ДНІПРОВСЬКИЙ ДЕРЖАВНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

КАФЕДРА «ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМ»

**КУРСОВА РОБОТА**

з дисципліни «Технології створення WEB-застосувань»

на тему: Розробка веб-застосунку «Список наукових публікацій» засобами React

Здобувача вищої освіти 4 курсу групи *ПЗ-21-1ду*

Спеціальності *121 – Інженерія програмного забезпечення Павлюченка В.О.*

Керівник *ст. викл. каф. ПЗС, Іскандарова-Мала А.О.*

Національна шкала

Кількість балів: Оцінка: ECTS

Члени комісії:

(підпис) (прізвище та ініціали)

(підпис) (прізвище та ініціали)

(підпис) (прізвище та ініціали)

Кам’янське 2023

Дніпровський державний технічний університет Факультет *комп’ютерних технологій та енергетики* Кафедра *Програмне забезпечення систем* Спеціальність *121 Інженерія програмного забезпечення*

**ЗАВДАННЯ**

**на курсову роботу**

з дисципліни *Технології створення WEB-застосувань*

здобувачу групи *ПЗ-21-1ду*

ПІБ *Павлюченку Владиславу Олексійовичу*

тема курсової роботи *Розробка веб-застосунку «Список наукових публікацій» засобами React*

дата видачі завдання

дата здавання

Завдання видав

керівник курсової роботи

ст. викл. каф. ПЗС А.О. Іскандарова-Мала

Завдання прийняв до виконання

здобувач гр. ПЗ-21-1ду В.О. Павлюченко

Міністерство освіти і науки України

Дніпровський державний технічний університет

Здобувач вищої освіти *Павлюченко В.О.* групи *ПЗ- 21-1ду*

Тема курсової роботи: «*Розробка веб-застосунку «Список наукових публікацій» засобами React»*

Оцінювання

|  |  |  |
| --- | --- | --- |
| Критерій оцінювання | Максимальна  оцінка, балів | Оцінка, балів |
| **ОЦІНКА ЕТАПУ ПІДГОТОВКИ РОБОТИ** | **20** |  |
| * Дотримання календарного плану * Відповідність роботи постановці задачі * Самостійність * Ініціативність | 5  5  5  5 |  |
| **ОЦІНКА ЗА ЗВІТ І** | **20** |  |
| * Структура, об’єм, оформлення * Реферат * Вступ, постановка задачі, перелік посилань * Опис предметної області (задачі) | 5  5  5  5 |  |
| **ОЦІНКА ЗА ЗВІТ ІІ** | **40** |  |
| * Логічна структура (компотенти) * Опис програмного продукту * Функціональна повнота * Інтерфейсна частина * Опис тестування * Керівництво користувача | 10  10  5  5  5  5 |  |
| **ОЦІНКА ЗА ПРАКТИЧНУ РЕАЛІЗАЦІЮ** | **10** |  |
| * Степінь закінченості * Демонстрація | 5  5 |  |
| **РІВЕНЬ СКЛАДНОСТІ (наявність спеціалізації)** | **10** |  |
| **Всього** | **100б** |  |

Захист відбувся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Загальна оцінка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Керівник курсової роботи \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.О. Іскандарова-Мала

**РЕФЕРАТ**

Курсова робота: 26 с., 10 рис., 1 табл., 4 джерела.

**Об’єкт дослідження** – процес керування онлайн списком.

**Мета роботи** – створення інтуїтивно зрозумілого та функціонального веб-додатку, здатного ефективно організовувати, фільтрувати та представляти інформацію про наукові публікації.

Для програмної реалізації клієнтської частини використано технологію React. Мова програмування – JavaScript. Бібліотека стилю – SASS.

У розділі 1 «Аналіз предметної області» наведено результат аналізу предметної області, опису вимог до веб-застосунку, що розробляється, та актуальність розробки.

Розділ 2 «Архітектура та проєктування» містить опис створеної структури проєкту, його компоненти, а також метод отримання даних.

Розділ 3 «Розробка веб-застосунку» містить опис усіх веб-сторінок системи та діалоги користувача з системою. Також описуються механізми створення навігації між сторінками.

СПИСОК НАУКОВИХ ПУБЛІКАЦІЙ, REACT, JAVASCRIPT, API, КОМПОНЕНТ, НАВІГАЦІЯ

**ЗМІСТ**

[ВСТУП 6](#_Toc153434482)

[1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 7](#_Toc153434483)

[2 АРХІТЕКТУРА ТА ПРОЄКТУВАННЯ 8](#_Toc153434484)

[2.1 Структура веб-застосунку 8](#_Toc153434485)

[2.2 Список компонентів 8](#_Toc153434486)

[2.3 Робота з даними 9](#_Toc153434487)

[3 РОЗРОБКА ВЕБ-ЗАСТОСУНКУ 10](#_Toc153434488)

[3.1 Навігація у веб-застосунку 10](#_Toc153434489)

[3.2 Веб-сторінки 10](#_Toc153434490)

[ВИСНОВКИ 16](#_Toc153434491)

[ПЕРЕЛІК ПОСИЛАНЬ 17](#_Toc153434492)

[ДОДАТОК А 18](#_Toc153434493)

[ДОДАТОК Б 23](#_Toc153434494)

ВСТУП

В сучасному інформаційному суспільстві створення ефективних і зручних веб-застосунків стає дедалі важливішим елементом розвитку наукової та освітньої спільноти. Веб-застосунки надають унікальні можливості для управління та представлення різноманітних даних, таких як наукові публікації, забезпечуючи дослідникам, науковцям і студентам ефективний доступ до актуальної інформації.

Актуальність даної роботи зумовлена стрімким розвитком наукових досліджень в умовах сучасного інформаційного суспільства. З появою нових технологій виникає потреба в ефективних засобах для систематизації та управління науковими публікаціями. Розробка веб-застосунка «Список наукових публікацій» на платформі React спрямована на задоволення цієї потреби. Нині вчені та дослідники потребують інструменту, який би полегшив процес організації, зберігання та моніторингу своїх наукових досягнень. Робота в цій галузі актуальна, оскільки вона пропонує інноваційний підхід до створення веб-додатка, здатного ефективно управляти науковим портфоліо і відповідати вимогам сучасної наукової спільноти.

Метою роботи є створення інтуїтивно зрозумілого та функціонального веб-додатку, здатного ефективно організовувати, фільтрувати та представляти інформацію про наукові публікації.

В поточній курсовій роботі для розробки всього функціоналу була використана технологія React [1], яка є бібліотекою мови програмування JavaScript, розроблена компанією Meta. Вона використовується для створення користувацьких інтерфейсів (UI) у веб-додатках. Для стилізації була використана бібліотека SASS [2]. React дозволяє створювати компоненти, які представляють собою багаторазові будівельні блоки інтерфейсу, що використовуються повторно, та ефективно керувати станом програми.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Предметна область «Список наукових публікацій» охоплює великий набір питань, пов’язаних з ефективним управлінням, організацією та відстеженням наукових публікацій з використанням технології React.

Щодня мільйони науковців зі всього світу працюють над науковими роботами, після чого надсилають їх на публікацію до відповідних наукових журналів. Тому база робіт постійно оновлюється з додаванням нових наукових праць. Через це використання API (Application Programming Interface) є ефективним варіантом, оскільки це створює можливість для багатьох розробників отримувати актуальні наукові роботи та відображати їх у необхідному їм застосунку.

В контексті організації наукових публікацій, робота передбачає створення веб-застосунку з використанням спеціального API [3], який надає доступ до всієї збереженої необхідної інформації.

Розробка веб-додатка також передбачає використання React для створення динамічного користувацького інтерфейсу. Застосування компонентного підходу, роботи зі станом і властивостями React, а також забезпечення адаптивності інтерфейсу для різних пристроїв є важливими аспектами проекту.

2 АРХІТЕКТУРА ТА ПРОЄКТУВАННЯ

2.1 Структура веб-застосунку

По завершенню процесу аналізу предметної області було виявлено, що веб-застосунок повинен містити наступні сторінки:

- головна сторінка;

- сторінка конкретної наукової публікації.

На головній сторінці містяться компоненти хедер, футер, а також увесь список наукових публікацій, який було отримано за допомогою запитів до зовнішнього API.

2.2 Список компонентів

Перелік усіх необхідних компонентів веб-застосунку наведений в таблиці 2.1.

Таблиця 2.1 – Перелік компонентів веб-застосунку

|  |  |  |
| --- | --- | --- |
| № | Компонент | Опис |
| 1 | App.js | Початковий компонент, який створюється разом з проєктом |
| 2 | MainPage.js | Компонент, який формує головну сторінку веб-застосунку |
| 3 | CardPage.js | Компонент, який формує сторінку окремої публікації |
| 4 | Header.js | Компонент, який формує загальний для сторінок хедер |
| 5 | Footer.js | Компонент, який формує загальний для сторінок футер |
| 6 | Cards.js | Компонент, який формує перелік усіх наукових публікацій, використовуючи веб-розмітку завдяки flex |
| 7 | Card.js | Компонент однієї наукової публікації |
| 8 | FullCard.js | Компонент, який використовується для відображення повної інформації про конкретну наукову публікацію на окремій сторінці |

2.3 Робота з даними

В поточній роботі дані були отримані завдяки запитам до зовнішнього API публичного сервісу RapidAPI [3].

API [3] показує всі останні наукові публікації галузі медицини про поширені захворювання з Національної бібліотеки NIH.

3 РОЗРОБКА ВЕБ-ЗАСТОСУНКУ

3.1 Навігація у веб-застосунку

Навігація у веб-застосунку виконується переважно завдяки хедеру та посиланню на конкретну наукову публікацію. Для налаштування переходів між сторінками необхідно визначити маршрути веб-застосунку завдяки бібліотеці react-router-dom та компонентам BrowserRouter, Routes, Route [4]:

function App() {

    return (

        <div>

            <BrowserRouter>

                <Routes>

                    <Route path='/' element={<MainPage />}></Route>

                    <Route path='/CardPage' element={<CardPage />}></Route>

                </Routes>

            </BrowserRouter>

        </div>

    )

}

На рисунку 3.1 зображений компонент хедера, на рисунку 3.2 – футера.



Рисунок 3.1 – Вигляд компоненту хедера

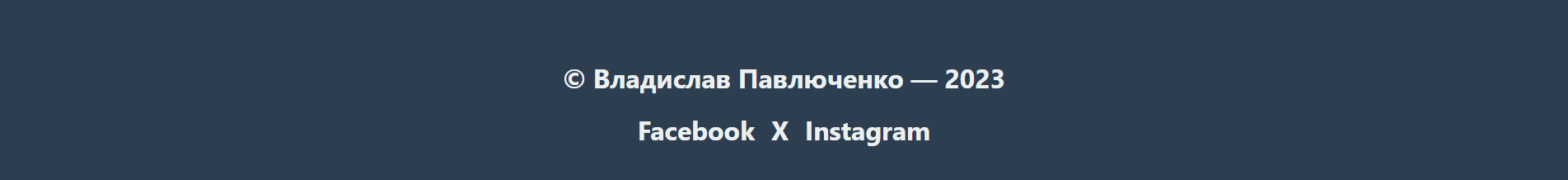


Рисунок 3.2 – Вигляд компоненту футера

3.2 Веб-сторінки

Вигляд стартової сторінки зображений на рисунку 3.3.

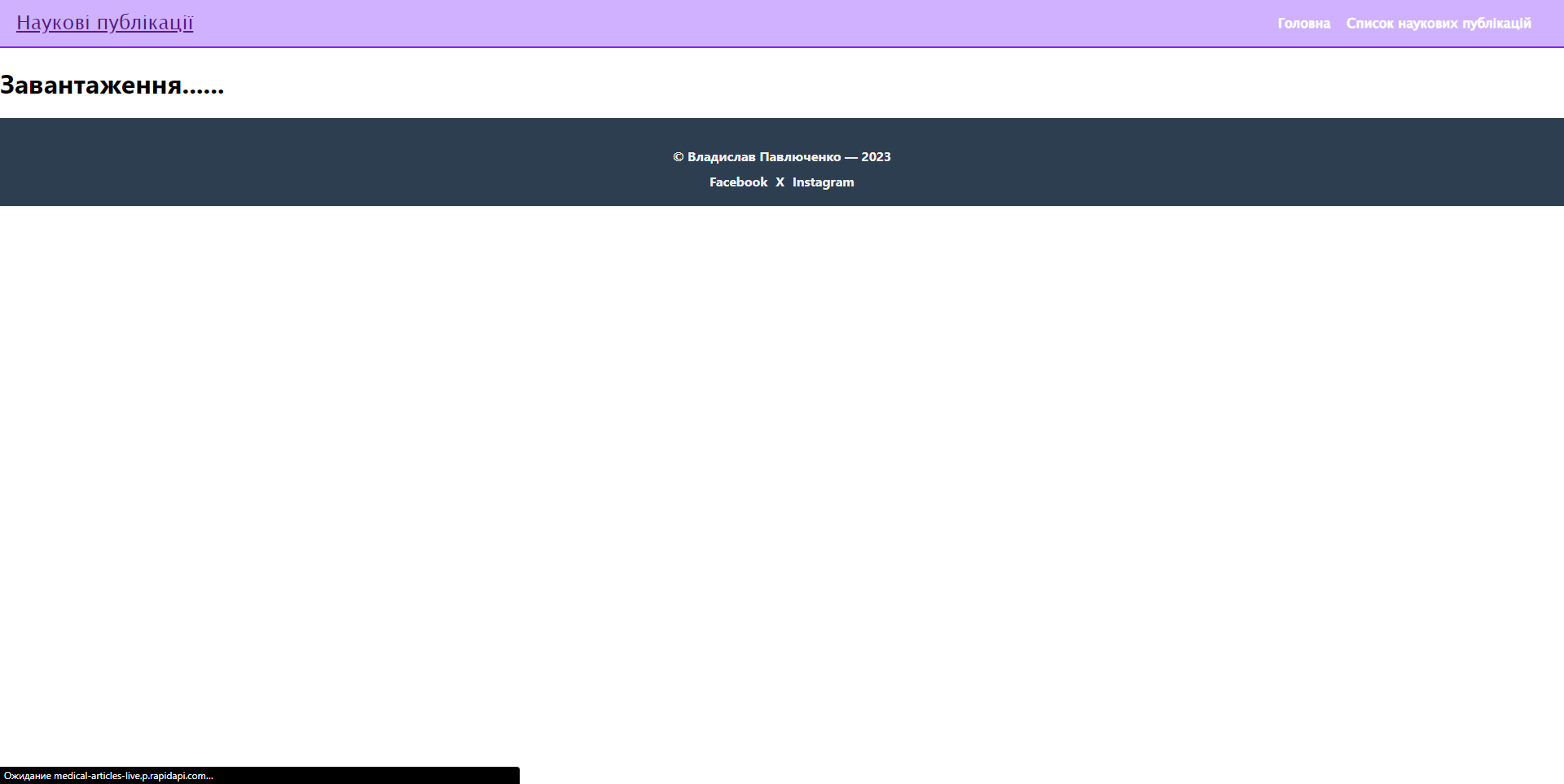


Рисунок 3.3 – Початковий вигляд стартової сторінки

На рисунку 3.3 використаний стан: поки дані не будуть отримані, користувач буде бачити надпис «Завантаження». Якщо дані успішно будуть завантажені завдяки API – користувач побачить список наукових публікацій. Якщо виникне помилка під час отримання інформації – користувач побачить повідомлення про помилку (рис. 3.4) та, на жаль, не зможе продовжити роботу. Слід зауважити, що можливих помилок небагато: або зникло підключення до інтернету, або API перестав надавати будь-яку інформацію.

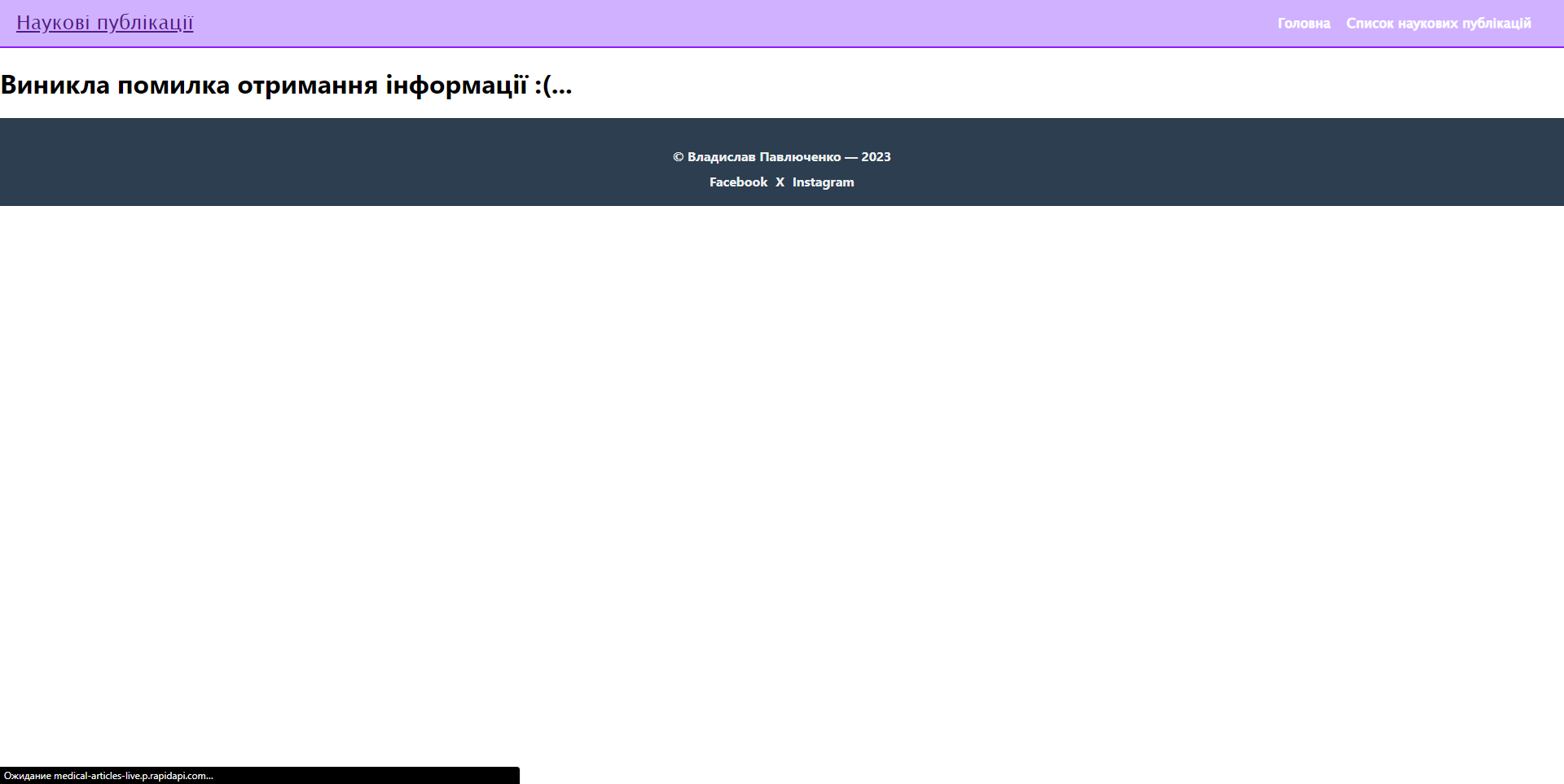


Рисунок 3.4 – Помилка при отриманні інформації через API

На рисунку 3.5 зображена веб-сторінка з усіма науковими публікаціями.



Рисунок 3.5 – Список наукових публікацій при успішному отриманні його через API

Для того, аби перейти до деталей будь-якої наукової публікації, необхідно просто натиснути на гіперпосилання. Після натискання користувача буде перенаправлено на іншу сторінку, яка призначення для відображення деталей про обрану наукову публікацію (рис. 3.6).

Зі сторінки, зображеної на рисунку 3.6, можна повернутися назад натисканням кнопки «Go Back».

Також веб-застосунку сторінки адаптовані до різних розмірів екрану (рис. 3.7-3.11).

Рисунок 3.6 – Сторінка з деталями про обрану наукову публікацію



Рисунок 3.7 – Головна сторінка для більш широкого екрану



Рисунок 3.8 – Головна сторінка для менш широкого екрану

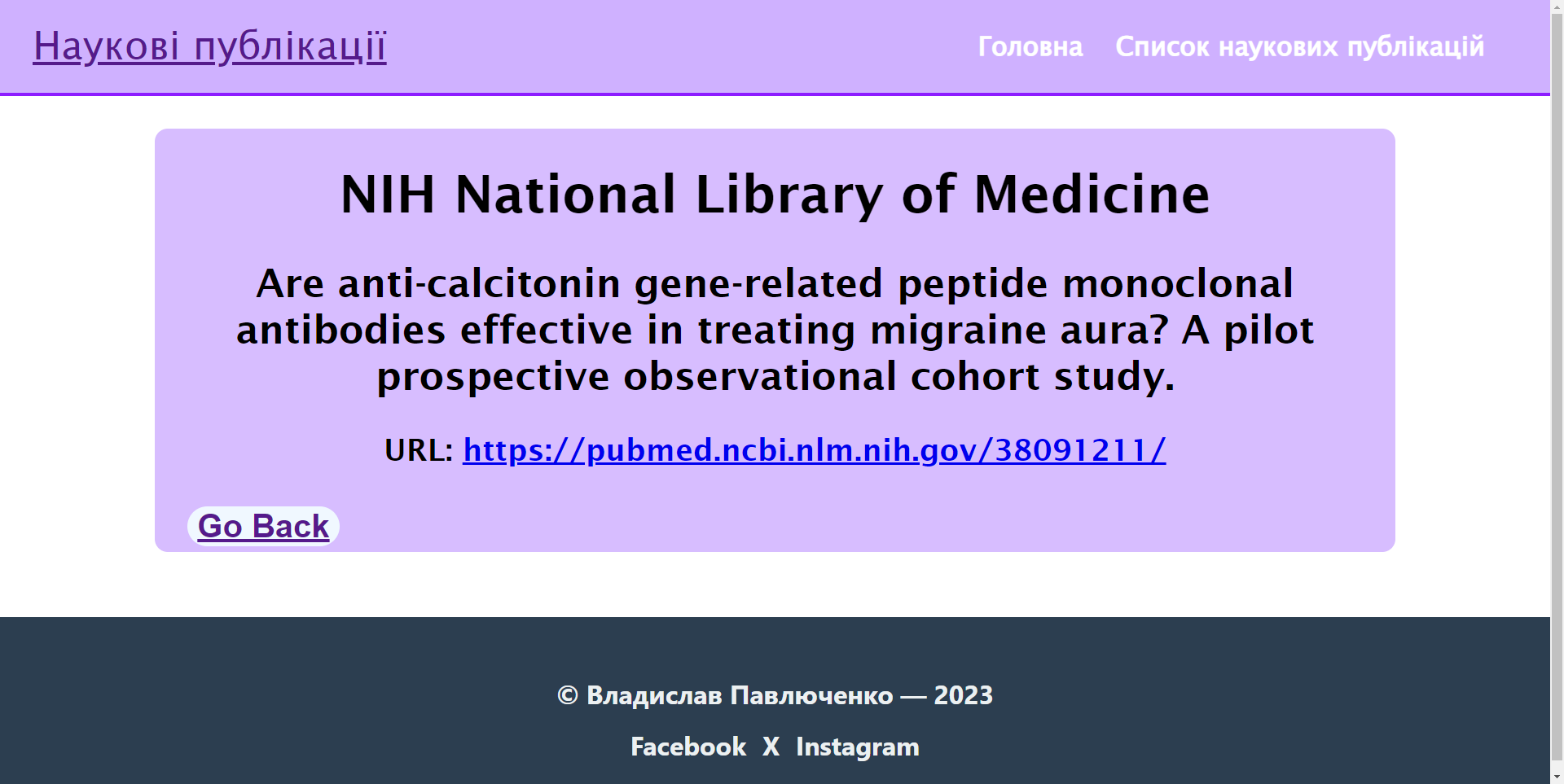


Рисунок 3.9 – Варіант адаптації другої сторінки для екрану, ширина якого менша за 1050 пікселів

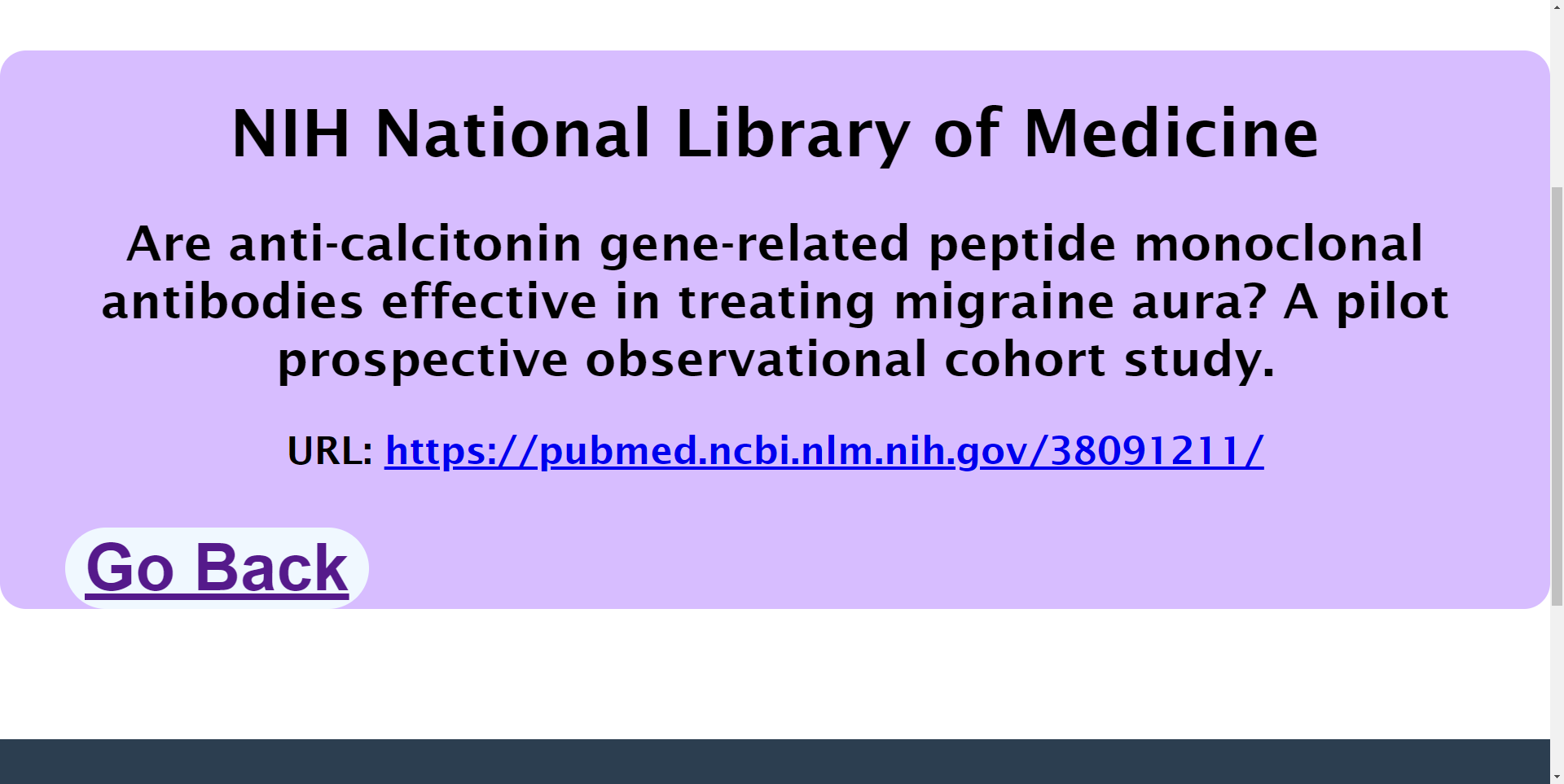


Рисунок 3.10 – Варіант адаптації другої сторінки для екрану, ширина якого менша за 480 пікселів

Адаптація під різні розміри екрану може бути виконана завдяки веб-розмітці або медіа-запитам:

@media (max-width: 1050px) {

        width: 80%;

    }

    @media (max-width: 730px) {

        width: 70%;

        font-size: 14px;

    }

    @media (max-width: 480px) {

        font-size: 10px;

        width: 100%;

        height: 50%;

    }

ВИСНОВКИ

**Об’єкт дослідження** – процес керування онлайн списком.

**Мета роботи** – створення інтуїтивно зрозумілого та функціонального веб-додатку, здатного ефективно організовувати, фільтрувати та представляти інформацію про наукові публікації.

В поточній курсовій роботі для розробки всього функціоналу була використана технологія React, яка є бібліотекою мови програмування JavaScript, розроблена компанією Meta. Вона використовується для створення користувацьких інтерфейсів (UI) у веб-додатках. Для стилізації була використана бібліотека SASS.

У розділі 1 «Аналіз предметної області» наведено результат аналізу предметної області, опису вимог до веб-застосунку, що було розроблено, та актуальність розробки.

Розділ 2 «Архітектура та проєктування» містить опис створеної структури проєкту, його компоненти, а також метод отримання даних.

Розділ 3 «Розробка веб-застосунку» містить опис усіх веб-сторінок системи та діалоги користувача з системою. Також описуються механізми створення навігації між сторінками.

ПЕРЕЛІК ПОСИЛАНЬ

1. React Documentation. URL: https://ru.legacy.reactjs.org/docs/getting-started.html (дата звернення: 12.12.2023).

2. SASS Documentation. URL: https://sass-lang.com/documentation/ (дата звернення: 12.12.2023).

3. Використане API для роботи. URL: https://rapidapi.com/ayronadas/api/medical-articles-live (дата звернення: 12.12.2023).

4. React Router. URL: https://reactrouter.com/en/main (дата звернення: 12.12.2023)

ДОДАТОК А

Лістинг програмного коду

**App.js**

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

import MainPage from './pages/MainPage';

import CardPage from './pages/CardPage';

function App() {

    return (

        <div>

            <BrowserRouter>

                <Routes>

                    <Route path='/' element={<MainPage />}></Route>

                    <Route path='/CardPage' element={<CardPage />}></Route>

                </Routes>

            </BrowserRouter>

        </div>

    )

}

export default App;

**Header.js**

import React from 'react';

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

import './Header.scss';

const Header = () => {

    return (

        <header>

            <div className="logo"><Link to="/">Наукові публікації</Link></div>

            <nav>

                <ul>

                    <li><Link to="/">Головна</Link></li>

                    <li><Link to="/">Список наукових публікацій</Link></li>

                </ul>

            </nav>

        </header>

    );

}

export default Header;

**Footer.js**

import React from 'react';

import './Footer.scss';

const Footer = () => {

    return (

        <footer>

            <div className="footer-content">

                <p>&copy; Владислав Павлюченко — 2023</p>

                <div className="social-links">

                    <a href="https://facebook.com" target="\_blank">Facebook</a>

                    <a href="https://x.com" target="\_blank">X</a>

                    <a href="https://instagram.com" target="\_blank">Instagram</a>

                </div>

            </div>

        </footer>

    );

}

export default Footer;

**Card.js**

import { Link } from 'react-router-dom'

import './Card.scss'

export default function Card(props) {

    return (

        <div title={props.article.title} className='article'>

            <h3 className='title'><Link to='CardPage' state={{ data: props.article.url.split('/').slice(-2)[0] }}>Наукова стаття №{props.index + 1}</Link></h3>

            <p>{props.article.source}</p>

        </div>

    )

}

**Cards.js**

import Card from './Card'

import './Cards.scss'

export default function Cards(props) {

    return (

        <div>

            <div className="articles">

                {

                    props.articles.map((article, index) =>

                        <Card article={article} index={index} />

                    )

                }

            </div>

        </div>

    )

}

**FullCard.js**

import { Link } from 'react-router-dom'

import './FullCard.scss'

export default function FullCard(props) {

    return (

        <div style={{display: 'flex', justifyContent: "center"}}>

            <div className='card'>

                <h1>{props.article.source}</h1>

                <h2>{props.article.title}</h2>

                <h3>URL: <Link to={props.article.url}>{props.article.url}</Link></h3>

                <button><Link to='/'>Go Back</Link></button>

            </div>

        </div>

    )

}

**MainPage.js**

import { useEffect, useState } from "react"

import Cards from "../components/Cards"

import axios from "axios"

import './MainPage.scss'

import Header from "../components/Header"

import Footer from "../components/Footer"

export default function MainPage() {

    const [message, setMessage] = useState('Завантаження...')

    const [articles, setArticles] = useState([])

    useEffect(() => {

        axios.get(

            'https://medical-articles-live.p.rapidapi.com/journals',

            {

                headers: {

                    "X-RapidAPI-Key": 'ee4f9cf3d5mshe450216970bd5c4p1e7f7bjsna37c7b1d3aee',

                    'X-RapidAPI-Host': 'medical-articles-live.p.rapidapi.com'

                }

            }

        ).then(res => {

            setArticles(res.data)

            setMessage(false)

        }).catch(res => {

            setMessage('Виникла помилка отримання інформації :(')

        })

    }, [])

    return (

        <div>

            <Header />

            {message && <h1>{message}...</h1>}

            {!message &&

                <div>

                    <hr />

                    <h2>Список наукових публікацій</h2>

                    <hr />

                    <Cards articles={articles} />

                </div>

            }

            <Footer />

        </div>

    );

}

**CardPage.js**

import axios from "axios";

import { useEffect, useState } from "react";

import { useLocation } from "react-router-dom"

import FullCard from "../components/FullCard";

import Header from "../components/Header";

import Footer from "../components/Footer";

export default function CardPage() {

    const [article, setArticle] = useState()

    const [message, setMessage] = useState('Завантаження....')

    const location = useLocation()

    const id = location.state?.data;

    useEffect(() => {

        axios.get(

            'https://medical-articles-live.p.rapidapi.com/journals',

            {

                headers: {

                    "X-RapidAPI-Key": 'ee4f9cf3d5mshe450216970bd5c4p1e7f7bjsna37c7b1d3aee',

                    'X-RapidAPI-Host': 'medical-articles-live.p.rapidapi.com'

                }

            }

        ).then(res => {

            setArticle(res.data.filter(d => d.url.includes(id))[0])

            setMessage(false)

        }).catch(res => {

            setMessage('Виникла помилка отримання інформації :(')

        })

    }, [])

    return (

        <div>

            <Header />

            {message && <h1>{message}</h1>}

            {!message && <div><FullCard article={article} /></div>}

            <Footer />

        </div>

    )

}

ДОДАТОК Б

Лістинг стилізації

**Card.scss**

.article {

    border-bottom: 4px solid rgb(219, 120, 255);

    box-shadow: 0px 0px 10px rgb(149, 0, 255);

    background-color: #eac2ff;

    width: 400px;

    height: 120px;

    border-radius: 8px;

    margin: 20px;

    font-family: 'Lucida Sans', 'Lucida Sans Regular', 'Lucida Grande', 'Lucida Sans Unicode', Geneva, Verdana, sans-serif;

    p {

        font-weight: bold;

        font-size: 18px;

        text-align: center;

    }

}

.title {

    text-align: center;

}

**Cards.scss**

.articles {

    display: flex;

    justify-content: center;

    flex-wrap: wrap;

    font-family: 'Lucida Sans', 'Lucida Sans Regular', 'Lucida Grande', 'Lucida Sans Unicode', Geneva, Verdana, sans-serif;

}

**Footer.scss**

footer {

    background-color: #2c3e50;

    color: #ecf0f1;

    padding: 20px;

    text-align: center;

    font-weight: bold;

    .footer-content {

        max-width: 1200px;

        margin: 0 auto;

        display: flex;

        flex-direction: column;

        align-items: center;

        p {

            margin-bottom: 10px;

        }

        .social-links {

            display: flex;

            gap: 10px;

            a {

                color: #ecf0f1;

                text-decoration: none;

                transition: color 0.3s ease;

                &:hover {

                    color: #3498db;

                }

            }

        }

    }

}

**FullCard.scss**

.card {

    width: 1000px;

    height: 260px;

    background-color: rgb(215, 189, 255);

    border-radius: 8px;

    margin-top: 20px;

    margin-bottom: 40px;

    font-family: 'Lucida Sans', 'Lucida Sans Regular', 'Lucida Grande', 'Lucida Sans Unicode', Geneva, Verdana, sans-serif;

    h1, h2, h3 {

        text-align: center;

    }

    button {

        outline: none;

        border: none;

        background-color: aliceblue;

        font-size: 20px;

        font-weight: bold;

        border-radius: 18px;

        margin-top: 5px;

        margin-left: 20px;

    }

    @media (max-width: 1050px) {

        width: 80%;

    }

    @media (max-width: 730px) {

        width: 70%;

        font-size: 14px;

    }

    @media (max-width: 730px) {

        font-size: 12px;

        height: 50%;

    }

    @media (max-width: 480px) {

        font-size: 10px;

        width: 100%;

        height: 50%;

    }

}

**Header.scss**

header {

    background-color: #cfb1ff;

    padding: 10px 20px;

    display: flex;

    justify-content: space-between;

    align-items: center;

    border-bottom: 2px solid rgb(141, 27, 255);

    font-family: 'Lucida Sans', 'Lucida Sans Regular', 'Lucida Grande', 'Lucida Sans Unicode', Geneva, Verdana, sans-serif;

    .logo {

        font-size: 24px;

        color: #fff;

        text-decoration: none;

    }

    nav {

        ul {

            list-style: none;

            margin: 0;

            padding: 0;

            display: flex;

            li {

                margin-right: 20px;

                a {

                    text-decoration: none;

                    color: #fff;

                    font-weight: bold;

                    font-size: 18   px;

                    transition: color 0.3s ease;

                    &:hover {

                        color: #ecf0f1;

                    }

                }

            }

        }

    }

}