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

Aplikacija koja je razvijena služi kao platforma za upravljanje i deljenje akademskih publikacija u okviru istraživačkih i obrazovnih sredina, poput univerziteta ili naučnih institucija. Glavni fokus aplikacije je olakšavanje skladištenja, preuzimanja i deljenja naučnih radova i istraživačkih materijala.

U osnovi, aplikacija nudi intuitivan interfejs za postavljanje istraživačkih dokumenata. Korisnici mogu podnositi svoje publikacije, uključujući bitne detalje kao što su naslov, autori, sažetak i ključne reči. Svaka publikacija može biti dodatno opremljena priloženim fajlovima, koji se mogu preuzeti .

Jedna od ključnih funkcionalnosti aplikacije je mogućnost pretraživanja publikacija. Korisnici mogu pretraživati sadržaj na osnovu različitih parametara, uključujući naslov, autore, sažetak i ključne reči, što omogućava lako pronalaženje specifičnih istraživačkih radova. Takođe, aplikacija omogućava sortiranje publikacija po abecednom redu, što doprinosi boljoj organizaciji i lakšem snalaženju.

# Implementacija

Kod ispod predstavlja komponentu "HomePage" u React aplikaciji.

Prvo, uvezene su potrebne biblioteke i komponente, uključujući React, useState i useEffect za upravljanje stanjem i efektima, axios za izvršavanje HTTP zahtjeva, te različite ikone iz biblioteke "react-icons" za prikazivanje ikona.

Zatim se definira funkcionalna komponenta "HomePage". U ovoj komponenti postoje delovi koda koji su namenjeni za prikazivanje i paginaciju "zanimljivih činjenica" koje su preuzete sa spoljnog API-ja.

Nakon toga, u samom HTML delu komponente, prikazuju se informacije o laboratoriji, ikone koje predstavljaju različite naučne oblasti, i nekoliko sekcija sa tekstom o značaju nauke i zanimljivim činjenicama.

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

import axios from 'axios';

import { FaMicroscope, FaFlask, FaAtom, FaBrain, FaLaughBeam, FaRegLaughBeam } from 'react-icons/fa';

import FunFact from './FunFact';

const HomePage = () => {

    // const [facts, setFacts] = useState([]);

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

    // const [factsPerPage] = useState(10);

    // useEffect(() => {

    //   const fetchFacts = async () => {

    //     try {

    //       const response = await axios.get('https://api.api-ninjas.com/v1/facts?limit=50', {

    //         headers: { 'X-Api-Key': '7xiJG3ZG/DVXBFQcpnUANw==DCKsOuWEdluVhptV' }

    //       });

    //       setFacts(response.data);

    //     } catch (error) {

    //       console.error('Error fetching facts:', error);

    //     }

    //   };

    //   fetchFacts();

    // }, []);

    // const lastFactIndex = currentPage \* factsPerPage;

    // const firstFactIndex = lastFactIndex - factsPerPage;

    // const currentFacts = facts.slice(firstFactIndex, lastFactIndex);

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

    // // Calculate the total number of pages

    // const pageCount = Math.ceil(facts.length / factsPerPage);

    // // Render pagination dots

    // const PaginationDots = () => {

    //   let dots = [];

    //   for (let i = 1; i < pageCount; i++) {

    //     dots.push(

    //       <span

    //         key={i}

    //         className={`dot ${i === currentPage ? 'active' : ''}`}

    //         onClick={() => paginate(i)}

    //       />

    //     );

    //   }

    //   return <div className="pagination">{dots}</div>;

    // };

  return (

    <div className="homepage">

     <h1><FaAtom /> Dobrodošli u našu naučno-istraživačku laboratoriju</h1>

      <p>

        Otkrijte svijet mikroskopski malih čuda i naprednih istraživanja. Naš tim

        stručnjaka je posvećen razumijevanju temelja znanosti kako bi

        unaprijedili medicinu, tehnologiju i naše razumijevanje svemira.

      </p>

      <div className="icons">

        <div><FaMicroscope /> Mikroskopija</div>

        <div><FaFlask /> Hemija</div>

        <div><FaAtom /> Fizika</div>

        <div><FaBrain /> Neuroscience</div>

      </div>

      <p>

        Otkrijte svijet mikroskopski malih čuda i naprednih istraživanja. Naš tim

        stručnjaka je posvećen razumijevanju temelja znanosti kako bi

        unaprijedili medicinu, tehnologiju i naše razumijevanje svemira.

      </p>

      <section className="science-importance">

        <h2><FaLaughBeam /> Nauka je važna i zabavna</h2>

        <p>

          U svijetu gdje se granice znanja neprestano pomjeraju, nauka nudi

          alat za razumijevanje složenosti oko nas. Osim što doprinosi našem

          svakodnevnom životu, nauka nas uči kritičkom razmišljanju i rješavanju problema.

          A istraživanje novih koncepta može biti izuzetno zabavno!

        </p>

      </section>

      <section className="fun-facts">

        <h2><FaRegLaughBeam /> Zanimljive činjenice</h2>

        <ul>

          {/\* {currentFacts.map((fact, index) => (

            <FunFact key={index} fact={fact.fact} />

          ))} \*/}

        </ul>

        {/\* <PaginationDots /> \*/}

      </section>

    </div>

  );

};

export default HomePage;

import React from 'react';

const FunFact = ({ fact }) => {

  return (

    <li className="fun-fact">

      <p>{fact}</p>

    </li>

  );

};

export default FunFact;

Dalje, imamo komponentu "PublicationUploadForm" koja se koristi za unos informacija o publikaciji i njeno preuzimanje kao datoteke.

Na početku se uvoze React i komponente: "InputField" i "TextareaField". Komponenta "InputField" koristi se za unos teksta u polje, dok "TextareaField" omogućava unos teksta u veće polje za unos teksta. Ove dve komponente predstavljaju reusable komponente.

Zatim se definiše funkcionalna komponenta "PublicationUploadForm". Unutar nje, koristi se React Hook "useState" kako bi se inicijalizovao objekat "publication" koji će sadržati podatke o publikaciji, uključujući naslov, autore, apstrakt, ključne reci i datoteku.

Imamo sledeće tri funkcije:

* "handleInputChange" - Ova funkcija se poziva kada se promeni sadržaj polja za unos teksta ili teksta unutar komponenti "InputField" i "TextareaField". Ona ažurira stanje objekta "publication" s novim vrednostima na temelju imena polja koje se menja.
* "handleFileChange" - Ova funkcija se poziva kada se odabere datoteka za unos. Ona ažurira datoteku u objektu "publication" s odabranom datotekom.
* "handleSubmit" - Ova funkcija se poziva kada se obrazac za unos podataka pošalje. Ona prvo proverava da li je odabrana datoteka, a zatim čita tu datoteku kao Base64 string i sprema je u lokalno skladište zajedno s ostalim informacijama o publikaciji. Nakon toga, prikazuje se poruka o uspešnom unosu ili poruka o grešci ako dođe do problema.

import React, { useState } from 'react';

import InputField from './InputField';

import TextareaField from './TextareaField';

const PublicationUploadForm = () => {

  const [publication, setPublication] = useState({

    title: '',

    authors: '',

    abstract: '',

    keywords: '',

    file: null

  });

  const handleInputChange = (e) => {

    const { name, value } = e.target;

    setPublication({ ...publication, [name]: value });

  };

  const handleFileChange = (e) => {

    setPublication({ ...publication, file: e.target.files[0] });

  };

  const handleSubmit = (e) => {

    e.preventDefault();

    if (publication.file) {

      const reader = new FileReader();

      reader.onload = function (event) {

        try {

          const base64File = event.target.result;

          const existingFiles = JSON.parse(localStorage.getItem('uploadedFiles') || '[]');

          const newFile = {

            title: publication.title,

            authors: publication.authors,

            abstract: publication.abstract,

            keywords: publication.keywords,

            file: base64File

          };

          existingFiles.push(newFile);

          localStorage.setItem('uploadedFiles', JSON.stringify(existingFiles));

          alert('Publication uploaded successfully!');

        } catch (error) {

          console.error('Error saving the file:', error);

          alert('Failed to upload the publication.');

        }

      };

      reader.onerror = function (error) {

        console.error('Error reading the file:', error);

      };

      reader.readAsDataURL(publication.file);

    }

    setPublication({

      title: '',

      authors: '',

      abstract: '',

      keywords: '',

      file: null

    });

  };

  return (

    <form onSubmit={handleSubmit}>

      <InputField

        label="Title"

        name="title"

        value={publication.title}

        onChange={handleInputChange}

      />

      <InputField

        label="Authors"

        name="authors"

        value={publication.authors}

        onChange={handleInputChange}

      />

      <TextareaField

        label="Abstract"

        name="abstract"

        value={publication.abstract}

        onChange={handleInputChange}

      />

      <InputField

        label="Keywords"

        name="keywords"

        value={publication.keywords}

        onChange={handleInputChange}

      />

      <label>

        Upload File:

        <input

          type="file"

          name="file"

          onChange={handleFileChange}

          required

        />

      </label>

      <button type="submit">Submit Publication</button>

    </form>

  );

};

export default PublicationUploadForm;

const TextareaField = ({ label, name, value, onChange }) => {

    return (

      <label>

        {label}:

        <textarea

          name={name}

          value={value}

          onChange={onChange}

          required

        />

      </label>

    );

  };

  export default TextareaField;

const InputField = ({ label, name, value, onChange, type = 'text' }) => {

    return (

      <label>

        {label}:

        <input

          type={type}

          name={name}

          value={value}

          onChange={onChange}

          required

        />

      </label>

    );

  };

  export default InputField;

Poslednja komponenta, "PublicationsList", koja se koristi za prikazivanje liste publikacija, omogućava pretragu i sortiranje tih publikacija. Takođe, koristi se komponenta "PublicationItem" za prikaz pojedinačnih stavki na listi publikacija. Pored toga, kod uključuje i funkciju "useSearch" koja se koristi za filtriranje publikacija na osnovu pretrage. Kratak pregled koda:

* useState i useEffect: Komponenta "PublicationsList" koristi "useState" i "useEffect" hook-ove kako bi upravljala stanjem komponente. Stanje "publications" čuva listu publikacija, "searchTerm" čuva trenutni unos za pretragu, a "sortAscending" označava trenutni redosled sortiranja. Unutar "useEffect" hook-a, komponenta čita prethodno spremljene publikacije iz lokalnog skladišta i postavlja ih u stanje "publications" kad se komponenta montira.
* downloadFile funkcija: Ova funkcija se koristi za preuzimanje datoteke publikacije. Kada korisnik klikne na dugme "Download File" pored publikacije, funkcija kreira privremeni link za preuzimanje i pokreće preuzimanje datoteke.
* handleSearchChange funkcija: Ova funkcija se poziva kada se promeni unos za pretragu. Ona ažurira stanje "searchTerm" s trenutnim unosom korisnika.
* handleSortToggle funkcija: Ova funkcija se poziva kada korisnik klikne na dugme za promenu redosleda sortiranja.
* useSearch funkcija: Ova funkcija je prilagođena kuka koja se koristi za filtriranje publikacija na osnovu unosa za pretragu. Ona prima listu publikacija i trenutni unos za pretragu, a zatim filtrira publikacije koje odgovaraju kriterijumima pretrage, kao što su naslov, autori, apstrakt i ključne reci. Filtrirane publikacije se čuvaju u stanju "filteredItems" i ažuriraju se svaki put kad se unos za pretragu promeni

# Korisničko uputstvo

Prilikom pokretanja aplikacije otvara se početna stranica kao na slici ispod

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

Opis je automatski generisan

Na početnoj stranici korsnici imaju stranicu dobrodošlice, kao i zanimljive činjenice koje se učitavaju sa spoljnog apija. Zbog velikog broja zanimljivih činjenica, dodata je i elegantna paginacija.

Sledeća stranica je stranica preko koje korisnici mogu da uploaduju svoje publikacije, popunjavajući formu koja je prikazana na slici ispod.

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

Opis je automatski generisan

Treća stranica nam prikazuje koje su sve objavljene publikacije. Na ovoj stranici korisnici mogu pretraživati ili sortirati publikacije.

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

Opis je automatski generisan