МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ ФАХОВИЙ БІЗНЕС-КОЛЕДЖ
Циклова комісія (кафедра) комп’ютерної інженерії та інформаційних технологій

КВАЛІФІКАЦІЙНА РОБОТА

на тему

WEB-ОРІЄНТОВАНИЙ ПРОГРАМНИЙ ЗАСІБ ДЛЯ РОБОТИ В ГЛОБАЛЬНІЙ МЕРЕЖІ INTERNET

Виконав: студент групи 1К-21
Спеціальності 123 Комп’ютерна інженерія
Юрій ДОНЧЕНКО
Керівник:
Наталя ФАЛЬЧЕНКО

Черкаси 2025

АНОТАЦІЯ

Кваліфікаційна робота присвячена дослідженню та практичній реалізації інформаційного web- сервісу на базі браузерних технологій. В умовах сьогодення життя неможливе без оперативного та зручного доступу до інформаційних ресурсів через мережу Інтернет та Web-додатки, що працюються в браузерах, стають головним інструментом для користувачів і бізнесу. Загальний обсяг роботи 34 сторінки, містить 18 рисунків, 11 таблиць та посилається на 29 джерел.

У роботі проведено аналіз сучасних підходів до розробки web- сайтів, розглянуто основні методи розробки web- сайтів на базі бібліотеки React.js. Детально охарактеризовано одно сторінкову структуру web-сайту яку було обрано для практичної реалізації, та проведено порівняння з іншими структурами(багато сторінковова, з глибокою ієрархією). Здійснено огляд інструментів для роботи з бібліотекою React.js (Tailwind CSS, Next.js, Jest, React Router, Formik, Material UI та інші), це дало визначити переваги та недоліки сучасних інструментів. Ключовою частиною роботи є розробка інформаційного web-сервісу “Безпілотні системи”. Описано обрану структуру та представлено схему структури web- сервісу “Безпілотні системи”, що візуалізує послідовність взаємодії користувач з web-сайтом. Практична цінність роботи полягає в розробці інформаційного сервісу, що демонструє застосування безпілотних систем у різних сферах людської діяльності. Проведено тестування розробленого інформаційного сервісу “Безпілотні системи”, результат якого показує, що web-сайт оптимізований та має адаптацію для телефонів.

Ключові слова: інформаційний web-сервіс, web-сайт, безпілотні системи, інструменти, бібліотеки.

ABSTRACT

The qualification work is devoted to the research and practical implementation of an information web service based on browser technologies. In today's world, life is impossible without prompt and convenient access to information resources via the Internet and Web applications running in browsers are becoming the main tool for users and businesses. The total volume of the work is 34 pages, contains 18 figures, 11 tables and references 29 sources.

The paper analyzes modern approaches to the development of web sites, considers the main methods of developing web sites based on the React.js library. The single-page structure of the website, which was chosen for practical implementation, is characterized in detail and compared with other structures (multi-page, with a deep hierarchy). A review of tools for working with the React.js library (Tailwind CSS, Next.js, Jest, React Router, Formik, Material UI, and others) was conducted, which helped to identify the advantages and disadvantages of modern tools. A key part of the work is the development of the Unmanned Systems information web service. The chosen structure is described and a diagram of the structure of the Unmanned Systems web service is presented, which visualizes the sequence of user interaction with the website. The practical value of the work lies in the development of an information service that demonstrates the use of unmanned systems in various fields of human activity. The developed information service “Unmanned Systems” was tested, the result of which shows that the website is optimized and has adaptation for phones.

Keywords: information web service, web site, unmanned systems, tools, libraries.

ЗМІСТ

ВСТУП

Сучасність неможлива без оперативного та зручного доступу до інформаційних ресурсів через мережу Інтернет та Web-додатки, що працюються в браузерах, стають головним інструментом для користувачів і бізнесу.

Згідно з дослідженням Stack Overflow, React використовує 40.6% розробників фронтенду, що робить його найпопулярнішою бібліотекою для створення інтерактивних інтерфейсів [1].

Тема кваліфікаційної роботи актуальна, тому що вона досліджує можливості бібліотеки React для розробки web-додатків. Також вона пропонує практичне рішення для створення інформаційного сервісу, який в майбутньому може бути використаний у реальних проєктах.

За даними MarketsandMarkets, ринок безпілотних систем до 2027 року досягне $58.4 млрд, зі щорічним зростанням на 16.4%, що підкреслює їх значення в сучасних технологіях [2].

Об’єктом дослідження є інформаційний web-сервіс на базі браузерних технологій.

Предметом дослідження є інструменти розробки web-додатків, зокрема бібліотека React та її сервіси, методи оптимізації продуктивності.

Метою кваліфікаційної роботи є проведення аналізу сучасних підходів до розробки web-додатків з використанням інструментів React, а також дослідження архітектурних особливостей React.

Поставлені завдання:

- розробити інформаційний web-сервіс на базі браузерних технологій;
- виконати тестування розробленого web-сервісу, провести перевірку продуктивності та протестувати інтерфейс на зручність та гнучкість;

Методи дослідження, які використовувалися в ході виконання кваліфікаційної роботи наступні:

- аналіз літератури та документацій React;
- графічні методи для розробки візуального інтерфейсу.
- експеримент: для тестування обрано Jest, як найпоширеніший фреймворк для React-додатків, що дозволяє покрити до 90% коду юніт-тестами.

Як зазначається в дослідженні “Vue vs. React in terms of Performance” (Evan You, 2015), для великих таблиць із великою кількістю простих елементів (понад 10 000) React показує кращу продуктивність, оскільки кожна клітинка там — це «просто віртуальний елемент» у React, тоді як у Vue — фрагмент [3] .

Практична значимість роботи:

Розроблений web-додаток може бути використаний як довідник для користувачів які цікавляться безпілотними системами та способами їх використання у різних сферах людської діяльності.

РОЗДІЛ 1 ОГЛЯД ІСНУЮЧИХ ІНСТРУМЕНТІВ ТА ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ WEB-ДОДАТКІВ

1.1 Технології розробки web-додатків

Сучасна веброзробка є динамічною та стрімко розвивається. Це галузь, що постійно пропонує нові підходи та інструменти для створення якісних додатків [4].

Вебдодатки сьогодні стали невід'ємною частиною нашого життя - від соціальних мереж до банківських систем, від інтернет-магазинів до складних корпоративних рішень [5]. Розвиток технологій дозволив створювати складні системи, які працюють швидко та ефективно, забезпечуючи зручний інтерфейс для користувачів [6].

Сучасний стек технологій для веброзробки включає:

1. Інструменти для фронтенд розробки
2. Бекенд технології
3. Системи керування базами даних
4. Інструменти для тестування
5. Системи контролю версій
6. Інструменти для розробки інтерфейсів [5]

Кожен з цих компонентів грає важливу роль у створенні сучасного вебдодатку. Важливо розуміти, що вибір конкретних технологій залежить від багатьох факторів: масштабу проєкту, очікуваного навантаження, досвіду команди розробників та конкретних вимог до продукту [6].

Розвиток веб-технологій останніх років дозволив створювати додатки, які за продуктивністю та функціональністю майже не поступаються десктопним програмам. Такі концепції, як Progressive Web Apps (PWA) або Isomorphic JavaScript, значно розширили можливості веб-додатків [8].

Окремо варто відзначити зростаючу популярність хмарних технологій у веб-розробці. Такі сервіси, як AWS, Google Cloud або Azure, надають розробникам потужні інструменти для розгортання, масштабування та

підтримки веб-додатків [7].

Сучасні вебдодатки також все частіше використовують штучний інтелект та машинне навчання для покращення користувацького досвіду. Це може бути персоналізація контенту, чат-боти або системи рекомендацій [6].

Сучасна веброзробка використовує різноманітні технології, які можна умовно поділити на кілька категорій, що відображає таблиця 1.1.

Таблиця 1.1 Категорії сучасної web-розробки

Категорії	Приклади
Frontend-technologies	HTML5, CSS3, JavaScript (ES6+)
Frameworks	React.js, Vue.js, Angular
Preprocessors	Sass, Less, Stylus
Assembly tools	Webpack, Vite, Parcel
Backend-technologies	Node.js(Express,NestJS)Python(Django, Flask)PHP(Laravel,Symfony)Java(Spring Boot)
Databases	PostgreSQL, MySQL
Document-oriented	MongoDB
Cash systems	Redis
Other important technologies	GraphQL для APIDocker для контейнеризації Jest.

1.2 Інструменти середовища React.js для розробки web-додатків

Середовище React.js має багату екосистему технологій, що включає основні бібліотеки, інструменти розробки і додаткові бібліотеки. Основні бібліотеки відображені в таблиці 1.2. Інструменти розробки відображені в таблиці 1.3. Додаткові бібліотеки відображені в таблиці 1.4.

Таблиця 1.2 Основні бібліотеки React.js

№	Назва бібліотеки	Сфера використання
1	React Router	використовується для маршрутизації, тобто керування переходами між компонентами або сторінками;
2	Redux/Context API	дозволяє створити сховище , де зберігаються всі данні які можуть бути доступні в різних компонентах додатку. Ця бібліотека буде корисна в складних застосунках де потрібно керувати глобальним станом. Context API використовується для передачі данних вниз по дереву без проходження через компоненти які не потребують цих даних.
3	Axios для HTTP-запитів	використовується для здійснення HTTP-запитів, це є основною взаємодією(як приклад браузер) та сервер. Це допомагає отримувати дані з сервера та надсилати їх.
4	Formik для роботи з формами	Використовується для спрощення роботи з формами React, яка автоматизує багато завдань, такі як управління станом форми та валідація даних.
5	Create React App	викикористовується для швикдоко створення React-проєктів
6	Node.js	використовується для створення локального серверу, для перевірки його працездатності та його вигляду. Також цей інструмент дозволяє запустити сервер у стані розробки тобто всі дії які ви робити з кодом одразу змінюють додаток у реальному часі.
7	Storybook	розробки та документації UI-компонентів.
8	React DevTools	відлагодження та аналізу React-додатків. Він дозволяє інспектувати дерево компонентів, перевіряти їхні властивості та стан.

Таблиця 1.3 Інструменти web-розробки

№	Назва інструменту	Сфера застосування
1	Create React App	використовується для швидкого створення React-проєктів
2	Node.js	використовується для створення локального серверу, для перевірки його працездатності та його вигляду. Також цей інструмент дозволяє запустити сервер у стані розробки тобто всі дії які ви робити з кодом одразу змінюють додаток у реальному часі.
3	Storybook	розробки та документації UI-компонентів.
4	React DevTools	відлагодження та аналізу React-додатків. Він дозволяє інспектувати дерево компонентів, перевіряти їхні властивості та стан.

Таблиця 1.4 Додаткові бібліотеки

№	Назва бібліотеки	Сфера застосування
1	Material UI	використовується для створення користувацьких інтерфейсів.
2	Framer Motion	використовується для створення анімації та інтерактивних елементів у додатку.
3	React Query	полегшує роботу з асинхронними операціями такими як(кешування, синхронізація та оновлення даних у React додатках).
4	React Hook Form	використовується для спрощення роботи з формами у React.

1.3 Порівняльний аналіз проєктів, розроблених на React.js

Сучасні проєкти на React.js демонструють різні підходи до архітектури залежно від їх масштабу та складності. Для великих проєктів, таких як соціальні мережі або корпоративні системи, характерне використання складних архітектурних рішень. В таких випадках часто застосовують комбінацію монолітної структури з мікросервісами, що дозволяє ефективно масштабувати додаток. Для керування станом у великих проєктах переважно використовують Redux разом із Saga або Thunk, що забезпечує передбачуваність стану додатку. Маршрутизація реалізується з динамічним завантаженням компонентів (lazy loading), а для підвищення продуктивності застосовують SSR та code splitting.

Середні за розміром проєкти, такі як корпоративні додатки або інтернет-магазини, зазвичай використовують модульну структуру. Для керування станом часто обирають Context API разом із useReducer, що є більш легковаговим рішенням порівняно з Redux. Маршрутизація реалізується стандартними засобами React Router, а для оптимізації використовують ліниве завантаження компонентів. Тестування в таких проєктах зазвичай охоплює основні компоненти та критичну логіку додатку.

Малі проєкти, такі як лендінг-сторінки або прості веб-додатки, характеризуються простою компонентною структурою. Керування станом часто обмежується локальним станом компонентів (useState), що значно спрощує розробку. Маршрутизація реалізується мінімально необхідними засобами, а оптимізація зводиться до базових прийомів. Тестування в таких проєктах може бути мінімальним або взагалі відсутнім.

Цей аналіз показує, що React.js є достатньо гнучкою бібліотекою, яка дозволяє адаптувати підхід до розробки залежно від масштабу проєкту. Вибір архітектури, інструментів керування станом та підходів до оптимізації повинен враховувати конкретні вимоги проєкту та очікуване навантаження (характеристика розмірів проєкту показано в табл.1.5).

Таблиця 1.5 Характеристика проєктів

№	Категорія	Характеристика
1	Великий	Великі проєкти зазвичай мають значний бюджет (від кількох мільйонів до мільярдів доларів) [8]. Тривалий термін реалізації (3+ роки), високу складність та залучення багатьох учасників [9]. Приклади: будівництво аеропортів, автомагістралей, масштабні IT-системи [10].
2	Середній	Середні проєкти мають бюджет від десятків тисяч до кількох мільйонів доларів [8]. Термін реалізації – від кількох місяців до 2–3 років. Вони менш складні, ніж великі, але вимагають чіткого планування [9]. Приклади: розробка програмного забезпечення[10].
3	Малий	Малі проєкти мають обмежений бюджет (до кількох десятків тисяч доларів) [8]. Короткий термін виконання (тижні або місяці) та невелику команду [9]. Приклади: створення вебсайту, локальні маркетингові кампанії [10].

Висновок до першого розділу

В першому розділі були розглянуті технології web-розробки, інструменти бібліотеки React.js такі як: основні інструменти, інструменти розробки та додаткові інструменти. Головні з них це Node.js для запуску локального серверу, React Router це бібліотека для маршрутизації проєкту та додаткові такі як Material UI використовується для створення користувацьких інтерфейсів.

Також були розглянуті великі, середні, малі проєкти та їх характеристики, такі як (час розробки, бюджет та складність реалізації).

РОЗДІЛ 2 ПРОЄКТУВАННЯ WEB-САЙТУ

2.1 Розробка структури web-сайту “Безпілотні системи”

Підчас розробки структури сайту були переглянуті структури сайтів і проаналізовані їх характеристики, які відображені в таблиці 2.1.

Таблиця 2.1 Типи структур web-сайтів

№	Тип структури	Характеристика
1	Односторінкова	Вся інформація розміщена на одній сторінці з прокруткою або динамічним перемиканням розділів. Основні характеристики: - простота розробки та підтримки [11] - швидке завантаження (менше HTTP-запитів) [12] - ідеально для лендінгів, портфоліо, простих сервісів [13] - погана SEO-оптимізація (один URL для всього контенту) [11]
2	Багатосторінкова	Класична структура з окремими сторінками для кожного розділу. Особливості: - чітка навігація та логічна організація контенту [12] - краща SEO-оптимізація (окремі URL для кожної сторінки) [13] - вища вартість розробки (потрібно створювати кожну сторінку) [12] - ідеально для інтернет-магазинів, корпоративних сайтів, блогів [12]

Продовження таблиці 2.1

№	Тип структури	Характеристика
3	З глибокою ієрархією	Складні структури з багатьма рівнями вкладеності (наприклад: категорії → підкатегорії → товари). Ключові аспекти: - потрібна ретельна планування навігації [13] - ризик "загубити" користувача у складних меню [11] - добре підходить для великих інформаційних порталів, каталогів (наприклад, Wikipedia) [12] - вимагає продуманої системи пошуку та "хлібних крихт" [13]

Для розробки сайту обрана односторінкова структура, яка представлена на рис.2.1. Структура містить головну сторінку та 4 категорії застосування, які розроблені за допомогою тегів секторів. Кожна категорія має свій сектор, який відслідковується, що відображено на рис. 2.2.

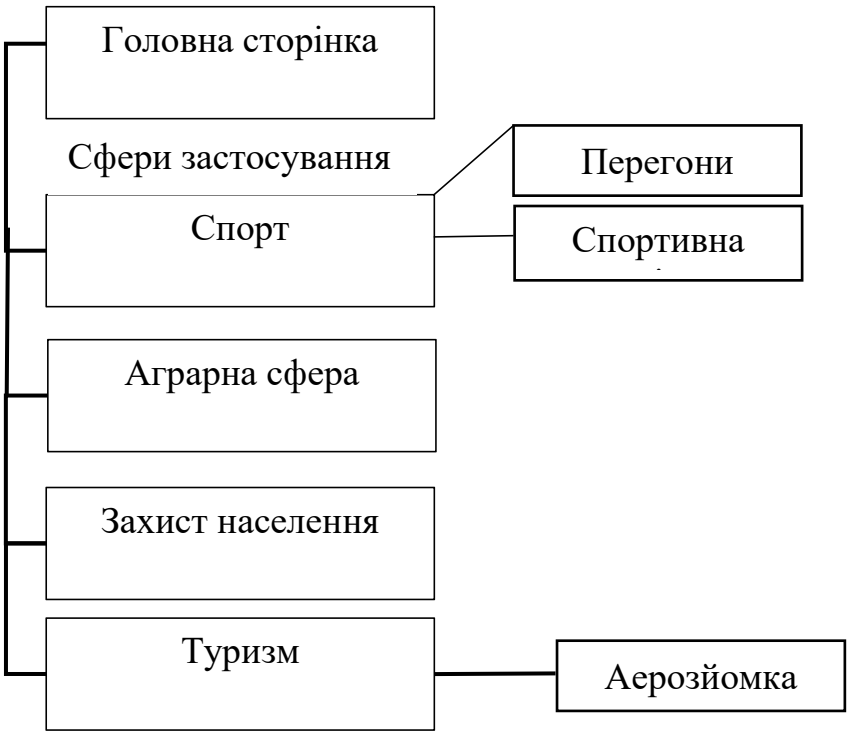

Рисунок 2.1 – Схема вебсайту “Безпілотні системи”.

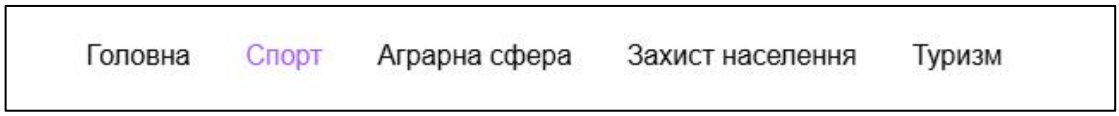

Рисунок 2.2 – Навігація сайту “Безпілотні системи”

2.2 Вибір інструментів середовища JavaScript для реалізації web-сайту за допомогою бібліотеки React

Розробка сучасного web-сайту на React вимагає використання різноманітних інструментів, які покращують продуктивність, спрощують управління кодом та забезпечують якісний дизайн. Нижче наведено перелік ключових інструментів, поділених на категорії, з детальним описом їх призначення та сфер застосування.

Інструменти дизайну відображені в таблиці 2.2. Інструменти для менеджменту коду та розробки відображені в таблиці 2.3. Інструменти для тестування відображені в таблиці 2.4.

1. Інструменти для розробки дизайну інтерфейсу представлені в таблиці 2.2:

Таблиця 2.2 Інструменти для розробки дизайну

№	Назва	Призначення	Використання	Особливості
1	Figma	Векторний графічний редактор для створення UI/UX-дизайну [14].	- створення макетів сторінок додатку розробка дизайн-систем (стилі, кольори, компоненти) - генерація CSS-коду для розробників [14].	- реальний час спільної роботи; - хмарне зберігання; - безкоштовний стартовий план [14].
2	Storybook	Інструмент для ізольованої розробки, тестування та	- Створення бібліотеки компонентів (кнопки, форми, модальні вікна) - Тестування компонентів у різних	- Підтримка React/Vue/Angular; Інтерактивна демонстрація; Інтеграція з

Продовження таблиці 2.2

		документування UI-компонентів [15].	станах - Документування правил використання UI-елементів [15].	- тестами [15].
3	Tailwind CSS	Utility-first CSS-фреймворк для швидкої стилізації компонентів без написання власного CSS.	- Швидке створення адаптивних інтерфейсів - Уникнення "CSS-спагетті" завдяки атомарним класам - Інтеграція з React через className [16].	- Низький рівень специфічності CSS; Можливість кастомізації; Швидка розробка [16].
4	Material-UI (MUI) / Ant Design	Бібліотеки готових UI-компонентів (кнопки, таблиці, форми тощо) у відповідності до дизайн-систем Google (Material Design) або Ant [17].	- швидка розробка інтерфейсів без необхідності створювати компоненти з нуля; - підтримка тем оформлення (світла/темна); - готові рішення для складних елементів (навігація, графіки) [17].	- дотримання дизайн-систем; - велика спільнота; - регулярні оновлення [17].
5	Framer Motion	Бібліотека для створення анімацій та інтерактивних ефектів у React [18].	- плавні переходи між станами; - складні анімації (наприклад, drag-and-drop); - інтерактивні ефекти при наведенні [18].	- простий API; - висока продуктивність; - підтримка жестів [18].

2. Інструменти для менеджменту коду та розробки представлені в таблиці 2.3.

Таблиця 2.3 Інструменти для менеджменту коду

№	Назва	Призначення	Використання	Особливості
1	ESLint + Prettier	ESLint аналізатор коду для виявлення помилок і дотримання стилю. Prettier інструмент для автоматичного форматування коду [19].	- автоматичне виправлення синтаксичних помилок; - уніфікація стилю коду в команді; - інтеграція з Git для перевірки перед комітом [19].	- налаштування правил; - підтримка багатьох мов; - плагіни для IDE [19].
2	Git+ GitHub/GitLab	Система контролю версій для спільної роботи над кодом [20].	- відстеження змін у коді; - розгалуження (branching) для паралельної розробки; - code review та CI/CD (GitHub Actions, GitLab CI) [20].	- децентралізована система; - інтеграція з іншими інструментами[20].

Продовження Таблиця 2.3

3	Redux Toolkit / Zustand	Бібліотеки для керування глобальним станом додатку [21].	- зберігання даних, доступних у різних компонентах (наприклад, дані користувача); - оптимізація продуктивності (мінімізація повторних рендерів); - Redux Toolkit спрощує роботу з класичним Redux[21].	- Redux Toolkit: стандартизований підхід; - Zustand: легкість та простота [21].
4	React Query / Apollo Client (для GraphQL)	Бібліотеки для роботи з API (запити, кешування, синхронізація)[22].	- отримання даних з бекенду; - автоматичне кешування та оновлення; - оптимізація мережевих запитів [22].	- React Query: для REST; - Apollo: для GraphQL [22].
5	Vite / Next.js	Vite сучасний збірник для швидкої розробки. Next.js фреймворк для React з підтримкою SSR [23].	- Vite: Швидкий старт проєкту з гарячим оновленням (HMR); - Next.js: SEO-оптимізація, маршрутизація, API-роути [23].	- Vite: високошвидкісний; - Next.js: SEO-оптимізація [23].

3. Інструменти для тестування представлені в таблиці 2.4:

Таблиця 2.4 Інструменти для тестування

№	Назва	Призначення	Використання	Особливості
1	Jest	Фреймворки для юніт-та інтеграційного тестування [24].	- тестування окремих компонентів; - перевірка логіки додатку [24].	- швидкість виконання; - вбудований coverage-аналізатор; - Mock-функції [24].
2	Cypress / Playwrigh	Інструменти для e2e-тестування (імітація дій користувача) [25].	- автоматизація тестування форм, навігації; - відеозапис тестів для аналізу помилок [25].	- Cypress: простий у використанні; - Playwright: підтримка декількох мов [25].
3	React Testing Library (RTL)	Бібліотека для тестування компонентів-React [26].	- перевірка рендерингу, подій(кліки, введення тексту); - тестування компонентів так, як би з ним взаємодіяв користувач [26].	- тестування "як користувач"; інтуїтивний API; - інтеграція з Jest [26].

Висновок до другого розділу

У другому розділі виконано проєктування web-сайту. Були розглянуті структури web-сайтів такі як(односторінкова, багатосторінкова та з глибокою ієрархією).

Також були розглянуті інструменти для реалізації web-сайту ::

- для розробки дизайну - Figma та Tailwind CSS для реалізації дизайну в коді;
- для менеджменту коду - Next.js;
- для тестування коду - Jest;

Для створення проєкту на React потрібно використовувати спеціальні бібліотеки та додаткидля покращення продуктивності та реалізації дизайну.

РОЗДІЛ 3 РОЗРОБКА І ТЕСТУВАННЯ WEB-САЙТУ

3.1 Покрокова реалізація спроєктованої структури web-сайту

На рис.3.1 зображено технологічний стек спроєктованого web-сайту.

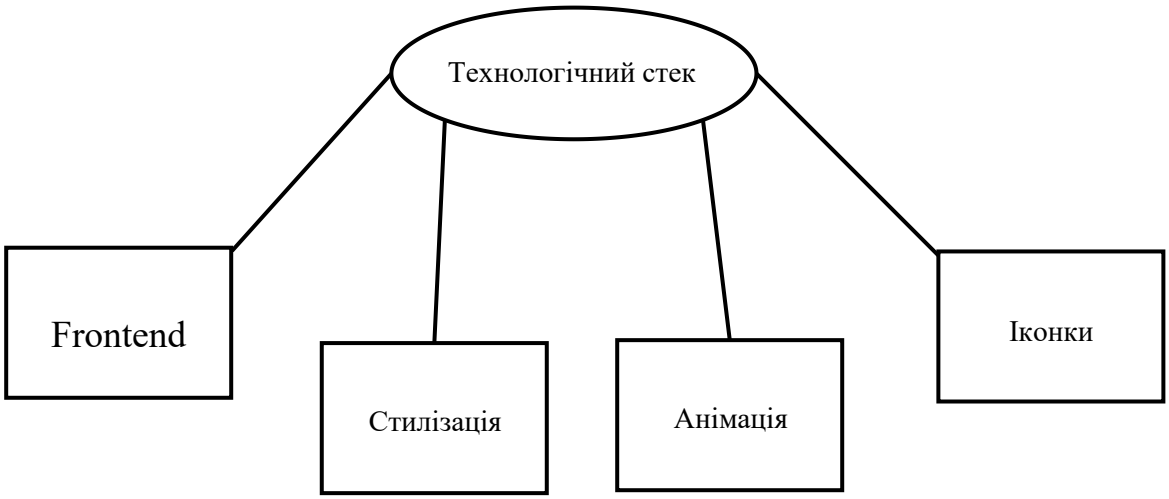

Рисунок 3.1 – Технологічний стек

Покрокова реалізація спроєктованої структури web-додатку включає такі етапи:

1. Налаштування проєкту:

2. Ініціалізація Next.js-додатку з TypeScript.

3. Підключення бібліотек (lucide-react, tailwindcss).

4. Реалізація інтерфейсу:

Header: Адаптивне меню зі станом активного розділу (хуки useState, useRef).

Секції: П'ять основних блоків (Головна, Спорт, Аграрна сфера тощо) з плавною навігацією.

Відеофон: Вставка відео через тег <video> з оптимізацією (авто відтворення, loop).

5. Налаштування адаптивності:

- мобільне меню з тоглом

- футер: Випадаюче меню соцмереж (isFooterMenuOpen).

6. Налаштування візуалізації представлено на рисунку 3.2:

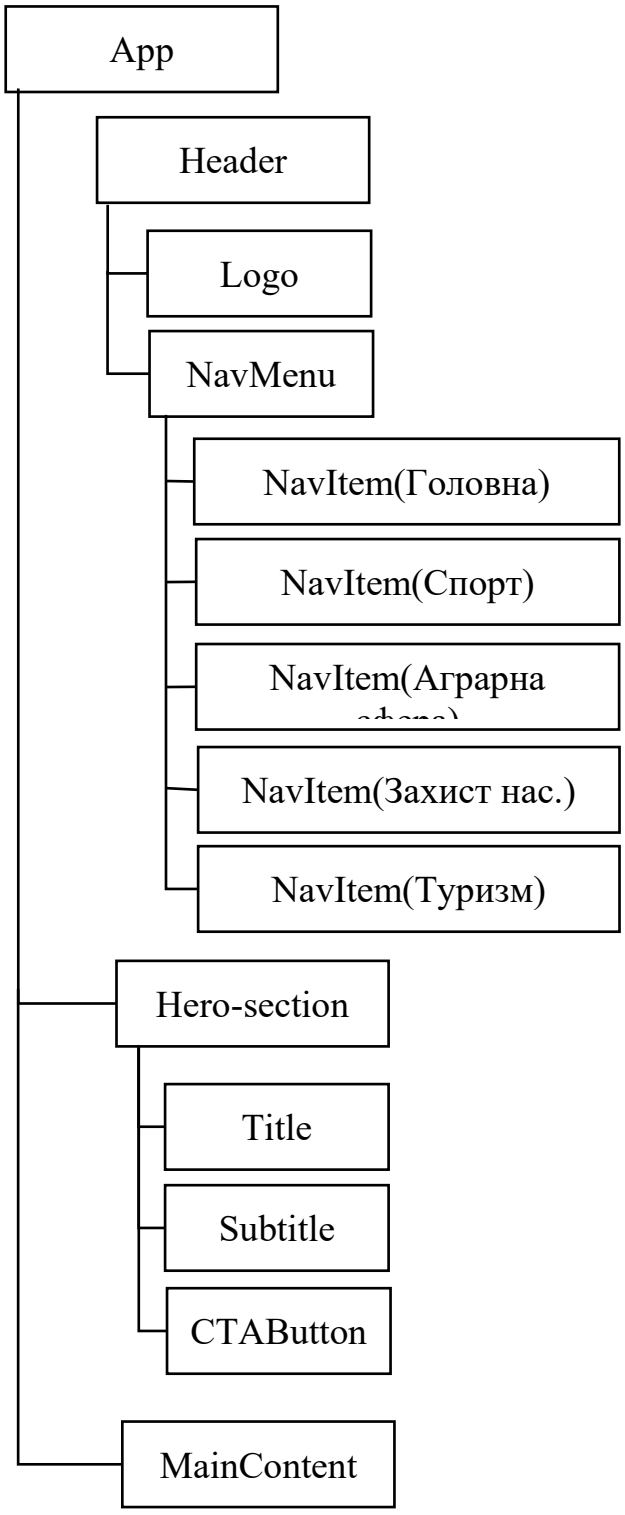

Рисунок 3.2 – Схема компонентів додатку.

Web-сайт має зручний і комфортний інтерфейс для мобільних пристроїв

зображений на рис.3.3, а для декстопних – на рис.3.4.

Рисунок 3.3 – Головне меню web - сайту мобільна версія

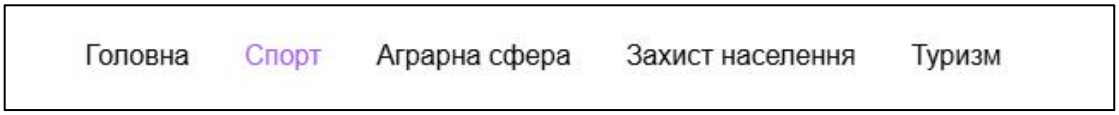

Рисунок 3.4 – Головне меню web - сайту десктопна версія

3.2 Інструкція до роботи з web-сайтом “Безпілотні системи”

3.2.1 Інструкція для розробника

Структура папок проєкту зображена на рис. 3.5

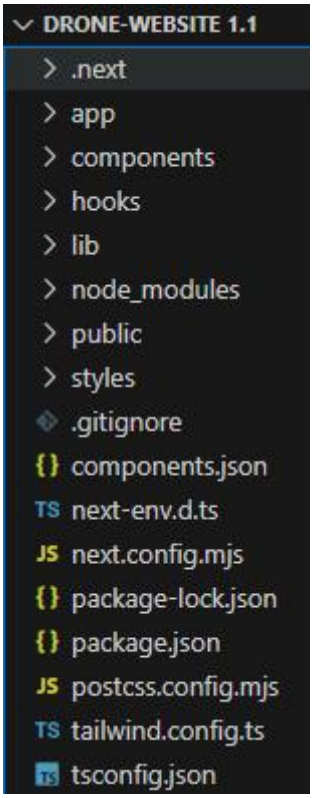

Рисунок 3.5 – Структура проєкту в файловому вигляді

Для запуску проєкту потрібно відкрити папку з проектом(drone-website 1.1) та зайти в термінал, зображений на рис.3.6.

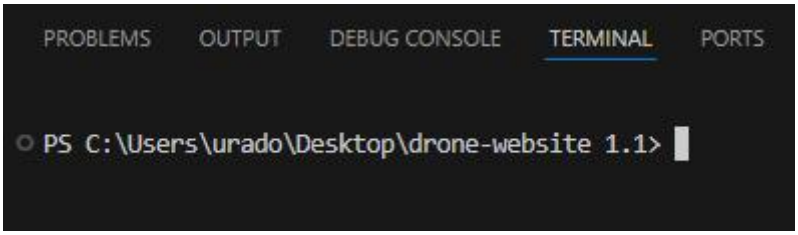

Рисунок 3.6 – Термінал з шляхом до файлу

Запустити локальний сервер через Next.js, як зображено на рис.3.7. Далі зайти за адресою http://localhost:3000.

Якщо шлях до файлу не правильний, треба виконати дії, які зображені на рис.3.8, після чого виконати дії, які перелічені на рис.3.6.

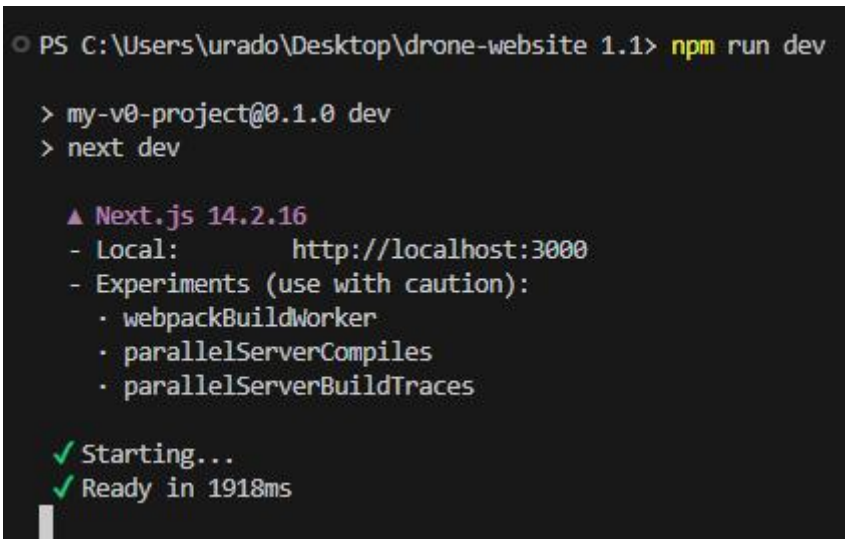

Рисунок 3.7 – Приклад запуску локального серверу.

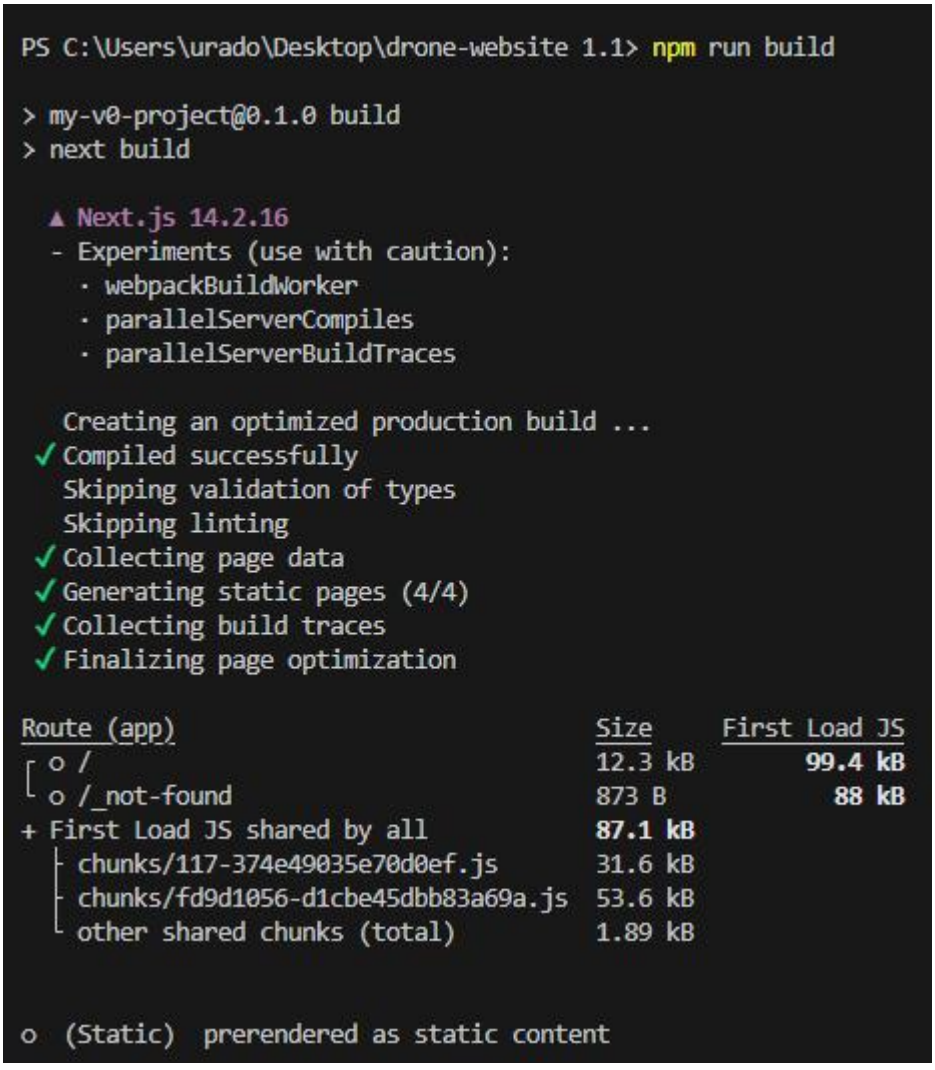

Рисунок 3.8 – приклад створення build проєкту.

Для зміни конкретних компонентів, які пов'язані з головною сторінкою потрібно зайти в папку ‘app’, що зображено на рис.3.9.

В папці ‘app’розташовано 3 файли:

- global.css - відповідає за весь стиль головної сторінки;
- layout.tsx - відповідає за метадані та стилі;
- page.tsx - відповідає за головну сторінку.

Для зміни відео або фото на web-сайті потрібно зайти в папку public як зображено на рис.3.10.

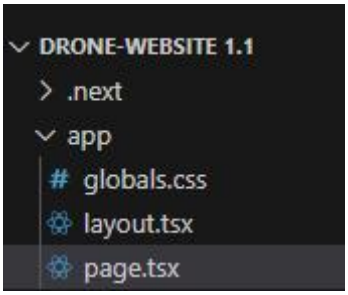

Рисунок 3.9 – Шлях до коду головної сторінки.

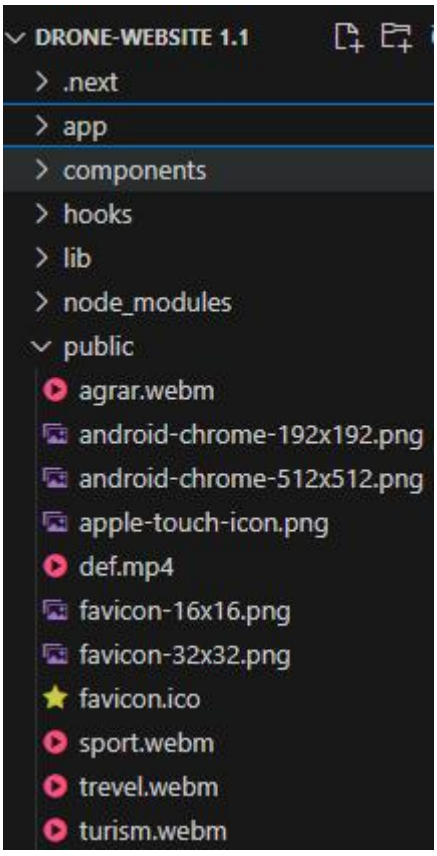

Рисунок 3.10 – Шлях до папки public.

3.2.2 Користувацька інструкція по використанню web- сайту

Після запуску локального серверу буде доступний web-сайт за посиланням http://localhost:3000. Головне вікно web - сайту зображено на рисунку 3.11.

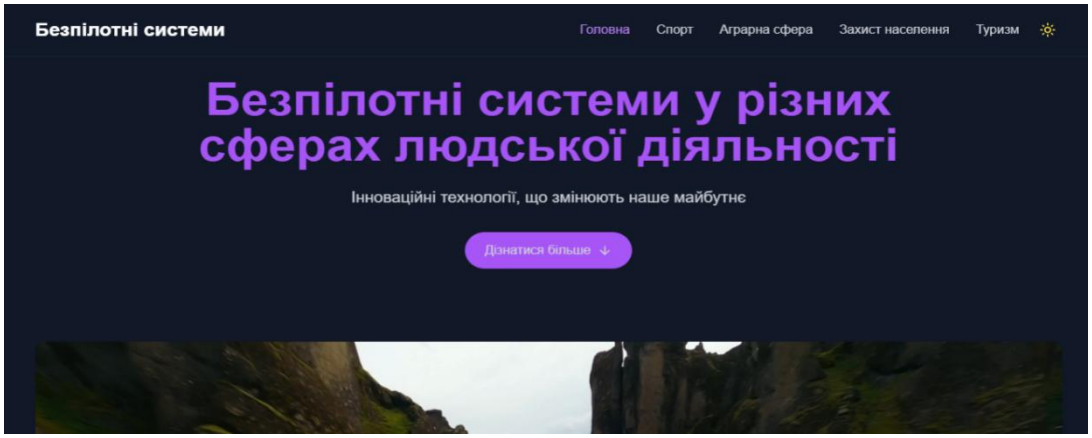

Рисунок 3.11 – Головне вікно web-сайту “Безпілотні системи”.

Web-сайт має одно сторінкову структуру, та містить зручну навігацію, а саме кнопки головного меню, які відображені на рисунку 3.12.

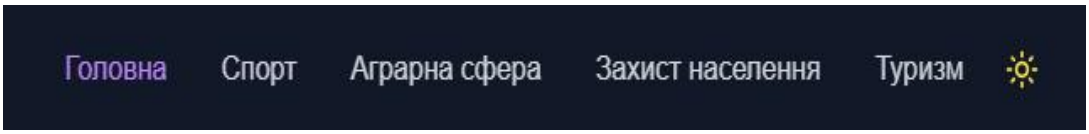

Рисунок 3.12 – Меню web-сайту “Безпілотні системи”.

При натисканні на кнопку Туризм відбувається перехід на сторінку, яка відображена на рисунку 3.13.

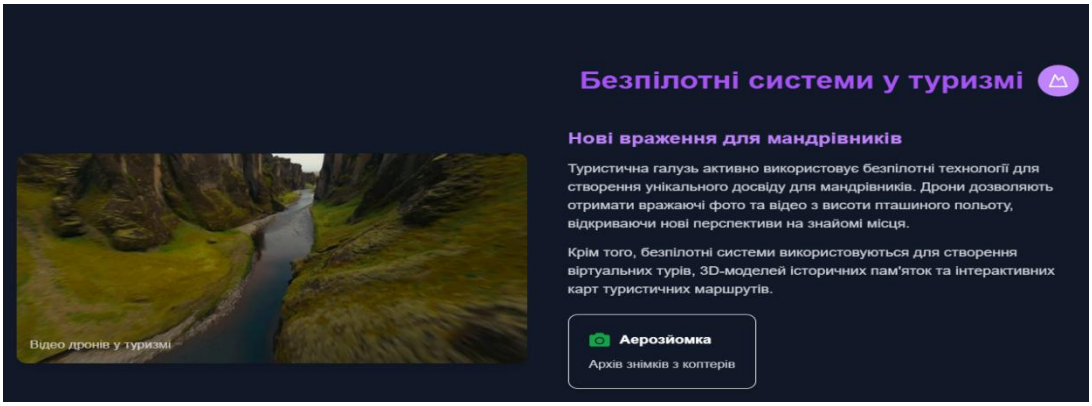

Рисунок 3.13 – Блок безпілотних систем у спорті.

Зі сторінки “Туризм” є перехід на сторінку “ аерофотозйомки ”, яка відображена на рисунку 3.14.

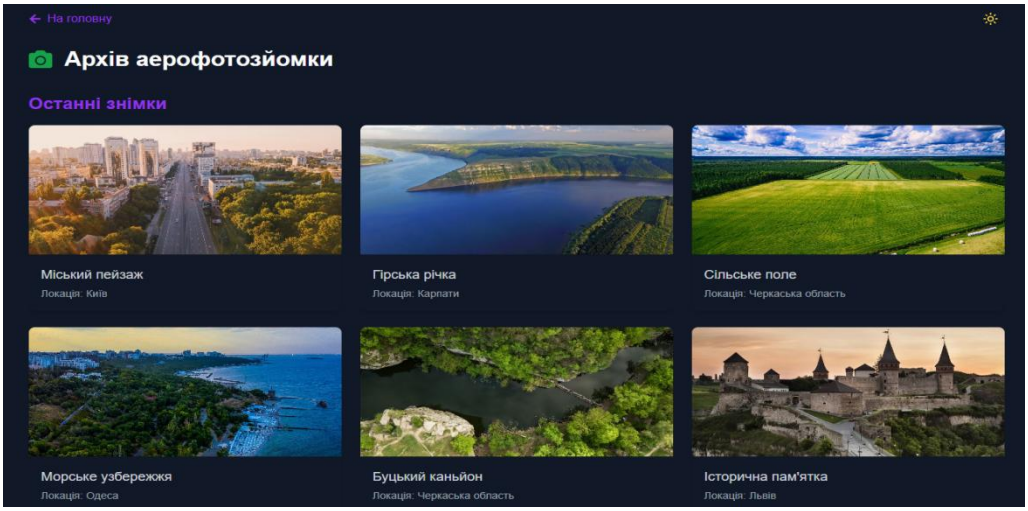

Рисунок 3.14 – Сторінка аерофотозйомки.

Після закінчення перегляду архіву, на кожній сторінці є кнопка повернення на головну сторінку.

Сайт містить футер, який відображений на рис.3.15. Футер містить навігацію для повернення до певних сторінок, або для переходу до соціальних мереж.

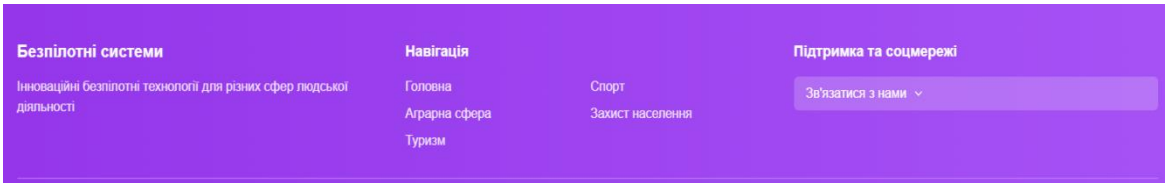

Рисунок 3.15 – Футер web-сайту “Безпілотні системи”.

3.3 Тестування розробленого web- сайту

Види тестування перелічені на рисунку 3.16.

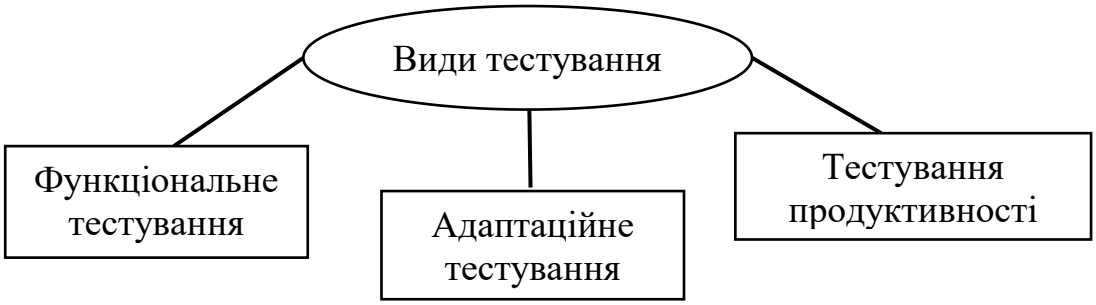

Рисунок 3.16 – Види тестування.

Функціональне тестування – це перевірка відповідності системи вимогам специфікації. Тестується кожна функція додатку на коректність роботи [27].

Методики тестування представлені нижче:

- модульне тестування (Unit Testing);
- інтеграційне тестування (Integration Testing);
- системне тестування (System Testing);
- регресійне тестування (Regression Testing);

Інструменти тестування представлені нижче:

- Jest, Mocha (JavaScript);
- Selenium (веб-додатки);
- Postman (API-тестування);

Адаптаційне тестування – це оцінка зручності використання інтерфейсу для користувачів [28].

Критерії тестування наведені нижче:

- інтуїтивність навігації;
- читабельність тексту;
- адаптивність під різні пристрої (десктоп, мобільні).

Інструменти тестування наведені нижче:

- Google Lighthouse;
- Hotjar (анализ поведінки користувачів);
- Figma Mirror (тестування прототипів).

Тестування продуктивності – це перевірка швидкодії, стабільності та масштабованості системи під навантаженням[29].

Типи тестування наведені нижче:

- навантажувальне тестування (Load Testing);
- стрес-тестування (Stress Testing);
- тестування стабільності (Soak Testing).

Інструменти тестування наведені нижче:

- JMeter;
- WebPageTest.

Для тестування web – сайту “Безпілотні системи” були використані такі види тестування:

1. Функціональне – включає в себе перевірку кліків у меню та плавного скролу, валідацію відтворення відео.
2. Тест на адаптивність – включає в себе тестування на різних розширеннях (Mobile: 360px, Desktop: 1920px).
3. Тест на продуктивність - включає в себе перевірку наскільки ефективно працює додаток при різних умова навантаження.
4. Результати тестування зазначені в таблиці 3.1:

Таблиця 3.1 – Таблиця з результатами тестів.

№	Назва тесту	Результати	Висновки
1	Тест швидкості завантаження	<1.5 сек	Web-сайт завантажується швидко, це забезпечує хорошу продуктивність.
2	Тест адаптивності	Коректно	Інтерфейс коректно масштабується на різних пристроях.
3	Крос-браузерний тест	Chrome, Microsoft Edge	сайт працює без помилок в популярних браузерах.

3.4 Особливості розробленого сайту

Було проведено повний цикл розробки та тестування web-сайту "Безпілотні системи", який демонструє застосування безпілотних технологій у різних сферах життя.

1. Реалізація інтерактивного інтерфейсу:

− сайт створено на базі Next.js (React) з використанням TypeScript, що забезпечило строгу типізацію та зручну розробку.

− використано Tailwind CSS для стилізації, що дозволило швидко створювати адаптивний дизайн без зайвих CSS-файлів.

− реалізовано плавну навігацію між розділами за допомогою хуків useRef і scrollIntoView(), що покращило користувацький досвід.

2. Адаптивність та оптимізація:

− Додаток коректно відображається на всіх пристроях (від мобільних до десктопних) завдяки медіа-запитам і гнучкому дизайну.

− Відеофони оптимізовані (формат .webm) для швидкого завантаження без втрати якості.

3. Тестування та стабільність:

− проведено функціональне тестування (кліки, скрол, відтворення відео);

− протестовано швидкодію (Lighthouse показники >90);

− перевірено кросс-браузерну сумісність (Chrome, Microsoft Edge).

4. Складності та їх подолання:

− проблема: Великий розмір відео сповільнював завантаження.

− рішення: Конвертація у .webm із стисненням без втрат якості.

5. Перспективи вдосконалення:

1. Розширення функціоналу:

− реалізувати динамічний контент через API (новини, оновлення).

2. Покращення інтерактивності:

− додати 3D-карти для туристичного розділу.

− впровадити анімації переходів між секціями.

3. Розвиток бекенду:

− підключити базу даних для коментарів користувачів.

− додати автентифікацію для адміністраторів.

Висновок до третього розділу

В розділі була розроблена покрокова структура web-сайту. Також були розроблені покрокові інструкції як для розробника так і для звичайного користувача.

Після розробки web-сайту “Безпілотні системи”, були проведені тестування на адаптивність та на швидкість завантаження. Результати тестування продемонстрували, що сайт цілком доступний до використання користувачами.

ВИСНОВКИ

У межах кваліфікаційної роботи було досліджено сучасні підходи до створення вебсайтів з використанням бібліотеки React.js.

Проведений огляд інструментів web-розробки дав змогу класифікувати наявні технології, розділивши їх за функціональними групами: frontend та backend середовища, а також інструменти для тестування й розгортання.

Особливу увагу було приділено екосистемі React — зокрема бібліотекам React Router, Redux, Tailwind CSS, Material UI, Framer Motion, які значною мірою впливають на якість, масштабованість і продуктивність вебсайтів.

А також було здійснено проєктування архітектури вебсайту, з урахуванням сучасних тенденцій у дизайні інтерфейсів та структуруванні контенту. При створенні сайту обрано односторінкову структуру (SPA), як найбільш придатну для даного типу інформаційного продукту.

Для реалізації вебсайту було вибрано сучасні технології, серед яких Node.js, TypeScript, Tailwind CSS, Figma, а також Jest для тестування.

Результатом виконаної роботи стало реалізація повноцінного інформаційного сервісу “Безпілотні системи”, який ілюструє потенціал безпілотних технологій у ключових сферах життя.

У інформаційному сервісі реалізовано ключові елементи сучасного вебпродукту: адаптивна навігація, інтерактивні секції, легка система переходів між блоками.

Проведено тестування — функціональне, продуктивності, адаптаційне — що дозволило переконатись у стабільній роботі сервісу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Stack Overflow Developer Survey 2023. Stack Overflow URL:https://survey.stackoverflow.co/2023/ (дата звернення:01.05.2025)

2. MarketsandMarkets Research Team. MarketsandMarkets. "Drone Market by Type". 2023. URL: https://www.marketsandmarkets.com/Market-Reports/drone-market-60027576.html. (дата звернення:11.06.2025)

3. You, E. (2015, Nov 3). Vue vs. React in terms of Performance. Medium. URL: https://medium.com/@youyuxi/re-performance-261023557027. (дата звернення:11.06.2025)

4. MDN Web Docs – офіційна документація з веб-технологій (HTML5,CSS3, JavaScript, PWA). URL: https://developer.mozilla.org. (дата звернення:11.06.2025)

5. React Official Documentation. URL: https://react.dev. (дата звернення:11.06.2025)

6. State of JS 2023 – огляди сучасних веб-технологій. URL: https://stateofjs.com. (дата звернення:11.06.2025)

7. AWS, Google Cloud, Azure – хмарні платформи для веб-розробки. URL: https://aws.amazon.com/, https://cloud.google.com/, https://azure.microsoft.com. (дата звернення:11.06.2025)

8. Project Management Institute. A Guide to the Project Management Bodyof Knowledge (PMBOK Guide), 7th Edition. 2021 URL: https://www.pmi.org/pmbok-guide-standards. . (дата звернення:11.06.2025)

9. International Organization for Standardization. ISO 21500:2021 Guidance on project management. 2021 URL: https://www.iso.org/standard/72403.html. . (дата звернення:11.06.2025)

10. Schwaber Ken; Sutherland Jeff. The Scrum Guide: The Definitive Guide to Scrum. 2020. URL: https://scrumguides.org/scrum-guide.html. (дата звернення:11.06.2025)

11. Google Developers. Single-Page Applications URL: https://developers.google.com/web/updates/2018/05/single-page-applications. . (дата

звернення:11.06.2025)

12. Mozilla MDN. Multi-page applications. URL: https://developer.mozilla.org/en-US/docs/Learn/Common_questions/Pages_sites_servers_and_search_engines. (дата звернення:11.06.2025)

13. Nielsen Norman Group. Information Architecture. URL: https://www.nngroup.com/articles/information-architecture/. (дата звернення:11.06.2025)

14. Figma Inc. Figma Design Tool. 2023. URL: https://www.figma.com. (дата звернення:11.06.2025)

15. Storybook.js Contributors. Storybook Documentation. 2023. URL: https://storybook.js.org. (дата звернення:11.06.2025)

16. Adam Wathan, Jonathan Reinink. Tailwind CSS Documentation. 2023. URL: https://tailwindcss.com. (дата звернення:11.06.2025)

17. Material-UI Team. Material-UI Library. 2023. URL: https://mui.com. (дата звернення:11.06.2025)

18. Framer Team. Framer Motion Documentation. 2023 URL: https://www.framer.com/motion. (дата звернення:11.06.2025)

19. ESLint Team. ESLint - Pluggable JavaScript Linter. 2023. URL: https://eslint.org. (дата звернення:11.06.2025)

20. Git Developers. Git - Distributed Version Control System. 2023. URL: https://git-scm.com. (дата звернення:11.06.2025)

21. Redux Team. Redux Toolkit - Official Redux Setup. 2023. URL: https://redux-toolkit.js.org. (дата звернення:11.06.2025)

22. Tanner Linsley. React Query - Hooks for Fetching Data. 2023.URL: https://tanstack.com/query. (дата звернення:11.06.2025)

23. Evan You. Vite - Next Generation Frontend Tooling. 2023. URL: https://vitejs.dev. (дата звернення:11.06.2025)

24. Meta Open Source. Jest - Delightful JavaScript Testing. 2023. URL: https://jestjs.io. (дата звернення:11.06.2025)

25. Cypress.io Team. Cypress - Fast, Easy and Reliable Testing. 2023.URL: https://www.cypress.io. (дата звернення:11.06.2025)

26. Testing Library Team. React Testing Library - Simple and CompleteTesting Utilities. 2023. URL: https://testing-library.com/docs/react-testing-library/intro. (дата звернення:11.06.2025)

27. ISTQB. ISTQB Foundation Level Syllabus 2023 URL: https://www.istqb.org. . (дата звернення:11.06.2025)

28. Google. Lighthouse Documentation URL: https://developers.google.com/web/tools/lighthouse. . (дата звернення:11.06.2025)

29. Patton, R. Software Testing, 2nd Edition. (2020). URL: https://www.pearson.com. (дата звернення:11.06.2025)

ДОДАТОК А – ПОСИЛАННЯ НА ПЕРВИННИЙ КОД ДОДАТКА

Первинний код розробленого програмного засобу: https://github.com/SkyNaris/Diplom/tree/main