Univerzitet u Beogradu

Fakultet organizacionih nauka

Katedra za elektronsko poslovanje

ReactJS

Domaći zadatak 3

|  |  |  |
| --- | --- | --- |
| Ime | Prezime | Broj indeksa |
|  |  |  |
| Link ka Github-u |  | |

Sadržaj

[1 Korisnički zahtev 3](#_Toc117932679)

[2 Implementacija 3](#_Toc117932680)

[3 Korisničko uputstvo 3](#_Toc117932681)

# Korisnički zahtev

Potrebno je kreirati aplikaciju za salon lepote, kako bi se olaksao proces rezervacije usluga. Korisnicima je omogućeno da se prijave na svoje naloge, pregledaju listu dostupnih usluga i zaposlenih, i kreiraju rezervacije za odabrane usluge, nakon cega im se salje potvrda o rezervaciji putem mejla. Takođe, mogu pretraživati postojeće rezervacije po različitim kriterijumima. Aplikacija takođe podržava zadržavanje sesije korisnika kako bi olakšala njihovo kretanje kroz različite delove aplikacije. Aplikacija omogućava efikasno upravljanje rezervacijama i pomaže korisnicima da brzo pronađu usluge i zaposlene koji odgovaraju njihovim potrebama.

# Implementacija

Login komponenta je odgovorna za formu za prijavu korisnika. Koristi se useState hook za praćenje stanja unetih podataka (email adrese i lozinke) i za praćenje tokena koji se dobija nakon uspešne prijave. Takođe, koristi se axios za slanje HTTP zahteva na server kako bi se izvršila prijava. Nakon uspešne prijave, korisnik se preusmerava na stranicu "/rezervacije". U slučaju greške, prikazuje se greška na ekranu.

InputField komponenta je jednostavna komponenta za unos teksta koja prikazuje ikonu (ikonu korisnika ili ikonu katanca, zavisno od toga da li se unosi email adresa ili lozinka) i input polje za unos teksta.

import React, { useState } from 'react';

import axios from 'axios';

import InputField from './InputField';

import { useNavigate } from 'react-router-dom';

function Login({setToken}) {

    let navigate = useNavigate();

  const [credentials, setCredentials] = useState({

    email: 'adonis.lind@example.org',

    password: 'password',

  });

  const handleChange = (e) => {

    const { name, value } = e.target;

    setCredentials({ ...credentials, [name]: value });

  };

  const handleSubmit = async (e) => {

    e.preventDefault();

    try {

      const response = await axios.post('http://127.0.0.1:8000/api/login', credentials);

      console.log(response.data);

        sessionStorage.setItem("token",response.data.token)

        sessionStorage.setItem("id",response.data.user.id)

        setToken(response.data.token);

        navigate('/rezervacije');

    } catch (error) {

      console.error('Login error', error);

        alert(error);

    }

  };

  return (

    <div className="login-container">

      <form onSubmit={handleSubmit} className="login-form">

        <h2>Sign In</h2>

        <InputField

          name="email"

          value={credentials.email}

          onChange={handleChange}

          placeholder="Username"

          type="email"

        />

        <InputField

          name="password"

          value={credentials.password}

          onChange={handleChange}

          placeholder="Password"

          type="password"

        />

        <div className="options">

          <button type="submit" className="login-button">Login</button>

        </div>

        <div className="footer">

          <a href="/signup">Don't have an account? Sign Up</a>

          <a href="/forgot-password">Forgot Password</a>

        </div>

      </form>

    </div>

  );

}

export default Login;

import React from 'react';

import { FontAwesomeIcon } from '@fortawesome/react-fontawesome';

import { faUser, faLock } from '@fortawesome/free-solid-svg-icons';

const InputField = ({ name, value, onChange, placeholder, type }) => {

  const icon = name === 'username' ? faUser : faLock;

  return (

    <div className="input-group">

      <FontAwesomeIcon icon={icon} className="input-icon" />

      <input

        type={type}

        name={name}

        value={value}

        onChange={onChange}

        placeholder={placeholder}

        required

      />

    </div>

  );

};

export default InputField;

Ovaj kod predstavlja stranicu "Rezervacije" u aplikaciji. Na ovoj stranici se prikazuju rezervacije korisnika, omogućeno je pretraživanje tih rezervacija, a podaci se dohvataju iz baze podataka. Na vrhu stranice se prikazuje naslov "Lista Rezervacija", ispod toga se nalazi polje za unos teksta kako bi se filtrirale rezervacije, a zatim se prikazuje tabela sa detaljima rezervacija.

Stranica koristi hookove poput useState i useEffect za upravljanje stanjem komponente i dohvatanje podataka sa servera. Takođe, koristi se i biblioteka lodash za implementaciju odložene pretrage kako bi se smanjilo opterećenje servera pri svakom unosu teksta.

Komponenta Rezervacije dohvata podatke o rezervacijama sa servera i prikazuje ih u tabeli. Takođe, omogućava korisniku da pretražuje rezervacije na osnovu unetog teksta. Ukoliko se podaci još uvek učitavaju sa servera, prikazuje se poruka "Loading...", a ako dođe do greške, prikazuje se odgovarajuća greška.

import React, { useState, useEffect } from 'react';

import axios from 'axios';

import { debounce } from 'lodash';

import RedTabele from './RedTabele';

const Rezervacije = () => {

  const [reservations, setReservations] = useState([]);

  const [filteredReservations, setFilteredReservations] = useState([]);

  const [error, setError] = useState(null);

  const [isLoading, setIsLoading] = useState(true);

  const [searchText, setSearchText] = useState('');

  useEffect(() => {

    const token = sessionStorage.getItem('token');

    const config = {

      headers: {

        Authorization: `Bearer ${token}`,

      },

    };

    axios.get('http://127.0.0.1:8000/api/rezervacije', config)

      .then((response) => {

        setReservations(response.data.data);

        setFilteredReservations(response.data.data);

        setIsLoading(false);

      })

      .catch((err) => {

        setError('Greška pri učitavanju rezervacija');

        setIsLoading(false);

      });

  }, []);

  const searchReservations = (text) => {

    setSearchText(text);

    const lowercasedText = text.toLowerCase();

    const filtered = reservations.filter(reservation => {

      return searchInObject(reservation, lowercasedText);

    });

    setFilteredReservations(filtered);

  };

  const debouncedSearch = debounce(searchReservations, 300);

  const handleSearchChange = (e) => {

    debouncedSearch(e.target.value);

  };

  const searchInObject = (obj, searchText) => {

    if (obj === null) return false;

    if (typeof obj === 'object') {

      return Object.values(obj).some(value => searchInObject(value, searchText));

    }

    return String(obj).toLowerCase().includes(searchText);

  };

  if (isLoading) {

    return <div>Loading...</div>;

  }

  if (error) {

    return <div>{error}</div>;

  }

  return (

    <div className='glavniContainer'>

      <h2>Lista Rezervacija</h2>

      <input

        type="text"

        placeholder="Pretraži rezervacije..."

        onChange={handleSearchChange}

      />

      <table>

        <thead>

          <tr>

            <th>ID</th>

            <th>Datum</th>

            <th>Vreme</th>

            <th>Korisnik</th>

            <th>Zaposleni</th>

            <th>Usluga</th>

          </tr>

        </thead>

        <tbody>

          {filteredReservations.map((reservation) => (

            <RedTabele key={reservation.id} reservation={reservation} />

          ))}

        </tbody>

      </table>

    </div>

  );

};

export default Rezervacije;

U ovoj aplikaciji imamo dve korisnicki definisane kuke, useUsluge i useZaposleni, koje se koriste za dohvatanje podataka o uslugama i zaposlenima sa servera. Ove funkcije koriste axios za slanje HTTP zahteva i useState i useEffect hookove za upravljanje stanjem komponenti.

Prva funkcija, useUsluge, dohvata podatke o uslugama i vraća ih kao rezultat. Slično tome, druga funkcija, useZaposleni, dohvata podatke o zaposlenima sa servera i takođe ih vraća kao rezultat.

import axios from "axios";

import { useEffect, useState } from "react";

const useUsluge = () => {

    const [usluge, setUsluge] = useState([]);

    useEffect(() => {

      axios.get('http://127.0.0.1:8000/api/usluge')

        .then((response) => {

          setUsluge(response.data.data);

        })

        .catch((error) => {

          console.error('Greška pri učitavanju usluga', error);

        });

    }, []);

    return [usluge, setUsluge];

  };

  export default useUsluge;

import axios from "axios";

import { useEffect, useState } from "react";

const useZaposleni = () => {

    const [zaposleni, setZaposleni] = useState([]);

    useEffect(() => {

      axios.get('http://127.0.0.1:8000/api/zaposleni')

        .then((response) => {

          setZaposleni(response.data.data);

        })

        .catch((error) => {

          console.error('Greška pri učitavanju zaposlenih', error);

        });

    }, []);

    return [zaposleni, setZaposleni];

  };

  export default useZaposleni;

Komponenta "CreateReservation" u aplikaciji koja omogućava korisnicima da kreiraju nove rezervacije.

Na ovoj stranici, korisnik može da unese podatke o usluzi, datumu, vremenu i zaposlenom za koga želi da napravi rezervaciju. Kada popuni sva obavezna polja i klikne na dugme "Create Reservation", podaci se šalju na server putem HTTP zahteva sa odgovarajućim autorizacionim tokenom koji se dobija iz sesije.

Ako je kreiranje rezervacije uspešno, korisnik će biti preusmeren na stranicu sa listom rezervacija. U slučaju bilo kakve greške prilikom slanja zahteva ili kreiranja rezervacije, greška će biti prijavljena u konzoli, i može se tretirati na odgovarajući način.

import React, { useState, useEffect } from 'react';

import axios from 'axios';

import { useNavigate } from 'react-router-dom';

import InputField from '../../login/InputField';

import useUsluge from '../hooks/useUsluge';

import useZaposleni from '../hooks/useZaposleni';

function CreateReservation( ) {

    const token= sessionStorage.getItem("token");

  const navigate = useNavigate();

  const [formData, setFormData] = useState({

    usluga\_id: '',

    datum: '',

    vreme: '',

    zaposleni\_id: '',

  });

  const [usluge] = useUsluge();

  const [zaposleni] = useZaposleni();

  const handleChange = (e) => {

    const { name, value } = e.target;

    setFormData({ ...formData, [name]: value });

  };

  const handleSubmit = (e) => {

    e.preventDefault();

    axios

      .post('http://127.0.0.1:8000/api/rezervacije', formData, {

        headers: {

          Authorization: `Bearer ${token}`,

        },

      })

      .then((response) => {

        console.log('Rezervacija uspešno kreirana', response.data);

        navigate('/rezervacije');

      })

      .catch((error) => {

        console.error('Greška pri kreiranju rezervacije', error);

        // Tretirajte grešku pri kreiranju prema potrebi

      });

  };

  return (

    <div className="login-container">

      <form onSubmit={handleSubmit} className="login-form">

        <h2>Create Reservation</h2>

        <div className="input-group">

            <select

                name="usluga\_id"

                value={formData.usluga\_id}

                onChange={handleChange}

            >

                <option value="" disabled>

                Select a service

                </option>

                {usluge.map((usluga) => (

                <option key={usluga.id} value={usluga.id}>

                    {usluga.naziv}

                </option>

                ))}

            </select>

            </div>

        <InputField

          name="datum"

          value={formData.datum}

          onChange={handleChange}

          placeholder="Date"

          type="date"

        />

        <InputField

          name="vreme"

          value={formData.vreme}

          onChange={handleChange}

          placeholder="Time"

          type="time"

        />

       <div className="input-group">

        <select

            name="zaposleni\_id"

            value={formData.zaposleni\_id}

            onChange={handleChange}

        >

            <option value="" disabled>

            Select an employee

            </option>

            {zaposleni.map((z) => (

            <option key={z.id} value={z.id}>

                {z.name}

            </option>

            ))}

        </select>

        </div>

        <div className="options">

          <button type="submit" className="login-button">

            Create Reservation

          </button>

        </div>

      </form>

    </div>

  );

}

export default CreateReservation;

# Korisničko uputstvo

Kako bi korisnici mogli da koriste ovu aplikaciju potrebno je da se uloguju

Slika na kojoj se nalazi tekst, snimak ekrana, Font, dizajn

Opis je automatski generisan

Ukoliko korisnici nisu ulogovani imaju mogucnost pregleda cenovnika

Slika na kojoj se nalazi tekst, snimak ekrana, Font

Opis je automatski generisan

Nakon prijave, korisnicima se prikazuju zakazivanja koja su oni kreirali

Slika na kojoj se nalazi tekst, snimak ekrana

Opis je automatski generisan

Odabirom opcije Dodaj iz navigacionog menija korisnicima se otvara forma za kreiranje nove rezervacije, nakon cega im stize mejl kao potvrda rezervacije

Slika na kojoj se nalazi tekst, snimak ekrana, dizajn

Opis je automatski generisan

Potvrda mejla

Slika na kojoj se nalazi tekst, snimak ekrana, Font, softver

Opis je automatski generisan