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

Ova aplikacija pruža korisnicima različite funkcionalnosti u zavisnosti od njihovih uloga i toga da li su Ova aplikacija pruža različite funkcionalnosti korisnicima u zavisnosti od njihovih uloga i prijavljenosti. Korisnici koji su ulogovani imaju pristup anketama i FDA podacima. Oni mogu pregledavati dostupne ankete, ocenjivati ih i davati odgovore na postavljena pitanja. Takođe, imaju mogućnost pregleda informacija o povlacima hrane sa tržišta i detalja o pojedinačnim proizvodima.

Administratori, sa svoje strane, imaju dodatne privilegije. Oni mogu pristupiti administratorskom panelu gde mogu upravljati anketama, uključujući kreiranje novih anketa i pregled prikupljenih odgovora. Ova funkcionalnost omogućava administratorima efikasno upravljanje anketama i analizu rezultata.

# Implementacija

Komponenta "Login" se koristi za prijavu korisnika. Komponenta prikazuje formular za unos e-pošte i lozinke, te omogućava korisniku da se prijavi na aplikaciju. Kada korisnik unese e-poštu i lozinku te klikne na dugme "Submit", podaci se šalju na određeni API endpoint putem Axios biblioteke. Ako je prijava uspešna, odgovor sa servera se obradi i token, kao i neki drugi podaci, se skladište u sesiji korisnika. Takođe, poziva se funkcija "setToken" koja postavlja token u stanje aplikacije kako bi se korisnika autentikovalo. Nakon toga, korisnik se preusmerava na stranicu "/ankete". U slučaju greške pri prijavi, ispisuje se odgovarajuća greška u konzoli.

Stilovi su definisani u objektu "styles" i primenjuju se na komponentu kako bi se postigao određeni izgled i raspored elemenata na ekranu. Pozadinska slika i boje su takođe deo stilizacije kako bi se postigao određeni estetski efekat.

import React, { useState } from 'react';

import axios from 'axios';

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

import InputField from '../reusable/InputField';

const Login = ({setToken}) => {

  const [email, setEmail] = useState('');

  const [password, setPassword] = useState('');

  const navigate = useNavigate();

  const handleSubmit = async (event) => {

    event.preventDefault();

    try {

      const response = await axios.post('http://127.0.0.1:8000/api/login', { email, password });

      sessionStorage.setItem('token', response.data.token);

      sessionStorage.setItem('user\_id', response.data.user.id);

      sessionStorage.setItem('user\_uloga', response.data.user.uloga);

      setToken(response.data.token)

      navigate('/ankete');

    } catch (error) {

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

    }

  };

  return (

    <div style={styles.container}>

      <form style={styles.form} onSubmit={handleSubmit}>

        <h2>Please Sign In</h2>

        <InputField

          type="email"

          placeholder="email"

          value={email}

          onChange={(e) => setEmail(e.target.value)}

        />

        <InputField

          type="password"

          placeholder="password"

          value={password}

          onChange={(e) => setPassword(e.target.value)}

        />

        <button type="submit" style={styles.button}>Submit</button>

      </form>

    </div>

  );

};

const styles = {

    container: {

      display: 'flex',

      justifyContent: 'center',

      alignItems: 'center',

      height: '100vh',

      background: 'linear-gradient(135deg, #e2dadb 25%, #dae2df 25%, #dae2df 50%, #a2a7a5 50%, #a2a7a5 75%, #6d696a 75%)',

    },

    form: {

      display: 'flex',

      flexDirection: 'column',

      width: '300px',

      backgroundColor: '#ffffff',

      padding: '20px',

      borderRadius: '5px',

      boxShadow: '0 4px 8px rgba(0, 0, 0, 0.1)',

    },

    input: {

      marginBottom: '10px',

      padding: '10px',

      border: '1px solid #ddd',

      borderRadius: '4px',

    },

    button: {

      padding: '10px',

      border: 'none',

      borderRadius: '4px',

      backgroundColor: '#a2a7a5',

      color: 'white',

      cursor: 'pointer',

    }

  };

  export default Login;

Komponenta nazvana "TabelaAnketa" se koristi za prikaz i manipulaciju listom anketa. Komponenta omogućava korisnicima pretragu anketa putem unosa u polje za pretragu, prikazuje listu anketa u tabeli sa osnovnim informacijama o svakoj anketi, i omogućava dodavanje novih anketa putem modalnog prozora.

U kodu se koristi useState i useEffect hook-ovi za upravljanje stanjem komponente i asinhrono dohvaćanje anketa sa servera putem Axios biblioteke. Takođe, koristi se "useSurveys" hook za upravljanje listom anketa u aplikaciji.

Komponenta takođe omogućava unos novih anketa putem modalnog prozora i slanje podataka na server putem POST zahteva. Prikazane su osnovne informacije o anketama kao što su naslov, opis, datum početka i završetka, te status.

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

import axios from 'axios';

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

import RedAnkete from '../../reusable/RedAnkete';

import Modal from 'react-modal';

import useSurveys from '../../useSurveys';

const TabelaAnketa = () => {

    let navigate = useNavigate();

  const {surveys,setSurveys} = useSurveys();

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

  const [isModalOpen, setIsModalOpen] = useState(false);

  const [newSurvey, setNewSurvey] = useState({

    naslov: 'novaanketa',

    opis: 'opis nove ankete',

    datum\_pocetka: '2024-01-01',

    datum\_kraja: '2025-01-04',

    status: 'otvoreno'

  });

  useEffect(() => {

    const fetchSurveys = async () => {

      try {

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

        setSurveys(response.data);

      } catch (error) {

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

      }

    };

    fetchSurveys();

  }, [isModalOpen]);

  const handleInputChange = (e) => {

    setNewSurvey(prevState => ({

      ...prevState,

      [e.target.name]: e.target.value

    }));

  };

  const handleSearchChange = (e) => {

    setSearchTerm(e.target.value);

  };

  const handleSubmit = async (e) => {

    e.preventDefault();

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

    try {

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

        headers: {

          'Authorization': `Bearer ${token}`

        }

      });

      setIsModalOpen(false);

    } catch (error) {

      console.error('Error creating survey:', error);

    }

  };

    const filteredSurveys = surveys.filter(survey =>

      survey.naslov.toLowerCase().includes(searchTerm.toLowerCase())

    );

  return (

    <div style={styles.container}>

      <div style={styles.controlsContainer}>

        <input

          type="text"

          placeholder="Pretraži ankete..."

          value={searchTerm}

          onChange={handleSearchChange}

          style={styles.searchInput}

        />

      </div>

       <button onClick={() => setIsModalOpen(true)}>Dodaj Novu Anketu</button>

       <Modal isOpen={isModalOpen} onRequestClose={() => setIsModalOpen(false)}>

        <form onSubmit={handleSubmit} style={modalStyles.form}>

          <input

            type="text"

            name="naslov"

            placeholder="Naslov"

            value={newSurvey.naslov}

            onChange={handleInputChange}

            style={modalStyles.input}

          />

          <input

            type="text"

            name="opis"

            placeholder="Opis"

            value={newSurvey.opis}

            onChange={handleInputChange}

            style={modalStyles.input}

          />

          <input

            type="date"

            name="datum\_pocetka"

            placeholder="Datum početka"

            value={newSurvey.datum\_pocetka}

            onChange={handleInputChange}

            style={modalStyles.input}

          />

          <input

            type="date"

            name="datum\_kraja"

            placeholder="Datum kraja"

            value={newSurvey.datum\_kraja}

            onChange={handleInputChange}

            style={modalStyles.input}

          />

          <input

            type="text"

            name="status"

            placeholder="Status"

            value={newSurvey.status}

            onChange={handleInputChange}

            style={modalStyles.input}

          />

          <button type="submit" style={modalStyles.submitButton}>Kreiraj Anketu</button>

        </form>

      </Modal>

      <div style={styles.tableContainer}>

        <table style={styles.table}>

        <thead>

        <tr>

            <th style={styles.th}>ID</th>

            <th style={styles.th}>Title</th>

            <th style={styles.th}>Description</th>

            <th style={styles.th}>Start Date</th>

            <th style={styles.th}>End Date</th>

            <th style={styles.th}>Status</th>

            <th style={styles.th}>Detalji</th>

        </tr>

        </thead>

        <tbody>

        {filteredSurveys.map(survey => (

              <RedAnkete key={survey.id} survey={survey} navigate={navigate} />

            ))}

        </tbody>

        </table>

      </div>

    </div>

  );

};

const styles = {

  container: {

    display: 'flex',

    flexDirection: 'column',

    alignItems: 'center',

    justifyContent: 'flex-start',

    height: 'auto',

    background: 'linear-gradient(135deg, #e2dadb 25%, #dae2df 25%, #dae2df 50%, #a2a7a5 50%, #a2a7a5 75%, #6d696a 75%)',

    padding: '20px 0'

  },

  controlsContainer: {

    display: 'flex',

    justifyContent: 'space-between',

    width: '80%',

    marginBottom: '20px',

  },

  tableContainer: {

    backgroundColor: '#ffffff',

    padding: '20px',

    borderRadius: '5px',

    boxShadow: '0 4px 8px rgba(0, 0, 0, 0.1)',

  },

  table: {

    width: '100%',

    borderCollapse: 'collapse',

    backgroundColor: 'white',

  },

  th: {

    padding: '15px 10px',

    borderBottom: '1px solid #ddd',

    textAlign: 'left',

    backgroundColor: '#dae2df',

    color: '#6d696a',

  },

  td: {

    padding: '20px 10px',

    borderBottom: '1px solid #ddd',

  },

  openStatus: {

    color: '#98FB98',

  },

  closedStatus: {

    color: '#FFB6C1',

  },

  searchContainer: {

    margin: '20px',

    textAlign: 'center',

  },

  searchInput: {

    padding: '10px',

    width: '300px',

    borderRadius: '4px',

    border: '1px solid #ddd',

  },

};

const modalStyles = {

  content: {

    top: '50%',

    left: '50%',

    right: 'auto',

    bottom: 'auto',

    marginRight: '-50%',

    transform: 'translate(-50%, -50%)',

    backgroundColor: '#ffffff',

    padding: '20px',

    borderRadius: '5px',

    boxShadow: '0 4px 8px rgba(0, 0, 0, 0.1)',

    width: '50%',

    border: '1px solid #ddd',

  },

  overlay: {

    backgroundColor: 'rgba(0, 0, 0, 0.75)'

  },

  form: {

    display: 'flex',

    flexDirection: 'column',

    gap: '10px'

  },

  input: {

    padding: '10px',

    border: '1px solid #ddd',

    borderRadius: '4px',

    width: '100%',

  },

  submitButton: {

    backgroundColor: '#dae2df',

    color: '#6d696a',

    padding: '10px 15px',

    border: 'none',

    borderRadius: '4px',

    cursor: 'pointer',

    fontWeight: 'bold',

    fontSize: '16px',

  }

};

export default TabelaAnketa;

Ovaj kod predstavlja funkciju "useSurveys" koja se koristi kao korisnicki definisani hook u aplikaciji. Ova funkcija je dizajnirana da omogući lako upravljanje listom anketa u aplikaciji. Koristi se "useEffect" hook kako bi se izvršio asinhroni zahtev ka serveru kako bi se dohvatili podaci o anketama. Zahtev se šalje prema određenom API endpointu (URL) putem Axios biblioteke. Kada se odgovor od servera primi, podaci se postavljaju u stanje "surveys" pomoću funkcije "setSurveys".

import { useState, useEffect } from 'react';

import axios from 'axios';

const useSurveys = () => {

  const [surveys, setSurveys] = useState([]);

  useEffect(() => {

    const fetchSurveys = async () => {

      try {

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

        setSurveys(response.data);

        console.log(response.data)

      } catch (error) {

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

      }

    };

    fetchSurveys();

  }, []);

  return {surveys, setSurveys};

};

export default useSurveys;

Kod ispod predstavlja React komponentu nazvanu "DetaljiAnkete" koja se koristi za prikaz detalja o odabranoj anketi. Komponenta prima "id" ankete kao deo URL-a putem "useParams" hook-a, a zatim koristi taj "id" da bi izvršila HTTP GET zahtev prema serveru putem Axios biblioteke kako bi dobila podatke o toj anketi. Podaci se zatim prikazuju korisniku, uključujući naslov, opis, datume početka i završetka, status ankete, kao i listu pitanja i odgovora koji pripadaju toj anketi.

U slučaju da se podaci o anketi još uvek nisu učitali, korisniku će biti prikazana poruka "Loading...". Sve stilizacije su definisane u kodu kako bi se postigao određeni izgled komponente, uključujući pozadinu i izgled detalja ankete. Ova komponenta je dizajnirana da bude estetski privlačna i korisniku prijatna za pregled detalja ankete.

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

import axios from 'axios';

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

const DetaljiAnkete = () => {

  const { id } = useParams();

  const [anketa, setAnketa] = useState(null);

  useEffect(() => {

    const fetchAnketa = async () => {

      try {

        const response = await axios.get(`http://127.0.0.1:8000/api/ankete/${id}/pitanjaodgovori`);

        setAnketa(response.data);

      } catch (error) {

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

      }

    };

    fetchAnketa();

  }, [id]);

  if (!anketa) {

    return <div>Loading...</div>;

  }

  return (

    <div style={styles.container}>

      <div style={styles.detailsContainer}>

        <h2>{anketa.naslov}</h2>

        <p>{anketa.opis}</p>

        <p>Start Date: {anketa.datum\_pocetka}</p>

        <p>End Date: {anketa.datum\_kraja}</p>

        <p>Status: {anketa.status}</p>

        <h3>Pitanja:</h3>

        <ul>

          {anketa.pitanja.map((pitanje) => (

            <div key={pitanje.id}>

              <li>{pitanje.tekst}</li>

              <ul>

                {pitanje.odgovori.map((odgovor) => (

                  <li key={odgovor.id}>{odgovor.odgovor}</li>

                ))}

              </ul>

            </div>

          ))}

        </ul>

      </div>

    </div>

  );

};

const styles = {

  container: {

    display: 'flex',

    justifyContent: 'center',

    alignItems: 'center',

    height: '100vh',

    background:

      'linear-gradient(135deg, #e2dadb 25%, #dae2df 25%, #dae2df 50%, #a2a7a5 50%, #a2a7a5 75%, #6d696a 75%)',

  },

  detailsContainer: {

    backgroundColor: '#ffffff',

    padding: '20px',

    borderRadius: '5px',

    boxShadow: '0 4px 8px rgba(0, 0, 0, 0.1)',

  },

};

export default DetaljiAnkete;

Ovaj kod predstavlja React komponentu nazvanu "GlasajAnketa" koja se koristi za omogućavanje korisnicima da glasaju u okviru odabrane ankete. Komponenta prima "id" ankete kao deo URL-a putem "useParams" hook-a i koristi ga da bi izvršila HTTP GET zahtev prema serveru putem Axios biblioteke kako bi dobila podatke o toj anketi.

U komponenti se koristi lokalno stanje kako bi se pratio trenutni indeks pitanja koje korisnik treba da oceni, kao i niz "userRatings" za praćenje ocena koje korisnik daje na svako pitanje. Takođe, postoji stanje "submitted" koje označava da su odgovori uspešno poslati na server.

Kada korisnik oceni pitanje i klikne na odgovarajući broj, ocena se šalje na server putem HTTP POST zahteva. Nakon toga, korisnik se preusmerava na sledeće pitanje ili, ako je ocenio sva pitanja, dobija obaveštenje da su odgovori uspešno poslati i može se vratiti na listu anketa.

Ako podaci o anketi nisu još uvek učitani, korisniku će biti prikazana poruka "Loading...". Kada su odgovori uspešno poslati, korisniku se prikazuje poruka da su odgovori uspešno poslati.

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

import axios from 'axios';

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

const GlasajAnketa = () => {

  const { id } = useParams();

  const [anketa, setAnketa] = useState(null);

  const [currentQuestionIndex, setCurrentQuestionIndex] = useState(0);

  const [userRatings, setUserRatings] = useState([]);

  const [submitted, setSubmitted] = useState(false);

    const navigate = useNavigate();

  useEffect(() => {

    const fetchAnketa = async () => {

      try {

        const response = await axios.get(`http://127.0.0.1:8000/api/ankete/${id}/pitanjaodgovori`);

        setAnketa(response.data);

      } catch (error) {

      }

    };

    fetchAnketa();

  }, [id]);

  const handleRatingClick = async (rating) => {

    const korisnikId = sessionStorage.getItem('user\_id');

    const odgovor = {

      anketa\_id: id,

      korisnik\_id: korisnikId,

      pitanje\_ankete\_id: anketa.pitanja[currentQuestionIndex].id+"",

      odgovor: rating+"",

    };

    try {

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

      console.log(response.data);

    } catch (error) {

      console.error('Error submitting answer:', error);

    }

    // Pređi na sledeće pitanje ili završi anketu

    if (currentQuestionIndex + 1 < anketa.pitanja.length) {

      setCurrentQuestionIndex(currentQuestionIndex + 1);

    } else {

        alert('Hvala što ste odgovorili na sva pitanja!');

        navigate('/ankete');

    }

  };

  const handleSubmit = async () => {

    const korisnikId = sessionStorage.getItem('user\_id');

    const odgovori = [];

    userRatings.forEach((rating, index) => {

      odgovori.push({

        anketa\_id: anketa.id,

        korisnik\_id: korisnikId,

        pitanje\_ankete\_id: anketa.pitanja[index].id+"",

        odgovor: rating,

      });

    });

    try {

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

      if (response.status === 201) {

        setSubmitted(true);

      }

    } catch (error) {

      console.error('Error submitting answers:', error);

    }

  };

  if (!anketa) {

    return <div>Loading...</div>;

  }

  if (submitted) {

    return <div>Answers submitted successfully!</div>;

  }

  return (

    <div style={styles.container}>

      <div style={styles.detailsContainer}>

        <h2>{anketa.naslov}</h2>

        <p>{anketa.opis}</p>

        <p>Start Date: {anketa.datum\_pocetka}</p>

        <p>End Date: {anketa.datum\_kraja}</p>

        <p>Status: {anketa.status}</p>

        <h3>Pitanje {currentQuestionIndex + 1}:</h3>

        <ul>

          <li>{anketa.pitanja[currentQuestionIndex].tekst}</li>

          <div style={styles.ratingButtons}>

            {[1, 2, 3, 4, 5].map((rating) => (

              <button

                key={rating}

                onClick={() => handleRatingClick(rating)}

                className={userRatings[currentQuestionIndex] === rating ? 'selected' : ''}

              >

                {rating}

              </button>

            ))}

          </div>

        </ul>

        {currentQuestionIndex === anketa.pitanja.length - 1 && (

          <button onClick={handleSubmit}>Submit Ratings</button>

        )}

      </div>

    </div>

  );

};

const styles = {

  container: {

    display: 'flex',

    flexDirection: 'column',

    alignItems: 'center',

    justifyContent: 'flex-start',

    height: 'auto',

    background: 'linear-gradient(135deg, #e2dadb 25%, #dae2df 25%, #dae2df 50%, #a2a7a5 50%, #a2a7a5 75%, #6d696a 75%)',

    padding: '20px 0',

  },

  detailsContainer: {

    width: '80%',

  },

  ratingButtons: {

    display: 'flex',

    gap: '5px',

  },

};

export default GlasajAnketa;

Komponenta TabelaAnketaKorisnik koristi "useSurveys" hook kako bi dobila listu anketa za prikaz.

Na ekranu se prikazuju informacije o svakoj anketi, uključujući ID, naslov, opis i status ankete. Takođe, za svaku anketu postoji opcija za glasanje, gde se korisnika preusmerava na odgovarajuću stranicu za glasanje kada klikne na "Glasaj" link.

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

import useSurveys from '../../useSurveys';

const TabelaAnketaKorisnik = () => {

  const { surveys, setSurveys } = useSurveys();

  return (

    <div style={styles.container}>

      <h2>Surveys</h2>

      <table style={styles.table}>

        <thead>

          <tr>

            <th style={styles.th}>ID</th>

            <th style={styles.th}>Title</th>

            <th style={styles.th}>Description</th>

            <th style={styles.th}>Status</th>

            <th style={styles.th}>Actions</th>

          </tr>

        </thead>

        <tbody>

          {surveys.map(survey => (

            <tr key={survey.id}>

              <td style={styles.td}>{survey.id}</td>

              <td style={styles.td}>{survey.naslov}</td>

              <td style={styles.td}>{survey.opis}</td>

              <td style={styles.td}>{survey.status}</td>

              <td style={styles.td}>

                <Link to={`/ankete/glasaj/${survey.id}`}>Glasaj</Link>

              </td>

            </tr>

          ))}

        </tbody>

      </table>

    </div>

  );

};

const styles = {

  container: {

    display: 'flex',

    flexDirection: 'column',

    alignItems: 'center',

    justifyContent: 'flex-start',

    height: 'auto',

    background: 'linear-gradient(135deg, #e2dadb 25%, #dae2df 25%, #dae2df 50%, #a2a7a5 50%, #a2a7a5 75%, #6d696a 75%)',

    padding: '20px 0'

  },

  table: {

    width: '80%',

    borderCollapse: 'collapse',

    backgroundColor: 'white',

    marginTop: '20px',

  },

  th: {

    padding: '15px 10px',

    borderBottom: '1px solid #ddd',

    textAlign: 'left',

    backgroundColor: '#dae2df',

    color: '#6d696a',

  },

  td: {

    padding: '20px 10px',

    borderBottom: '1px solid #ddd',

  },

};

export default TabelaAnketaKorisnik;

Ispod imamo komponentu nazvanu "OpenFDAComponent" koja se koristi za prikaz podataka o povlacima hrane sa tržišta, dobijenih putem OpenFDA API-ja. Komponenta učitava podatke sa API-ja i prikazuje ih korisnicima u obliku paginirane liste.

Prvo, komponenta koristi React hookove "useState" i "useEffect" kako bi upravljala stanjem podataka, prikaza i eventualnih grešaka. Podaci se preuzimaju sa OpenFDA API-ja, ograničeni na prvih 50 povlaka hrane.

Zatim, komponenta koristi "ReactPaginate" komponentu za paginaciju, omogućavajući korisnicima da pregledaju različite strane podataka. Broj stavki po stranici je postavljen na 5, a ukupan broj stranica se računa na osnovu ukupnog broja podataka.

Nakon što se podaci preuzmu i obrade, prikazuju se u obliku kartica za svaki povlačen proizvod. Ove kartice prikazuju informacije kao što su naziv proizvoda, razlog povlačenja i datum izveštaja.

Komponenta takođe prikazuje navigaciju između stranica paginacije sa opcijama "Prethodna" i "Sledeća", omogućavajući korisnicima da pregledaju različite stranice sa podacima.

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

import axios from 'axios';

import FDAKartica from './FDAKartica';

import ReactPaginate from 'react-paginate';

import './Pagination.css';

const OpenFDAComponent = () => {

  const [data, setData] = useState([]);

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

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

  const [currentPage, setCurrentPage] = useState(0);

  const itemsPerPage = 5;

  const pageCount = Math.ceil(data.length / itemsPerPage);

  const handlePageClick = (event) => {

    setCurrentPage(event.selected);

  };

  const currentItems = data.slice(currentPage \* itemsPerPage, (currentPage + 1) \* itemsPerPage);

  useEffect(() => {

    const fetchData = async () => {

      try {

        const result = await axios.get('https://api.fda.gov/food/enforcement.json?limit=50');

        setData(result.data.results);

      } catch (error) {

        setError(error);

      }

      setIsLoading(false);

    };

    fetchData();

  }, []);

  if (isLoading) return <div>Loading...</div>;

  if (error) return <div>Error: {error.message}</div>;

  return (

    <div style={styles.container}>

      <h2>OpenFDA Food Recall Data</h2>

      <ul>

        {currentItems.map((item, index) => (

            <FDAKartica

            key={index}

            proizvod={item.product\_description}

            razlogOpoziva={item.reason\_for\_recall}

            datumObjave={item.report\_date}

            product\_description={item.product\_description}

        />

        ))}

      </ul>

      <ReactPaginate

            previousLabel={"Prethodna"}

            nextLabel={"Sledeća"}

            pageCount={pageCount}

            onPageChange={handlePageClick}

            containerClassName={"pagination"}

            activeClassName={"activePageItem"}

            pageLinkClassName={"pageItem"}

            />

    </div>

  );

};

const styles = {

    container: {

      display: 'flex',

      flexDirection: 'column',

      alignItems: 'center',

      justifyContent: 'flex-start',

      height: 'auto',

      background: 'linear-gradient(135deg, #e2dadb 25%, #dae2df 25%, #dae2df 50%, #a2a7a5 50%, #a2a7a5 75%, #6d696a 75%)',

      padding: '20px 0'

    },

};

export default OpenFDAComponent;

# Korisničko uputstvo

Kako bi korisnici mogli bilo sa da rade na nasoj aplikaciji potrebno je da se prvo uloguju

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

Opis je automatski generisan

Nakon sto se uloguje korisnik prikazuje mu se stranica gde ima pregled svih trenutno otvorenih anketa

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

Opis je automatski generisan

Klikom na dugme glasaj korisnik se odvodi na stranicu na kojoj moze da glasa

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

Opis je automatski generisan

Odabirom opcije FDA korisnik moze da vidi informacije o proizvodima koji su povuceni iz prodaje iz bezbedonosnih razloga

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

Opis je automatski generisan

Ukoliko se prijavi korisnik koji ima ulogu admin, njemu se otvara stranica za prikaz svih anketa

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

Opis je automatski generisan

Klikom na dugme detalji adminu se prikazuju odgovori prikupljeni na ankete

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

Opis je automatski generisan

Klikom na dugme za dodavanje nove ankete adminu se otvara forma za unos nove ankete u bazu

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

Opis je automatski generisan