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 koja je namenjena za upravljanje i praćenje fizičkih vežbi i fitness aktivnosti. Funkcionalnosti koje bi aplikacija trebalo da ima su date ispod:

* Registracija i prijava: Korisnici mogu da se registruju i prijave kako bi pristupili aplikaciji. Registracijom se stvara korisnički nalog, dok prijava omogućava pristup personalizovanim funkcijama.
* Pregled vežbi: Aplikacija omogućava korisnicima da pretražuju različite vežbe sa detaljnim opisima, slikama i video materijalom. Ovo pomaže korisnicima da pronađu odgovarajuće vežbe za svoje potrebe i ciljeve.
* Dodavanje i uređivanje vežbi: Registrovani korisnici mogu da dodaju nove vežbe u bazu podataka ili da urede postojeće informacije o vežbama. Ovo doprinosi širenju baze podataka i prilagodbi vežbi.
* Pretraga i filtriranje: Korisnici mogu da pretražuju vežbe po različitim kriterijumima kao što su tip vežbe, mišićna grupa, oprema i težina. Ovo omogućava personalizovan pristup vežbama.
* Brisanje vežbi: Korisnici mogu da obrišu vežbe koje više ne žele da čuvaju u bazi podataka
* Paginacija: prikaz vežbi kako bi olakšala pregled i navigaciju kroz veliku količinu informacija.
* Odjava: Ulogovani korisnici mogu se odjaviti iz aplikacije kada završe sa radom

# Implementacija

U ovom poglavlju će biti dat kratak pregled najbitnijih delova koda

import './App.css';

import { BrowserRouter as Router, Routes, Route } from 'react-router-dom';

import LoginForm from './komponente/LoginRegistracija/LoginForm';

import RegisterForm from './komponente/LoginRegistracija/RegisterForm';

import Vezbe from './komponente/vezbe/Vezbe';

import VezbeApi from './komponente/vezbe/VezbeApi';

import HomePage from './komponente/HomePage/HomePage';

import Navbar from './komponente/nav/Navbar';

import { useState } from 'react';

function App() {

  const [token,setToken]=useState(null);

  return (

    <Router>

      <Navbar token={token} setToken={setToken}  />

      <Routes>

      <Route path="/" element={<HomePage />} />

        <Route path="/login" element={<LoginForm setToken={setToken}/>} />

        <Route path="/vezbe" element={<Vezbe />} />

        <Route path="/vezbeapi" element={<VezbeApi/>} />

        <Route path="/register" element={<RegisterForm />} />

      </Routes>

    </Router>

  );

}

export default App;

Komponenta "RegisterForm" je odgovorna za prikaz i upravljanje registracionim obrascem za fitness aplikaciju. Na početku definiše lokalno stanje (formData) koje sadrži različita polja za unos kao što su ime, e-mail, lozinka, visina, težina, pol, datum rođenja, fitness ciljevi i napomene. Takođe koristi hookove useState za praćenje promena u ovim poljima i useNavigate za upravljanje rutama.

Kada korisnik unese podatke i pritisne dugme "Register", poziva se funkcija handleSubmit koja šalje podatke na server putem HTTP zahteva koristeći Axios biblioteku. Na osnovu odgovora servera, korisnika se preusmerava na stranicu za prijavu ili se prikazuje poruka o neuspešnoj registraciji.

Komponenta takođe koristi komponente "InputField" i "GenderSelect" kako bi olakšala unos i odabir pola. "InputField" komponenta prikazuje polja za unos sa odgovarajućim oznakama, dok "GenderSelect" komponenta omogućava odabir pola između "M" (Muško) i "Z" (Žensko). Ove komponente čine formu preglednijom i korisnički prijateljskom.

import React, { useState } from 'react';

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

import axios from 'axios';

import InputField from './InputField';

import GenderSelect from './GenderSelect';

const RegisterForm = () => {

  const [formData, setFormData] = useState({

    name: '',

    email: '',

    password: '',

    height: '',

    weight: '',

    gender: '', // 'M' for Male or 'Z' for Female

    date\_of\_birth: '',

    fitness\_goals: '',

    notes: ''

  });

  let navigate = useNavigate();

  const handleChange = (e) => {

    const { name, value } = e.target;

    setFormData(prevState => ({

      ...prevState,

      [name]: value

    }));

  };

  const handleSubmit = async (e) => {

    e.preventDefault();

    try {

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

      const { data } = response;

      if (response.status === 201) {

        console.log('Registration successful', data);

        navigate('/login');

      } else {

        console.error('Registration failed:', data.message);

      }

    } catch (error) {

      console.error('Error during registration:', error.response ? error.response.data : error.message);

    }

  };

  return (

    <div className="login-container">

      <div className="login-header">

        FITNESS REGISTRATION FORM

      </div>

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

        <InputField label="Name" type="text" name="name" value={formData.name} onChange={handleChange} />

        <InputField label="Email" type="email" name="email" value={formData.email} onChange={handleChange} />

        <InputField label="Password" type="password" name="password" value={formData.password} onChange={handleChange} />

        <InputField label="Height" type="number" name="height" value={formData.height} onChange={handleChange} />

        <InputField label="Weight" type="number" name="weight" value={formData.weight} onChange={handleChange} />

        <div className="input-container">

          <label>Gender</label>

          <GenderSelect

            selectedGender={formData.gender}

            onChange={handleChange}

          />

        </div>

        <InputField label="Date of Birth" type="date" name="date\_of\_birth" value={formData.date\_of\_birth} onChange={handleChange} />

        <InputField label="Fitness Goals" type="text" name="fitness\_goals" value={formData.fitness\_goals} onChange={handleChange} />

        <InputField label="Notes" type="text" name="notes" value={formData.notes} onChange={handleChange} />

        <div className="form-footer">

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

        </div>

      </form>

    </div>

  );

};

export default RegisterForm;

Komponenta "LoginForm" je odgovorna za prikaz i upravljanje obrascem za prijavu u aplikaciju. Ova komponenta koristi se za unos e-mail adrese i lozinke kako bi se omogućila prijava korisnika.

U unutrašnjosti komponente, koristi se hook "useState" za definisanje stanja "formData" koje sadrži polja za unos e-mail adrese i lozinke. Takođe se koristi "let" da bi se definisala promenljiva "navigate" za kasnije upravljanje preusmeravanjem korisnika.

Metoda "handleChange" se koristi za praćenje promena u poljima za unos. Kada korisnik unese e-mail ili lozinku, ova funkcija ažurira odgovarajuće vrednosti u "formData" stanju.

Metoda "handleSubmit" se aktivira kada korisnik pritisne dugme za prijavu. Ova funkcija šalje HTTP POST zahtev na server sa unetim podacima za e-mail i lozinku. Nakon dobijanja odgovora, ako je prijava uspešna (status 200), token se čuva u sesiji korisnika ("sessionStorage") i postavlja se pozivom "setToken". Korisnik se preusmerava na stranicu "/vezbe" i ispisuje se poruka o uspešnoj prijavi. U slučaju neuspešne prijave, ispisuje se odgovarajuća greška.

import React, { useState } from 'react';

import axios from 'axios';

import './LoginForm.css';

import InputField from './InputField';

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

const LoginForm = ({setToken}) => {

  const [formData, setFormData] = useState({

    email: '',

    password: ''

  });

  let navigate = useNavigate();

  const handleChange = (e) => {

    const { name, value } = e.target;

    setFormData(prevState => ({

      ...prevState,

      [name]: value

    }));

  };

  const handleSubmit = async (e) => {

    e.preventDefault();

    try {

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

        email: formData.email,

        password: formData.password

      });

      const { data } = response;

      if (response.status === 200) {

        sessionStorage.setItem('authToken', data.access\_token);

        setToken(data.access\_token);

        console.log('Login successful');

        navigate('/vezbe');

      } else {

        console.error('Login failed:', data.message);

      }

    } catch (error) {

      if (error.response) {

        console.error('Login failed:', error.response.data);

      } else if (error.request) {

        console.error('No response:', error.request);

      } else {

        console.error('Error:', error.message);

      }

    }

  };

  return (

    <div className="login-container">

      <div className="login-header">

        FITNESS LOGIN FORM

      </div>

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

        <InputField

            label="Email"

            type="email"

            name="email"

            value={formData.email}

            onChange={handleChange}

          />

          <InputField

            label="Password"

            type="password"

            name="password"

            value={formData.password}

            onChange={handleChange}

          />

        <div className="form-footer">

          <button type="submit" className="login-button">Log In</button>

          {/\* <a href="#forgot" className="forgot-password">Forgot Password?</a> \*/}

        </div>

      </form>

    </div>

  );

};

export default LoginForm;

Komponenta "VezbeApi" je odgovorna za prikazivanje vežbi i njihovo filtriranje na osnovu određenih kriterijuma. U prva dva hooka useState definisani su "exercises" (niz vežbi) i "filter" (objekat za filtriranje). "exercises" čuva podatke o vežbama dobijene sa servera, dok "filter" čuva vrednosti kriterijuma za filtriranje.

U hooku useEffect, definisana je funkcija "fetchExercises" koja se izvršava nakon prvog renderovanja komponente. Ova funkcija šalje HTTP GET zahtev serveru da bi dobila slučajne vežbe. Ako je zahtev uspešan, podaci se smeštaju u "exercises" stanje. U slučaju greške, ispisuje se odgovarajuća greška.

Nakon definisanja "fetchExercises", koristi se fetchExercises funkcija unutar useEffect kako bi se inicijalno dobile vežbe prilikom prvog renderovanja komponente.

Dalje, definisane su funkcije handleFilterChange koja se koristi za praćenje promena u filterima i filteredExercises niz koji se dobija filtriranjem vežbi na osnovu kriterijuma definisanih u "filter" objektu.

Na kraju, komponenta se renderuje sa dropdown menijima za odabir tipa, mišićne grupe, opreme i težine vežbe. Takođe se prikazuju kartice vežbi koje zadovoljavaju postavljene kriterijume filtriranja.

Kartice vežbi se prikazuju pomoću komponente "VezbaKartica", koja prima podatke o svakoj vežbi i prikazuje ih u kartici. Klikom na "Show more" se otvara modal sa detaljnim uputstvima za vežbu.

import React, { useState } from 'react';

import { FaDumbbell, FaHeartbeat, FaTools, FaUserGraduate } from 'react-icons/fa';

import Modal from './Modal';

const VezbaKartica = ({ exercise }) => {

  const [showModal, setShowModal] = useState(false);

  const imageUrl = `https://source.unsplash.com/featured/?gym,exercise/${encodeURIComponent(exercise.name)}/${Date.now()}`;

  const previewInstructions = exercise.instructions.substring(0, 30) + '...';

  return (

    <div className="exercise-card">

      <img

        src={imageUrl}

        alt={exercise.name}

        className="exercise-image"

      />

      <h3>{exercise.name}</h3>

      <p><FaDumbbell /> Type: {exercise.type}</p>

      <p><FaHeartbeat /> Muscle: {exercise.muscle}</p>

      <p><FaTools /> Equipment: {exercise.equipment}</p>

      <p><FaUserGraduate /> Difficulty: {exercise.difficulty}</p>

      <p>Instructions: {previewInstructions}

        <span className="show-more" onClick={() => setShowModal(true)}>Show more</span>

      </p>

      {showModal && <Modal content={exercise.instructions} onClose={() => setShowModal(false)} />}

    </div>

  );

};

export default VezbaKartica;

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

import axios from 'axios';

import VezbaKartica from './VezbaKartica';

import './Exercises.css';

const VezbeApi = () => {

  const [exercises, setExercises] = useState([]);

  const [filter, setFilter] = useState({ type: '', muscle: '', equipment: '', difficulty: '' });

  useEffect(() => {

    const fetchExercises = async () => {

      try {

        const response = await axios.get('http://127.0.0.1:8000/api/exercises/getRandomExercises');

        setExercises(response.data);

      } catch (error) {

        console.error('Error fetching exercises:', error);

      }

    };

    fetchExercises();

  }, []);

  const handleFilterChange = (e) => {

    setFilter({ ...filter, [e.target.name]: e.target.value });

  };

  const filteredExercises = exercises.filter(exercise => {

    return (

      (filter.type ? exercise.type === filter.type : true) &&

      (filter.muscle ? exercise.muscle === filter.muscle : true) &&

      (filter.equipment ? exercise.equipment === filter.equipment : true) &&

      (filter.difficulty ? exercise.difficulty === filter.difficulty : true)

    );

  });

  return (

    <div className="exercises-container">

      <div className="filters">

        <select name="type" onChange={handleFilterChange}>

          <option value="">Select Type</option>

          <option value="strength">strength</option>

          <option value="olympic\_weightlifting">olympic\_weightlifting</option>

          <option value="strongman">strongman</option>

        </select>

        <select name="muscle" onChange={handleFilterChange}>

          <option value="">Select Muscle Group</option>

          <option value="abdominals">abdominals</option>

          <option value="quadriceps">quadriceps</option>

          <option value="forearms">forearms</option>

          <option value="middle\_back">middle\_back</option>

          <option value="lats">lats</option>

          <option value="lower\_back">lower\_back</option>

          <option value="shoulders">shoulders</option>

          <option value="biceps">biceps</option>

        </select>

        <select name="equipment" onChange={handleFilterChange}>

          <option value="">Select Equipment</option>

          <option value="machine">machine</option>

          <option value="barbell">barbell</option>

          <option value="dumbbell">dumbbell</option>

          <option value="other">other</option>

        </select>

        <select name="difficulty" onChange={handleFilterChange}>

          <option value="">Select Difficulty</option>

          <option value="beginner">beginner</option>

          <option value="intermediate">intermediate</option>

          <option value="advanced">advanced</option>

        </select>

      </div>

      <div className="exercises-cards">

            {filteredExercises.map((exercise, index) => (

            <VezbaKartica key={index} exercise={exercise} />

            ))}

        </div>

    </div>

  );

};

export default VezbeApi;

Komponenta "Vezbe" ima ulogu prikazivanja i upravljanja listom vežbi. Na početku komponente definišemo stanje za trenutnu stranicu (currentPage), broj vežbi koje se prikazuju po stranici (exercisesPerPage), pretragu vežbi (searchTerm), filtrirane vežbe (filteredExercises), prikazivanje moda za dodavanje vežbi (showModal) i token za autentikaciju korisnika.

Koristi se hook useEffect kako bi se odmah pri inicijalizaciji komponente preuzeli podaci o vežbama sa servera korišćenjem custom hooka useVezbe. Takođe, filteredExercises se postavlja na inicijalnu listu vežbi.

Kreirana je funkcija paginate za promenu trenutne stranice i funkcija handleSearch za praćenje promena u pretrazi vežbi.

U drugom useEffect bloku, filtrirane vežbe se ažuriraju na osnovu pretrage koju korisnik unosi u input polje. Funkcija filtrira vežbe na osnovu različitih atributa vežbi kao što su naziv, opis, URL videa, kalorije i kategorija.

Komponenta takođe omogućava brisanje vežbi. Klikom na dugme za brisanje (<FaRegTrashAlt />), poziva se funkcija deleteExercise, koja šalje HTTP DELETE zahtev na server i uklanja vežbu iz prikaza.

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

import axios from 'axios';

import './Vezbe.css';

import useVezbe from '../hooks/useVezbe';

import ExerciseRow from './ExerciseRow';

import DodajVezbuModal from './DodajVezbuModal';

const Vezbe = () => {

  const [currentPage, setCurrentPage] = useState(1);

  const [exercisesPerPage] = useState(5);

  const [searchTerm, setSearchTerm] = useState('');

  const [filteredExercises, setFilteredExercises] = useState([]);

  const [showModal, setShowModal] = useState(false);

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

  const vezbe = useVezbe('http://127.0.0.1:8000/api/exercises', token);

  useEffect(() => {

    setFilteredExercises(vezbe);

  }, [vezbe]);

  const paginate = (pageNumber) => setCurrentPage(pageNumber);

  const handleSearch = (e) => {

    setSearchTerm(e.target.value);

    setCurrentPage(1);

  };

  useEffect(() => {

    const lowerCaseSearchTerm = searchTerm.toLowerCase();

    const newFilteredExercises = vezbe.filter((vezba) => {

      const { name, description, video\_url, average\_calories\_burned, category } = vezba;

      return (

        name.toLowerCase().includes(lowerCaseSearchTerm) ||

        description.toLowerCase().includes(lowerCaseSearchTerm) ||

        video\_url.toLowerCase().includes(lowerCaseSearchTerm) ||

        average\_calories\_burned.toString().includes(lowerCaseSearchTerm) ||

        category.toLowerCase().includes(lowerCaseSearchTerm)

      );

    });

    setFilteredExercises(newFilteredExercises);

  }, [searchTerm, vezbe]);

  const deleteExercise = async (id) => {

    try {

      await axios.delete(`http://127.0.0.1:8000/api/exercises/${id}`, {

        headers: {

          Authorization: `Bearer ${token}`,

        },

      });

      const updatedExercises = filteredExercises.filter((vezba) => vezba.id !== id);

      setFilteredExercises(updatedExercises);

      alert("Uspesno obrisano!")

    } catch (error) {

      console.error("Došlo je do greške prilikom brisanja vežbe", error);

    }

  };

  const pageNumbers = [];

  for (let i = 1; i <= Math.ceil(filteredExercises.length / exercisesPerPage); i++) {

    pageNumbers.push(i);

  }

  const indexOfLastExercise = currentPage \* exercisesPerPage;

  const indexOfFirstExercise = indexOfLastExercise - exercisesPerPage;

  const currentExercises = filteredExercises.slice(indexOfFirstExercise, indexOfLastExercise);

  return (

    <div className="vezbe-container">

      <h1>Vezbe</h1>

      <button onClick={() => setShowModal(true)}>Dodaj Novu Vezbu</button>

      <div className="search-container">

        <input

          type="text"

          placeholder="Pretraži vezbe..."

          value={searchTerm}

          onChange={handleSearch}

        />

      </div>

      <table className="vezbe-table">

        <thead>

          <tr>

            <th>ID</th>

            <th>Naziv</th>

            <th>Opis</th>

            <th>Video URL</th>

            <th>Kalorije</th>

            <th>Kategorija</th>

            <th>Obrisi</th>

          </tr>

        </thead>

        <tbody>

          {currentExercises.map((vezba) => (

            <ExerciseRow key={vezba.id} vezba={vezba} deleteExercise={deleteExercise} />

          ))}

        </tbody>

      </table>

      <nav>

        <ul className="pagination">

          {pageNumbers.map((number) => (

            <li key={number} className={`page-item ${currentPage === number ? 'active' : ''}`}>

              <button onClick={() => paginate(number)} className="page-link">

                {number}

              </button>

            </li>

          ))}

        </ul>

      </nav>

      <DodajVezbuModal show={showModal} handleClose={() => setShowModal(false)} token={token} />

    </div>

  );

};

export default Vezbe;

import React from 'react';

import { FaRegTrashAlt } from 'react-icons/fa';

const ExerciseRow = ({ vezba, deleteExercise }) => {

  return (

    <tr>

      <td>{vezba.id}</td>

      <td>{vezba.name}</td>

      <td>{vezba.description}</td>

      <td>{vezba.video\_url}</td>

      <td>{vezba.average\_calories\_burned}</td>

      <td>{vezba.category}</td>

      <td>

        <button onClick={() => deleteExercise(vezba.id)}><FaRegTrashAlt /></button>

      </td>

    </tr>

  );

};

export default ExerciseRow;

Komponenta Navbar ima ulogu navigacionog menija kroz aplikaciju. Ovde je značajno napomenuti funkciju "handleLogout" koja se koristi za obradu korisnikove odjave. Klikom na dugme "Logout", ova funkcija šalje HTTP POST zahtev na server kako bi odjavila korisnika. Nakon uspešne odjave, token se postavlja na "null" putem properti "setToken" kako bi se korisniku onemogućio pristup zaštićenim resursima, a zatim se koristi "history" funkcija za preusmeravanje korisnika na početnu stranicu ("/").

import React from 'react';

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

import axios from 'axios';

function Navbar({ token, setToken }) {

  const history = useNavigate();

  const handleLogout = async () => {

    try {

      await axios.post('http://127.0.0.1:8000/api/logout', null, {

        headers: {

          Authorization: `Bearer ${token}`,

        },

      });

      setToken(null);

      history('/');

    } catch (error) {

      console.error('Greška prilikom odjave:', error);

    }

  };

  return (

    <nav className="navbar">

      <ul className="nav-list">

        <li className="nav-item">

          <Link to="/" className="nav-link">Početna</Link>

        </li>

        {token === null && (

          <>

            <li className="nav-item">

              <Link to="/login" className="nav-link">Login</Link>

            </li>

            <li className="nav-item">

              <Link to="/register" className="nav-link">Registracija</Link>

            </li>

            <li className="nav-item">

              <Link to="/vezbeapi" className="nav-link">Vežbe API</Link>

            </li>

          </>

        )}

        {token !== null && (

          <>

            <li className="nav-item">

              <Link to="/vezbe" className="nav-link">Vežbe</Link>

            </li>

            <li className="nav-item">

              <button onClick={handleLogout} className="nav-link">Logout</button>

            </li>

          </>

        )}

      </ul>

    </nav>

  );

}

export default Navbar;

# Korisničko uputstvo

Početna stranica izgleda kao na slici ispod.

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

Opis je automatski generisan

Stranica za registraciju novih korisnika

Slika na kojoj se nalazi tekst, snimak ekrana, dizajn

Opis je automatski generisan

Stranica za prijavu korisnika

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

Opis je automatski generisan

Stranica za prikaz vežbi sa spoljnog apija

Slika na kojoj se nalazi tekst, odeća, snimak ekrana, obuća

Opis je automatski generisan

Stranica za admina – prikaz, brisanje i dodavanje vežbi

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

Opis je automatski generisan