МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ВОДНОГО ГОСПОДАРСТВА ТА ПРИРОДОКОРИСТУВАННЯ

Навчально-науковий інститут кібернетики, інформаційних технологій та інженерії

Кафедра комп'ютерних наук та прикладної математики

"*До захисту допущена*"

Зав. кафедри комп'ютерних наук та прикладної математики

д.т н., проф. Ю.В. Турбал

«____» _____ 2025 р.

КВАЛІФІКАЦІЙНА РОБОТА

Розробка вебдодатку з надання послуг будівельною компанією з використанням фреймворку React

Виконав: Циганчук Максим Сергійович

група ІПЗ-41

Керівник: к.т.н., доцент, доцент Остапчук О. П

Рівне - 2025

ЗМІСТ

РЕФЕРАТ

Кваліфікаційна робота: 41 с., 11 рисунків, 26 джерел

Мета роботи: розробка функціонального вебдодатку для надання послуг будівельною компанією із застосуванням фреймворку React, що забезпечить зручний інтерфейс для користувачів та ефективне управління замовленнями.

Об'єкт дослідження - процес розробки вебдодатків для автоматизації надання послуг будівельними компаніями на основі сучасних JavaScript-фреймворків.

Предмет дослідження - технологічні та методологічні підходи до створення інтерактивних вебінтерфейсів із використанням React, зокрема організація компонентної структури, управління станом, оптимізація продуктивності та інтеграція із зовнішніми сервісами.

Методи розробки базуються на технології React, сучасному JavaScript-фреймворку, який забезпечує створення швидких, масштабованих та інтерактивних вебдодатків.

Побудовано інформаційну модель для автоматизації процесів надання послуг будівельною компанією та спроєктовано відповідний вебдодаток. Використовуючи фреймворк React, здійснено програмну реалізацію розробленої інформаційної системи із сучасним, адаптивним інтерфейсом. Проведено серію експериментів щодо функціональності, продуктивності та зручності використання вебдодатку, а також виконано детальний аналіз отриманих результатів.

Ключові слова: ВЕБДОДАТОК, REACT, NEXT.JS, СЕРВЕРНИЙ РЕНДЕРИНГ, АДАПТИВНИЙ ІНТЕРФЕЙС, ІНТЕРАКТИВНІСТЬ, FRONTEND-РОЗРОБКА.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

DOM - Document Object Model (структура, яка представляє HTML- або XML-документ у вигляді дерева об’єктів)

JSX - JavaScript XML (розширення синтаксису JavaScript, що дозволяє писати елементи інтерфейсу у вигляді XML-подібного коду у React)

SPA - Single Page Application (односторінковий додаток)

SR - Server Side Rendering (серверний рендеринг)

UI - User Interface (користувацький інтерфейс)

UX - User Experience (користувацький досвід)

ВСТУП

Розвиток інформаційних технологій суттєво впливає на всі сфери бізнесу, зокрема на будівельну галузь. Вебдодатки стають основним інструментом для автоматизації процесів надання послуг, підвищення ефективності взаємодії з клієнтами та оптимізації внутрішніх операцій будівельних компаній. Використання сучасних фреймворків, таких як React, дозволяє створювати швидкі, масштабовані та зручні у використанні вебдодатки, що відповідають вимогам ринку та користувачів.

Актуальність теми зумовлена зростаючим попитом на сучасні вебтехнології, що забезпечують ефективну взаємодію з клієнтами та оптимізацію бізнес-процесів. В умовах цифровізації саме React дозволяє створювати швидкі, масштабовані та зручні у використанні вебдодатки, які відповідають сучасним вимогам до продуктивності, безпеки та користувацького досвіду

Метою даної кваліфікаційної роботи є розробка функціонального вебдодатку для надання послуг будівельною компанією із застосуванням фреймворку React, що забезпечить зручний інтерфейс для користувачів та ефективне управління замовленнями.

Для досягнення поставленої мети необхідно виконати такі завдання:

1. Провести аналітичний огляд наукових джерел та сучасних практик застосування фреймворку React у розробці вебдодатків для будівельної сфери.
2. Дослідити стан розвитку технологій React та їх застосування у будівельних вебдодатках.
3. Проаналізувати потреби цільової аудиторії та сформулювати основні вимоги до функціоналу вебдодатку.
4. Визначити функціональні вимоги до програмного продукту.
5. Розробити архітектуру системи, що базується на компонентному підході React.
6. Спроектувати інтерфейс користувача, який забезпечить інтуїтивну взаємодію та адаптивність для різних пристроїв.

7. Спроектувати загальну структуру проєкту.

8. Описати та реалізувати основні компоненти вебдодатку, забезпечивши їх функціональність та взаємодію.

Об’єктом дослідження є процес розробки вебдодатків для автоматизації надання послуг будівельними компаніями на основі сучасних JavaScript-фреймворків.

Предметом дослідження є технологічні та методологічні підходи до створення інтерактивних вебінтерфейсів із використанням React, зокрема організація компонентної структури, управління станом, оптимізація продуктивності та інтеграція із зовнішніми сервісами.

Теоретична значущість полягає у систематизації знань про можливості React для створення сервісних вебдодатків у будівництві. Практична цінність роботи полягає у створенні прототипу вебдодатку, який може бути використаний будівельними компаніями для автоматизації процесів замовлення та управління послугами, що сприятиме підвищенню їх конкурентоспроможності.

У роботі використані методи аналізу літературних джерел, системного проектування, моделювання програмних систем, а також практичні методи розробки програмного забезпечення з використанням React.

Кваліфікаційної робота складається зі вступу, трьох розділів, висновків, списку використаних джерел та додатків. У першому розділі розглядаються теоретичні основи розробки вебдодатків та огляд технології React. Другий розділ присвячений аналізу вимог і проектуванню системи. У третьому розділі описано процес розробки та тестування вебдодатку.

Результати кваліфікаційної роботи апробовані шляхом участі у науковій конференції [26], де було представлено доповідь щодо особливостей розробки сучасних вебдодатків із використанням React та впровадження їх у сфері будівельних послуг.

Наукова новизна роботи - запропоновано комплексний підхід до розробки вебдодатку для будівельної компанії з використанням компонентної архітектури

React, що дозволяє ефективно інтегрувати різні сервіси, масштабувати функціонал і забезпечувати високу продуктивність системи.

Практична значущість - розроблений вебдодаток сприяє підвищенню цифрової зрілості будівельної компанії, автоматизації взаємодії з клієнтами та оптимізації внутрішніх процесів. Запропоновані рішення можуть бути адаптовані для інших сфер послуг, що розширює можливості їх практичного застосування.

РОЗДІЛ 1
ТЕОРЕТИЧНІ АСПЕКТИ ЗАСТОСУВАННЯ REACT У БУДІВЕЛЬНИХ ВЕБДОДАТКАХ

1.1. Аналітичний огляд джерел щодо застосування React у будівельних вебдодатках

Компонентна архітектура React є ключовим фактором для створення модульних інтерфейсів, що особливо актуально для складних систем будівельних компаній. Вона дозволяє розробникам швидко створювати та оновлювати окремі елементи без необхідності змінювати всю систему, що значно підвищує гнучкість і підтримуваність проєктів [1; 2]. З огляду на це, можна стверджувати, що використання компонентної архітектури у будівельних вебдодатках є оптимальним рішенням для забезпечення масштабованості та ефективності.

Віртуальний DOM у React суттєво покращує продуктивність додатків, що працюють з великими обсягами даних, такими як проектна документація чи каталоги матеріалів. Оптимізація процесу рендерингу зменшує навантаження на браузер і прискорює відображення інформації, що є критично важливим для користувачів будівельних платформ [2]. Це свідчить про те, що React здатен ефективно підтримувати складні бізнес-процеси у будівельній сфері.

Інтеграція React з бекенд-технологіями, зокрема Node.js і MongoDB, відкриває можливості для створення повноцінних full-stack рішень, де фронтенд взаємодіє з API для обробки запитів користувачів. Такий підхід спрощує розробку і забезпечує гнучкість у реалізації функціоналу, що є важливим для динамічних будівельних вебдодатків [1; 3].

Досвід практичного застосування React у будівельній галузі, зокрема дослідження Slashdev.io (2024), демонструє перспективність використання бібліотек для 3D-візуалізації (наприклад, Three.js). Це дозволяє створювати інтерактивні моделі будівель, що значно покращує презентацію проектів та взаємодію з клієнтами [3]. Крім того, реалізація інтерактивних планів поверхів із

відображенням змін у реальному часі підкреслює можливості React у створенні динамічних інтерфейсів для будівельних проєктів.

Аналіз досвіду компанії GetTrusted свідчить про те, що перехід на React дозволив зменшити час завантаження сторінок на 50% завдяки серверному рендерингу (SSR), підвищити мобільний трафік на 60% через адаптивність інтерфейсів і покращити SEO-показники, що є важливим для будівельних компаній, які залежать від онлайн-запитів [2]. Це підтверджує ефективність React у підвищенні якості користувацького досвіду та маркетингової видимості.

Наукові джерела, зокрема IJRPR (2023), виділяють такі переваги React, як спрощення створення динамічних інтерфейсів через JSX, зменшення ризику помилок завдяки односпрямованому потоку даних, а також можливість інтеграції додаткових функцій через екосистему бібліотек (Redux, GraphQL) [1; 4]. Отож, React є потужним інструментом для розробки складних будівельних вебдодатків із багатофункціональним інтерфейсом.

На основі аналізу джерел можна зробити висновки, що React є оптимальним вибором для будівельних вебдодатків завдяки своїй здатності ефективно обробляти складну логіку та великі обсяги даних. Особливо перспективним напрямом є використання 3D-візуалізації для архітектурних компаній, що потребують інтерактивних презентацій проектів. Подальші дослідження доцільно спрямувати на інтеграцію штучного інтелекту для автоматизації кошторисів та застосування AR/VR технологій для віддаленого огляду будівельних майданчиків, що відкриває нові можливості для інновацій у будівельній галузі.

1.2. Аналіз класифікацій та підходів на тему роботи

React є надзвичайно ефективним інструментом для створення швидких, масштабованих і інтерактивних вебдодатків, що має важливе значення для бізнесів, які прагнуть забезпечити високий рівень користувацького досвіду. Аналізуючи особливості React, можна стверджувати, що розподіл інтерфейсу на прості функціональні та складні класові компоненти сприяє оптимізації процесів

розробки й підтримки коду, що підвищує гнучкість і якість програмних продуктів [5].

Використання додаткових бібліотек і фреймворків, таких як Ant Design, Material UI та Semantic UI, значно розширює функціональні можливості React. Це дає змогу будівельним компаніям створювати привабливі та зручні інтерфейси з готовими компонентами, що економить час розробки і покращує якість кінцевого продукту. З огляду на специфіку будівельної галузі, де важлива швидкість і надійність взаємодії з користувачем, застосування таких інструментів є виправданим і перспективним [6].

З технічної точки зору, React базується на компонентній архітектурі, яка передбачає розбиття інтерфейсу на незалежні, повторно використовувані блоки. Цей підхід забезпечує модульність, масштабованість і полегшує підтримку коду, що є критично важливим для складних вебдодатків будівельних компаній із великою кількістю інтерактивних форм, каталогів послуг і планів проектів. Поділ компонентів на функціональні (з хуками) та класові дає гнучкість у виборі підходу залежно від складності задачі, що відповідає сучасним вимогам розробки [7; 8].

Щодо типів вебдодатків, найбільш поширеними є односторінкові додатки (SPA), які забезпечують швидку навігацію та динамічний інтерфейс без перезавантаження сторінок, що особливо важливо для сервісів замовлення будівельних послуг. Серверний рендеринг (SSR) за допомогою фреймворків типу Next.js сприяє покращенню SEO та швидкості завантаження, що є вагомим для будівельних компаній, які орієнтуються на залучення клієнтів через пошукові системи [7]. Наш аналіз підтверджує, що поєднання SPA з SSR може значно підвищити конкурентоспроможність вебресурсів у будівельній сфері.

Функціонально React дозволяє створювати інтерактивні каталоги та форми для вибору послуг і розрахунку кошторисів, що підвищує зручність користувачів. Інтеграція з 3D-бібліотеками, такими як Three.js, відкриває перспективи для візуалізації будівельних проектів у реальному часі, що є важливим для презентації складних архітектурних рішень. Адміністративні

панелі, які реалізуються з урахуванням складної логіки і масштабованості, дозволяють ефективно управляти замовленнями, користувачами та графіками робіт [7]. Ці функціональні можливості є ключовими для підвищення ефективності будівельних сервісів.

Інтеграція React з додатковими бібліотеками, такими як Redux, Material UI, Ant Design, а також застосування TypeScript, підвищує надійність коду і покращує UX/UI. Такий підхід дозволяє розширювати функціональність додатків і підвищувати якість користувацького досвіду, що є важливим для складних і динамічних систем [7].

Основні фактори, що впливають на застосування React у будівельних вебдодатках, включають продуктивність і швидкодію завдяки віртуальному DOM і оптимізації рендерингу, масштабованість через компонентну архітектуру, що дозволяє розширювати функціонал без значних змін у коді, а також потребу у високій інтерактивності для замовлення послуг, перегляду проектів і комунікації з клієнтами. Важливою є також SEO-оптимізація через серверний рендеринг. Водночас слід враховувати складність навчання і підтримки React, що може впливати на швидкість впровадження. Інтеграція з бекендом і сторонніми сервісами для обробки даних і бізнес-логіки є невід'ємною складовою успішних рішень [7].

Наукові підходи підкреслюють, що ефективність React у системах управління проектами зумовлена його компонентною архітектурою, високою продуктивністю завдяки віртуальному DOM, односпрямованим потоком даних та сучасними можливостями, такими як хуки і JSX. Ці характеристики забезпечують гнучкість, масштабованість і швидкість розробки, що є критично важливим для створення складних і динамічних систем управління проектами. Аналіз підтверджує, що React є надійним вибором для корпоративних систем, що потребують стабільності та масштабованості, як це демонструють приклади Facebook та Instagram .

Таким чином, React є оптимальним інструментом для будівельних вебдодатків. Його архітектурні та функціональні особливості створюють міцну

основу для подальших досліджень, зокрема у напрямках інтеграції штучного інтелекту для автоматизації кошторисів та застосування AR/VR технологій для віддаленого огляду будівельних майданчиків, що відкриває перспективи для інновацій у галузі.

1.3. Стан і розвиток досліджуваного об'єкта

React постійно розвивається, впроваджуючи нові функціональні можливості, такі як хуки, Suspense та серверний рендеринг, що робить його більш потужним і зручним для створення складних вебдодатків, зокрема у сфері будівельних сервісів. З огляду на це, можна стверджувати, що сучасний розвиток React сприяє підвищенню ефективності розробки та підтримки програмних продуктів у будівельній галузі. Зростання кількості готових UI-бібліотек і спеціалізованих інструментів значно спрощує процес розробки і підвищує якість кінцевого продукту . Поширення TypeScript та сучасних підходів до управління станом і асинхронними процесами підвищує надійність і підтримуваність кодової бази, що є критично важливим для довготривалих і масштабних проєктів [7; 16; 17].

Сучасні дослідження свідчать про зростаючу популярність React у різних галузях, зокрема в будівельних компаніях, що зумовлено численними перевагами цього інструменту . На нашу думку, стабільне лідерство React серед фронтенд-фреймворків, випередження Angular, Vue.js та інших, а також висока оцінка серед розробників, підтверджують його актуальність і перспективність [17].

Особливу увагу заслуговує компонентна архітектура React, що дозволяє розбивати інтерфейс на невеликі незалежні частини, що значно полегшує розробку, повторне використання коду та підтримку, забезпечуючи гнучкість у створенні складних систем . Використання віртуального DOM мінімізує зміни в реальному DOM, що підвищує продуктивність додатків, а ефективні алгоритми оновлення забезпечують оптимальне використання ресурсів. Саме ці

особливості роблять React ефективним інструментом для проєктів різного масштабу [17].

Активна підтримка спільноти та розвинена екосистема бібліотек і плагінів сприяють швидкому розвитку React і розширенню його функціональності . Це дозволяє використовувати React для розробки веб-застосунків будь-якого масштабу - від малих сайтів до великих корпоративних порталів, що дає змогу легко масштабувати проєкти та інтегрувати їх з іншими технологіями. Динамічні та плавні інтерфейси, які можна створювати за допомогою React, підвищують зацікавленість користувачів, що особливо важливо для будівельних компаній, які прагнуть забезпечити зручність і привабливість своїх веб-ресурсів [17].

Варто також відзначити, що React користується популярністю серед студентів і молодих розробників, що свідчить про його перспективність і затребуваність у майбутньому. Аналіз ринку праці підтверджує, що знання React є однією з найбільш затребуваних навичок серед розробників, що відкриває широкі можливості для працевлаштування [17].

Водночас React має певні виклики. Зокрема, складність навчання для новачків, особливо тих, хто не має досвіду з JavaScript, може уповільнювати впровадження технології . Крім того, швидкий розвиток екосистеми вимагає постійного оновлення знань і освоєння нових інструментів, що є важливим аспектом для команд розробників [17].

Перспективним напрямком подальших досліджень є розвиток інструментів і бібліотек, які спрощують розробку на React і підвищують продуктивність розробників. Подальша адаптація React до специфічних потреб будівельної галузі, зокрема автоматизація кошторисів і впровадження AR/VR технологій, має великий потенціал для підвищення ефективності галузевих процесів.

Таким чином, сучасний стан досліджуваного об’єкта підтверджує перспективність React для використання у будівельних компаніях, проте для повноцінної реалізації його потенціалу необхідні подальші дослідження та адаптація під конкретні галузеві потреби.

РОЗДІЛ 2
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПРОЄКТУВАННЯ АРХІТЕКТУРИ ПРОГРАМНОГО ПРОДУКТУ

2.1. Аналіз потреб і постановка задачі

У цьому пункті здійснюється аналітичне обґрунтування необхідності створення програмного забезпечення для будівельної компанії, що прагне підвищити ефективність комунікації з клієнтами шляхом автоматизації презентації послуг, демонстрації прикладів реалізованих проєктів, попереднього розрахунку вартості робіт та збору зворотного зв'язку. Аналіз сучасного ринку будівельних послуг показує, що високий рівень конкуренції змушує компанії впроваджувати інноваційні підходи до взаємодії з потенційними клієнтами. Традиційні методи презентації, такі як друковані каталоги або усні консультації, дедалі частіше виявляються недостатньо ефективними у контексті швидко мінливих потреб ринку та зростаючих очікувань споживачів. Клієнти сьогодні прагнуть отримати максимально повну інформацію про послуги, ознайомитися з портфоліо компанії та попередньо оцінити вартість робіт ще до особистої зустрічі з представниками організації.

Сучасний стан будівельного ринку України характеризується стійким зростанням, незважаючи на виклики воєнного часу та економічні труднощі. За даними Property Times, у 2024 році обсяг ринку зріс на 6 % порівняно з 2023 роком і склав близько 170 млрд грн (€3,9 млрд), при цьому найбільшим попитом користуються складські та торгові об'єкти, а також регіони Київської та Львівської областей, які мають вигідне розташування та значне населення [18; 19].

Водночас, за інформацією GMK Center, у 2022-2023 роках собівартість будівництва зросла на 53 %, що спричинило підвищення цін на первинну нерухомість, а ринок переорієнтувався на західні регіони країни, де відбувається активна відбудова інфраструктури [18].

Значну роль у розвитку галузі відіграє відновлення інфраструктурних об’єктів, зокрема доріг, мостів та соціальних споруд, що фінансується як державою, так і міжнародними донорами. Інноваційні технології, такі як BIM, 3D-друк і модульне будівництво, активно впроваджуються у великих проєктах, що дозволяє знизити собівартість та скоротити терміни реалізації. За даними Forbes, у 2023 році обсяг будівельних робіт зріс на 22,6 %, зокрема інфраструктурне будівництво збільшилося на 32,9 %, тоді як житлове будівництво зазнало скорочення. Це свідчить про зміщення пріоритетів у бік відновлення та розвитку критично важливої інфраструктури [20].

За інформацією BDO Україна, будівельна галузь у 2024 році стикається з викликами, такими як дефіцит кваліфікованої робочої сили та високі ціни на будівельні матеріали через глобальну інфляцію та проблеми з постачанням. Ці фактори впливають на вартість проєктів і строки їх завершення, що потребує від компаній впровадження ефективних інструментів управління та цифрових рішень [21].

Компанії, які забезпечують зручний онлайн-доступ до інформації про свої послуги, отримують значні конкурентні переваги. Це пов’язано з тим, що потенційні замовники можуть самостійно ознайомитися з пропозиціями, переглянути приклади виконаних робіт та отримати попередню оцінку вартості у зручний для них час, незалежно від робочого графіку компанії.

Серед основних вимог до майбутнього вебзастосунку ми виділили інтуїтивність, адаптивність, швидкодія, інтерактивність та зручність використання на різних типах пристроїв. Інтуїтивність інтерфейсу має забезпечувати легку орієнтацію користувача в структурі сайту без потреби у попередньому навчанні чи детальному вивченні інструкцій. Адаптивність є необхідною для коректного відображення контенту на екранах різних розмірів - від смартфонів до великих моніторів настільних комп’ютерів.

З огляду на поставлені цілі, завданням розробника є створення вебінтерфейсу, який ефективно представлятиме перелік будівельних послуг, демонструватиме портфоліо реалізованих проєктів, забезпечуватиме попередній

розрахунок вартості робіт відповідно до параметрів, введених користувачем, а також надаватиме можливість встановлення контакту із представниками компанії.

Детальний аналіз функціональних вимог дозволяє виокремити ключові секції вебзастосунку. Вітальна секція (HeroSection) має створювати позитивне перше враження та чітко інформувати про основну діяльність компанії, включаючи назву, стислий опис, візуальні елементи та заклик до дії. Секція проєктів (ProjectsSection) покликана демонструвати портфоліо у вигляді карток з фотографіями, описами та основними характеристиками реалізованих об'єктів. Секція послуг (ServicesSection) структуровано представлятиме весь спектр будівельних послуг із супроводом іконок, назв і коротких описів для зручності сприйняття. Калькулятор (CalculatorSection) є критично важливою складовою, що автоматизує процес попереднього розрахунку вартості, забезпечуючи інтерактивний вибір послуг, введення параметрів і динамічний підрахунок загальної суми. Секція контактів (ContactSection) сприятиме збору зворотного зв'язку та надаватиме можливість встановлення зв'язку з компанією через форму або прямі контактні дані.

Таким чином, створення такого вебзастосунку відповідає сучасним потребам будівельної компанії і відкриває перспективи для підвищення ефективності комунікації з клієнтами, що є необхідною передумовою для подальшого розвитку бізнесу.

2.2. Визначення функціональних вимог

З огляду на сформульовані вимоги до системи, а також сучасні тенденції веброзробки, було прийнято рішення про використання фреймворку Next.js версії 14 з архітектурою App Router. Цей фреймворк надає гнучкі можливості для реалізації серверного рендерингу, оптимізованої маршрутизації та високої продуктивності. Отже, Next.js 14 з App Router є оптимальним вибором завдяки інтуїтивній файловій маршрутизації, де структура директорій автоматично

відповідає URL-адресам сайту, що значно спрощує організацію проєкту та робить його більш зрозумілим для розробки та підтримки [22; 23].

Фреймворк також пропонує вбудовані можливості для оптимізації продуктивності, включаючи автоматичне розділення коду, оптимізацію зображень та підтримку серверного рендерингу, що є критично важливим для забезпечення швидкої роботи вебзастосунку6. Архітектура App Router дозволяє чітко структурувати проект за принципом модульності, де кожна сторінка та компонент мають своє чітко визначене місце у файловій системі, що особливо важливо для проєктів із багатьма секціями та компонентами, як у випадку з веб-сайтом будівельної компанії [22; 23; 24].

Для розробки логіки інтерфейсу обрано мову TypeScript, яка забезпечує статичну типізацію та полегшує виявлення помилок ще на етапі компіляції. Це дозволяє гарантувати коректність передачі даних між компонентами та запобігає runtime-помилкам, що є важливим при роботі з інтерактивними елементами, такими як калькулятор вартості6. Конфігурація TypeScript через tsconfig.json забезпечує консистентність коду по всьому проєкту та полегшує розробку завдяки інтелектуальному автодоповненню та навігації в IDE [24].

У ролі засобу стилізації використовується Tailwind CSS - утилітарна CSS-бібліотека, що дозволяє безпосередньо в JSX-розмітці визначати стилі, мінімізуючи потребу у зовнішніх CSS-файлах. Такий підхід значно прискорює процес розробки інтерфейсу та забезпечує консистентність дизайну. Конфігурація Tailwind через tailwind.config.ts дозволяє адаптувати палітру кольорів, анімації та інші параметри теми відповідно до фірмової ідентичності компанії, що сприяє створенню уніфікованого стилю сайту [24].

Для побудови адаптивних компонентів інтерфейсу додатково інтегрована бібліотека ShadCN/UI, яка надає набір готових стилізованих компонентів, що легко налаштовуються та добре інтегруються з Tailwind CSS. Це дозволяє ефективно створювати такі елементи, як картки, поля введення, чекбокси, кнопки, які використовуються в калькуляторі та формах зворотного зв’язку [24].

Управління залежностями та конфігурація проєкту здійснюється через package.json, де визначено основні бібліотеки та скрипти для розробки, збірки та запуску. Додаткові налаштування Next.js через next.config.ts забезпечують підтримку зовнішніх зображень, ESLint для перевірки коду та інтеграцію з TypeScript, що підвищує безпеку і якість коду [24; 25].

Таким чином, вибір Next.js 14 з App Router у поєднанні з TypeScript, Tailwind CSS та ShadCN/UI створює потужний і сучасний стек технологій, який відповідає вимогам масштабованості, продуктивності та зручності розробки, що є ключовими для успішної реалізації вебзастосунку будівельної компанії.

2.3. Архітектура системи

На етапі архітектурного проєктування вебзастосунку було обрано модель, що базується на принципах модульної структури та чіткому розділенні обов'язків між складовими системи. Наш підхід до організації логіки взаємодії полягає у застосуванні компонентного підходу, де кожен функціональний елемент реалізується як окремий компонент. Така модель дозволяє забезпечити гнучкість, масштабованість і простоту підтримки проєкту.

Загальна структура проєкту організована навколо кореневої директорії src/, що містить чітко структуровані піддиректорії відповідно до їх функціонального призначення. Директорія app/ відповідає за основну логіку маршрутизації та рендерингу сторінок. Головна сторінка реалізована у файлі app/page.tsx, який виконує роль точки входу для кореневого маршруту сайту, забезпечуючи композицію основних секцій та їх коректне відображення. Важливим елементом є піддиректорія actions/, що містить серверні дії для обробки форм та інших операцій, що дозволяє ізолювати серверну логіку від клієнтської частини та підтримує принцип розділення відповідальності.

Директорія components/ організована за ієрархічним принципом із урахуванням функціонального призначення компонентів. Компоненти загальної структури сторінки, такі як Header та Footer, розміщені у components/layout/, що забезпечує єдине оформлення і консистентність інтерфейсу по всьому

застосунку. Основні функціональні секції головної сторінки, серед яких HeroSection, ProjectsSection, ServicesSection, CalculatorSection та ContactSection, розміщені у components/sections/. Важливою складовою є components/ui/, де зосереджені дрібніші елементи інтерфейсу з бібліотеки ShadCN/UI, які використовуються як базові будівельні блоки для більших функціональних секцій.

Директорія lib/ призначена для зберігання утилітарних функцій, які не мають безпосереднього відношення до візуального інтерфейсу - це функції обробки даних, конвертації значень, форматування валют та інші допоміжні обчислення. Такий поділ сприяє підтримці чистоти коду та полегшує його тестування.

Власна позиція полягає в тому, що обрана архітектурна організація забезпечує високий рівень модульності, де кожен компонент має чітко визначену відповідальність і може розроблятися, тестуватися або змінюватися незалежно від інших частин системи. Це відповідає принципу єдиної відповідальності (Single Responsibility Principle), що є фундаментальним для масштабованих і підтримуваних проєктів. Компонентна архітектура також сприяє повторному використанню коду, особливо UI-компонентів, що зменшує дублювання і забезпечує єдність інтерфейсу, що є критично важливим для підтримки високої якості користувацького досвіду.

Інформація про архітектуру вебзастосунку базується на принципах компонентного підходу та модульної структури, які детально описані в офіційній документації Next.js. Зокрема, використання App Router у Next.js 14 забезпечує інтуїтивну файлову маршрутизацію та розділення логіки між серверними та клієнтськими компонентами, що значно спрощує організацію проєкту . Практичні рекомендації щодо структурування директорій і застосування принципу єдиної відповідальності (Single Responsibility Principle) підтверджують ефективність такого підходу для масштабованих вебзастосунків . Крім того, інтеграція TypeScript і використання утилітарної CSS-бібліотеки Tailwind CSS, а

також UI-компонентів із бібліотеки ShadCN/UI, відповідають сучасним кращим практикам розробки і підвищують якість коду та інтерфейсу.

Таким чином, запропонована архітектура створює надійну основу для подальшої розробки і впровадження функціональних можливостей вебзастосунку, а також формує передумови для проведення власних досліджень щодо оптимізації структури і підвищення продуктивності системи.

2.4. Проєктування інтерфейсу користувача

Особливе значення у функціональній структурі вебзастосунку має проєктування взаємодії користувача з різними секціями сайту, особливо з інтерактивними елементами, такими як калькулятор вартості та форма зворотного зв'язку. Наш підхід до розробки інтерфейсу базується на створенні інтуїтивно зрозумілих і зручних для користувача компонентів, що сприяють підвищенню залученості та ефективності взаємодії.

Проєктування HeroSection спрямоване на формування позитивного першого враження. Ця секція включає заголовок компанії, стислий опис діяльності та візуальні елементи, які миттєво передають суть бізнесу. Особливу увагу приділено call-to-action кнопкам, які спрямовують користувача до ключових дій на сайті, що, на нашу думку, є важливим для підвищення конверсії та зручності навігації.

Логіка секції ProjectsSection передбачає відображення портфоліо у вигляді карток (Card компонентів), кожна з яких представляє окремий проєкт. Дані про проєкти можуть бути як статично закодовані, так і динамічно завантажуватися з зовнішніх джерел, що забезпечує гнучкість і масштабованість. Секція адаптована для комфортного перегляду на різних пристроях, що відповідає сучасним вимогам до UX.

Архітектура ServicesSection базується на гнучкій сітковій системі з використанням Tailwind CSS, що дозволяє ефективно відображати перелік послуг у вигляді окремих блоків з іконкою, заголовком та описом. Такий підхід

забезпечує адаптивність макета під різні розміри екранів і підтримує єдність стилю.

Центральним інтерактивним елементом є CalculatorSection з компонентом CostCalculator. Логіка цього компонента побудована на використанні React hooks: useState для управління станом вибраних послуг і їх кількостей, useMemo для оптимізації перерахунку вартості лише при зміні релевантних даних. Інтерактивність забезпечується чекбоксами для вибору послуг та полями введення кількості, а форматування результатів реалізовано через функцію formatCurrency, що відображає суму у зручному для українського користувача форматі з позначенням гривні. Такий підхід, на нашу думку, забезпечує високу продуктивність і зручність користування калькулятором.

Проєктування ContactSection передбачає збір якісних контактних даних через форму з базовою валідацією полів і можливість додатково розміщувати прямі контактні дані компанії. Інтеграція з серверними діями, розміщеними у директорії app/actions/, дозволяє безпечно обробляти форми на серверній стороні без перезавантаження сторінки, що підвищує зручність і безпеку взаємодії.

2.5. Структура даних та моделі

Структурна модель системи забезпечує чітку організацію зв'язків між компонентами, зберігаючи при цьому гнучкість для розширення функціональності. Кожен модуль взаємодіє з іншими виключно через визначені інтерфейси, що запобігає надмірним залежностям і спрощує підтримку системи.

Композиція головної сторінки реалізується у файлі app/page.tsx, який виконує роль оркестратора секцій, послідовно імпортуючи та рендерячи HeroSection, ProjectsSection, ServicesSection, CalculatorSection і ContactSection. Така послідовність створює логічний потік користувацького досвіду - від ознайомлення з компанією до детального ознайомлення з послугами, розрахунку вартості та встановлення контакту.

Передача даних між компонентами базується на React props та callback функціях, що забезпечує односпрямований потік даних і передбачуваність

поведінки системи. Це, на мою думку, є ключовим для підтримки стабільності і зрозумілості коду.

Повторне використання компонентів реалізується завдяки бібліотеці ShadCN/UI, яка надає універсальні елементи інтерфейсу (Card, Input, Button, Checkbox), що застосовуються в різних секціях, зокрема у ProjectsSection і ServicesSection. Такий підхід значно знижує дублювання коду і сприяє підтримці консистентності дизайну.

Утилітарні функції, розміщені у директорії lib/, забезпечують загальну функціональність, спільну для різних компонентів. Зокрема, функція formatCurrency використовується для форматування грошових сум у калькуляторі, а валідаційні функції - у формах. Це дозволяє централізовано керувати логікою і підвищує якість коду.

Інтеграція з серверними діями через app/actions/ забезпечує безшовну взаємодію компонентів форм із серверною логікою, що підвищує безпеку і зручність користування.

Стилізація та тематизація реалізовані на основі єдиної конфігурації Tailwind CSS, що гарантує візуальну цілісність і узгодженість між усіма компонентами. Використання responsive утиліт Tailwind CSS забезпечує коректне відображення на різних пристроях без необхідності додаткових медіа-запитів.

Таким чином, запропоновані проєктні рішення створюють міцну основу для реалізації сучасного вебдодатку, що відповідає вимогам адаптивності, інтерактивності та продуктивності. Модульна архітектура і чіткий розподіл відповідальності між компонентами формують передумови для подальших власних досліджень, спрямованих на оптимізацію структури та підвищення ефективності системи.

РОЗДІЛ 3
ПРОЄКТУВАННЯ ПРОГРАМНОГО ПРОДУКТУ

У цьому розділі детально розглянуто структуру реалізованого вебдодатку (рис. 3.1), його основні технічні файли, логіку компонування та розподіл відповідальності між модулями. Проєкт реалізовано за допомогою сучасного фреймворку Next.js, використовуючи App Router та інші сучасні підходи до розробки (Tailwind CSS, серверні дії, UI бібліотека ShadCN/UI тощо), що повністю відповідає архітектурним рішенням, визначеним у попередньому розділі.

3.1. Загальна структура проєкту

Уся кодова база розташована у папці src/, що містить головні функціональні частини вебзастосунку відповідно до архітектурної моделі, розробленої на етапі проєктування:

- app/ - маршрути, компоненти сторінок і макетів, що реалізують архітектуру App Router;
- components/ - перевикористовувані елементи інтерфейсу, організовані за модульним принципом;
- lib/ - утилітарні функції, що забезпечують загальну функціональність системи;
- app/actions/ - серверні дії для обробки форм та серверних операцій.

Файл package.json визначає залежності проєкту, доступні скрипти та основну інформацію про застосунок, відображаючи всі технологічні рішення, прийняті на етапі аналізу. Файл next.config.ts використовується для конфігурації Next.js відповідно до специфічних потреб проєкту, а tailwind.config.ts - для налаштувань теми Tailwind CSS згідно з фірмовою ідентичністю компанії.

Така організаційна структура забезпечує реалізацію принципу єдиної відповідальності (Single Responsibility Principle), визначеного в архітектурному

проєктуванні, де кожна директорія має чітко визначене функціональне призначення та область відповідальності.

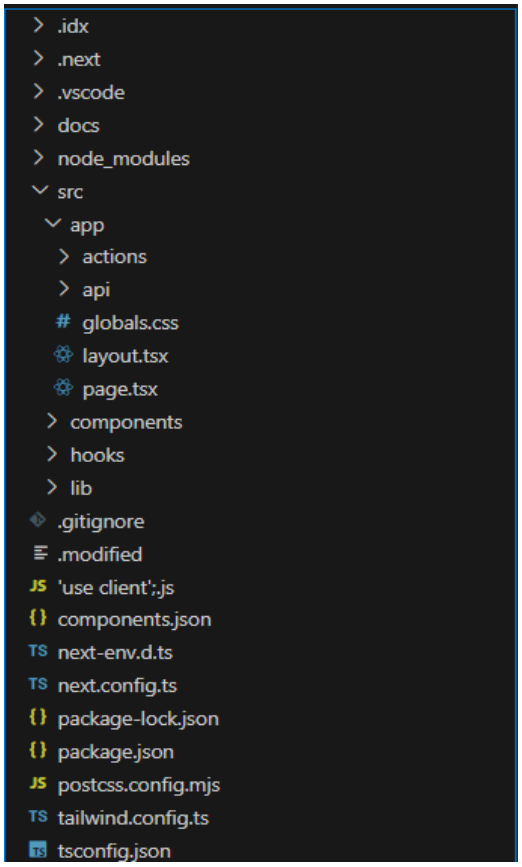

Рис. 3.1. Структура проекту у файловому дереві

3.2. Основні конфігураційні файли

package.json

У файлі package.json зберігається основна інформація про застосунок: його назва, версія, а також список зовнішніх бібліотек, що використовуються в проєкті. Одними з головних залежностей є next, react та react-dom, які забезпечують базову функціональність фреймворку та реактивного інтерфейсу відповідно до технологічного стеку, обраного на етапі аналізу предметної області.

Конфігурація залежностей включає також TypeScript для статичної типізації, Tailwind CSS для утилітарної стилізації, ShadCN/UI для готових компонентів інтерфейсу, а також додаткові бібліотеки для оптимізації роботи з формами, валідацією та іншими функціональними аспектами системи. Кожна

залежність була ретельно обрана відповідно до функціональних вимог, сформульованих у попередньому розділі.

tsconfig.json

Типізація коду у проєкті забезпечується завдяки використанню мови TypeScript, що реалізує стратегію статичної типізації, визначену в архітектурному проєктуванні. У конфігураційному файлі tsconfig.json вказано параметри компіляції, шляхи до кореневих директорій, а також опції для перевірки типів. Це дозволяє виявляти помилки ще на етапі написання коду та покращує підтримуваність проєкту у довготривалій перспективі.

Конфігурація включає строгі параметри типізації, що забезпечують високу якість коду та запобігають runtime-помилкам, особливо критичним для інтерактивних компонентів, таких як калькулятор вартості. Налаштування path mapping дозволяють використовувати абсолютні імпорти, що спрощує навігацію по проєкту та підвищує читабельність коду.

tailwind.config.ts

Конфігураційний файл tailwind.config.ts використовується для адаптації Tailwind CSS до потреб проєкту відповідно до utility-first підходу, обраного на етапі вибору технологій. У ньому вказано, які файли скануються на наявність класів Tailwind, а також задано власну палітру кольорів, ступінь заокруглення кутів, анімації та інші параметри теми. Це дозволяє створити уніфікований стиль сайту, який відповідає фірмовій ідентичності будівельної компанії.

Кастомізація теми включає визначення кольорової схеми, що відображає професійність та надійність будівельних послуг, типографічних стилів для оптимальної читабельності контенту, а також адаптивних точок перелому для забезпечення коректного відображення на всіх типах пристроїв. Конфігурація також включає налаштування анімацій та переходів, що підвищують інтерактивність інтерфейсу.

next.config.ts

У файлі next.config.ts задаються параметри, які впливають на поведінку всього проєкту та реалізують оптимізаційні стратегії, закладені в архітектурному проєктуванні. Зокрема, у ньому дозволено використання зображень з зовнішніх джерел для інтеграції портфоліо проєктів, увімкнено перевірку коду за ESLint для підтримки якості коду, а також встановлено підтримку TypeScript для статичної типізації. Таке налаштування дозволяє інтегрувати зовнішні сервіси, зберігаючи при цьому контроль над якістю та безпекою коду.

Додаткові налаштування включають оптимізацію збірки для продуктивності, конфігурацію серверного рендерингу для покращення SEO та швидкості завантаження, а також налаштування безпеки для захисту від потенційних вразливостей при роботі з зовнішніми ресурсами.

3.3. Побудова сайту на базі Next.js

Розроблений вебдодаток побудований із використанням сучасного фреймворку Next.js, а саме його версії 14, що використовує архітектуру App Router відповідно до технологічних рішень, обґрунтованих у попередньому розділі. Цей підхід дозволяє організувати структуру проєкту максимально логічно та ефективно, використовуючи файлову маршрутизацію на основі директорії app (рис. 3.2). Саме тут розміщено головні файли, які відповідають за маршрути і рендеринг сторінок, реалізуючи принципи модульної архітектури.

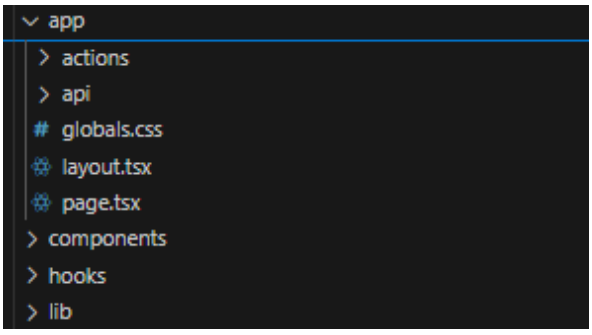

Рис. 3.2. Директорія app

Архітектура App Router забезпечує інтуїтивну організацію маршрутів, де структура директорій автоматично відповідає URL-адресам застосунку. Це спрощує навігацію по проєкту та робить його більш зрозумілим як для розробки, так і для подальшої підтримки. Використання серверних компонентів за замовчуванням оптимізує продуктивність завдяки серверному рендерингу, що особливо важливо для SEO та швидкості завантаження сторінок.

Головна сторінка сайту реалізована у файлі app/page.tsx (рис 3.3), що є точкою входу для кореневого маршруту відповідно до проєктованої архітектури. Саме цей файл викликається маршрутизатором Next.js у відповідь на запит до кореневої сторінки сайту та здійснює композицію всіх основних секцій. Всередині page.tsx відбувається рендеринг головного вмісту сторінки - викликаються основні компоненти, що формують її структуру відповідно до логічного flow користувацького досвіду, визначеного в проєктуванні.

До таких компонентів належать: секція привітання (Hero), блок із прикладами виконаних проєктів (Projects), секція з переліком послуг (Services), інтерактивний калькулятор вартості (Calculator) та форма для зворотного зв'язку (Contact). Кожен із цих компонентів є окремим, самодостатнім модулем, що відповідає лише за свою частину інтерфейсу і розташовується у папці components/sections, реалізуючи принцип єдиної відповідальності.

```tsx

import { HeroSection } from '@/components/sections/hero-section';
import { ProjectsSection } from '@/components/sections/projects-section';
import { ServicesSection } from '@/components/sections/services-section';
import { CalculatorSection } from '@/components/sections/calculator-section';
import { ContactSection } from '@/components/sections/contact-section';

export default function Home() {
  return (
    <>
      <HeroSection />
      <ProjectsSection />
      <ServicesSection />
      <CalculatorSection />
      <ContactSection />
    </>
  );
}

```

Рис. 3.3. Код файлу page.tsx

Загальна організація проєкту передбачає чіткий поділ усіх компонентів відповідно до їх функціонального призначення, що відображає ієрархічну структуру, розроблену на етапі архітектурного проєктування. Компоненти, які стосуються загального макета сторінки, такі як шапка або підвал сайту, винесені в директорію components/layout для забезпечення консистентності оформлення по всьому застосунку. Основні розділи сторінки - ті, які відображаються на головній, - згруповані в директорії components/sections (рис. 3.4), а дрібніші елементи, що використовуються як частини інших компонентів, зберігаються у components/ui.

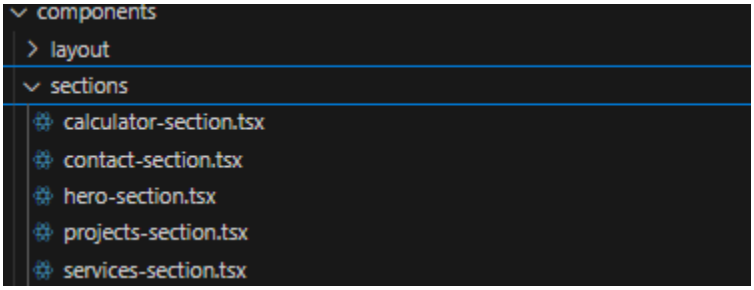

Рис. 3.4. Вигляд директорії components

Такий підхід дозволяє легко орієнтуватися у структурі проєкту та сприяє масштабованості, оскільки кожен елемент інтерфейсу може розроблятись, змінюватися або тестуватися окремо. Модульна організація також полегшує командну розробку, де різні розробники можуть працювати над окремими компонентами без конфліктів та взаємних залежностей.

Уся стилізація додатку реалізована за допомогою утилітарної бібліотеки Tailwind CSS, яка дозволяє застосовувати стилі безпосередньо в JSX-розмітці, використовуючи короткі класові імена відповідно до utility-first підходу. Такий підхід значно пришвидшує розробку інтерфейсу та полегшує підтримку стилів, усуваючи необхідність у великих CSS-файлах та потенційних конфліктах стилів. Окрім Tailwind, у проєкті також застосовується бібліотека ShadCN/UI, яка містить набір готових адаптивних і стилізованих компонентів інтерфейсу, що легко налаштовуються та добре інтегруються з Tailwind.

Завдяки комбінації Next.js, Tailwind CSS та модульної архітектури компонентів, сайт не лише виглядає сучасно, а й є технічно зручним у підтримці та розширенні, що є важливою вимогою до будь-якого сучасного вебрішення. Така технологічна база забезпечує високу продуктивність, SEO-оптимізацію та можливість легкого масштабування функціональності у майбутньому.

3.4. Опис компонентів

HeroSection

Компонент HeroSection розташований у директорії components/sections/hero-section.tsx (рис 3.5) і є першим візуальним блоком, що відображається на головній сторінці відповідно до проєктованого користувацького flow. Він виконує роль вітального банера сайту та створює перше враження для користувача, реалізуючи вимоги сформульовані в аналізі потреб. Візуально цей блок містить назву будівельної компанії, короткий опис її діяльності, привабливий фон або зображення, а також заклик до дії (CTA), що направляє користувачів до ключових функцій сайту.

Компонент написаний з використанням сучасного підходу до верстки із Tailwind CSS, що забезпечує утилітарний підхід до стилізації. Основна розмітка включає в себе заголовок першого рівня з назвою компанії, підзаголовок з описом діяльності, контейнер для кнопок call-to-action та обгортковий контейнер, у якому все це розміщено з урахуванням адаптивності. Код організовано таким чином, щоб компонування було адаптивним до різних розмірів екранів і зручним для редизайну у майбутньому без структурних змін.

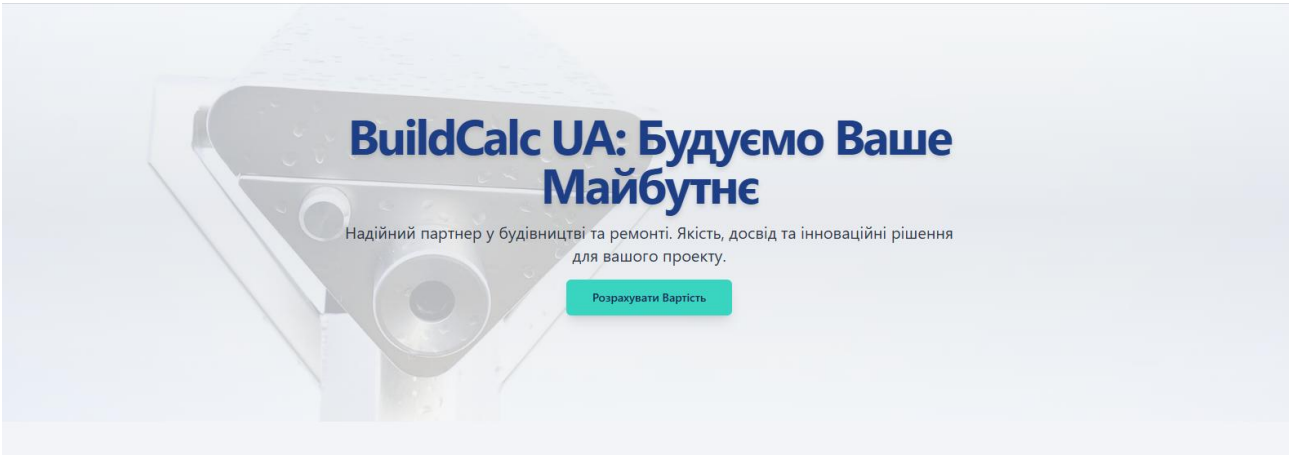

Рис. 3.5. Зовнішній вигляд hero-section

Стилізація компонента використовує градієнтні фони, сучасну типографіку та адаптивні відступи, що створюють професійний вигляд, відповідний будівельній тематиці. Інтерактивні елементи, такі як кнопки, мають hover-ефекти та плавні переходи, що підвищують користувацький досвід та створюють відчуття якісного, сучасного інтерфейсу.

ProjectsSection

Компонент ProjectsSection знаходиться у файлі components/sections/projects-section.tsx (рис 3.6) та відображає перелік реалізованих проєктів компанії, реалізуючи функціональну вимогу демонстрації портфоліо (рис 3.7). Цей компонент є критично важливим для будівельної компанії, оскільки дозволяє потенційним клієнтам оцінити якість та масштаб виконаних робіт, що впливає на прийняття рішення про співпрацю.

```
import Image from 'next/image';
import { Card, CardContent } from '@/components/ui/card';
import {
  Carousel,
  CarouselContent,
  CarouselItem,
  CarouselNext,
  CarouselPrevious,
} from '@/components/ui/carousel'; // Ensure carousel is installed via shadcn/ui

const projects = [
  { id: 1, title: 'Сучасний Житловий Комплекс', description: 'Будівництво багатоквартирного будинку з нуля.', image: 'https://picsum.photos/600/400?random=1' },
  { id: 2, title: 'Реконструкція Офісної Будівлі', description: 'Повна модернізація фасаду та інтер\'єрів.', image: 'https://picsum.photos/600/400?random=2' },
  { id: 3, title: 'Приватний Будинок "Мрія"', description: 'Індивідуальний проект котеджу.', image: 'https://picsum.photos/600/400?random=3' },
  { id: 4, title: 'Торговий Центр "Галерея"', description: 'Завершення великого торгового комплексу.', image: 'https://picsum.photos/600/400?random=4' },
  { id: 5, title: 'Ремонт Квартири "Лофт"', description: 'Дизайнерський ремонт у стилі лофт.', image: 'https://picsum.photos/600/400?random=5' },
];
```

Рис. 3.6. Фрагмент коду projects-section

Компонент може отримувати дані про проєкти у вигляді масиву об'єктів з зовнішніх джерел або використовувати статичні дані, захардкожені у коді для демонстраційних цілей. Кожен проєкт представлений у вигляді картки (Card компонента з ShadCN/UI), що містить зображення, назву проєкту, короткий опис, основні характеристики (площа, тип будівництва, термін виконання) та посилання для детального перегляду.

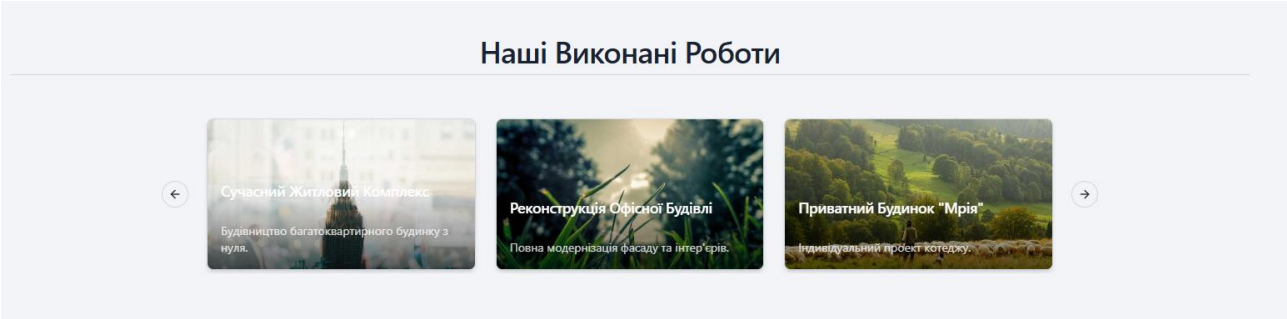

Рис. 3.7. Вигляд projects-section на сайті

Архітектура компонента передбачає гнучку сіткову систему, яка автоматично адаптується під різні розміри екранів: на великих екранах відображається 3-4 картки в ряд, на планшетах - 2, на мобільних пристроях - по одній. Кожна картка є самостійною візуальною сутністю з консистентним оформленням, що забезпечує професійний вигляд портфоліо та легкість сприйняття інформації користувачами.

ServicesSection

Компонент ServicesSection, що зберігається у файлі components/sections/services-section.tsx, демонструє список послуг, які надає будівельна компанія, реалізуючи функціональну вимогу структурованого представлення спектру послуг (рис. 3.8). Кожна послуга відображається у вигляді блоку з тематичною іконкою, заголовком та детальним описом, що дозволяє користувачам швидко зорієнтуватися в пропозиціях компанії.

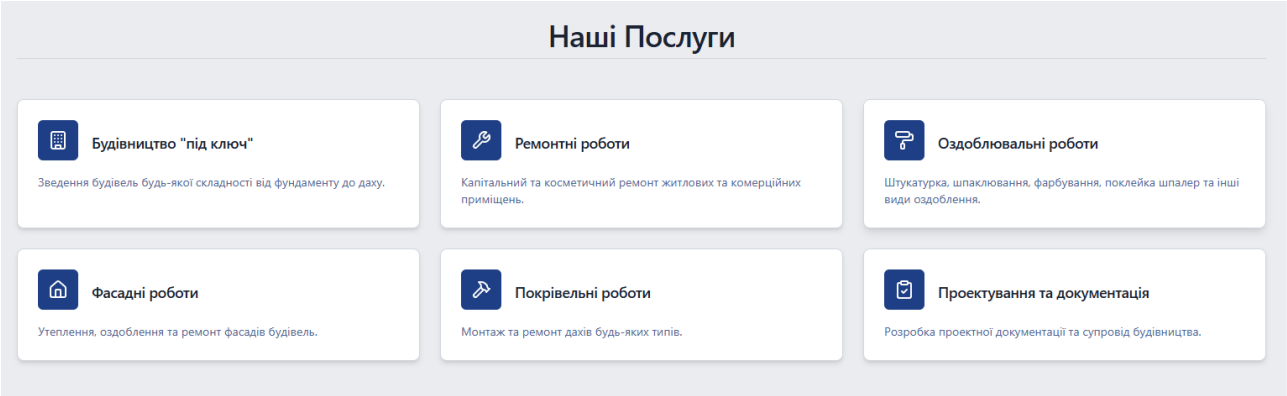

Рис. 3.8. Вигляд services-section на сайті

Компонент використовує гнучку сітку для розміщення елементів, що дозволяє зручно адаптувати вигляд секції під різні пристрої відповідно до адаптивних вимог. Весь контент розміщується у контейнері з розміткою, оптимізованою для користувацького досвіду та читабельності. Іконки послуг взяті з бібліотеки Lucide React, що забезпечує консистентний стиль та професійний вигляд.

Структура послуг включає основні категорії будівельних робіт: проєктування, фундаментні роботи, зведення стін, покрівельні роботи, внутрішнє оздоблення, ландшафтний дизайн та інші спеціалізовані послуги. Кожна послуга супроводжується інформацією про одиниці виміру та орієнтовну вартість, що підготовлює користувачів до роботи з калькулятором вартості.

CalculatorSection

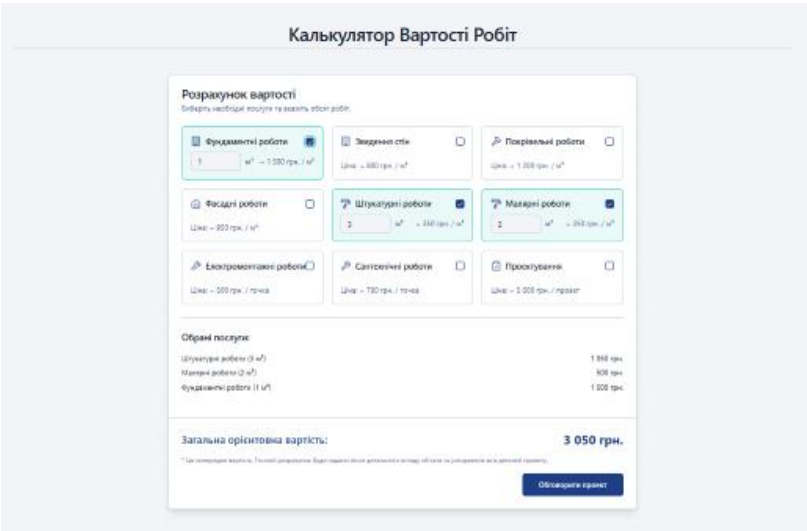

Рис. 3.9. Оформлення calculator-section

Компонент CalculatorSection є найбільш технічно складним та інтерактивним елементом сайту, що дозволяє користувачам виконати попередній розрахунок вартості послуг відповідно до критично важливої функціональної вимоги автоматизації розрахунків (рис. 3.9). Він розміщується у файлі components/sections/calculator-section.tsx та реалізує складну логіку взаємодії з користувачем.

Компонент використовує локальний стан React (через useState hook), щоб зберігати введені користувачем значення та проводити обчислення в режимі реального часу. Результат розрахунку оновлюється динамічно без перезавантаження сторінки, що забезпечує високу інтерактивність та зручність використання відповідно до вимог до швидкодії системи.

Основний функціонал побудований на компоненті CostCalculator, який є ключовим елементом та реалізує наступну архітектуру взаємодії:

Вибір послуг за допомогою чекбоксів - кожна будівельна послуга відображається з тематичною іконкою, зрозумілою назвою, одиницею виміру (м², м³, точок, проектів тощо) та ціною за одиницю в українських гривнях (рис 3.10). Чекбокси мають сучасний дизайн та інтуїтивно зрозумілі стани активації.

Динамічне введення кількості - при активації чекбокса автоматично з'являється поле введення для кількості з відповідною одиницею виміру, значення якого миттєво враховується в розрахунках. Поля мають валідацію для запобігання введенню некоректних значень та автоматичне форматування для зручності використання.

Обчислення загальної вартості - виконується через useMemo hook, що забезпечує ефективне перерахування вартості тільки при зміні обраних послуг або їх кількостей, оптимізуючи продуктивність та запобігаючи зайвим обчисленням.

Форматування ціни - за допомогою спеціальної функції formatCurrency з директорії lib/ сума виводиться у зручному для українського користувача

форматі з розділенням тисяч пробілами та позначенням "грн", що відповідає локальним стандартам.

Оформлення інтерфейсу - реалізовано через компоненти UI-бібліотеки ShadCN/UI (Card, Input, Checkbox, Button), що забезпечує стильний, адаптивний вигляд та консистентність з рештою сайту.

```
const availableServices = [
  { id: 'foundation', name: 'Фундаментні роботи', basePrice: 1500, unit: 'м³', icon: Building },
  { id: 'walls', name: 'Зведення стін', basePrice: 800, unit: 'м²', icon: Building },
  { id: 'roofing', name: 'Покрівельні роботи', basePrice: 1200, unit: 'м²', icon: Hammer },
  { id: 'facade', name: 'Фасадні роботи', basePrice: 950, unit: 'м²', icon: Home },
  { id: 'plastering', name: 'Штукатурні роботи', basePrice: 350, unit: 'м²', icon: PaintRoller },
  { id: 'painting', name: 'Малярні роботи', basePrice: 250, unit: 'м²', icon: PaintRoller },
  { id: 'electrical', name: 'Електромонтажні роботи', basePrice: 500, unit: 'точка', icon: Wrench },
  { id: 'plumbing', name: 'Сантехнічні роботи', basePrice: 700, unit: 'точка', icon: Wrench },
  { id: 'design', name: 'Проектування', basePrice: 5000, unit: 'проект', icon: ClipboardCheck },
];
```

Рис. 3.10. Фрагмент коду калькулятору

У нижній частині блоку відображається підсумкова вартість з великим, помітним шрифтом та пояснення, що розрахунок є попереднім і фінальна вартість може відрізнятися після детальної консультації. Також присутня кнопка для переходу до форми обговорення проєкту, що забезпечує плавний перехід до наступного етапу взаємодії з компанією.

Таким чином, CalculatorSection реалізує гнучкий та зручний інтерфейс взаємодії з користувачем, дозволяючи швидко зорієнтуватися в орієнтовній вартості будівельних послуг та приймати обґрунтовані рішення про співпрацю з компанією.

ContactSection

Компонент ContactSection, розміщений у файлі components/sections/contact-section.tsx, відповідає за виведення контактної інформації компанії, а також реалізацію форми зворотного зв'язку відповідно до функціональної вимоги забезпечення можливості встановлення контакту (рис. 3.11). Цей компонент є завершальним елементом користувацького шляху на сайті, призначеним для конвертації зацікавлених відвідувачів у потенційних клієнтів.

Форма зворотного зв'язку містить поля для введення імені користувача, електронної пошти для зворотного зв'язку, детального повідомлення з описом проєкту або запитання, а також кнопку «Надіслати» з привабливим дизайном. Всі поля мають відповідне placeholder-текст та labeling для покращення користувацького досвіду та доступності.

Форма має комплексну валідацію як на клієнтській, так і на серверній стороні: перевірку на заповнення обов'язкових полів, валідацію формату електронної пошти, обмеження довжини повідомлення та санітизацію введених даних для безпеки. Після натискання на кнопку дані можуть надсилатися на електронну пошту компанії або зберігатися у базі даних через серверні дії з директорії app/actions/, залежно від подальшої реалізації бек-енд функціональності.

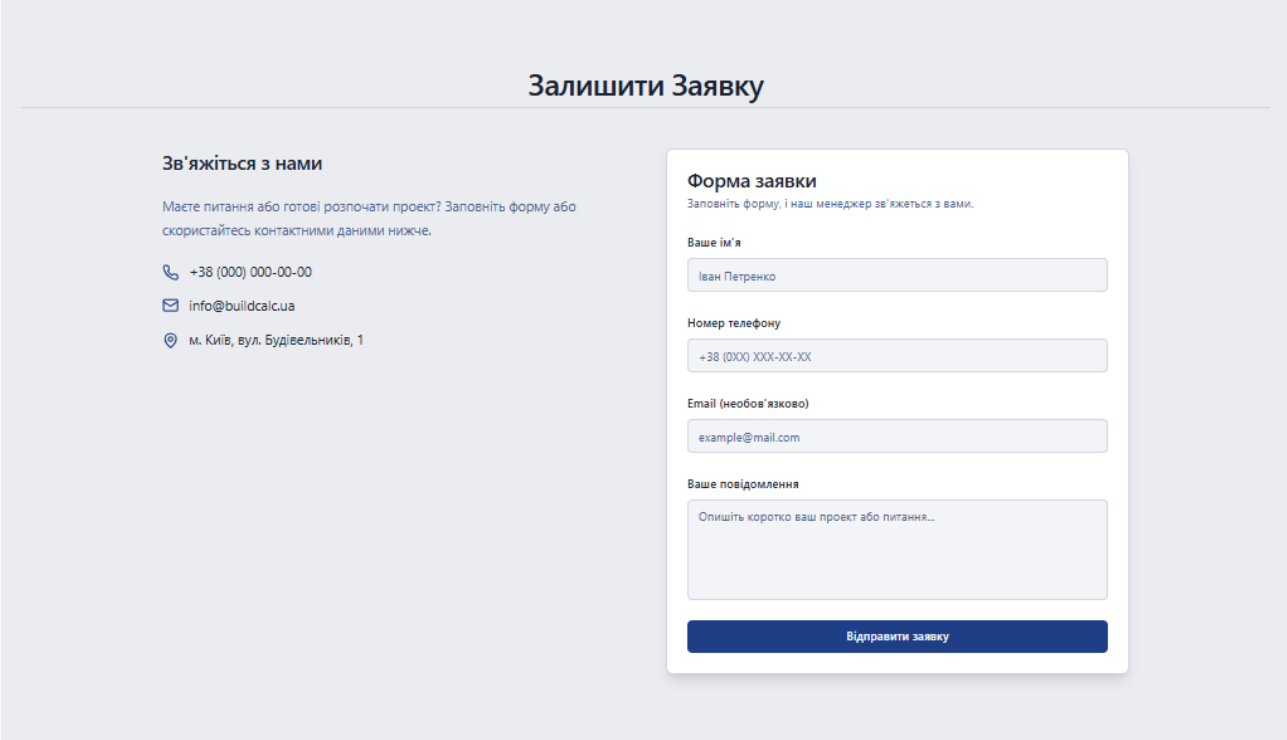

Рис. 3.11. Зовнішній вигляд компоненту contact-section

Окрім форми, секція містить блок з прямими контактними даними компанії: номер телефону з можливістю прямого виклику на мобільних пристроях, електронну адресу з mailto-посиланням, фізичну адресу офісу компанії та посилання на профілі у соціальних мережах (Facebook, Instagram,

LinkedIn). Контактна інформація оформлена з використанням іконок та структурована для легкого сприйняття.

Компонент також включає інтерактивну карту розташування офісу (за необхідності) та робочий графік компанії, що додатково полегшує контакт для потенційних клієнтів та демонструє професійний підхід до обслуговування.

Інтеграція компонентів

Усі описані компоненти формують єдину, логічно структуровану головну сторінку сайту, що поєднує в собі візуальну привабливість, функціональність та зручність для кінцевого користувача відповідно до архітектурних принципів, закладених у проєктуванні. Послідовність розташування компонентів забезпечує природний flow від знайомства з компанією до конкретного розрахунку вартості та встановлення контакту.

Взаємодія між компонентами здійснюється через загальні стилі Tailwind CSS, використання спільних UI-компонентів з ShadCN/UI та консистентну кольорову схему, що створює цілісний користувацький досвід. Кожен компонент оптимізований для швидкого завантаження та ефективного рендерингу, забезпечуючи високу продуктивність всього застосунку.

Така архітектура компонентів дозволяє легко вносити зміни в окремі секції без впливу на решту сайту, додавати нові функції або модифікувати існуючі відповідно до змінних потреб бізнесу або відгуків користувачів.

ВИСНОВКИ

У кваліфікаційній роботі виконано комплексне дослідження та розробку вебдодатку для надання послуг будівельною компанією із застосуванням сучасного фреймворку React. У процесі виконання роботи досягнуто поставленої мети - створено функціональний, адаптивний та зручний у використанні вебдодаток, що відповідає сучасним вимогам до продуктивності, безпеки та користувацького досвіду.

На основі аналізу джерел можна зробити висновки, що React є оптимальним вибором для будівельних вебдодатків завдяки своїй здатності ефективно обробляти складну логіку та великі обсяги даних. Особливо перспективним напрямом є використання 3D-візуалізації для архітектурних компаній, що потребують інтерактивних презентацій проєктів. Подальші дослідження доцільно спрямувати на інтеграцію штучного інтелекту для автоматизації кошторисів та застосування AR/VR технологій для віддаленого огляду будівельних майданчиків, що відкриває нові можливості для інновацій у будівельній галузі.

Наукові підходи підкреслюють, що ефективність React у системах управління проєктами зумовлена його компонентною архітектурою, високою продуктивністю завдяки віртуальному DOM, односпрямованим потоком даних та сучасними можливостями, такими як хуки і JSX. Таким чином, на основі проведеного огляду можна стверджувати, що React є оптимальним інструментом для будівельних вебдодатків. Його архітектурні та функціональні особливості створюють міцну основу для подальших досліджень, зокрема у напрямках інтеграції штучного інтелекту для автоматизації кошторисів та застосування AR/VR технологій для віддаленого огляду будівельних майданчиків, що відкриває перспективи для інновацій у галузі.

Стабільне лідерство React серед фронтенд-фреймворків, випередження Angular, Vue.js та інших, а також висока оцінка серед розробників, підтверджують його актуальність і перспективність. Водночас React має певні

виклики. Зокрема, складність навчання для новачків, особливо тих, хто не має досвіду з JavaScript, може уповільнювати впровадження технології. Крім того, швидкий розвиток екосистеми вимагає постійного оновлення знань і освоєння нових інструментів, що є важливим аспектом для команд розробників [17].

Серед основних вимог до майбутнього вебзастосунку можна виділити інтуїтивність, адаптивність, швидкодія, інтерактивність та зручність використання на різних типах пристроїв. Інтуїтивність інтерфейсу має забезпечувати легку орієнтацію користувача в структурі сайту без потреби у попередньому навчанні чи детальному вивченні інструкцій. Адаптивність є необхідною для коректного відображення контенту на екранах різних розмірів - від смартфонів до великих моніторів настільних комп'ютерів. Створення такого вебзастосунку відповідає сучасним потребам будівельної компанії і відкриває перспективи для підвищення ефективності комунікації з клієнтами, що є необхідною передумовою для подальшого розвитку бізнесу.

Для розробки логіки інтерфейсу обрано мову TypeScript, яка забезпечує статичну типізацію та полегшує виявлення помилок ще на етапі компіляції. Це дозволяє гарантувати коректність передачі даних між компонентами та запобігає runtime-помилкам, що є важливим при роботі з інтерактивними елементами, такими як калькулятор вартості6. Конфігурація TypeScript через tsconfig.json забезпечує консистентність коду по всьому проєкту та полегшує розробку завдяки інтелектуальному автодоповненню та навігації в IDE [24]. Таким чином, вибір Next.js 14 з App Router у поєднанні з TypeScript, Tailwind CSS та ShadCN/UI створює потужний і сучасний стек технологій, який відповідає вимогам масштабованості, продуктивності та зручності розробки, що є ключовими для успішної реалізації вебзастосунку будівельної компанії.

На етапі архітектурного проєктування вебзастосунку було обрано модель, що базується на принципах модульної структури та чіткому розділенні обов'язків між складовими системи. Наш підхід до організації логіки взаємодії полягає у застосуванні компонентного підходу, де кожен функціональний

елемент реалізується як окремий компонент. Така модель дозволяє забезпечити гнучкість, масштабованість і простоту підтримки проєкту.

Особливе значення у функціональній структурі вебзастосунку має проєктування взаємодії користувача з різними секціями сайту, особливо з інтерактивними елементами, такими як калькулятор вартості та форма зворотного зв'язку. Підхід до розробки інтерфейсу базується на створенні інтуїтивно зрозумілих і зручних для користувача компонентів, що сприяють підвищенню залученості та ефективності взаємодії. Інтеграція з серверними діями, розміщеними у директорії app/actions/, дозволяє безпечно обробляти форми на серверній стороні без перезавантаження сторінки, що підвищує зручність і безпеку взаємодії.

Структурна модель системи забезпечує чітку організацію зв'язків між компонентами, зберігаючи при цьому гнучкість для розширення функціональності. Кожен модуль взаємодіє з іншими виключно через визначені інтерфейси, що запобігає надмірним залежностям і спрощує підтримку системи. Таким чином, запропоновані проєктні рішення створюють міцну основу для реалізації сучасного вебдодатку, що відповідає вимогам адаптивності, інтерактивності та продуктивності. Модульна архітектура і чіткий розподіл відповідальності між компонентами формують передумови для подальших власних досліджень, спрямованих на оптимізацію структури та підвищення ефективності системи.

Конфігурація включає строгі параметри типізації, що забезпечують високу якість коду та запобігають runtime-помилкам. Загальна організація проєкту передбачає чіткий поділ усіх компонентів відповідно до їх функціонального призначення. Компоненти, які стосуються загального макета сторінки, такі як шапка або підвал сайту, винесені в директорію components/layout для забезпечення консистентності оформлення по всьому застосунку. Такий підхід дозволяє легко орієнтуватися у структурі проєкту та сприяє масштабованості, оскільки кожен елемент інтерфейсу може розроблятись, змінюватися або тестуватися окремо. Уся стилізація додатку реалізована за допомогою

утилітарної бібліотеки Tailwind CSS. Такий підхід значно пришвидшує розробку інтерфейсу та полегшує підтримку стилів.

Створений вебдодаток служить цифровою платформою для будівельної компанії, забезпечуючи комплексне представлення її діяльності в онлайн-просторі. Основне призначення додатку полягає у налагодженні ефективної комунікації з клієнтами через надання доступу до інформації про послуги, демонстрацію портфоліо реалізованих проєктів та забезпечення можливості швидкого зв'язку через інтегровану систему зворотного зв'язку.

Технічне рішення характеризується адаптивним дизайном та інтуїтивним інтерфейсом, що формує позитивне враження про компанію та підвищує довіру потенційних клієнтів. Платформа функціонує як ефективний маркетинговий інструмент, дозволяючи залучати нових клієнтів через пошукову оптимізацію та інтеграцію з соціальними мережами.

Впровадження CTA-елементів стимулює активну взаємодію користувачів з платформою, зокрема через можливість оформлення попередніх замовлень та отримання консультацій. Таким чином, вебдодаток не тільки виконує інформаційну функцію, але й сприяє цифровій трансформації бізнес-процесів компанії, забезпечуючи її конкурентоспроможність на сучасному ринку.

СПИСКИ ВИКОРИСТАНИХ ДЖЕРЕЛ

1. React Development. SmartTek Solutions. URL: https://smarttek.solutions/technologies/react-development/ (дата звернення: 23.03.2025).

2. Sharma R., Gupta P. React.js: A Modern JavaScript Library for Building User Interfaces // International Journal of Research Publication and Reviews. 2024. Vol. 5, Issue 4. C. 123-130. URL: https://ijrpr.com/uploads/V5ISSUE4/IJRPR24825.pdf (дата звернення: 22.04.2025).

3. How to Build a Custom Construction Architectural Design Software in React in 2024.Slashdev. URL: https://slashdev.io/-how-to-build-a-custom-construction-architectural-design-software-in-react-in-2024 (дата звернення: 22.03.2025).

4. Transforming Digital Presence: A React Development Success Story. GetTrusted. URL: https://gettrusted.io/posts/transforming-digital-presence-a-react-development-success-story-15063/ (дата звернення: 23.03.2025).

5. Що таке React JS і для чого він потрібен . DAN.IT Education. URL: https://dan-it.com.ua/uk/blog/chto-takoe-react-js-i-dlja-chego-on-nuzhen/ (дата звернення: 22.04.2025).

6. 8 інструментів для розробки додатків на React. Alternative Science. URL: https://alternativescience.net/programming/194-8-instrumentiv-dlya-rozrobky-dodatkiv-na-react/ (дата звернення: 22.03.2025).

7. Що таке React JS і як почати вивчати React: навички для React Developer. Cases Media. URL: https://cases.media/article/sho-take-react-js-yak-pochati-vivchati-reakt-navichki-dlya-react-developer (дата звернення: 22.04.2025).

8. Що таке React JS, як почати вивчати React. Brainlab. URL: https://brainlab.com.ua/uk/blog-uk/shho-take-react-js-yak-pochaty-vyvchaty-reakt (дата звернення: 24.04.2025).

9. Що таке React JS і для чого він потрібен . DAN.IT Education. URL: https://dan-it.com.ua/uk/blog/chto-takoe-react-js-i-dlja-chego-on-nuzhen/ (дата звернення: 23.03.2025).

10. Архітектура React-додатка. Foxminded. URL: https://foxminded.ua/arkhitektura-react-dodatka/ (дата звернення: 21.05.2025).

11. React. Вікіпедія. URL: https://uk.wikipedia.org/wiki/React (дата звернення: 22.03.2025).

12. Що таке React JS, як почати вивчати React. Brainlab. URL: https://brainlab.com.ua/uk/blog-uk/shho-take-react-js-yak-pochaty-vyvchaty-reakt (дата звернення: 21.03.2025).

13. Козловський О.І. Сучасні інструменти розробки додатків на React // Інформаційні технології. 2024. № 2. С. 45-52. URL: https://journals.maup.com.ua/index.php/it/article/view/2096 (дата звернення: 23.03.2025).

14. React in Action: Real-World Applications of the Popular JavaScript Library . PNN. URL: https://pnn.com.ua/ua/blog/detail/react-in-action-real-world-applications-of-the-popular-javascript-library (дата звернення: 24.03.2025).

15. Thinking in React. React Documentation. URL: https://uk.reactjs.org/docs/thinking-in-react.html (дата звернення: 22.03.2025).

16. Thinking in React. React Documentation. URL: https://uk.reactjs.org/docs/thinking-in-react.html (дата звернення: 21.03.2025).

17. Що таке React JS і як почати вивчати React: навички для React Developer . Cases Media. URL: https://cases.media/article/sho-take-react-js-yak-pochati-vivchati-reakt-navichki-dlya-react-developer (дата звернення: 28.03.2025).

18. Драйвером зростання будівництва в Україні у 2024 році став … - GMK Center. URL: https://gmk.center/ua/posts/drajverom-zrostannia-budivnytstva-v-ukraini-u-2024-rotsi-stav-nezhytlovyj-sehment/ (дата звернення: 05.06.2025).

19. Тенденції будівельного ринку України у 2024 році - Property Times. URL:

https://propertytimes.com.ua/blogs/andriy_ozeychuk/tendentsiyi_budivelnogo_rinku_ukrayini_u_2024_rotsi (дата звернення: 05.06.2025).

20. Forbes Україна. Ринок будівництва: тенденції та прогнози 2023. URL: https://forbes.ua/industry/construction-trends-2023 (дата звернення: 05.06.2025).

21. Будівельний ринок України 2024 року - BDO Україна. URL: https://www.bdo.ua/uk-ua/insights-2/information-materials/2024/ukrainian-construction-market-in-2024 (дата звернення: 05.06.2025).

22. React 19: что нового? Habr. URL: https://habr.com/ru/articles/788898/ (дата звернення: 10.06.2025).

23. Обговорення: React 19 - що змінилось? DOU. URL: https://dou.ua/forums/topic/48621/ (дата звернення: 10.06.2025).

24. Front-end без Next.js - як спорткар без двигуна: аналізуємо переваги та недоліки. Wezom. URL: https://wezom.com.ua/blog/front-end-bez-nextjs-kak-sportkar-bez-dvigatelya-analiziruem-preimuschestva-i-nedostatki (дата звернення: 10.06.2025).

25. Документація Next.js 14. IntLayer. URL: https://intlayer.org/ru/doc/environment/nextjs/14 (дата звернення: 10.06.2025).

26. Циганчук М.С., студент; Остапчук О.П., керівник. Розробка вебдодатку з надання послуг будівельною компанією з використанням фреймворку REACT. *Збірник тез студентської науково-практичної конференції навчально-наукового інституту кібернетики, інформаційних технологій та інженерії* (Рівне, травень 2025 року). Рівне: НУВГП, 2025. Прийнято до друку.