod i prihod pojedine jedinice izračunat temeljem broja dana rezervacija na svakom smještaju.

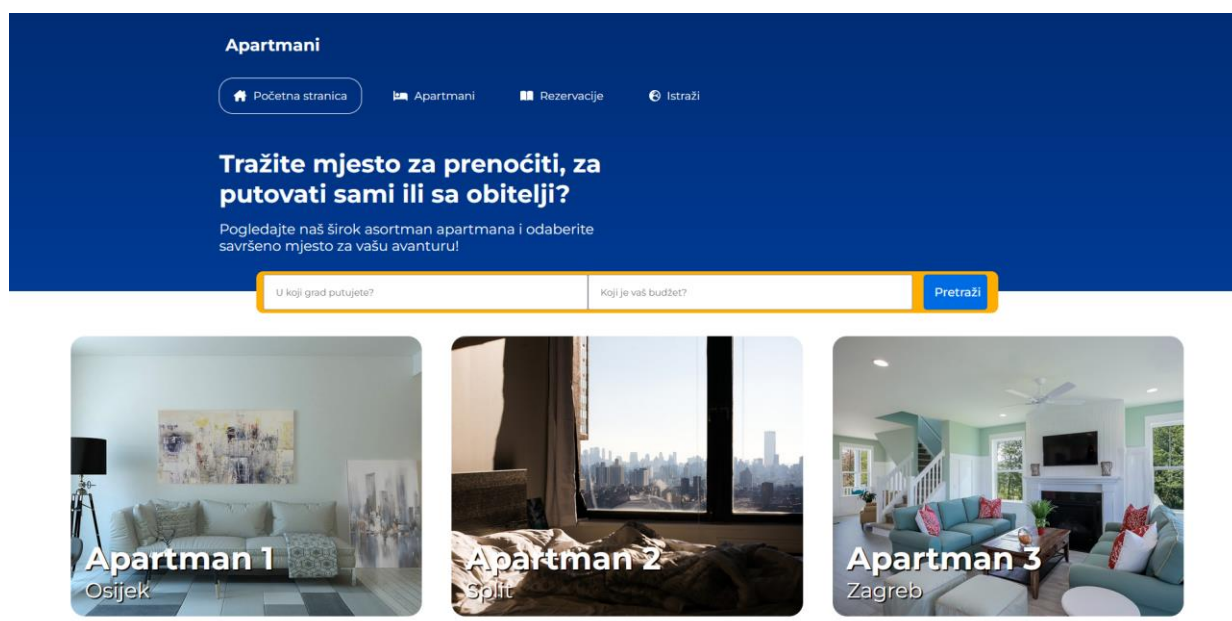
Ovi podatci prikazani su na tortastom dijagramu iz biblioteke „PieChart“ koja nam omogućava stvaranje ovakvih dijagrama korištenjem gotovih blokova koda.

```
<PieChart width={500} height={500}>
  <Pie
    data={pieData}
    dataKey='value'
    nameKey='name'
    cx='50%'
    cy='50%'
    outerRadius={150}
    fill='#8884d8'
    label
  >
    {pieData.map((entry, index) => (
      <Cell key={`cell-${index}`} fill={entry.fill} />
    ))}
  </Pie>
  <Tooltip formatter={(value) => `$$${value.toFixed(0)}$`} />
  <Legend />
</PieChart>
```

Slika 15 Kod tortaskog dijagrama

## 5.2. Sustav za rezervaciju

Sustav za rezervaciju ima svrhu prikazati spremljene apartmane i omogućiti njihovo rezerviranje.



Slika 16 Početna stranica klijentske web stranice

Stranica se sastoji od više dijelova kojima je cilj prikazati apartmane na različite načine kako bi se vidjela stvarna primjena korištenja zapisa iz baze podataka.

Ti dijelovi su:

- Početna stranice sa više prikaza, navigacijom, filterom i podnožjem
- Apartmani (stranica za prikazivanje svih apartmana)
- Rezervacije (rezerviranje nekog smještaja)
- Istraži (stranica za prikaz turističkih atrakтивности)
- Detaljniji prikaz apartmana klikom na karticu (i rezerviranje)

**Rezervirajte apartman:**

Ime

Email

Phone

Država

< May 2024 >

1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

Od: 08/05/2024  
Do: 14/05/2024  
Cijena: 700€

Slika 17 Modal forma za rezervaciju

Klikom na karticu apartmana otvara se detaljniji prikaz sa slikom i svim podacima o apartmanu. Klikom na gumb „REZERVIRAJTE APARTMAN“ otvara se modal, prozor koji se prikazuje iznad ostatka web stranice, na kojemu se nalazi forma za prijavu korisnika i biranje datuma za rezervaciju. Rezervirani datumi prikazuju se crveno, a odabrani su zeleno.

Klikom na gumb rezerviraj stvara se novi unos u bazu podataka u kolekciju za rezervacije.

```

const renderCalendar = () => {
  const startOfMonth = new Date(currentYear, currentMonth, 1);
  const endOfMonth = new Date(currentYear, currentMonth + 1, 0);
  const days = [];

  for (let i = 1; i <= endOfMonth.getDate(); i++) {
    const date = new Date(currentYear, currentMonth, i);
    const isSelected = startDate && endDate && date >= startDate && date <= endDate;
    days.push(
      <div>
        {key={i}}
        {className={`day ${isSelected ? 'booked' : ''} ${!isSelected ? 'selected' : ''}`}}
        {onClick={() => handleDateClick(date)}}
      </div>
    );
  }

  return days;
};

const formatDate = (date) => {
  return date.toLocaleDateString('en-GB', { day: '2-digit', month: '2-digit', year: 'numeric' }).replace(/\\/g, '/');
};

return (
  <div className="date-picker">
    <div className="month-year-browser">
      <button onClick={handlePrevMonth}></button>
      <span>{new Date(currentYear, currentMonth).toLocaleString('default', { month: 'long' })} {currentYear}</span>
      <button onClick={handleNextMonth}></button>
    </div>
    <div className="calendar">
      {renderCalendar()}
    </div>
    <div className="selected-dates">
      <p>Od: {startDate ? formatDate(startDate) : 'None'}</p>
      <p>Do: {endDate ? formatDate(endDate) : 'None'}</p>
    </div>
  </div>
);

```

Slika 18 Komponenta za biranje datum

Komponenta za biranje datuma kao svojstvo (eng. property) prima objekt rezervacije od kojeg uzima rezervirane datume. Rezervirani datumi će se prikazati kao crveni, oni ne mogu biti odabrani. Klikom na prvi datum i drugi datum svi označeni datumi se bojaju u zeleno te se sprema zabilježen datum rezervacije i računa cijena ovisno o cijeni noćenja apartmana.



## 6. Zaključak

Izrada web sustava za upravljanje i iznajmljivanje apartmana obuhvaća sve dijelove razvoja modernih web stranica. Predstavlja sveobuhvatan i koristan projekt primjenjiv u stvarnom svijetu. Kroz ovaj rad obrađeni su sve faze razvoja web aplikacije, određivanje funkcionalnosti, odabir tehnologija, backend razvoj i frontend razvoj.

Na početku rada raspisali smo sve funkcionalnosti i zadaće koje naš sustav treba obaviti. Ovo je ključno radi ispravnog odabira tehnologija i daljnjeg razvoja aplikacije. Web sjedište se sastoji od jedne stranice za upravljanje imanjima, njihovim prihodom, podacima, gostima te rezervacijama i druge web stranice za prikaz tih apartmana te njihovo rezerviranje.

Za korištene tehnologije odabrao sam React za frontend razvoj. React je JavaScript biblioteka koja omogućuje jednostavnu integraciju kompleksnih funkcionalnosti i ubrzava i olakšava razvojni proces kroz upotrebu komponenata. U sklopu React projekta korišten je HTML, CSS i JavaScript. Za izradu baze podataka korišten je Google-ov sustav Firebase i bazu podataka Firestore. Firestore je mrežna baza podataka kojom je moguće upravljati kroz korisničko sučelje ili kod. Omogućava nam spremanje apartmana, gostiju i rezervacija za daljnju obradu.

Izvršena je integracija baze podataka u našu aplikaciju korištenjem React biblioteke za Firebase. Nakon spajanja na bazu putem biblioteke možemo vršiti operacije za upravljanje bazom, te njeno prikazivanje. Iz baze podataka vučemo podatke o spremljenim apartmanima koji se prikazuju u klijentskom dijelu, i upravljamo njima u aplikaciji za administratore.

Na kraju dobijemo funkcionalan dvodijelni sustav za upravljanje imanjima i njihovu rezervaciju koji obuhvaća sve korisničke potrebe u navedenom kontekstu. Također koristi kao odličan primjer full stack web aplikacije i predstavlja sveobuhvatan projekt za učenje rada modernih web stranica.

## LITERAURA

- [1.] <https://techbootcamps.utexas.edu/blog/html-css-javascript/>
- [2.] <https://react.dev/>
- [3.] <https://firebase.google.com/>