МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра комп'ютерних наук

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: **«РОЗРОБКА НАВЧАЛЬНОГО ВЕБРЕСУРСУ "ЗАХИСТ УКРАЇНИ" ДЛЯ ЗДОБУВАЧІВ СЕРЕДНЬОЇ ОСВІТИ»**

Виконав: студент 2 курсу, групи 8.1222-з

спеціальності 122 комп'ютерні науки
(шифр і назва спеціальності)

освітньої програми комп'ютерні науки
(назва освітньої програми)

О. А Алєйкін
(ініціали та прізвище)

Керівник доцент кафедри комп'ютерних наук, доцент к.т.н., Матвіїшина Н.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент доцент кафедри програмної інженерії, доцент, к.т.н., Мухін В.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя
2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет Математичний

Кафедра комп'ютерних наук

Рівень вищої освіти Магістр

Спеціальність 122 комп'ютерні науки

Освітня програма комп'ютерні науки

ЗАТВЕРДЖУЮ

В.о. завідувача кафедри комп'ютерних наук, д.т.н., доцент

_____ Шило Г.М.

(підпис)

« « « « «

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Алєйкіну Олександру Анатолійовичу

(прізвище, ім'я та по– батькові)

1. Тема роботи (проекту) Розробка навчального вебресурсу "Захист України" для здобувачів середньої освіти

керівник роботи (проекту) Матвіїшина Надія Вікторівна, к.т..н., доцент

(прізвище, ім'я та по– батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 01 » 05 2023 року № 643- с

2. Строк подання студентом роботи 01.12.2023

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково– пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі, аналіз предметної області.
2. Моделювання та проектування програмного доповнення.
3. Реалізація системи.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Презентація

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7.Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану виконання кваліфікаційної роботи магістра.	15.05.2023	Виконано
2.	Збір вихідних даних та аналіз предметної області.	20.06.2023	Виконано
3.	Обробка методичних та теоретичних джерел.	15.07.2023	Виконано
4.	Специфікація вимог до системи. Робота над першим розділом.	29.07.2023	Виконано
5.	Моделювання та проектування системи. Робота над другим розділом.	19.08.2023	Виконано
6.	Реалізація та тестування системи. Робота над третім розділом.	27.09.2023	Виконано
7.	Оформлення та нормоконтроль кваліфікаційної роботи магістра.	11.12.2023	Виконано
8.	Захист кваліфікаційної роботи магістра.	13.12.2023	Виконано

Студент _____ О.А Алєйкін
(підпис) (ініціали та прізвище)

Керівник роботи _____ Н.В. Матвіїшина
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ О.Г. Спиця
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота магістра «Розробка навчального вебресурсу “Захист України” для здобувачів середньої освіти»: 46 с., 34 рис., 20 джерел.

API, React, JAVASCRIPT, MONGODB, MONGOOSE, JSON, HTML, CSS.

Об’єкт дослідження – фреймворк/бібліотека React, бібліотеки для роботи з ним, засоби обробки даних на стороні клієнта та на сервері.

Мета роботи: розробка навчального вебресурсу “Захист України” для здобувачів середньої освіти

Метод дослідження – методи збору та аналізу вимог до програмного забезпечення, методи моделювання, проектування, конструювання та тестування програмного забезпечення.

Предмет дослідження – створення платформи, в якій розглянуто різноманітні функції, які дозволяють користувачам використовувати її ефективно. Вони включають можливість реєструватися в системі, входити до свого існуючого акаунту, переглядати доступні теми курсу, перевірити свої знання пройшовши тест. Всі ці функції допомагають підтримувати активну спільноту користувачів та створюють зручне середовище для навчання.

При розробці вебресурсу був проведений аналіз предметної області, обрано та спроєктовано архітектуру системи. Реалізовано серверну та клієнтську частини з використанням React.

В результаті роботи отримано програмне забезпечення для організації освітнього процесу на уроках з навчального предмета «Захист України» у закладах загальної середньої освіти.

SUMMARY

Master's Qualification Theses "Development of the Educational Web Resource "Defense of Ukraine" for Secondary Education Students": 46 pages, 34 figures, 20 sources.

API, REACT, JavaScript, MONGODB, MONGOOSE, JSON, HTML, CSS.

The subject of study is the React framework/library, accompanying libraries, client-side and server-side data processing tools. The objective of this project is to create an educational website, "Defense of Ukraine," for secondary school students.

The focus of this study is the development of a platform that incorporates multiple functions to enhance user effectiveness. These functions encompass the ability to register within the system, access an existing account, browse available course topics, and assess one's knowledge through testing. These features collectively contribute to fostering an engaged user community and facilitating a conducive learning environment.

During the development of the website, an analysis was conducted on the subject area, and the system architecture was chosen and designed. The server and client components were then implemented using React. The end result of this project is a software that facilitates the organization of educational lessons on the topic of "Defense of Ukraine" in general secondary education institutions.

ЗМІСТ

Завдання на кваліфікаційну роботу 2

Реферат 4

Summary 5

Зміст 6

Вступ 7

1 Огляд інтернет ресурсів “Захист України” 8

2 Розробка навчального вебресурсу “Захист України” 15

2.1 Розробка дизайну 15

2.2 Файлова структура проєкту 17

2.3 Розробка веб-сайту 19

2.4 База даних 24

2.5 Реалізація бази даних 26

3 Демонстраційний режим навчального вебресурсу “Захист України” 29

Висновки 34

Перелік посилань 35

Додаток А Головна сторінка 37

Додаток Б Організація сторінки курсу 39

Додаток В Організація сторінки тесту 41

ВСТУП

Сучасна епоха характеризується стрімким розвитком технологій та поширенням цифрових інструментів у всіх сферах життя суспільства. Освіта не залишається осторонь цього процесу, але скористатися усією потужністю інновацій для підвищення якості навчання можливо лише завдяки розробці та впровадженню сучасних освітніх вебресурсів.

Навчальні вебресурси стають необхідністю в освітньому процесі, оскільки вони дозволяють створити доступне та ефективне навчання, забезпечуючи здобувачів освіти інтерактивними матеріалами, ігровими елементами та можливістю індивідуалізації навчання. У цьому контексті, розробка навчального вебресурсу "Захист України" для здобувачів середньої освіти набуває особливого значення.

Тема "Захист України" не лише актуальна в контексті сучасних викликів, що стоять перед українським суспільством, але й має стратегічне значення для формування патріотичної свідомості та готовності молоді до захисту України. Розробка такого навчального вебресурсу дозволить створити унікальну інтерактивну платформу, яка сприятиме розширенню знань про історію та сучасні виклики у сфері оборони, розвиватиме навички стратегічного мислення та сприятиме формуванню позитивного ставлення до служби в армії та резерві.

У даній кваліфікаційній роботі буде розглянуто процес розробки навчального вебресурсу "Захист України", враховуючи сучасні підходи до організації навчального процесу та використання інформаційно-комунікаційних технологій у сфері освіти.

1 ОГЛЯД ІНТЕРНЕТ-РЕСУРСІВ “ЗАХИСТ УКРАЇНИ”

Організація освітнього процесу на уроках з предмета «Захист України» у закладах загальної середньої освіти у 2023-2024 н.р. здійснюватиметься відповідно до законів України «Про освіту», «Про повну загальну середню освіту», «Про військовий обов’язок і військову службу», «Про національну безпеку України», Стратегії воєнної безпеки України, Стратегії національно патріотичного виховання, Державного стандарту базової і повної загальної середньої освіти, Положення про допризовну підготовку [1].

Учителям предмета «Захист України» рекомендується використовувати такі Інтернет-ресурси:

а) сайт для викладачів та здобувачів освіти з предмету “Захист України” (рисунок 1.1.) [2];

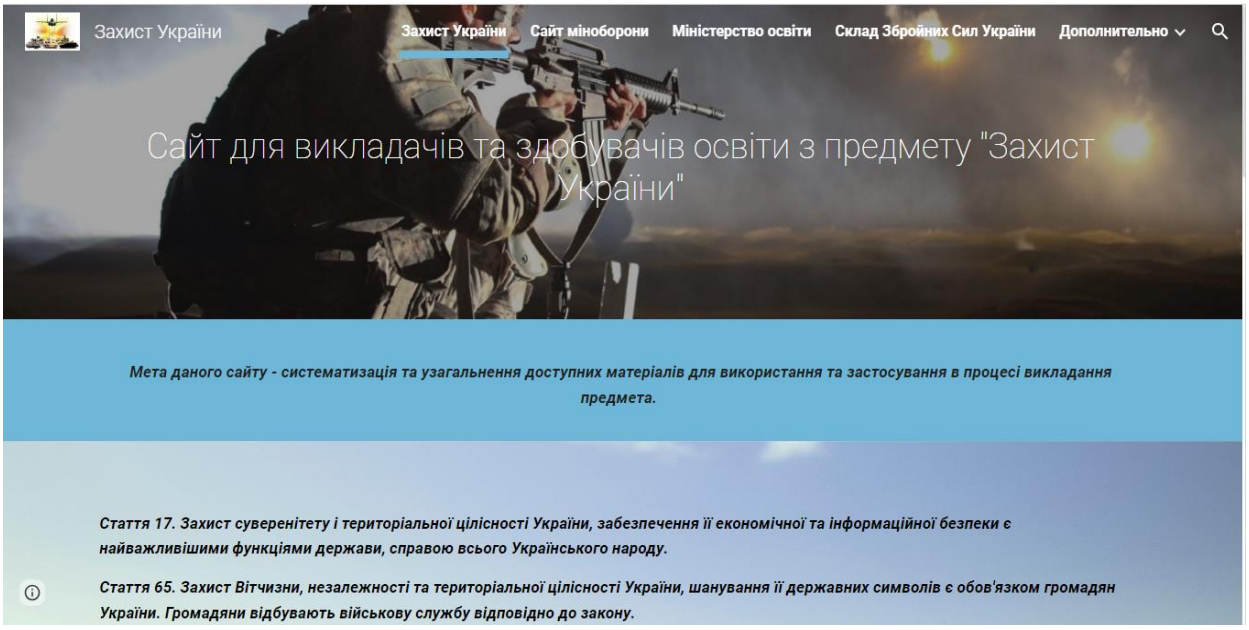

Рисунок 1 1– Сайт для викладачів та здобувачів освіти з предмету “Захист України”

б) «ЗАХИСТ УКРАЇНИ від Ганни Ягнич (рисунок 1.2) [3];

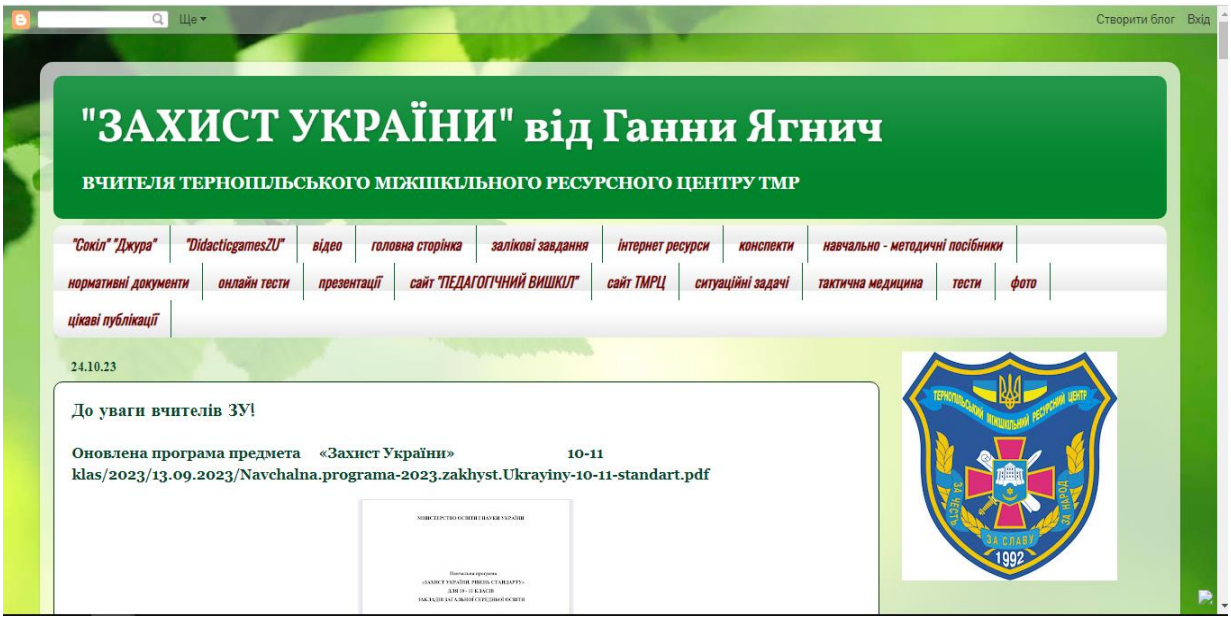

Рисунок 1 2– Сайт «ЗАХИСТ УКРАЇНИ від Ганни Ягнич

в) #ЗАХИСТИМОУКРАЇНУ! (рисунок 1.3) [4];

Рисунок 1 3–#ЗАХИСТИМОУКРАЇНУ!

г) електронний збірник дидактичних ігор із предмета «Захист України» (рисунок 1.4) [5];

Рисунок 1 4– Електронний збірник дидактичних ігор

д) навчальний курс - "Навчання з попередження ризиків від вибухонебезпечних предметів" (EORE) (рисунок 1.5) [6];

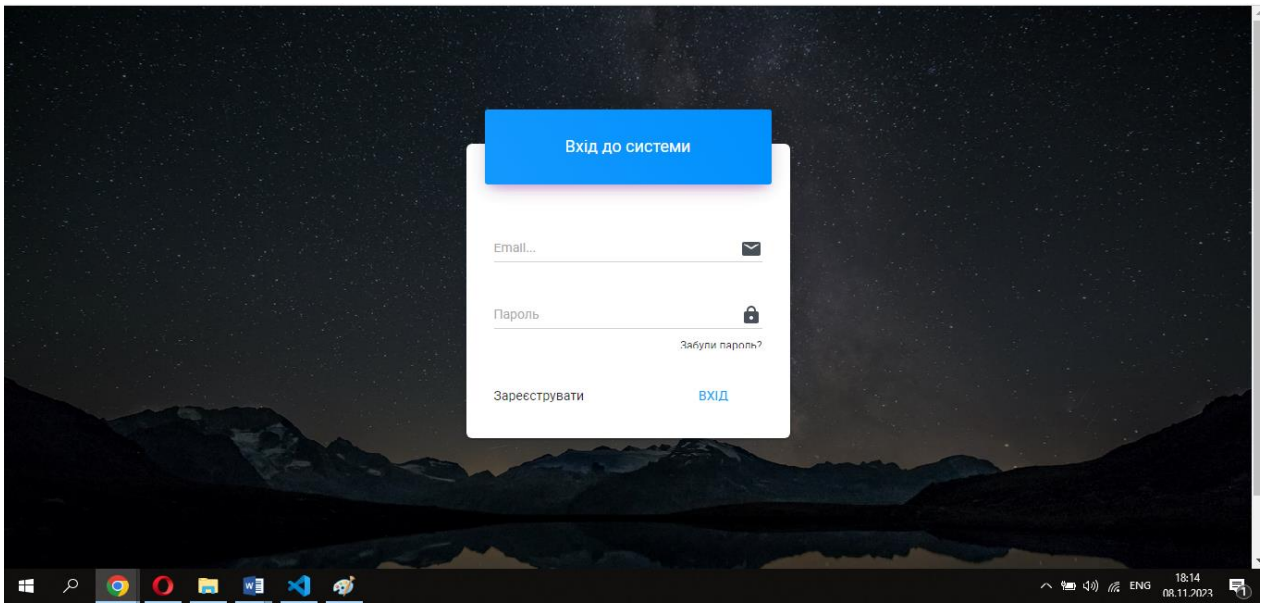

Рисунок 1 5– "Навчання з попередження ризиків від вибухонебезпечних предметів" (EORE)

е) #ВБЕЗПЕЦІ) (рисунок 1.6) [7];

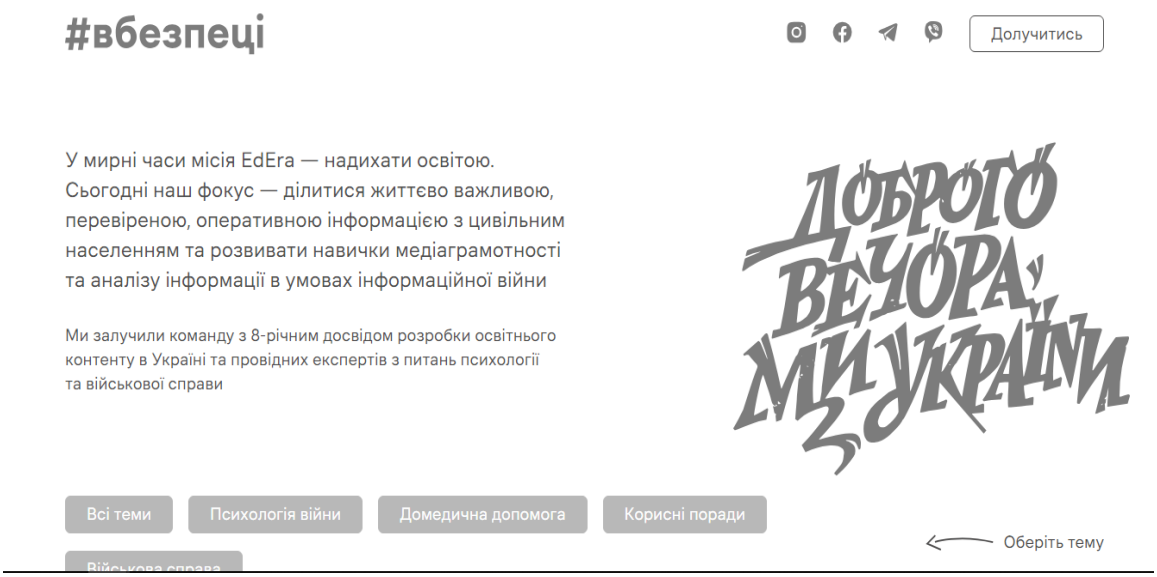

Рисунок 1 6– #ВБЕЗПЕЦІ

ж) G7 Сили територіальної оборони ЗСУ готові до спротиву (рис. 1.7) [8];

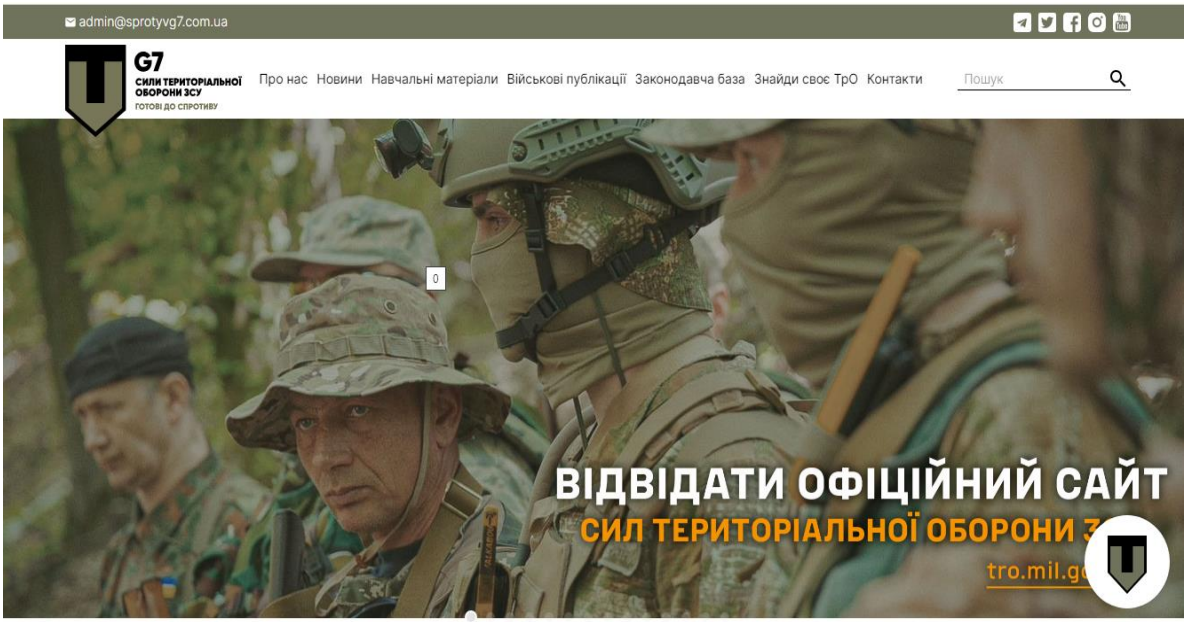

Рисунок 1 7– G7 Сили територіальної оборони ЗСУ готові до спротиву

з) стандартизоване навчання TCCC для військових (рисунок 1.8) [9];

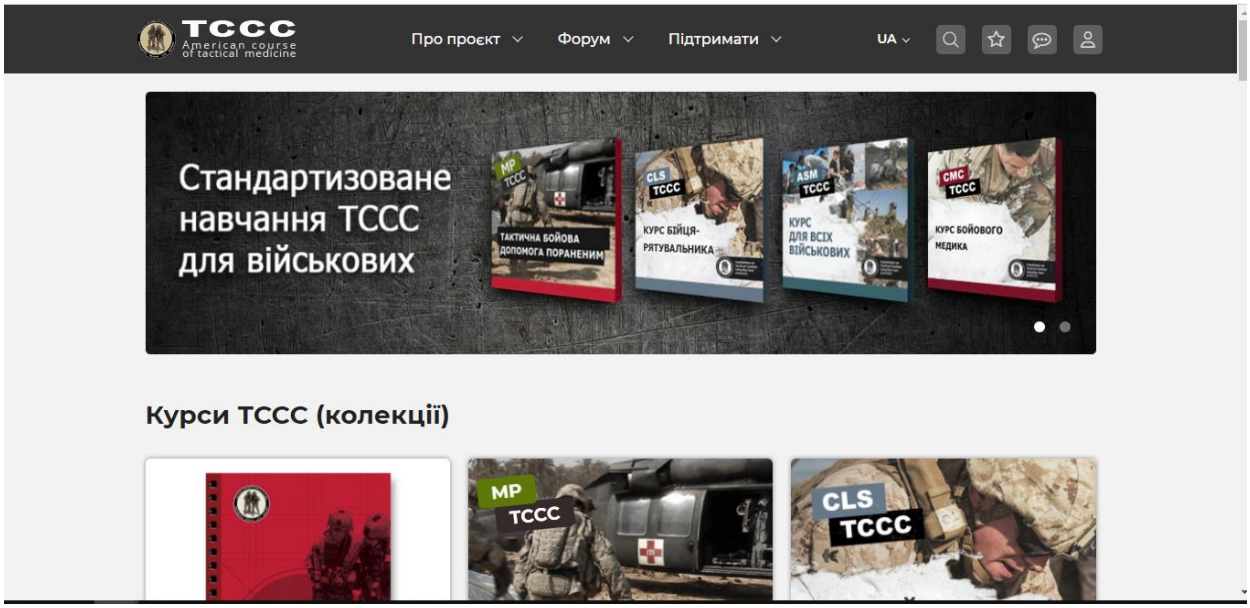

Рисунок 1 8– TCCC

и) агенція Екстреної Медичної Допомоги (рисунок 1.9) [10];

Рисунок 1 9– Агенція Екстреної Медичної Допомоги

к) ВСЕ ПРО МІННУ БЕЗПЕКУ (рисунок 1.10) [11];

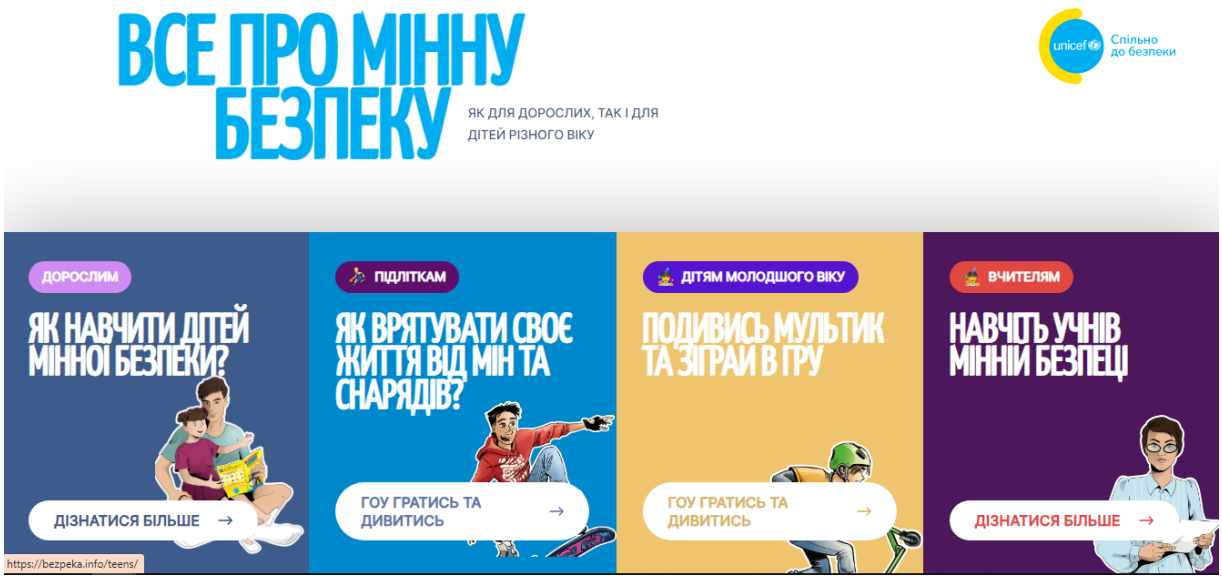

Рисунок 1 10– ВСЕ ПРО МІННУ БЕЗПЕКУ

л) Електронна військова бібліотека (рисунок 1.11) [12];

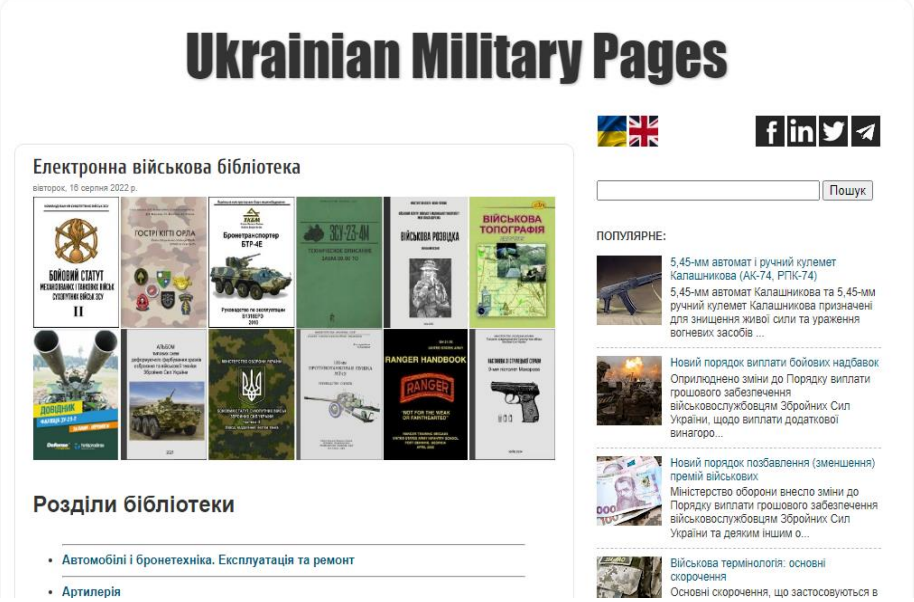

Рисунок 1 11– Електронна військова бібліотека

м) Курс тактичної медицини СБУ: протокол TCCC, алгоритми MARCH та PAWS (рисунок 1.12) [13];

н)

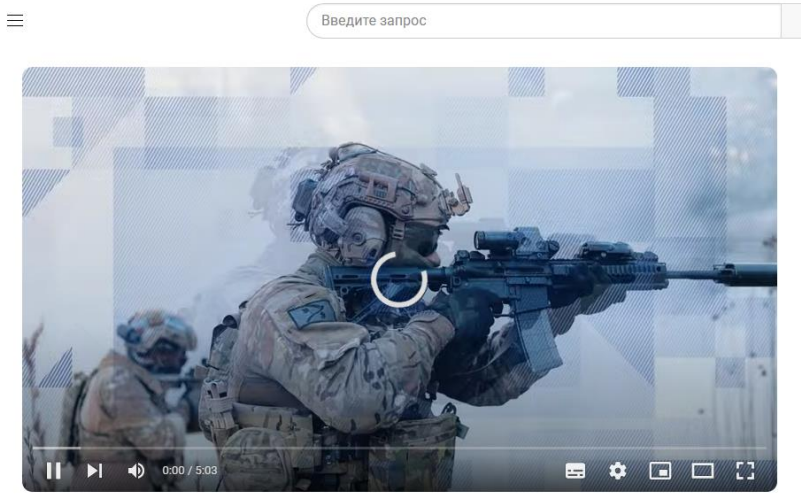

Рисунок 1 12– Курс тактичної медицини СБУ

У даній кваліфікаційній роботі використано процес розробки навчального вебресурсу "Захист України", враховуючи найкращі сучасні підходи до організації навчального процесу та використання інформаційно-комунікаційних технологій у сфері освіти.

2 РОЗРОБКА НАВЧАЛЬНОГО ВЕБРЕСУРСУ “ЗАХИСТ УКРАЇНИ”

2.1 Розробка дизайну

Дизайн сайту – це одна з найголовніших частин у проєкті, адже він може як відштовхувати, так і приваблювати нових користувачів.

Дизайн сайту є надзвичайно важливим з кількох причин:

- візуальне враження. Перше, що бачать люди, коли заходять на сайт – зовнішній вигляд. Хороший дизайн створює гарне враження у користувачів;
- зручність у використанні. Якісний дизайн спрощує навігацію сайтом і робить його інтуїтивно зрозумілим. Користувачі зможуть швидко знайти потрібну інформацію, що збільшить їхнє задоволення від використання сайту;
- унікальність та індивідуальність. Відмінний дизайн сайту допомагає виділитися серед конкурентів і змушує відвідувачів запам'ятати ваш бренд.

Для того, щоб задовольнити смаки кожного при створенні дизайну, було обрано мінімалізм як оптимальний стиль.

Мінімалізм (у WEB-дизайні) – це у веб-дизайні означає спрощення інтерфейсу шляхом видалення зайвих елементів. Іншими словами, для досягнення більшого результату за допомогою обмежених ресурсів [14].

Веб-додаток створений у темних тонах та з мінімалістичним дизайном. На рисунку 2.1 зображена головна сторінка сайту.

Отже, дизайн створений у мінімалістичному стилі, що забезпечує простоту та лаконічність у розташуванні елементів на сайті. Це полегшує користувачам орієнтуватись та швидко знаходити необхідну інформацію. Також мінімалістичний дизайн зазвичай має менше графічних та мультимедійних елементів, що допомагає сайту завантажуватись швидше,

допомагає підвищувати рівень довіри до сайту та його власника, оскільки він виглядає більш професійним та дбайливо продуманим.

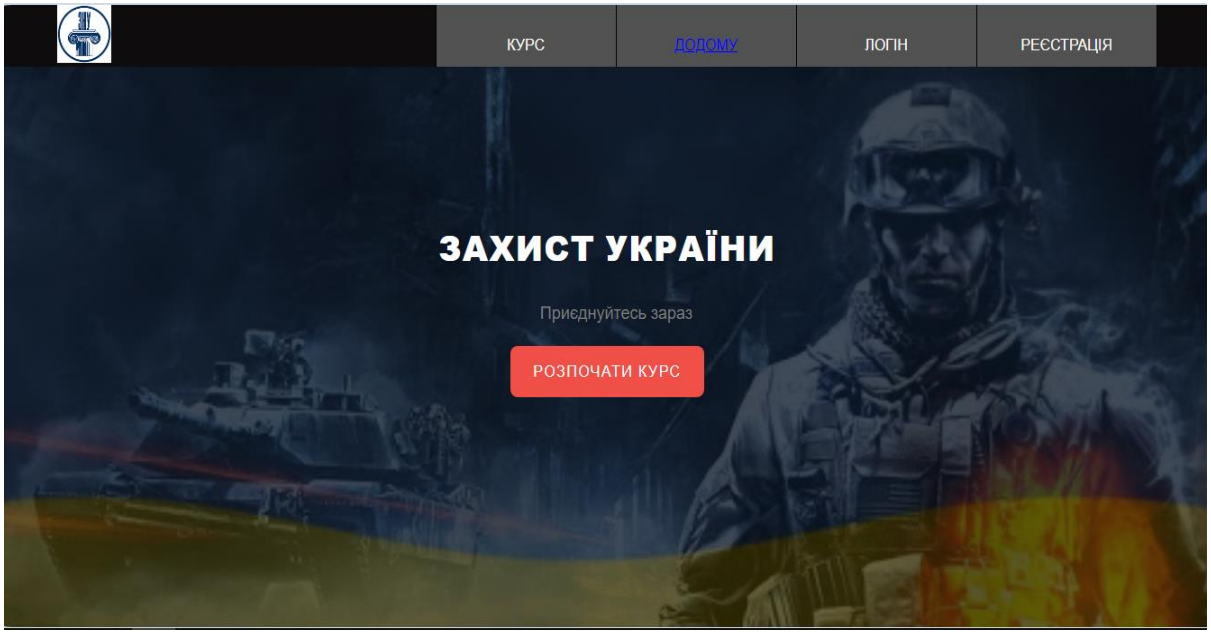

Рисунок 2.1 – Головна сторінка сайту

Стиль тем курсу зображений на рис. 2.2.

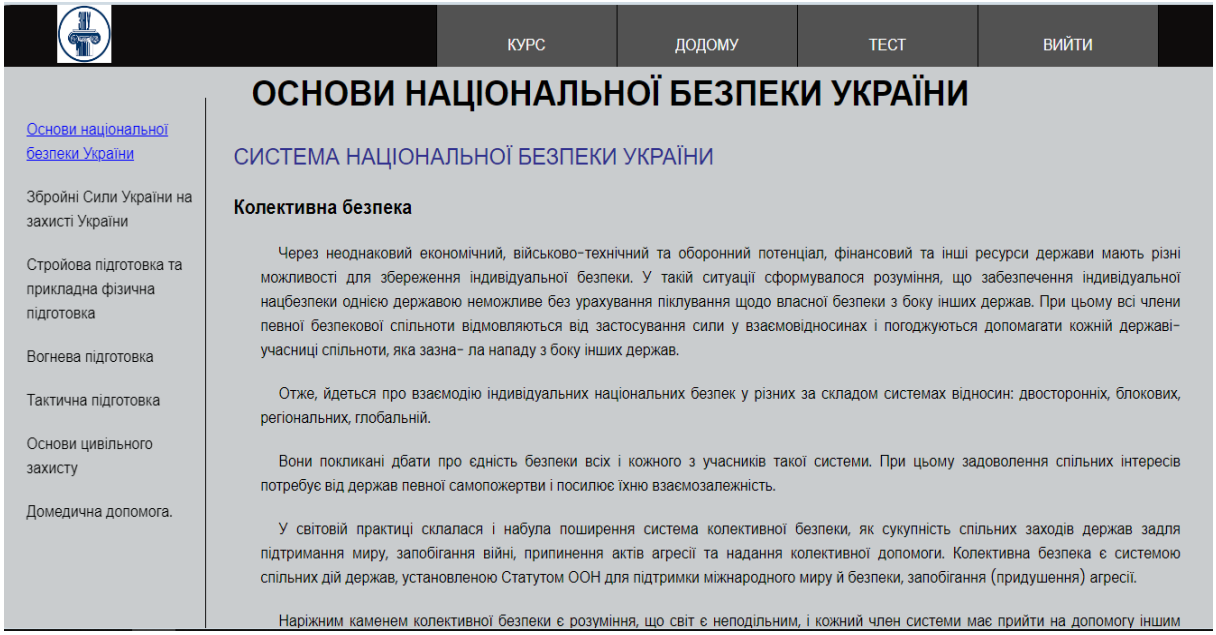

Рисунок 2.2 – Стиль тем сторінки курсів

2.2 Файлова структура проєкту

Фреймворк/бібліотека React має свою архітектуру файлів проєкту. Один із популярних підходів – розміщення файлів CSS, JS і тестів у папках, згрупованих за функціональністю або маршрутом [15].

Файлова структура серверної частини проєкту зображена на рисунку 2.4.

Рисунок 2.4 – Файлова структура серверної частини проєкту

В серверній папці знаходяться такі директорії:

– routes містить в собі файли, які допомагають в маршрутизації, переходити на сторінки без перезавантаження;
– models містить в собі файли, для взаємодії з базою даних;
– midllewares містить в собі файл, для проміжних функцій, що мають доступ до об'єкта запиту (req), об'єкта відповіді (res) і до наступної функції проміжної обробки в циклі «запит-відповідь» застосунку.

Файлова структура клієнтської частини проєкту зображена на рисунку 2.5.

У папці public зберігаються зображення, шрифти, які використовуються у клієнтський частині проєкту.

Рисунок 2.5 – Файлова структура клієнтської частини проєкту

src – це основна папка клієнтської частини в якій знаходяться наступні каталоги:

- markup містить в собі всі сторінки курсів та їх css-файли.
- pages містить сторінки всього проєкту: курси (меню зліва), домашня сторінка, сторінка логіну та реєстрації.
- redux – тут знаходяться файли, в яких використовується ця бібліотека, а саме авторизація, коментарі, що спрощує керування станом, полегшує відлагодження та тестування, сприяє розширюваності та підтримує розробку складних додатків.
- validators містить в собі один файл, де дані перевіряються на валідність.

У процесі роботи з проєктом, папка "node_modules" автоматично створюється під час встановлення залежностей, які вказані у файлі "package.json". Ця папка містить всі залежності, такі як бібліотеки та пакети, які використовуються у проекті. Вона знаходиться у двох директоріях: одна в серверній частині проекту, а інша - у клієнтській частині.

2.3 Розробка веб-сайту

Розробка веб-сайту за допомогою React виконується шляхом створення компонентів, які відповідають за різні частини і функціональність сайту. Кожен компонент містить свою логіку, стиль та може містити вкладені компоненти. React забезпечує декларативний підхід до розробки, де стан компонентів оновлюється автоматично при зміні даних, що сприяє більш простій та ефективній розробці.

За допомогою наведеного на рисунку 2.7 коду було створено головну сторінку веб-сайту. Цей код містить необхідні елементи та компоненти, які відображають контент та функціональність сторінки. Шляхом використання HTML, CSS та React-специфічних елементів та властивостей, розробники зуміли створити вигляд та поведінку головної сторінки, забезпечуючи зручну навігацію та доступ до важливої інформації для користувачів. Загальний результат роботи коду полягає в створенні привабливої та функціональної головної сторінки, яка ефективно взаємодіє з користувачем та задовольняє їх потреби.

```
import React from 'react';
import Header from '../../components/Header/Header';
import styles from './Home.module.css';
import { Link } from 'react-router-dom';
import DropDownMenu from '../../components/DropDownMenu/DropDownMenu';
import { navItems } from '../../components/NavItems/NavItems';
import { useSelector } from 'react-redux';
import { checkAuth } from '../../redux/features/auth/authSlice';
//import img from '../../img/title.jpg'

console.log(navItems);

export default function Home() {
  const isAuth = useSelector(checkAuth)
  return (
    <div className={styles.body}>
      <section className={styles['main-content']}>
        <div className={styles['main-article']}>
          <h1>ЗАХИСТ УКРАЇНИ</h1>
          <p >Приєднуйтесь зараз </p>
          {isAuth ? <Link to='Course/html/introduction' className={styles['main-btn']}
          <a className={styles['main-btn']} title='потрібно зареєструватися'>Розпочати
        </div>
      </section>
        <div className={styles.homeBox}>
          <div className={styles.content}>
```

Рисунок 2.7 – Код головної сторінки

Наведений код відповідає за створення головної сторінки сайту. Він включає імпорт необхідних модулів та компонентів, таких як "Header", "Link", "DropDownMenu", а також використання функцій і змінних з бібліотеки React та Redux. В середині компонента відбувається рендеринг різних елементів, таких як заголовок, текстові блоки, кнопки, зображення тощо. Також проводиться перевірка наявності аутентифікації користувача та відповідно до цього відображається відповідний текст або посилання.

Також, як приклад, можна навести код Header, зображено на рисунку 2.8

```
import { checkAuth, logout } from '../../redux/features/auth/authSlice';
import {toast} from 'react-toastify'

export default function Header() {
  const [dropDown, setDropDown] = useState(false);
  const isAuth = useSelector(checkAuth)
  const dispatch = useDispatch()

  const logoutHandler = () => {
    dispatch(logout())
    window.localStorage.removeItem('token')
    toast('Ви вийшли з системи.')
  }

  return (
    <>
      <div className={styles.header}>
        <Link to='/'><img className={styles['logo']} src='/img/logo.png'/></Link>
        <ul className={styles.menu}>
            <li className={styles['header-li']} onMouseEnter={()=>{setDropDown(!dropDow
            {dropDown ? <DropDownMenu obj={navItems}/> : null}</li>
            <CustomLink to='/'><li className={styles['header-li']}>Додому</li></CustomL
```

Рисунок 2.8 – Код Header

У тілі компонента відбувається рендеринг різних елементів, таких як логотип, список з пунктами меню, кнопки входу/виходу. В компоненті також використовуються стан "dropDown", який відповідає за відображення розкривного меню при наведенні на пункт "Курс". Залежно від статусу автентифікації користувача, відображаються відповідні пункти меню, такі як "Тест", "Логін", "Зареєструватися".

Також в компоненті присутній обробник події для кнопки "Вийти" ("logoutHandler"), який виконує вихід з системи, видаляє збережений токен з локального сховища та виводить повідомлення за допомогою бібліотеки "react-toastify".

Отже, код створює верхню частину сайту, яка містить логотип, меню навігації та кнопки входу/виходу, і забезпечує функціональність для взаємодії з ними.

Серверна частина або back-end, у контексті React, відповідає за обробку запитів, збереження та отримання даних з бази даних, аутентифікацію користувачів та інші серверні операції, необхідні для роботи додатка. Для реалізації серверної частини використовуються інші технології та інструменти. У своєму проєкті я використовую Express.

Нижче наведений код буде демонструвати використання Express в моєму проєкті. Рисунок 3.9, демонструє модулі, які будуть використані.

```
1  import  express  from "express";
2  import cors from 'cors';
3  import authRouter from './routes/auth.js'
4  import postRouter from './routes/post.js'
5  import commentRouter from './routes/comment.js'
6  import fileUpload from "express-fileupload";
```

Рисунок 2.9 – Імпорт необхідних модулів

Цей розділ імпортує необхідні модулі, такі як express, cors, express-fileupload, а також маршрутизатори authRouter, postRouter і commentRouter, які визначені в окремих файлах.

На рисунку 2.10 зображено створення і конфігурація сервера.

Рисунок 2.10 – Створення і конфігурація сервера

Тут створюється екземпляр сервера Express та налаштування його. Метод app.use() використовується для додавання проміжного ПЗ (middleware). У даному випадку, використовується express.json() для обробки запитів JSON, cors() для управління політикою міжсайтових запитів (CORS), fileUpload() для обробки завантаження файлів та express.static() для надання статичних файлів з папки "upload".

Визначення маршрутів зображено на рисунку 2.11.

Рисунок 2.11 – Визначення маршрутів

Тут використовується маршрутизатори authRouter, postRouter і commentRouter, які були імпортовані раніше, і прив'язуємо їх до шляху "/api". Це означає, що всі запити, які починаються з "/api", будуть оброблятись цими маршрутами.

Експорт серверного додатка відбувається на рисунку 2.12.

Рисунок 2.12 – Експорт серверного додатку

Остаточно, експортується створений серверний додаток як модуль за допомогою синтаксису export default, що дозволяє використовувати його в інших частинах додатка або при запуску сервера.

Middleware – це проміжний шар програмного забезпечення, який діє як посередник між вхідними запитами та обробниками запитів. Він знаходиться між прийомом запиту сервером і виконанням обробки цього запиту. Застосування middleware у моєму проєкті зображено на рис. 2.13.

server > midllewares > JS checkAuth.js > checkAuth

```
1   import jwt from 'jsonwebtoken';
2
3   export const checkAuth = (req, res, next) => {
4       const token = (req.headers.authorization || '').replace(/Bearer\s?/, '')
5
6       if(token){
7           try{
8               const result = jwt.verify(token, process.env.JWT_SECRET)
9
10              req.userId = result.id
11
12              next()
13          }catch(err){
14              return res.json({message: 'Немає доступа.'})
15          }
16      }else{
17          return res.json({
18              message: 'Немає доступа.'
19          })
20      }
21  }
```

Рисунок 2.13 – Застосування middleware

Наведений код виконує перевірку автентифікації користувача з використанням токенів JWT (JSON Web Tokens). В функції checkAuth, спочатку отримується токен з заголовків запиту і виконується його перевірка. Якщо токен існує, він розшифровується за допомогою jwt.verify з використанням секретного ключа process.env.JWT_SECRET. Результат розшифрування містить ідентифікатор користувача, який потім додається до об'єкта req для подальшого використання.

Якщо перевірка токену успішна, виконується наступна функція next(), що передає управління наступному обробнику в ланцюжку маршрутизації. У випадку помилки перевірки токену або якщо токен не існує, повертається відповідь JSON з повідомленням про відсутність доступу.

Отже, загальний опис коду полягає у створенні сервера Express з налаштуваннями middleware, визначенні маршрутів та експорту створеного серверного додатка.

2.4 База даних

База даних – це сукупність інформації, яка організована таким чином, що до неї можна легко отримати доступ, керувати та оновлювати. Комп’ютерні бази даних зазвичай містять сукупності записів даних або файлів, що містять інформацію про операції продажу або взаємодії з конкретними клієнтами.

Головною перевагою БД є швидкість внесення та використання потрібної інформації. Завдяки спеціальним алгоритмам, які використовуються для баз даних, можна легко знаходити необхідні дані всього за декілька секунд. Також в базі даних існує певний взаємозв'язок інформації: зміна в одному рядку може спричинити зміни в інших рядках — це допомагає працювати з інформацією простіше і швидше [9].

Існують різні типи баз даних:

– ієрархічна база даних – виконує порядок ранжування або взаємозв’язок батьків і дітей для структурування даних;
– мережева база даних – схожа на ієрархічну базу даних, але з деякими змінами. Мережева база даних дозволяє дочірньому запису зв’язуватися з різними батьківськими записами, отже, дозволяючи двонаправлені відносини;

- об’єктно-орієнтована база даних – інформація зберігається об’єктно-подібним чином;
- реляційна база даних – орієнтована на таблиці, де кожен біт даних має зв’язок з кожним іншим бітом даних;
- нереляційна база даних або база даних NoSQL. Жодна база даних SQL не використовує різноманітні формати, такі як документи, графіки тощо, що забезпечує більшу гнучкість і масштабованість для проєктування бази даних.

Однак бази даних широко поділяються на два основні типи, а саме: реляційні бази даних та нереляційні бази даних.

Реляційні бази даних – це БД, що зберігають дані у таблиці і рядках, які також називаються записами, наприклад, PostgreSQL, SQLite, MySQL та MariaDB.

Нереляційні бази даних відрізняються від реляційних баз даних тим, що вони не використовують таблиці, рядки, первинні або зовнішні ключі. Замість цього, нереляційна база даних оперує моделлю сховища, яка оптимізована для задоволення конкретних вимог до типу даних, які зберігаються.

Для збереження даних користувачів, я обрав нереляційну базу даних MongoDB [16].

MongoDB – це документо-орієнтована база даних, яка зберігає дані у вигляді документів в форматі JSON (JavaScript Object Notation). Вона дозволяє зберігати інформацію у вигляді складних об'єктів, які можуть містити різноманітні поля, що дозволяє зберігати дані у більш структурованому вигляді порівняно з традиційними реляційними базами даних.

MongoDB має багато переваг перед реляційними базами даних, зокрема:

- висока швидкість роботи: MongoDB працює швидше, ніж більшість реляційних баз даних, оскільки документи можуть бути збережені в одному дисковому блоку;

- простота використання: MongoDB дуже легко встановлювати та використовувати. Крім того, вона має приємний інтерфейс для роботи з даними;
- гнучкість: MongoDB дозволяє зберігати будь-яку кількість полів у документах, що робить його більш гнучким ніж традиційні реляційні бази даних.

Існують також інші документо-орієнтовані бази даних, які можуть бути альтернативами MongoDB. Декілька з них:

- Couchbase – є документо-орієнтованою базою даних з високою продуктивністю, що працює у розподіленому середовищі;
- Apache Cassandra – є ключ-значенням базою даних з високою доступністю, яка дозволяє зберігати великі обсяги даних у розподілених середовищах;
- Amazon DynamoDB – є повністю керованою базою даних з розміщенням у реальному часі та відмінною масштабованістю.

Я вважаю, що MongoDB є доречним варіантом бази даних для мого проєкту з таких причин: вона має гнучку структуру даних, що дозволяє зручно працювати з різними типами і форматами інформації; вона також відзначається простотою використання, що спрощує процес розробки та підтримки; крім того, MongoDB демонструє високу продуктивність, що забезпечує швидкий доступ до даних; і, нарешті, вона масштабована, що означає, що вона може легко розширюватись і впоратись зі зростаючим обсягом даних.

2.5 Реалізація бази даних

MongoDB використовує схему бази даних, яку називають «гнучкою схемою». MongoDB не має фіксованих таблиць і зміни схеми можуть бути виконані безпосередньо в документах бази даних. Замість табличної структури

MongoDB використовує колекції документів, де кожен документ представляє сутність з набором ключ-значення полів. Схема бази даних зображена на рисунку 2.14.

Рисунок 2.14 – Схема бази даних

Для підключення до БД використовую mongoose. Mongoose – це об'єктно-реляційна бібліотека для роботи з MongoDB у JavaScript-додатках. Вона дозволяє встановлювати з'єднання з MongoDB, визначати моделі даних і виконувати різноманітні операції з базою даних, такі як додавання, оновлення, видалення та запити. Mongoose надає зручний спосіб взаємодії з MongoDB, допомагаючи спростити розробку і керування даними в програмах на JavaScript.

На рис. 2.15 продемонстровано підключення до бази даних.

```
async function start(){
    try{
        await mongoose.connect(`mongodb+srv://${DB_USER}:${DB_PASSWORD}@cluster0.gsetepn.mongodb.net/${DB_NAME}?retryWrites=true&w=majority`)

        server.listen(PORT, ()=>{
            console.log(`Server started on port ${PORT}`)
```

Рисунок 2.15 – Підключення до бази даних

Отже, MongoDB використовує гнучку схему бази даних, що дозволяє змінювати схему безпосередньо в документах бази даних. Замість таблиць,

MongoDB використовує колекції документів, де кожен документ має набір ключ-значення полів. Для зручної роботи з MongoDB у JavaScript-додатках можна використовувати бібліотеку Mongoose, яка дозволяє підключатися до бази даних, визначати моделі даних і виконувати різноманітні операції з даними. Підключення до бази даних можна здійснити за допомогою методів, які надає Mongoose. Використання Mongoose допомагає спростити розробку і керування даними в програмах на JavaScript.

3 ДЕМОНСТРАЦІЙНИЙ РЕЖИМ НАВЧАЛЬНОГО ВЕБРЕСУРСУ “ЗАХИСТ УКРАЇНИ”

Нижче представлені скріншоти екрану вебресурсу, який працює у демонстраційному режимі.

Перший екран, який зустріне користувач, є головною сторінкою, де він матиме можливість зареєструватися або увійти до свого акаунту, оскільки доступ до курсів обмежений для незареєстрованих користувачів (рисунок 3.1).

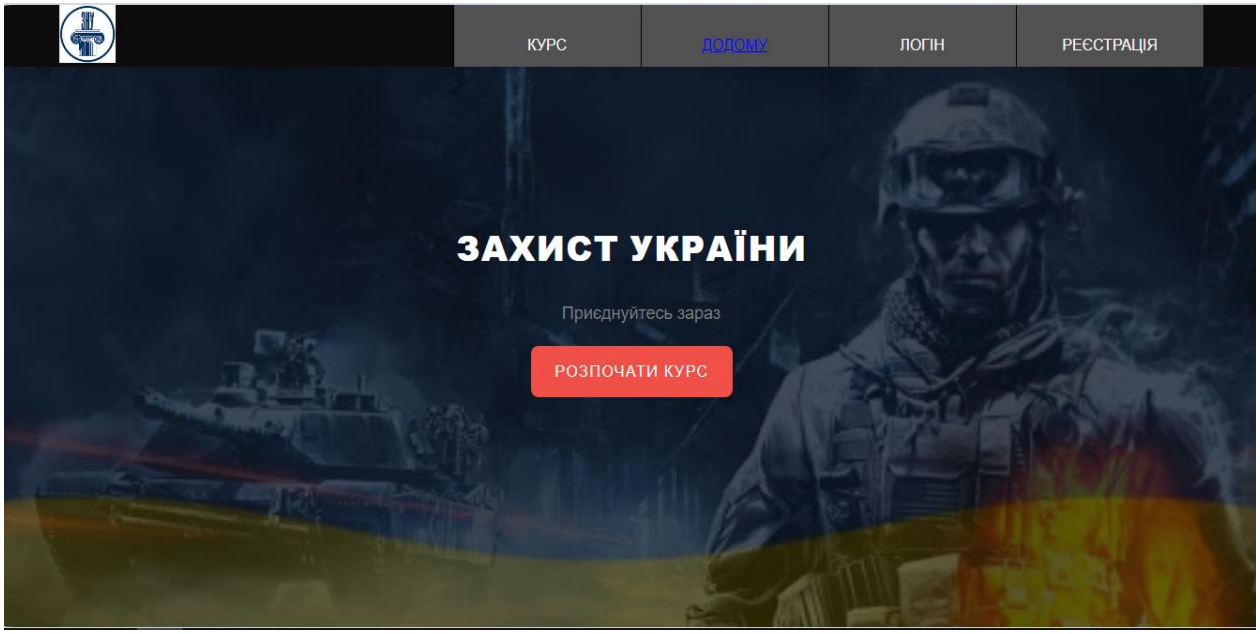

Рисунок 3.1 – Головний екран вебресурсу

На рисунку 3.2 зображена форма Логіну, яка надає користувачам можливість ввести свої облікові дані для отримання доступу до системи.

На рисунку 3.3 показано можливість зареєструватися для користувачів, які ще не мають облікового запису. Цей варіант надає їм можливість створити новий аккаунт і отримати повний доступ до всіх функцій системи. Для переходу до цієї форми, користувач може натиснути на посилання "Немає облікового запису?" або перейти безпосередньо на сторінку реєстрації (Реєстрація).

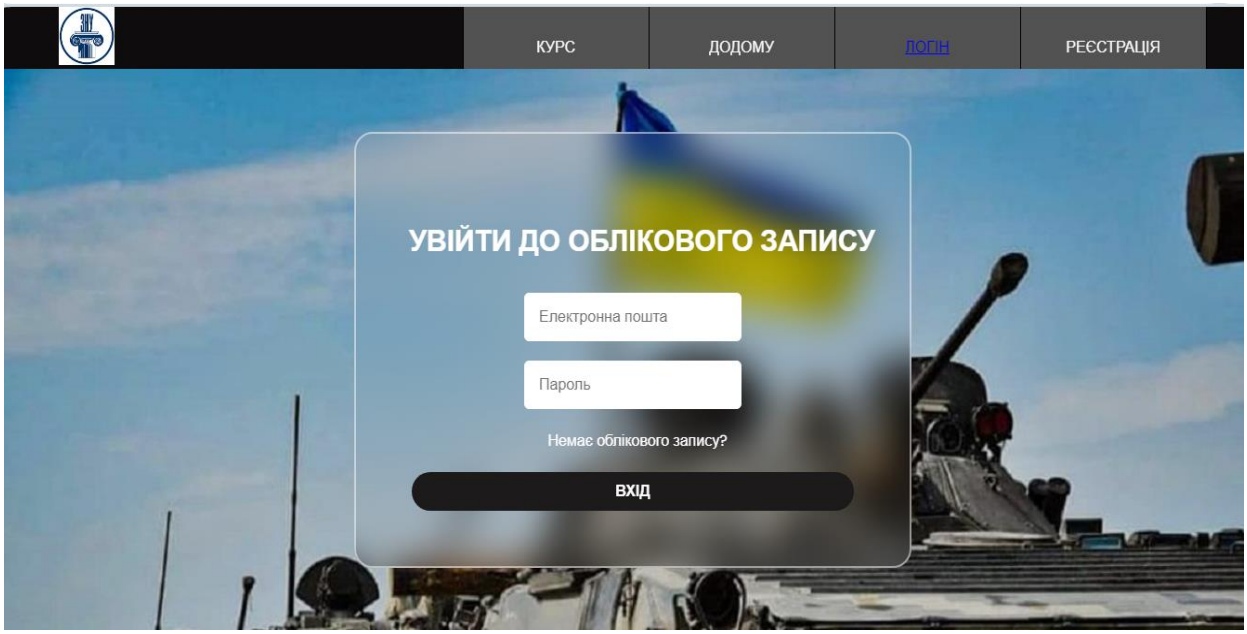

Рисунок 3.2 – Форма логіну

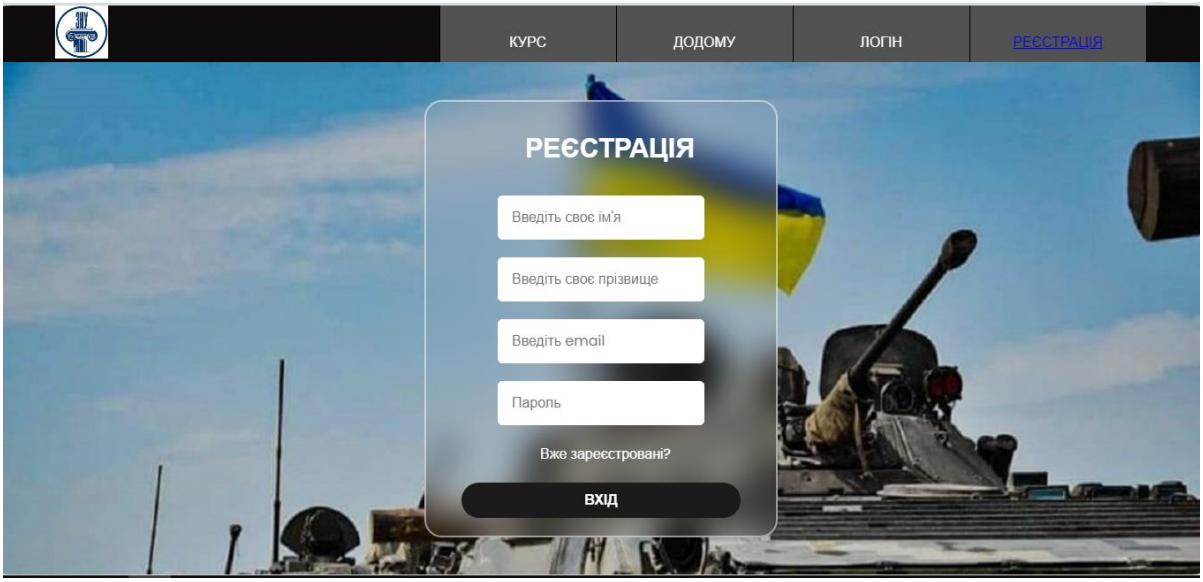

Рисунок 3.3 – Форма реєстрації

Після успішного входу в особистий аккаунт, меню веб-сайту змінюється, додавши нові пункти, які стають доступними для зареєстрованих користувачів. На рисунку 4.4 зображено оновлене меню, яке надає розширені функції та ресурси. Зареєстровані користувачі можуть звертатися до цих нових пунктів для отримання інформації про доступні курси та матеріали, можливість перевірити свої знання пройшовши тест. Це оновлене меню

забезпечує зручну навігацію та розширені можливості для користувачів після входу в їх особистий аккаунт.

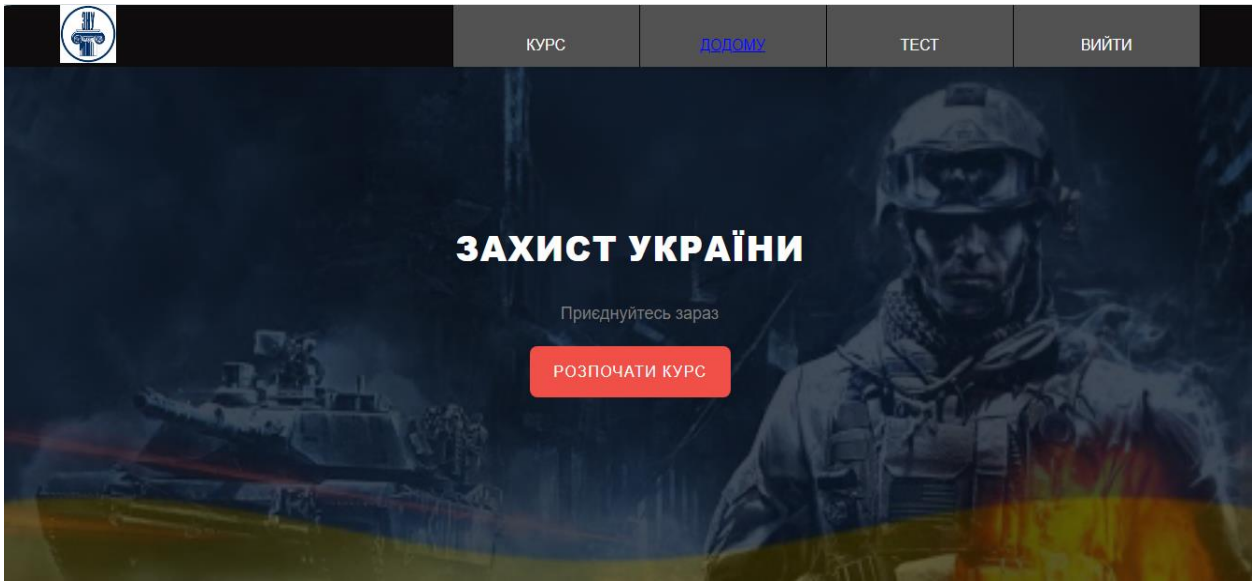

Рисунок 3.4 – Нові пункти меню

При наведенні курсору на пункт "Курс" в меню, з'являється випадаюче меню, яке дозволяє користувачеві обрати розділи курсу, яке зображено на рисунку 3.5. Це надає можливість вибору конкретного розділу залежно від інтересів та потреб користувача.

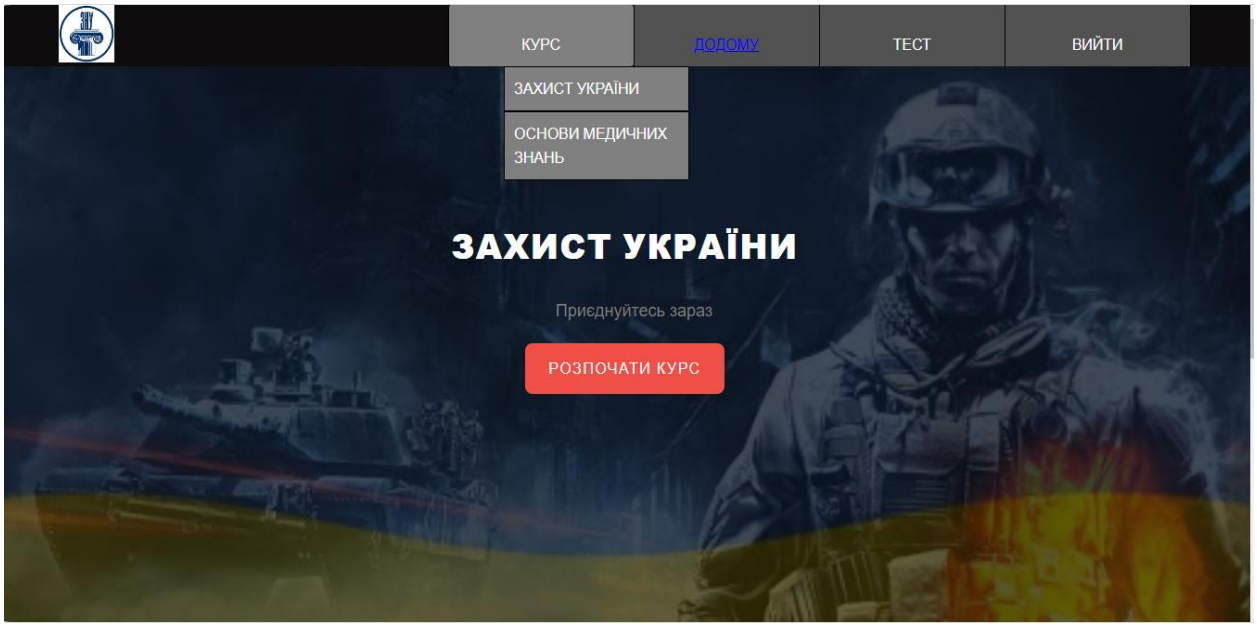

Рисунок 3.5 – Випадаюче меню розділів

Після вибору користувачем потрібного розділу, відкривається сторінка з матеріалом курсу, де ліворуч розташовані теми, а праворуч міститься відповідна інформація до кожної з цих тем, рисунок 3.6. Ця організація сторінки дозволяє користувачеві легко навігуватися по різних темах курсу та отримувати необхідну інформацію. Матеріали для наповнювання курсу були взяті з джерел [17-20]. За допомогою лівої панелі з темами, користувачі можуть швидко переходити між різними частинами курсу, а праворуч вони знаходять детальну інформацію, матеріали що стосуються обраної теми. Ця структура сторінки забезпечує зручний доступ до вмісту курсу та полегшує процес навчання користувачів.

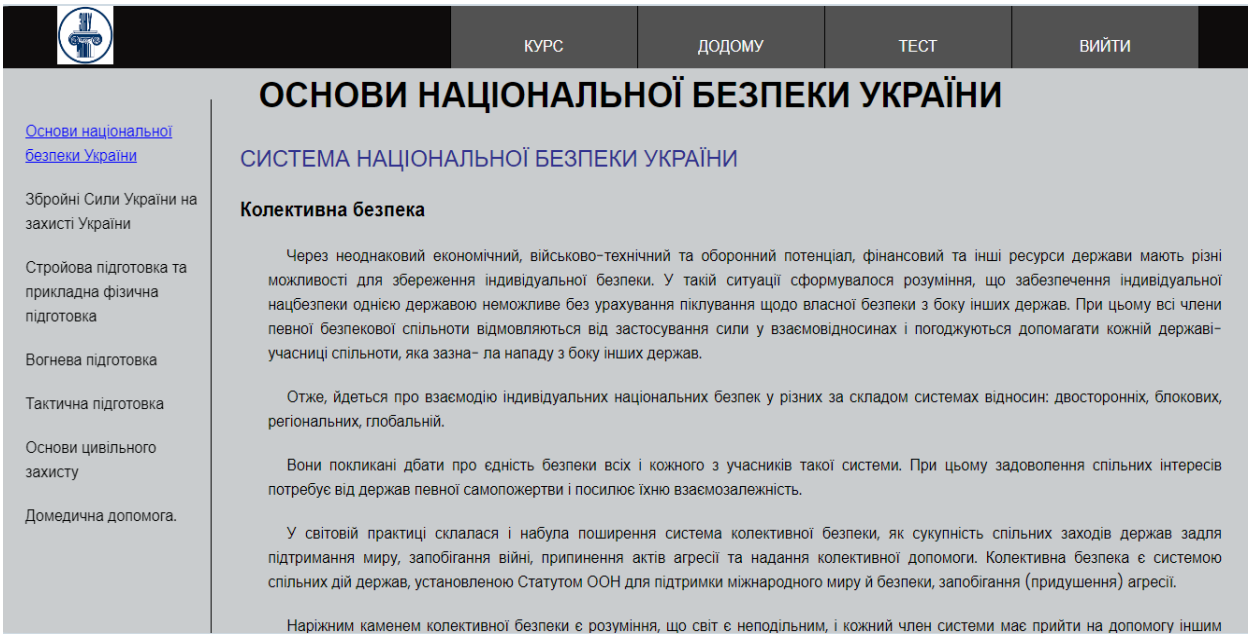

Рисунок 3.6 – Організація сторінки курсу

При виборі меню "Тест" замість курсів, користувач переходить на сторінку з тестом, за допомогою якого він може перевірити свої знання на розглянуту тему (рисунок 3.7).

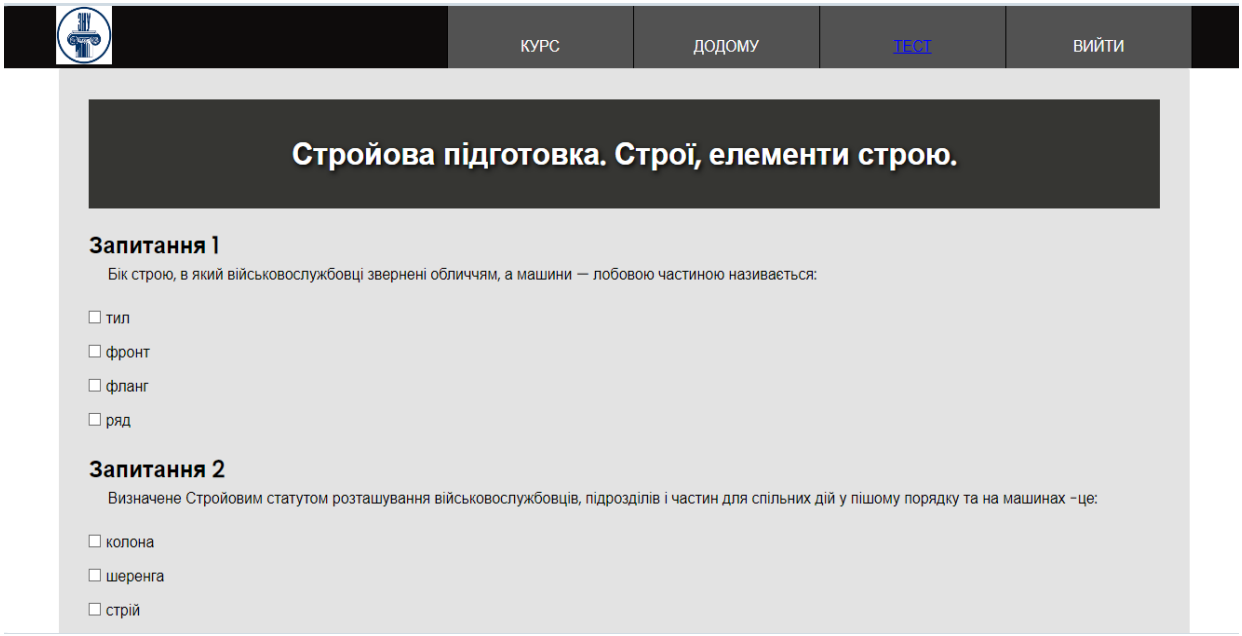

Рисунок 3.7 – Організація сторінки тесту

ВИСНОВКИ

В ході написання кваліфікаційної роботи було проведено огляд інтернет ресурсів рекомендованих для застосування при вивченні курсу “Захист України” для середньої освіти.

У даному проєкті вебресурсу було використано фреймворк/бібліотеку React, розглянуто різноманітні функції, які дозволяють користувачам використовувати платформу ефективно. Вони включають можливість реєструватися в системі, входити до свого існуючого акаунту, переглядати доступні теми курсу, перевірити свої знання пройшовши тест. Всі ці функції допомагають підтримувати активну спільноту користувачів та створюють зручне середовище для навчання. Користувачі можуть максимально використовувати потенціал проєкту і здобувати нові знання та навички.

Під час реалізації розробки вебресурсу, отримано практичні та теоретичні навички роботи з різними бібліотеками, як бекенд частини, так і клієнтського застосунку. Отримано знання про різні архітектурні рішення в обох частинах систем. В рамках розробки застосунку поглиблено знання про складові фреймворку/бібліотеки React

ПЕРЕЛІК ПОСИЛАНЬ

1. Інструктивно-методичні рекомендації щодо викладання навчальних предметів у закладах загальної середньої освіти у 2023/2024 навчальному році “ЗАХИСТ УКРАЇНИ” URL : https://mon.gov.ua/storage/app/media/zagalna%20serednya/metodichni%20recomendazii/2023/09/12/IMR-2023-2024-Zakhyst.Ukrayiny.12.09.2023.pdf (дата звернення : 05.07.2023).

2. Сайт для викладачів та здобувачів освіти з предмету “Захист України” URL : https://sites.google.com/site/zahist56/ (дата звернення : 10.07.2023).

3. “ЗАХИСТ УКРАЇНИ” від Ганни Ягнич URL : https://jahnu4.blogspot.com/ (дата звернення : 15.07.2023).

4. #ЗАХИСТИМОУКРАЇНУ! URL : https://viktorpoli.blogspot.com/ (дата звернення : 16.07.2023).

5. Електронний збірник дидактичних ігор із предмета «Захист України» URL : https://didacticgameszu.blogspot.com/ (дата звернення : 20.07.2023).

6. Навчальний курс "Навчання з попередження ризиків від вибухонебезпечних предметів" (EORE) URL : https://eoreplatform.web.app/dashboard (дата звернення : 30.07.2023).

7. #ВБЕЗПЕЦІ URL : https://safe.edera.com/ (дата звернення : 04.08.2023).

8. G7 Сили територіальної оборони ЗСУ готові до спротиву URL : https://sprotyvg7.com.ua/ (дата звернення : 15.08.2023).

9. TCCC URL : https://tccc.org.ua/ (дата звернення : 18.08.2023).

10. Агенція Екстреної Медичної Допомоги URL : https://aemc.org.ua/ (дата звернення : 25.08.2023).

11. ВСЕ ПРО МІННУ БЕЗПЕКУ URL : https://bezpeka.info/ (дата звернення : 29.08.2023).

12. Електронна військова бібліотека URL : https://www.ukrmilitary.com/p/military-library.html (дата звернення : 02.09.2023).

13. Курс тактичної медицини СБУ : протокол TCCC, алгоритми MARCH та PAWS URL : https://www.youtube.com/watch?v=T58mNfJ1Y4c (дата звернення : 07.09.2023).

14. Best Practices for Minimalist Website Design URL : https://medium.muz.li/best-practices-for-minimalist-website-design-9e8ea07e17c2?gi=1bf93b8c1c71#:~:text=Minimalism%20in%20web%20design%20means,words%2C%20do%20more%20with%20less (дата звернення : 17.09.2023).

15. React JavaScript-библиотека для создания пользовательских URL : https://uk.legacy.reactjs.org/ (дата звернення : 20.09.2023).

16. mongodb URL : https://www.mongodb.com/ (дата звернення : 24.09.2023).

17. Фука М. М., Пашко К. О., Гарасимів І. М., Гудима А. А., Чуткий С. І., Мельник Р. М., Білах Б. І. ЗАХИСТ УКРАЇНИ (рівень стандарту, «Основи медичних знань»). 10 клас. Астон, 2022, 305 с.

18. Лелека В. М. «ЗАХИСТ УКРАЇНИ (рівень стандарту, «Основи медичних знань»)». 10 клас. Ранок, 2023, 205 с.

19. Фука М. М., Пашко К. О., Гарасимів І. М., Гудима А. А., Чуткий С. І., Мельник Р. М., Білах Б. І. ЗАХИСТ УКРАЇНИ (рівень стандарту)» підручник для 10 класу закладів загальної середньої освіти. Астон, 2023, 380 с.

20. Фука М. М., Гарасимів І. М., Свідерський Ю. Ю., Пашко К. О., Свідерська Н. Ю. «Захист України (профільний рівень)» підручник для 10 класу закладів загальної середньої освіти. Астон, 2022, 328 с.

ДОДАТОК А

Головна сторінка

```
import React from 'react';
import Header from '../../components/Header/Header';
import styles from './Home.module.css';
import { Link } from 'react-router-dom';
import DropDownMenu from
'../../components/DropDownMenu/DropDownMenu';
import { navItems } from '../../components/NavItems/NavItems';
import { useSelector } from 'react-redux';
import { checkAuth } from '../../redux/features/auth/authSlice';
//import img from '../../img/title.jpg'

console.log(navItems);

export default function Home() {
  const isAuth = useSelector(checkAuth)
  return (
    <div className={styles.body}>
      <section className={styles['main-content']}>
        <div className={styles['main-article']}>
          <h1>ЗАХИСТ УКРАЇНИ</h1>
          <p >Приєднуйтесь зараз </p>
          {isAuth ? <Link to='Course/html/introduction' className={styles['main-
btn']}>Розпочати курс</Link> :
          <a className={styles['main-btn']} title='потрібно
зареєструватися'>Розпочати курс</a> }
```

```
        </div>
      </section>
       <div className={styles.homeBox}>
          <div className={styles.content}>
            <div className={styles.title}>
              <div className={styles.course}>
                <p align="justify" > Завдання цього курсу — оволодіння базовими знаннями про Збройні сили України, юридичними основами міжнародного гуманітарного права, домедичної допомоги, основ цивільного захисту, дотримання власної безпеки під час надзвичайних ситуацій мирного та воєнного часу.</p>
              </div>
              <img src="./img/title4.jpg" alt='title photo'/>
            </div>
          </div>
       </div>

    </div>
  )
}
```

ДОДАТОК Б

Організація сторінки курсу

```
import React from 'react';
import styles from './Course.module.css';
import { Link, Outlet } from 'react-router-dom';
import CustomLink from '../../components/CustomLink';


 function Topic1Page() {
 return (
   <div className={styles['courses-body']}>
   <div className={styles['courses-content']}>
     <ul className={styles['courses-navbar']}>
      <li><CustomLink to='/Course/topic1/introduction'
colorText='black'>Основи національної безпеки України  </CustomLink></li>

      <li><CustomLink to='/Course/topic1/Armed_forces' colorText='black'>
Збройні Сили України на захисті України  </CustomLink></li>
      <li><CustomLink to='/Course/topic1/Military_training' colorText='black'>
Стройова підготовка та прикладна фізична підготовка </CustomLink></li>
      <li><CustomLink to='/Course/topic1/Fire_preparation'
colorText='black'>Вогнева підготовка </CustomLink></li>
      <li><CustomLink to='/Course/topic1/Tactical_training'
colorText='black'>Тактична підготовка  </CustomLink></li>
      <li><CustomLink to='/Course/topic1/Civil_Protection'
colorText='black'>Основи цивільного захисту  </CustomLink></li>
```

```
        <li><CustomLink to='/Course/topic1/Home_medical_assistance'
colorText='black'>Домедична допомога.   </CustomLink></li>

  </ul>
         <Outlet/>
    </div>
</div>
  )
}

export default Topic1Page;
```

ДОДАТОК В

Організація сторінки тесту

```
import React, { useState, useEffect } from 'react';
import { useFormik } from 'formik';
import './test.css';
import { useDispatch, useSelector } from 'react-redux';
import { getMe } from '../../redux/features/auth/authSlice';

const Test = () => {
  const dispatch = useDispatch();
  const userData = useSelector(state => state.auth.user);

  useEffect(() => {
    dispatch(getMe());
  }, [dispatch]);

  const questions = [
    {
      id: 1,
      text: ' Бік строю, в який військовослужбовці звернені обличчям, а машини — лобовою частиною називається:',
      options: ['тил', 'фронт', 'фланг', 'ряд'],
      correctAnswer: ['тил'],
    },
    {
      id: 2,
```

```
    text: 'Визначене Стройовим статутом розташування військовослужбовців, підрозділів і частин для спільних дій у пішому порядку та на машинах -це:',
    options: ['колона', 'шеренга', 'стрій', 'дистанція'],
    correctAnswer: ['колона'],
  },
  {
    id: 3,
    text: ' Колона" - це...',
    options: ['стрій, у якому військовослужбовці вишикувані один біля одного на одній лінії згідно з визначеними інтерваламиКолона',
     'стрій, у якому військовослужбовці розташовані один за одним, а підрозділи (машини) один (одна) за одним (за одною) — на відстанях, визначених Стройовим статутом або командиром',
     'визначене Стройовим статутом розташування військовослужбовців, підрозділів і частин для спільних дій у пішому порядку та на машинах',
     'два військовослужбовці, які стоять у двошеренговому строю один за одним'],
    correctAnswer: ['стрій, у якому військовослужбовці вишикувані один біля одного на одній лінії згідно з визначеними інтерваламиКолона'],
  },
  {
    id: 4,
    text: ' Колона" - це...',
    options: ['стрій, у якому військовослужбовці вишикувані один біля одного на одній лінії згідно з визначеними інтерваламиКолона',
     'стрій, у якому військовослужбовці розташовані один за одним, а підрозділи (машини) один (одна) за одним (за одною) — на відстанях, визначених Стройовим статутом або командиром',
     'визначене Стройовим статутом розташування військовослужбовців, підрозділів і частин для спільних дій у пішому порядку та на машинах',
```

```
      'два військовослужбовці, які стоять у двошеренговому строю один за одним'],
    correctAnswer: ['стрій, у якому військовослужбовці вишикувані один біля одного на одній лінії згідно з визначеними інтервалами Колона'],
   },
   // Add more questions as needed
 ];

 const [results, setResults] = useState(null);
 const [completionTime, setCompletionTime] = useState(null);

 const formik = useFormik({
  initialValues: {
   answers: Array(questions.length).fill([]),
  },
  onSubmit: (values) => {
   const correctAnswers = values.answers.reduce((total, answer, index) => {
    const isCorrect =
     JSON.stringify(answer) === JSON.stringify(questions[index].correctAnswer);
    return isCorrect ? total + 1 : total;
   }, 0);

   setResults(correctAnswers);

   const currentTime = new Date();
   setCompletionTime(currentTime.toLocaleString());

   const currentDateString = currentTime.toISOString().split('T')[0];
   const currentTimeString = currentTime.toLocaleTimeString().replace(/:/g, '-');
```

```
    // Создаем имя файла с учетом фамилии пользователя, даты и времени
    const fileName =
`Result_${userData.lastname}_${currentDateString}_${currentTimeString}.txt`;

    // Создаем содержимое файла
    const fileContent = `
      РЕЗУЛЬТАТ
      Правильні відповіді: ${correctAnswers}
      Дата і час проходження: ${currentTime.toLocaleString()}
      Ім'я користувача: ${userData.firstname} ${userData.lastname}
      Email: ${userData.email}
    `;

    // Создаем Blob (Binary Large Object) из строки
    const blob = new Blob([fileContent], { type: 'text/plain' });

    // Создаем ссылку для скачивания
    const link = document.createElement('a');
    link.href = window.URL.createObjectURL(blob);
    link.download = fileName;

    // Добавляем ссылку в DOM и эмулируем клик для скачивания
    document.body.appendChild(link);
    link.click();

    // Удаляем ссылку из DOM
    document.body.removeChild(link);
  },
 });
```

```
const handleCheckboxChange = (questionIndex, option) => {
  const currentAnswers = formik.values.answers[questionIndex];
  const updatedAnswers = currentAnswers.includes(option)
    ? currentAnswers.filter((ans) => ans !== option)
    : [...currentAnswers, option];

  formik.setFieldValue(`answers[${questionIndex}]`, updatedAnswers);
};

return (
  <div className='test-wrapper'>
    <h1 className="title">Стройова підготовка. Строї, елементи строю.</h1>
    <form onSubmit={formik.handleSubmit}>
      {questions.map((question, index) => (
        <div key={question.id}>
          <p></p>
          <h2> Запитання {question.id}</h2>
          <p>{question.text}</p>
          <div>
            {question.options.map((option) => (
              <div key={option} style={{ marginBottom: '10px' }}>
                <label>
                  <input
                    type="checkbox"
                    name={`answers[${index}]`}
                    value={option}
                    checked={formik.values.answers[index].includes(option)}
                    onChange={() => handleCheckboxChange(index, option)}
                    style={{ marginRight: '5px' }}
```

```
                />
                {option}
              </label>
            </div>
          ))}
        </div>
      </div>
    ))}
    <input className='start-button' type='submit' value='ЗАВЕРШИТИ'/>
  </form>
  {results !== null && (
    <div>
      <h2 align='center'>РЕЗУЛЬТАТ</h2>
      <p align='center'>Правильні відповіді: {results}</p>
      <p align='center'>Дата і час проходження: {completionTime}</p>

      {userData && (
        <div>
          <p>{userData.firstname} {userData.lastname}</p>
          <p>{userData.email}</p>
        </div>
      )}
    </div>
  )}
 </div>
 );
};

export default Test;
```

Декларація
академічної доброчесності
здобувача ступеня вищої освіти ЗНУ

Я, Алєйкін Олександр Анатолійович

студент(ка) 2 курсу, заочної форми навчання, математичного факультету, спеціальності 122 комп'ютерні науки, адреса електронної пошти alejkynaleksandr@gmail.com, підтверджую, що написана мною кваліфікаційна робота магістра на тему «Розробка вебресурсу «Захист України» для здобувачів середньої освіти» відповідає вимогам академічної доброчесності та не містить порушень, що визначені у ст. 42 Закону України «Про освіту», зі змістом яких ознайомлений/ознайомлена;

– заявляю, що надана мною для перевірки електронна версія роботи є ідентичною її друкованій версії;

– згоден/згодна на перевірку моєї роботи на відповідність критеріям академічної доброчесності у будь-який спосіб, у тому числі за допомогою інтернет-системи, а також на архівування моєї роботи в базі даних цієї системи.

Студент

11.12.2023		Алєйкін О. А.
(дата)	(підпис)	(прізвище, ініціали)

Науковий керівник

(дата)	(підпис)	(прізвище, ініціали)