МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА
Навчально-науковий інститут
«Інститут інформаційних технологій в економіці»
Кафедра інформаційних систем в економіці

ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА «КОМП'ЮТЕРНІ НАУКИ»
галузь знань 12 «Інформаційні технології»
спеціальність 122 «Комп'ютерні науки»

Форма навчання: денна

КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА

на тему: «РОЗРОБЛЕННЯ САЙТУ-ПОРТФОЛІО З ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ JAVASCRIPT І REACT»

здобувача Пружанського Олексія Володимировича
(ПІБ) *(Підпис)*

Науковий керівник:

к.е.н., доцент

_____Лозовик Ю.М.

Робота допущена до захисту перед екзаменаційною комісією з атестації здобувачів вищої освіти

завідувач кафедри:

к.е.н., доцент

_____Тішков Б.О.

Київ 2025

Міністерство освіти і науки України
Київський національний економічний університет імені Вадима Гетьмана
Навчально-науковий інститут «Інститут інформаційних технологій в економіці»
Кафедра інформаційних систем в економіці

ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА «КОМП'ЮТЕРНІ НАУКИ»
галузь знань 12 «Інформаційні технології»
спеціальність 122 «Комп'ютерні науки»

ПОГОДЖЕНО:	ЗАТВЕРДЖУЮ:
Керівник проєктної групи(гарант)	Завідувач кафедри
освітньо-професійної програми	_____Тішков Б.О.
_____Помазун О.М.	"____" _____2025 р.
"____" _____2025 р.	

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

здобувача вищої освіти **Пружанського Олексія Володимировича**

очної (денної) форми навчання

на підготовку кваліфікаційної бакалаврської роботи
на тему: «РОЗРОБЛЕННЯ САЙТУ-ПОРТФОЛІО З ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ JAVASCRIPT І REACT»

Тему затверджено наказом ректора Університету від « *7* » *березня 2025* р. № *466- ст*.

Кваліфікаційна бакалаврська робота виконується на матеріалах

_____

План кваліфікаційної бакалаврської роботи

Розділ І Характеристика та аналіз предметної області

_____

Розділ ІІ Розробка вимог і моделювання інформаційної системи

_____

Розділ ІІІ Проектування та реалізація компонентів системи

_____

Об'єкт це процес створення персоналізованого сайту-портфоліо за допомогою JavaScript та бібліотеки React

Предмет дослідження - сукупність теоретичних, методичних і прикладних аспектів створення персоналізованого сайту-портфоліо з використанням сучасних технологій веб-розробки.

Мета кваліфікаційної бакалаврської є проєктування, розробка та впровадження адаптивного односторінкового сайту-портфоліо на базі технологій JavaScript і React, який забезпечує ефективну цифрову презентацію навичок та проєктів розробника.

Конкретні завдання, які здобувач повинен виконати для досягнення поставленої мети:

У розділі I потрібно: провести аналіз предметної області та виявити основні бізнес-процеси; зробити огляд та порівняльний аналіз існуючих інформаційних систем та обґрунтувати доцільність розробки нової інформаційної системи

У розділі II необхідно: визначити функціональні та нефункціональні вимоги до інформаційної системи; розробити архітектуру вебсайту з урахуванням адаптивності та UX-дизайну; змоделювати структуру і поведінку системи засобами UML; реалізувати основний функціонал односторінкового додатку за допомогою React.

У розділі III слід: впровадити функцію перемикання теми (світла/темна); забезпечити обробку та надсилання повідомлень із форми зворотного зв'язку; провести тестування працездатності та стабільності системи; обґрунтувати практичну значущість і потенціал використання розробленого рішення.

Завдання підготував
науковий керівник _____ **Лозовик Юрій Миколайович**
« 10 » березня 2025 р.

Завдання одержав
здобувач _____ **Пружанський Олексій Володимирович**
« 10 » березня 2025 р..

Відгук
на кваліфікаційну бакалаврську роботу
здобувача навчально-наукового інституту
«Інститут інформаційних технологій в економіці»
освітньо-професійної програми
«Комп'ютерні науки»
Пружанського Олексія Володимировича

Тема **«РОЗРОБЛЕННЯ САЙТУ-ПОРТФОЛІО З ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ JAVASCRIPT І REACT»**

1. У сучасному цифровому світі наявність особистого веб-портфоліо стала необхідною умовою для фахівців у сфері IT, дизайну, медіа, освіти та інших галузей. Портфоліо не лише презентує навички та досягнення, але й виступає як засіб самореклами, дозволяючи потенційним роботодавцям або замовникам оперативно ознайомитися з професійним рівнем спеціаліста. Використання сучасних вебтехнологій, зокрема JavaScript та React, дозволяє створити динамічний, зручний та адаптивний інтерфейс, що відповідає актуальним вимогам веб-розробки. React забезпечує ефективну побудову компонентної структури, що сприяє масштабованості та повторному використанню коду, а JavaScript — реалізацію інтерактивного функціоналу. Створення сайту-портфоліо з використанням технологій JavaScript і React є актуальним завданням, яке відповідає сучасним трендам у сфері веб-розробки.

2. Позитивні риси кваліфікаційної магістерської роботи:

Автором проведено ґрунтовне теоретико-практичне дослідження та аналіз підходів щодо створення сайтів-портфоліо, проаналізовані аналоги програмних, визначені їх переваги та недоліки та здійснено проектування власної системи. Автор роботи чітко визначив функціональні та нефункціональні вимоги до системи, розробив основні UML-діаграми (use-case, діяльності, послідовностей, класів, компонентів, трасування, прецедентів), даталогічну модель БД, необхідні

для реалізації власної інформаційної системи. Автор розробив прототип веб-застосунку.

3. Наявність самостійних розробок автора

Робота виконана самостійно. У роботі чітко розмежовано текст автора та запозичення, наявні посилання на використані джерела. Автором роботи, Пружанським Олексієм Володимировичем, проаналізовані наявні аналоги, спроєктовано власну інформаційну систему сайту-портфоліо та створено її прототип. Значну увагу приділено етапу проєктування веб-застосунку.

4. Цінність теоретичних висновків та практичних рекомендацій:

Автором опрацьовано значний обсяг інформації, здійснено систематизацію та узагальнення відомостей щодо проєктування і створення прототипу веб-застосунку.

5. Наявність недоліків:

- відсутність економічного обґрунтування створення нової інформаційної системи, а також порівняння розробленої системи з наявними аналогами, проаналізованими в теоретичному розділі роботи;
- відсутній опис серверної частини застосунку (наведений лише фрагмент коду);
- реалізований лише фрагмент БД;
- відсутнє модульне, системне та інтеграційне тестування серверної частини додатку;
- у роботі наведені лише візуальні тест-кейси;
- висновки подані у формі анотацій.

6. Загальна оцінка кваліфікаційної магістерської роботи та її допущення до захисту перед ЕК:

Зазначене вище дозволяє констатувати, що дипломна робота **Пружанського Олексія Володимировича** є самостійно завершеним дослідженням. Автор роботи вміє розробляти веб-застосунки з використанням сучасного стеку технологій (React, MongoDB, Next.js, Git). Дипломна робота відповідає вимогам щодо написання та оформлення бакалаврських робіт і може бути рекомендована до захисту в ЕК.

Робота може бути оцінена на відміно (90 б).

Науковий керівник,
доцент кафедри інформаційних систем
в економіці, к. е. н. Лозовик Ю.М.

_____ _____

(підпис) *(прізвище, ініціали)*

“11” червня 2025 р.

Рецензія
на кваліфікаційну бакалаврську роботу
здобувача вищої освіти
Пружанського Олексія Володимировича

(прізвище, ім'я, по батькові)

Тема **«РОЗРОБЛЕННЯ САЙТУ-ПОРТФОЛІО З ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ JAVASCRIPT І REACT»**

Актуальність теми кваліфікаційної бакалаврської роботи і доцільність її розроблення

Тема є актуальною в умовах стрімкого розвитку цифрових технологій і зростання попиту на фахівців IT-галузі. Розробка персонального сайту-портфоліо дозволяє ефективно презентувати навички розробника, сприяє підвищенню конкурентоспроможності на ринку праці та відповідає сучасним тенденціям у сфері веброзробки. Використання технологій JavaScript і React забезпечує відповідність розробки вимогам до інтерактивності, адаптивності та зручності користування.

Якість проведеного дослідження

Робота виконана на високому рівні, містить структурований аналіз предметної галузі, постановку задачі, детальну специфікацію вимог та реалізацію з використанням сучасного технологічного стеку. Здобувач продемонстрував вміння працювати з бібліотеками React, TailwindCSS, інтеграцією з backend, а також здатність до обґрунтованого вибору архітектурних рішень.

Позитивні риси кваліфікаційної бакалаврської роботи: чітка структура та логічна послідовність викладу матеріалу; використання сучасних інструментів розробки (React, Vercel, EmailJS); реалізація темної/світлої теми, адаптивного дизайну та форми зворотного зв'язку; побудова моделей у відповідності до стандартів UML/SysML; візуальне та технічне оформлення проєкту на високому рівні.

Зауваження. в окремих розділах доцільно було б розширити порівняльний аналіз альтернативних фреймворків; варто посилити розділ тестування з прикладами кейсів.

Практична значимість висновків і рекомендацій Отримані результати мають високу практичну цінність та можуть бути застосовані для створення персональних сайтів IT-фахівцями, студентами та фрилансерами. Застосовані технології дозволяють масштабувати розробку та інтегрувати додатковий функціонал. Методичні підходи, продемонстровані у роботі, можуть бути використані в навчальному процесі або під час виконання інших проєктів.

Місце роботи та посада рецензента

Науковий ступінь, учене звання (*за наявності*)_____

(підпис, ПІБ)

Зав. відділення «Програмування»
Відокремленого структурного підрозділу
«Фаховий коледж інформаційних
систем і технологій» _____ Світлана Миколаївна ЯНЧУК

“10” червня 2025 р.

АНОТАЦІЯ

кваліфікаційної бакалаврської роботи

здобувача першого (бакалаврського) рівня вищої освіти, 4 курсу,

виконаної на тему: «Розроблення сайту-портфоліо з використанням технологій JavaScript і React»

Київ: кафедра інформаційних систем в економіці, 2025 р.

Бакалаврська дипломна робота присвячена актуальній проблемі створення персоналізованого веб-сайту-портфоліо з використанням сучасних вебтехнологій JavaScript та React. У роботі розглядається підхід до побудови динамічного інтерфейсу, здатного адаптуватися до вподобань користувача, а також забезпечення зворотного зв’язку через інтерактивні веб-форми.

Бакалаврська дипломна робота складається з трьох розділів, логічно пов’язаних між собою.

У першому розділі надано характеристику предметної галузі – сфери веброзробки персональних сайтів – та проаналізовано сучасні підходи й приклади реалізації онлайн-портфоліо. Проведено огляд актуальних тенденцій та інструментів, що використовуються для створення таких рішень.

Другий розділ є проєктним і присвячений формуванню вимог до інформаційної підсистеми сайту-портфоліо, постановці задачі та моделюванню її компонентів. Наведено функціональні та нефункціональні вимоги до системи, UML-діаграми та опис сценаріїв взаємодії користувача з інтерфейсом.

У третьому розділі подано інформаційне, технічне та програмне забезпечення системи. Висвітлено реалізацію клієнтської частини з можливістю перемикання між темною та світлою темами, а також описано механізм передачі даних із контактної форми на сервер. Окрему увагу приділено структурі бази даних, формам документів та організації системи зберігання даних.

У висновках надано обґрунтування ефективності розробленої системи, її доцільності та перспектив використання для представлення професійних навичок розробника у сучасному цифровому середовищі.

РЕФЕРАТ

Кваліфікаційна бакалаврська робота містить 53 сторінок, 12 таблиць, 40 рисунків, список використаних джерел з 31 найменувань, 2 додатків.

Об’єктом дослідження виступає процес створення сучасних односторінкових вебзастосунків (SPA) у сфері персонального позиціонування розробника в Інтернеті.Предметом дослідження є процес розробки односторінкового вебсайту-портфоліо з інтерактивними функціональними можливостями.

Предметом дослідження є методи, технології та інструменти розробки інтерактивного сайту-портфоліо з використанням JavaScript і React, що включає реалізацію адаптивного дизайну, перемикання теми оформлення та форму зворотного зв’язку з передачею даних на сервер.

Мета бакалаврської дипломної роботи полягає в розробці та впровадженні адаптивного сайту-портфоліо на базі технологій JavaScript і React для ефективної цифрової презентації навичок і проєктів розробника, з реалізацією перемикання теми оформлення (світлої/темної) та інтеграцією функціональної форми зворотного зв’язку. Очікуваний економічний ефект полягає у підвищенні конкурентоспроможності розробника на ринку праці, зменшенні витрат на самопрезентацію, а соціальний – у сприянні професійній самореалізації, доступності цифрових сервісів самопросування та розвитку особистого бренду.

Завданнями бакалаврського дипломного проекту є:

- аналіз вимог до функціональності та дизайну персонального портфоліо;
- розробка архітектури сайту;
- реалізація ключових компонентів у React;
- забезпечення адаптивності та навігації;
- інтеграція зворотного зв’язку з передачею даних на бекенд;
- реалізація перемикання теми;

- тестування роботи системи.

Апаратні та програмні засоби, що використовувались при проектуванні: персональний комп’ютер, браузер Google Chrome, середовище розробки Visual Studio Code, Node.js, бібліотека React, Tailwind CSS, сервіс Vercel для деплою.

Результати досягнуті в процесі роботи: створено повнофункціональний адаптивний сайт-портфоліо з можливістю перемикання теми, формою зворотного зв’язку з валідацією, організовано структуру компонентів згідно з принципами повторного використання. Новизна полягає в інтеграції зворотного зв’язку та теми в SPA-додатку без використання зовнішніх CMS.

Ступінь упровадження: експериментальне використання для особистої презентації в мережі. Одержані результати можуть бути використані для персонального просування розробника, у процесі подачі заявок на роботу, створення фриланс-профілів, а також як шаблон для створення аналогічних рішень іншими фахівцями. Рішення характеризуються високою адаптивністю, швидкодією та простотою розгортання.

Рік виконання бакалаврського дипломного проекту: 2025

Рік захисту бакалаврського дипломного проекту: 2025

Перелік ключових слів: портфоліо, сайт, веброзробка, React, JavaScript, SPA, адаптивність, форма зворотного зв’язку, перемикання теми, front-end, користувач, компоненти, дизайн, інтерфейс

ЗМІСТ

ВСТУП 3

РОЗДІЛ 1 6

ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ 6

1.1 Характеристика предметної галузі та об’єкта дослідження 6

1.2 Аналіз літературних джерел та практичного досвіду використання ІС і технологій в предметній галузі 8

РОЗДІЛ 2 11

РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ 11

2.1 Аналіз і специфікація вимог до інформаційної системи 11

2.2 Постановка задачі 17

2.2.1 Характеристика задачі 17

2.2.2 Алгоритм розв’язання задачі 20

2.3 Моделювання інформаційної системи 24

2.3.1 Моделювання поведінки системи 24

2.3.2 Моделювання структури системи 27

2.3.3 Розподіл вимог за компонентами системи 28

РОЗДІЛ 3 34

ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ 34

3.1 Інформаційне забезпечення 39

3.2 Технічне забезпечення 39

3.2.1 Загальні положення та схема автоматизації 39

3.2.2 Структура комплексу технічних засобів 40

3.2.3 Опис автоматизованого робочого місця 41

3.3 Програмне забезпечення 43

3.3.1 Структура програмного забезпечення 43

3.3.2 Системне програмне забезпечення 44
3.3.3 Прикладне програмне забезпечення 45
3.3.4 Програмна документація 46
3.4 Результати реалізації інформаційної системи 46
ВИСНОВКИ 49
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 51
ДОДАТКИ 53

ВСТУП

У сучасному цифровому світі наявність власного веб-сайту-портфоліо стає необхідною умовою для IT-фахівців, зокрема для веброзробників. Такий ресурс виконує не лише презентаційну функцію, а й слугує інструментом професійного самовираження, просування особистого бренду та взаємодії з потенційними замовниками чи роботодавцями. У зв’язку з цим тема створення персонального сайту-портфоліо набуває особливої актуальності, адже потребує не просто розробки типової візитки, а впровадження гнучких, адаптивних та інноваційних підходів до проєктування інтерфейсу та взаємодії з користувачем.

Аналіз останніх досліджень і публікацій показує зростання уваги до розробки SPA-додатків (Single Page Application) з використанням бібліотек React, Vue та Angular. У роботах Гросса Дж. та Максвелла А., Wieruch R., а також в освітніх матеріалах Mozilla Developer Network, GitHub Docs і Vercel Docs висвітлюються актуальні технології фронтенд-розробки, методи оптимізації продуктивності та забезпечення доступності інтерфейсу. Проте, попри численні інструкції та приклади, недостатньо уваги приділяється структурованому проєктуванню особистих вебпортфоліо з підтримкою перемикання тем, контактною формою і серверною інтеграцією. Саме цим аспектам і присвячено даний дипломний проєкт.

Предметна галузь проєкту охоплює веброзробку, фронтенд-технології та користувацький досвід. Об’єктом дослідження виступає процес створення персоналізованого сайту-портфоліо за допомогою JavaScript та бібліотеки React (див. [6], [7], [8], [9]). Особливість розробленої системи полягає у поєднанні сучасного дизайну, динамічного інтерфейсу, функціональності, що відповідає вимогам користувачів. Зокрема, реалізовано можливість перемикання теми оформлення (світла/темна), що забезпечує комфортне сприйняття контенту, а також створено інтерактивну контактну форму, дані з якої передаються на бекенд для подальшої обробки.

Метою дипломного проєкту є проєктування, розробка та впровадження

адаптивного односторінкового сайту-портфоліо на базі технологій JavaScript і React, який забезпечує ефективну цифрову презентацію навичок та проєктів розробника, підтримує перемикання теми (світлої/темної), інтегрує функціональну форму зворотного зв'язку, а також сприяє підвищенню конкурентоспроможності здобувача на ринку праці.

Відповідно до поставленої мети визначено такі завдання:

- здійснити аналіз предметної галузі та сучасних підходів до розробки веб-портфоліо;
- визначити функціональні та нефункціональні вимоги до інформаційної системи;
- розробити архітектуру вебсайту з урахуванням адаптивності та UX-дизайну;
- змоделювати структуру і поведінку системи засобами UML;
- реалізувати основний функціонал односторінкового додатку за допомогою React;
- впровадити функцію перемикання теми (світла/темна);
- забезпечити обробку та надсилання повідомлень із форми зворотного зв'язку;
- провести тестування працездатності та стабільності системи;
- обґрунтувати практичну значущість і потенціал використання розробленого рішення.

У процесі виконання роботи використовувались такі методи дослідження: структурно-логічний аналіз, порівняльний аналіз функціональних рішень, моделювання за допомогою UML, а також вербально-описовий метод для представлення архітектурних рішень.

Практична значущість роботи полягає в тому, що результатом дослідження є реальний, функціонуючий програмний продукт, який може бути використаний як основа для власного онлайн-портфоліо будь-якого фахівця у сфері IT або як шаблон для комерційних проєктів.

Теоретична значущість полягає в дослідженні підходів до побудови

інтерактивних SPA-додатків, адаптивного дизайну, динамічного керування інтерфейсом користувача та інтеграції клієнтської і серверної частин у межах одного проєкту.

Робота складається з трьох розділів. Перший містить аналіз предметної галузі та літературних джерел. У другому розділі наведено постановку задачі та моделювання підсистеми. Третій розділ присвячений реалізації основних компонентів сайту.

РОЗДІЛ 1
ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Характеристика предметної галузі та об’єкта дослідження

Предметна галузь проєкту охоплює сферу веброзробки – одну з найбільш динамічно зростаючих і критично важливих галузей сучасних інформаційних технологій. У контексті цього дослідження вона включає в себе засоби, методи, інструменти та технології, що використовуються для створення, підтримки та розвитку інтерфейсів взаємодії між користувачем і програмною системою, зокрема персональними сайтами-портфоліо.

Об’єктом дослідження є процес розробки та реалізації персонального веб-сайту-портфоліо з використанням бібліотеки React, який включає динамічні інтерфейсні рішення, передачу даних на сервер і адаптивність до користувацьких уподобань.

Вебпортфоліо – це цифрова візитівка розробника, що дозволяє презентувати власні навички, проєкти, сертифікати, контакти й інші важливі елементи. В умовах високої конкуренції на IT-ринку, ефективна презентація себе через інтернет стає одним з ключових факторів працевлаштування, самозайнятості чи просування особистого бренду. Таким чином, створення вебсайту-портфоліо – це не лише технічне завдання, а й частина ширшого процесу цифрової самореалізації.

Серед об’єктів предметної галузі можна виокремити:

- Індивідуальні розробники, фрилансери, початківці в IT-сфері;
- Малі IT-компанії, що створюють шаблонні або кастомні портфоліо;
- Платформи для розміщення портфоліо (GitHub Pages, Netlify, Vercel тощо);
- Фреймворки та бібліотеки (React, Vue, Next.js), системи керування контентом;
- Служби збору відгуків та аналітики (Google Analytics, Email API,

Formspree тощо).

У предметній галузі приймаються такі типові рішення:

- Вибір технологічного стеку (React, CSS-in-JS, Tailwind, тощо);
- Розробка структури контенту: секції “Про себе”, “Проєкти”, “Контакти” тощо;
- Реалізація інтерактивних елементів: контактні форми, перемикання тем, анімації;
- Оптимізація інтерфейсу під мобільні пристрої (адаптивна верстка);
- Підключення аналітики та налаштування SEO.

Серед факторів, що впливають на прийняття рішень у цій галузі, можна виділити:

- Зростання вимог до UI/UX якості з боку користувачів;
- Швидкий розвиток JavaScript-екосистеми;
- Популярність open-source підходів і доступність шаблонів;
- Потреба в персоналізації інтерфейсу;
- Вимоги до SEO-оптимізації для персональних брендів.

Взаємозв’язки прийнятих рішень мають прямий вплив на подальші етапи розробки: вибір фреймворку визначає особливості архітектури, структура сайту – спосіб навігації, реалізація теми – компонування стилів тощо. Кожне рішення впливає на якість взаємодії користувача з системою.

Інформація, необхідна для прийняття обґрунтованих рішень, охоплює:

- Емпіричні дані (аналітика поведінки користувачів, показники відмов);
- Об’єктивні джерела (документація React, Bootstrap, Tailwind CSS);
- Результати евристичного аналізу (порівняння інших портфоліо);
- Практичні кейси реалізації SPA-додатків.

Основною методологією оброблення інформації виступають:

- Алгоритмізація процесів (навігація, обробка форм, тема);
- Імітаційне моделювання інтерфейсу користувача;
- Вербально-описовий та структурно-логічний підходи;

- Використання компонування і багаторазового використання коду (component reusability).

У контексті типів рішень, що приймаються, дослідження належить до відкритих задач із підвищеною гнучкістю – при наявності великої кількості технологічних і дизайнерських варіантів. Це ситуації за умов невизначеності (вибір стеку, стилістики, інтеграцій), що вирішуються шляхом побудови прототипів, моделювання та адаптивного дизайну.

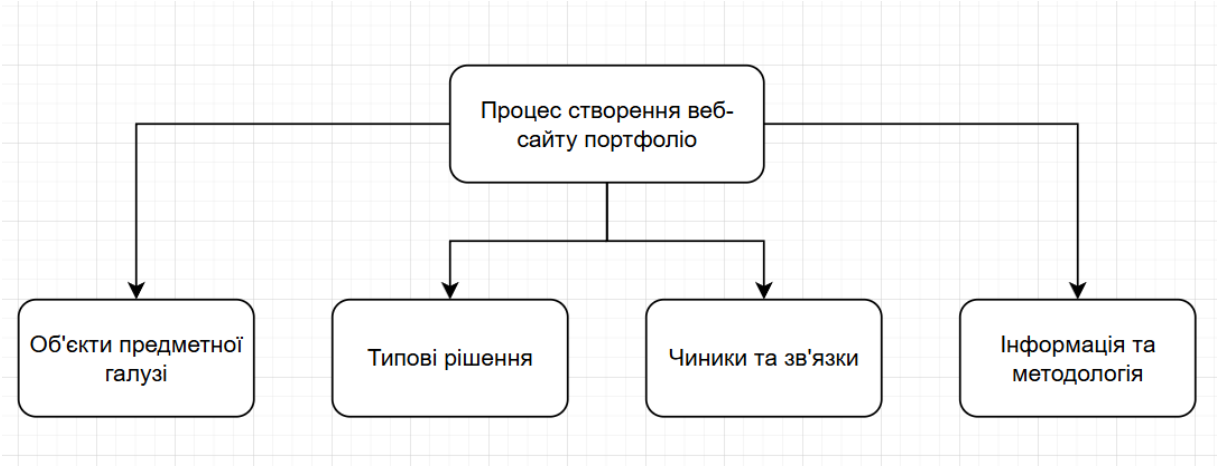

Рисунок 1.1.1 – Організаційна структура процесу створення веб-сайту-портфоліо

1.2 Аналіз літературних джерел та практичного досвіду використання ІС і технологій в предметній галузі

Сфера веброзробки, зокрема створення персональних сайтів-портфоліо, об’єктом активного теоретичного та практичного вивчення. Розробка інтерфейсів взаємодії, формування архітектури односторінкових додатків (SPA), а також впровадження інструментів зворотного зв’язку вимагають комплексного підходу з використанням сучасних інформаційних технологій, таких як JavaScript (див. [6], [7], [8], [9]), React (див. [6], [7], [8], [9]), REST API, UI/UX-дизайн (див. [25]), адаптивна верстка та інтерактивні елементи.

Сучасні літературні джерела, зокрема праці таких авторів, як Robin Nixon, Eric Freeman, Alex Banks та Eve Porcello, детально описують процеси створення

сучасних вебінтерфейсів. У книгах «Learning React», «Eloquent JavaScript» та «Modern Full-Stack Development» аналізуються шаблони розробки компонентів, управління станом, принципи функціонального програмування у React, а також підходи до побудови масштабованої структури SPA-додатків.

Також слід відзначити роботи Степанова О.С. та Коваленка І.В., які досліджують практичні аспекти впровадження компонентно-орієнтованих підходів у веброзробці, а також розглядають досвід українських IT-компаній у впровадженні сучасних фреймворків.

Аналіз джерел дозволяє виокремити кілька домінуючих підходів до реалізації сайтів-портфоліо:

- використання готових CMS-платформ (WordPress, Wix, Tilda), що дозволяє швидко створити сайт без глибоких знань коду;
- шаблонні рішення на HTML/CSS/JS, які є статичними і обмеженими функціонально;
- розробка SPA-додатків з використанням React або Vue – найбільш гнучкий, масштабований та професійний підхід.

Недоліками першого підходу є обмежена кастомізація, низька продуктивність і складність реалізації унікальних сценаріїв взаємодії. У той час як React дає змогу реалізувати розширену функціональність, зокрема:

- інтерактивне перемикання теми (dark/light mode);
- реалізація клієнт-серверної комунікації (наприклад, з API або бекендом Node.js);
- адаптивний дизайн і мобільна оптимізація.

Серед основних вхідних інформаційних потоків можна виділити дані, що надходять від користувача (з форми зворотного зв’язку), а також параметри взаємодії з сайтом (перемикання теми, вибір мови, навігація між секціями). Вихідні потоки включають виведення персональної інформації, досвіду, реалізованих проєктів, а також підтвердження про надсилання повідомлення або помилки при обробці форми.

Особливої уваги заслуговує підхід до розділення обов’язків між

технічними компонентами системи: React відповідає за відображення інтерфейсу та взаємодію з користувачем, у той час як серверна частина (наприклад, реалізована на Node.js або Express) приймає, обробляє і зберігає дані.

Основні користувачі таких систем – розробники, рекрутери, роботодавці, замовники, яким важливо отримати швидкий і повний доступ до портфоліо, а також мати можливість зв’язатися з розробником. Важливо, щоб взаємодія була інтуїтивною, швидкою і технічно надійною – саме цим вимогам відповідає обрана технологічна архітектура.

Практичний досвід, зокрема приклади з репозиторіїв GitHub (див. [10], [11]), свідчить про поширене використання інтеграції темного та світлого режимів із збереженням у localStorage, застосування адаптивних сіток за допомогою CSS-фреймворків на кшталт Tailwind CSS (див. [13]) або Bootstrap, а також динамічного формування контенту на основі JSON-файлів або API.

Підсумовуючи, можна зробити висновок, що в умовах зростаючої конкуренції серед IT-спеціалістів персональний сайт-портфоліо має не лише відображати навички розробника, але й демонструвати володіння сучасними технологіями та підходами. Реалізація можливості перемикання теми і передача даних на бекенд є прикладом інтеграції інноваційних елементів у межах вебдодатку та свідчить про високий рівень технічної підготовки розробника.

РОЗДІЛ 2
РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Аналіз і специфікація вимог до інформаційної системи

Процес визначення вимог є критично важливим етапом у розробці будь-якої інформаційної системи, що забезпечує її цілеспрямовану, ефективну та передбачувану реалізацію. Вимоги – це умови, які повинна задовольняти система або її компоненти для досягнення поставленої мети, забезпечення працездатності, відповідності технічним стандартам і задоволення очікувань кінцевого користувача.

Ієрархія бізнес-вимог (діаграма вимог типу «requirement diagram»)

Головна бізнес-вимога: BR1. Надати користувачам доступ до інформації про розробника

- BR1.1: Відображати особисту інформацію (ПІБ, фото, контактні дані)
- BR1.2: Надати розділ «Про себе» (опис навичок, мети, освіти тощо)
- BR1.3: Продемонструвати реалізовані проєкти
- BR1.4: Інтегрувати форму зворотного зв’язку

Зв’язки з іншими елементами моделі

- Джерело вимог: потенційний користувач сайту, роботодавець.
- Бізнес-цілі: особистий маркетинг, просування послуг.
- Обмеження: використання лише відкритих веб-технологій (HTML/CSS/JS/React), некомерційне застосування.

Таблиця 2.1.1 – Бізнес-цілі

Стейкхолдер	**Бізнес-ціль**	**Бізнес-вимоги до ІС**
Замовник (розробник)	Презентація навичок	Сайт має містити окремі розділи: "Про мене", "Навички", "Портфоліо", "Контакти"

Продовження таблиці 2.1.1

Стейкхолдер	Бізнес-ціль	Бізнес-вимоги до ІС
Користувачі	Легко знайти інформацію	Сайт має мати навігацію з чітким меню та адаптивний дизайн для мобільних пристроїв
Рекрутери	Швидка оцінка профілю	На головній сторінці мають бути стисло представлені ключові технології та посилання на GitHub / LinkedIn
Технічний консультант	Перевірка коду	Код сайту має бути написаний за стандартами чистого коду, проект повинен бути задеплоєний і відкритий на GitHub

Аналіз потреб користувача показує, що важливо забезпечити зрозумілу та логічну навігацію, швидке завантаження сторінок, коректне відображення на різних пристроях. Дизайн має бути естетичним і адаптивним, із можливістю перемикання між світлою та темною темами відповідно до вподобань користувача. Також необхідна функціональна форма зворотного зв’язку з перевіркою на помилки та підтвердженням успішного надсилання. Робота системи не повинна вимагати встановлення додаткового програмного забезпечення.

Основні функціональні вимоги показані у таблиці 2.1.2.

Таблиця 2.1.2 – Функціональні вимоги

Ідентифікатор	Назва	Повна назва
FR1	Головна сторінка	Відображення основної інформації про користувача
FR2	Розділ «Про себе»	Надання короткої біографічної інформації
FR3	Галерея проєктів	Перелік реалізованих проєктів із посиланнями та описами

Продовження таблиці 2.1.2

Ідентифікатор	Назва	Повна назва
FR4	Форма контакту	Надсилання повідомлень через форму
FR5	Адаптивність	Коректне відображення на мобільних пристроях

Таблиця 2.1.3 – Нефункціональні вимоги

Ідентифікатор	Назва	Повна назва	Статус	Складність	Пріоритет	Ризик	Примітки
NFR1	Продуктивність	Завантаження за 2 сек.	Затвер-джено	Висо-ка	Висо-ий	Висо-кий	Опти-мі-зація коду
NFR2	Надійні-сть	Немає збоїв при переході між сторін-ками	Затверджено	Середня	Високий	Середній	Тесту-вання навіга-ції
NFR3	Безпека	Забезпеч е-ння захисту від спаму у формі зворотного зв’язку	Заплановано	Середня	Середній	CAPTCHA	NFR3
NFR4	Універсальність	Підтримка популярних браузерів	Затверджено	Низька	Середній	Кросбраузерність	NFR4

Продовження таблиці 2.1.3

Ідентифікатор	**Назва**	**Повна назва**	**Статус**	**Складність**	**Пріоритет**	**Ризик**	**Примітки**
NFR5	Масштабованість	Легке додавання нових розділів і компонентів	Затверджено	Висока	Високий	Компонентна архітектура React	NFR5

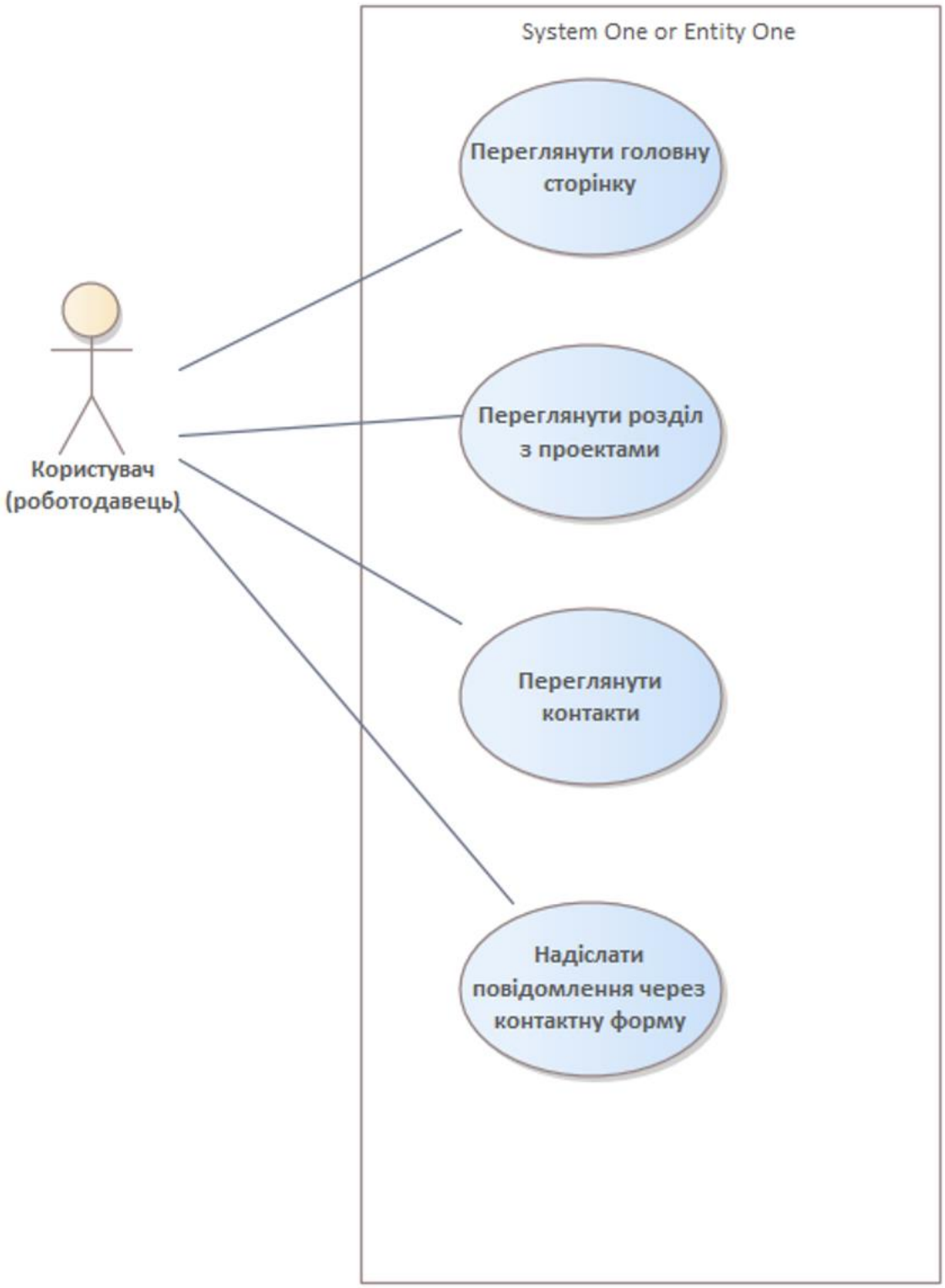

Рисунок 2.1.1 - Діаграма вимог UseCase діаграма

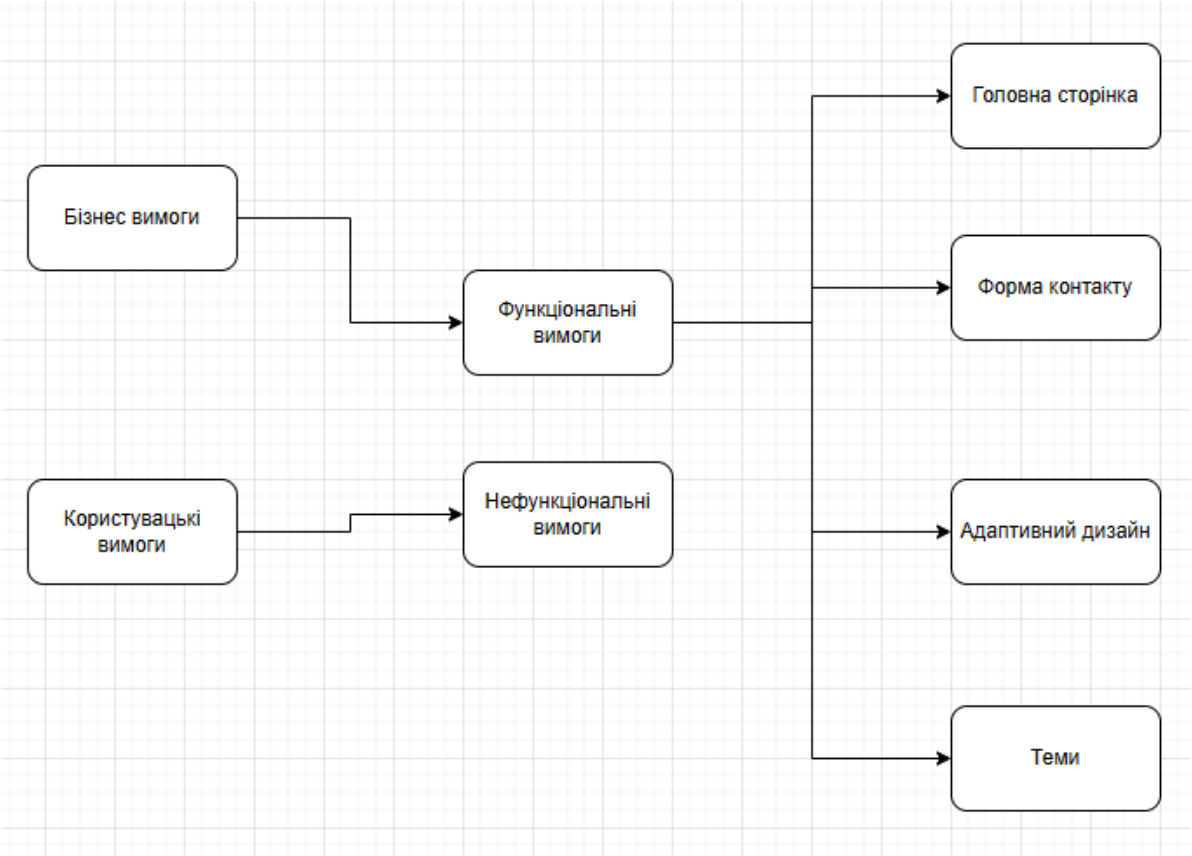

Рисунок 2.1.2 – Діаграма зв’язків між вимогами та елементами системи

Усі вимоги сформульовані згідно з принципами якості: вони є чіткими, завершеними, послідовними, не суперечать одна одній, мають однозначне тлумачення та можуть бути перевірені після реалізації. Наприклад, вимога щодо темного режиму має чіткий критерій перевірки – наявність перемикача і візуальна зміна кольорової схеми.

У процесі розробки використовуються такі методи:

- структурно-логічний аналіз для формування архітектури;
- вербально-описовий підхід для формулювання вимог;
- моделювання (UML/SysML) – для представлення системних компонентів і зв'язків між ними.

Розроблена специфікація вимог стане основою подальшого проєктування інформаційної підсистеми та реалізації її компонентів у рамках дипломного проєкту.

2.2 Постановка задачі

2.2.1. Характеристика задачі

Метою задачі є розробка інформаційної підсистеми – персонального сайту-портфоліо розробника – яка забезпечує представлення професійної інформації, навичок, проєктів та контактних даних. Розв’язання задачі передбачає автоматизацію відображення контенту, взаємодію з користувачем через форму зворотного зв’язку та дотримання сучасних стандартів веброзробки.

Техніко-економічна сутність полягає у створенні цифрового інструменту для особистого просування розробника без додаткових витрат на зовнішні ресурси. Розв’язання задачі засобами інформаційної системи дозволяє скоротити час на створення презентаційних матеріалів, забезпечити інтерактивну взаємодію з потенційними роботодавцями або замовниками, а також підвищити рівень персонального бренду.

Об’єкти, за управління якими розв’язується задача: веб-інтерфейс (сторінки сайту); навігаційні елементи; форма зворотного зв’язку; механізм перемикання теми; серверна частина для обробки повідомлень.

Використання вихідної інформації: вихідна інформація використовується для візуалізації контенту (біографія, проєкти, контакти), відображення повідомлень користувача після надсилання форми та зберігання статистичних даних для покращення UX.

Періодичність: задача виконується під час кожного звернення користувача до сайту. Оновлення відбувається автоматично під час завантаження сторінки або взаємодії з елементами (навігація, перемикання теми, відправка форми).

Умови припинення: автоматизоване розв’язання задачі припиняється у разі закриття сторінки або відсутності активності користувача.

Зв’язки з іншими задачами: задача пов’язана з задачами обробки форм,

збереження теми оформлення у браузері, реалізацією маршрутизації сторінок (React Router), інтеграцією з GitHub/LinkedIn API.

Розподіл дій: персонал здійснює первинне наповнення сайту, внесення змін до вмісту, адміністрування репозиторію. Технічні засоби відповідають за рендеринг контенту, маршрутизацію, обробку форм і UX.

Рисунок 2.2.1 - Інформаційна модель задачі

Вихідна інформація – це дані, які виводяться на екран користувачеві під час взаємодії з сайтом: вміст сторінок, повідомлення про успішне або помилкове надсилання форми, статус теми (світла/темна), навігація.

Таблиця 2.2.1 - Перелік вихідних повідомлень

№з/п	**Назва повідомлення**	**Ідентифікатор**	**Форма подання**	**Періодичність**	**Термін / затримка**	**Користувачі**
1.	Контактне повідомлення	OUT1	Виводиться в модальному вікні	При надсиланні форми	Миттєво	Користувач
2.	Повідомлення про помилку	OUT2	Виводиться на сторінці	У разі помилки	Миттєво	Користувач
3.	Список проєктів	OUT3	Список карток/блоків	Постійно	Після завантаження сторінки	Усі
4.	Візуалізація теми	OUT4	Графічне оформлення сайту	За перемиканням	Миттєво	Користувач

Вхідна інформація – це дані, які користувач вводить або які отримуються системою автоматично: ім’я, email, повідомлення з форми, вибір теми, дані навігації.

Таблиця 2.2.2 - Перелік вхідних повідомлень

№ з/п	**Назва повідомлення**	**Ідентифікатор**	**Форма подання**	**Термін / частота**	**Джерело**
1	Дані форми	IN1	Текстові поля	Кожне заповнення	Користувач
2	Вибір теми	IN2	Кнопка/перемикач	Будь-який момент	Користувач
3	Дії навігації	IN3	Події кліків	Будь-який момент	Користувач
4	Дані авторизації (опц.)	IN4	Token/URL (у GitHub API)	У разі інтеграції	GitHub API

Ескізи форм вхідних та вихідних повідомлень представлені на рис.2.2.1–2.2.3

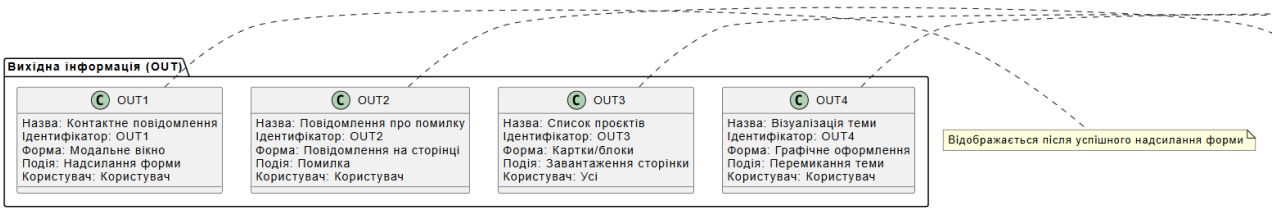

Рисунок 2.2.1 – Ескізи форм вхідних та вихідних повідомлень

Рисунок 2.2.2 – Наступний фрагмент ескізу форм вхідних та вихідних повідомлень

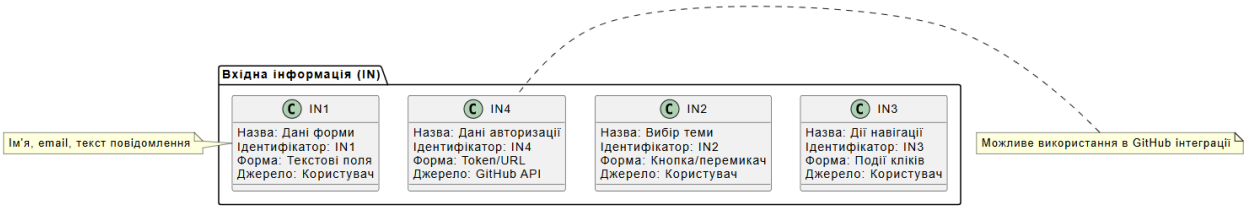

Рисунок 2.2.3 – Наступний фрагмент ескізу форм вхідних та вихідних повідомлень

2.2.2. Алгоритм розв’язання задачі

У межах задачі розробки сайту-портфоліо використовуються масиви вхідної інформації, отримані від користувача (дані з форми зворотного зв’язку, вибір теми оформлення, навігаційні події), а також дані, що зберігаються в коді або передаються у форматі JSON (перелік проєктів, контактна інформація, зміст секцій).

Таблиця 2.2.3 - Перелік масивів використовуваної інформації

Масив	Ідентифікатор	Максимальна кількість записів
Дані з форми	IN_FORM	100
Дані проєктів	IN_PROJECTS	50
Дані навігації	IN_NAV	10
Параметри теми	IN_THEME	2 (light, dark)
Посилання на соцмережі	IN_SOCIALS	10

Результати обчислень формуються у вигляді повідомлень, рендерингу вмісту сторінок, підтвердження дій користувача. Всі ці результати надаються у графічному форматі без потреби у генерації друкованих звітів, проте логіка сайту забезпечує їхній миттєвий вивід на екран.

Таблиця 2.2.4 – Перелік масивів результатної інформації

Масив	Ідентифікатор	Максимальна кількість записів
Повідомлення про надсилання	OUT_CONFIRM	1
Повідомлення про помилки	OUT_ERROR	10
Відображені проєкти	OUT_PROJECTS	50
Тематичне оформлення	OUT_THEME	1
Навігаційні сторінки	OUT_NAV	10

Задача не потребує складного математичного моделювання, проте можна формалізувати деякі частини, наприклад логіку вибору теми або валідацію форми.

Формула вибору теми оформлення:

$$T = \begin{cases} \text{'dark'}, \text{якщо localStorage.theme} = \text{'dark'} \\ \text{'light'}, \text{в інших вападках} \end{cases} \quad (2.1)$$

Перевірка валідності email:

$$E_{val} = \begin{cases} 1, \text{якщо email.match(Regex)} \neq \emptyset \\ 0, \text{інакше} \end{cases} \quad (2.2)$$

Алгоритм розв’язання задачі реалізується на клієнтській стороні в середовищі браузера за допомогою бібліотеки React і охоплює низку послідовних етапів. Спочатку завантажується початковий стан сайту та визначається тема оформлення шляхом зчитування значення з localStorage. Далі виконується рендеринг основних компонентів: головної сторінки, розділів "Про себе", "Портфоліо" та "Контакти".

Після цього реєструються обробники подій для навігації та взаємодії з формою. Залежно від вибраного маршруту відображається відповідний розділ, а введені у форму дані проходять валідацію. У разі успішної перевірки здійснюється надсилання інформації на бекенд і обробка відповіді сервера – з

виведенням повідомлення про успіх або помилку. Зміна теми призводить до оновлення значення у localStorage та повторного рендерингу з новими стилями.

Алгоритм враховує типові ситуації, як-от відсутність з’єднання з інтернетом, помилки валідації чи обробку неочікуваних даних. Структура рішення є модульною, що дає змогу легко масштабувати функціонал без необхідності внесення змін до основної архітектури.

Послідовність етапів алгоритму відображено на рисунках 2.2.4-2.2.5

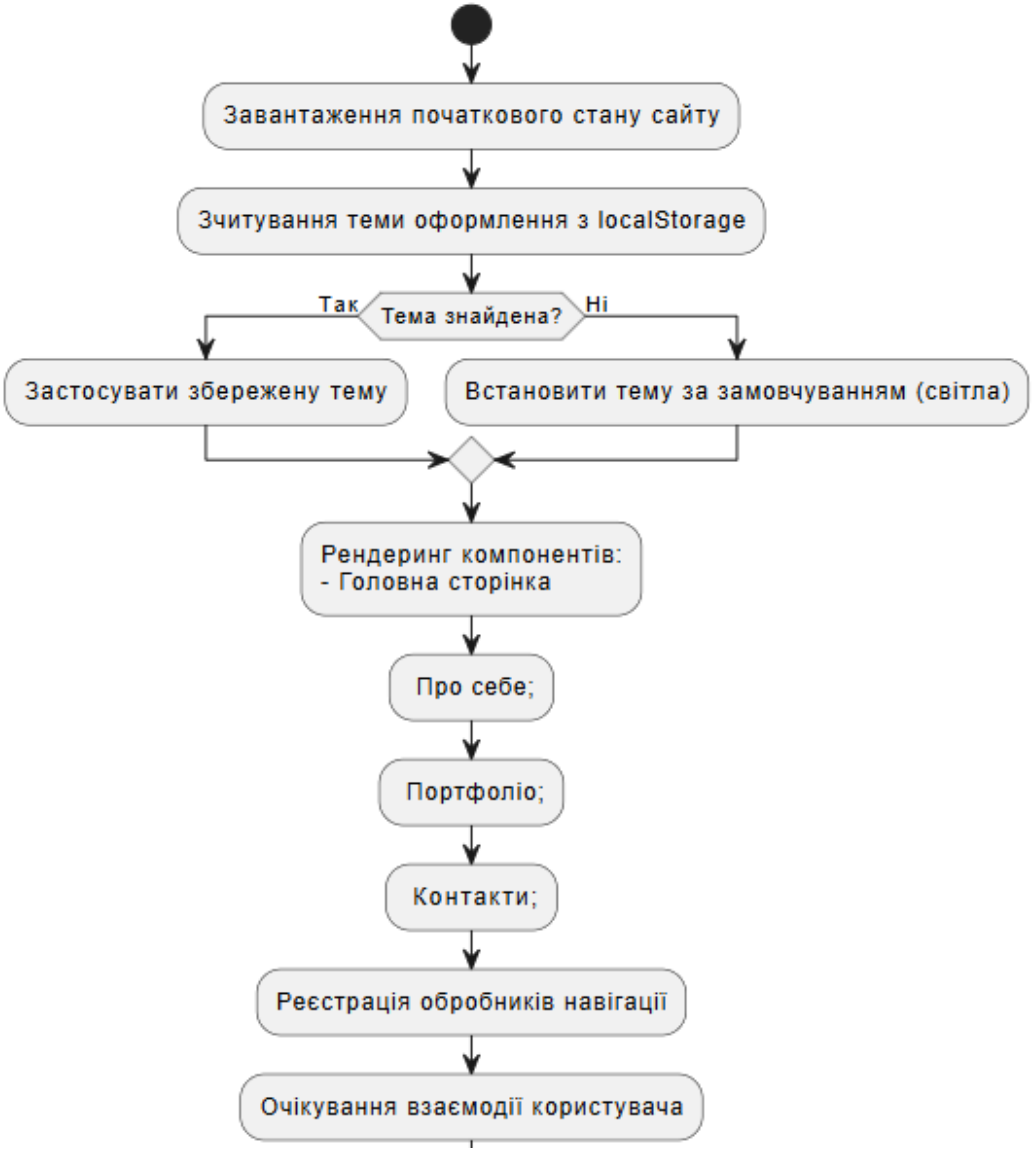

Рисунок 2.2.4 – Схема алгоритму розв’язання задачі

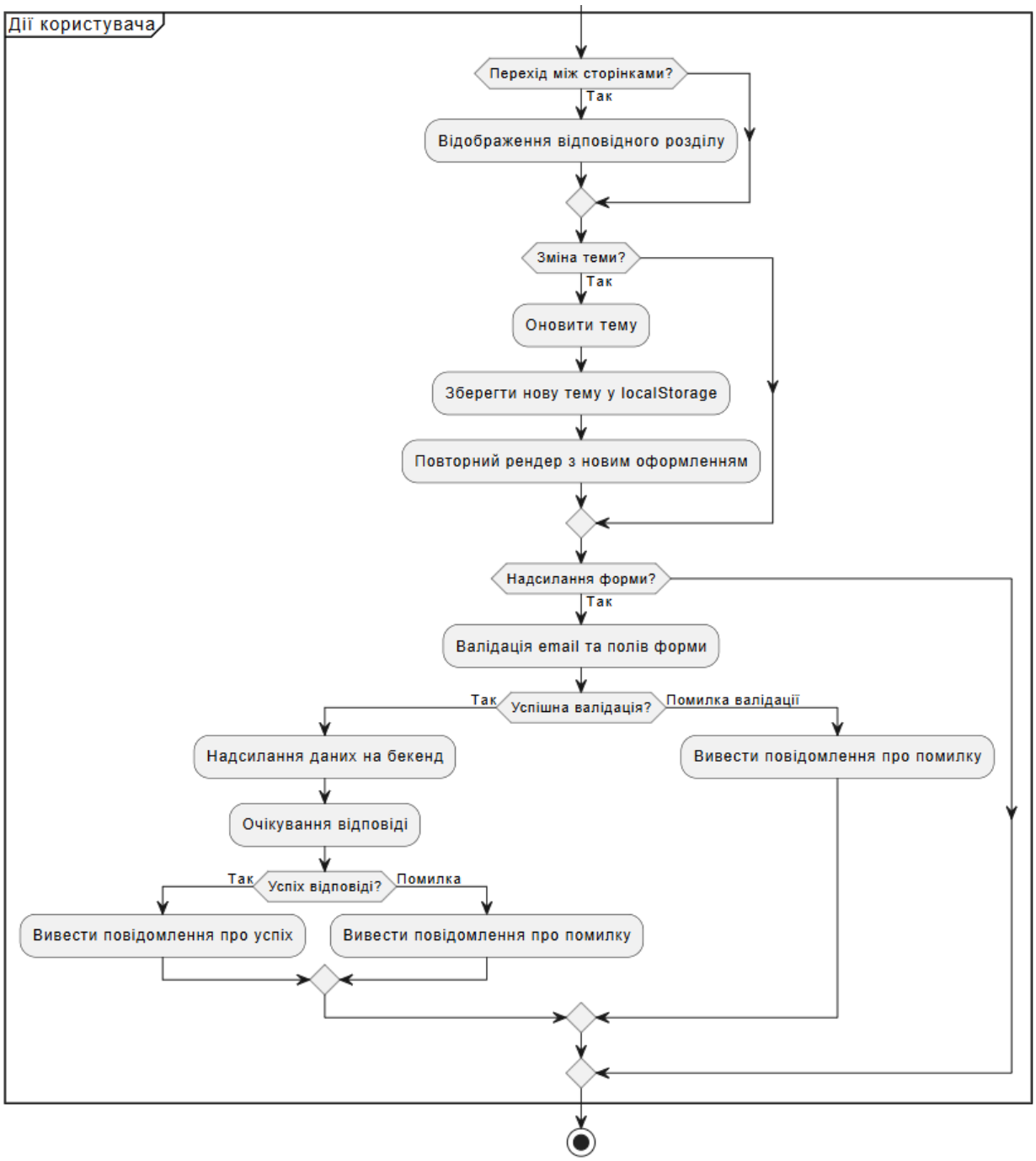

Рисунок 2.2.5 – Продовження схеми алгоритму розв’язання задачі

2.3 Моделювання інформаційної системи

Моделювання інформаційної системи є важливим етапом розробки сайту-портфоліо, що дозволяє формалізувати поведінку та структуру системи до початку її реалізації. Основу моделювання становить мова UML (див. [1], [2]),

яка забезпечує єдину візуальну мову опису різних аспектів системи, включаючи взаємодію користувача з інтерфейсом, логіку функціонування та компонування системних елементів.

Метод моделе-орієнтованого інжинірингу (Model-Driven Engineering, MDE) дає змогу описати архітектуру системи через набір узгоджених моделей, які далі використовуються для проектування, аналізу та валідації архітектурних рішень.

2.3.1. Моделювання поведінки системи

На діаграмі прецедентів відображено взаємодію основних акторів (користувач, рекрутер, розробник) із сайтом (див. [1]). Основні варіанти використання: перегляд інформації, перемикання теми, надсилання форми зворотного зв’язку, перегляд проєктів.

Рисунок 2.3.1 - Діаграма прецедентів

Для кожного з визначених прецедентів створено діаграми послідовності, які демонструють поетапну взаємодію об’єктів у межах системи. У випадку надсилання форми зворотного зв’язку послідовність виглядає так: користувач взаємодіє з формою, далі відбувається валідація введених даних, після чого запит надсилається на сервер і обробляється відповідь.

Під час перемикання теми натискання кнопки користувачем ініціює зміну теми, після чого новий стан зберігається у localStorage.

У процесі перегляду проєктів маршрутизатор спрямовує запит до відповідного компонента, який виконує рендеринг карток проєктів.

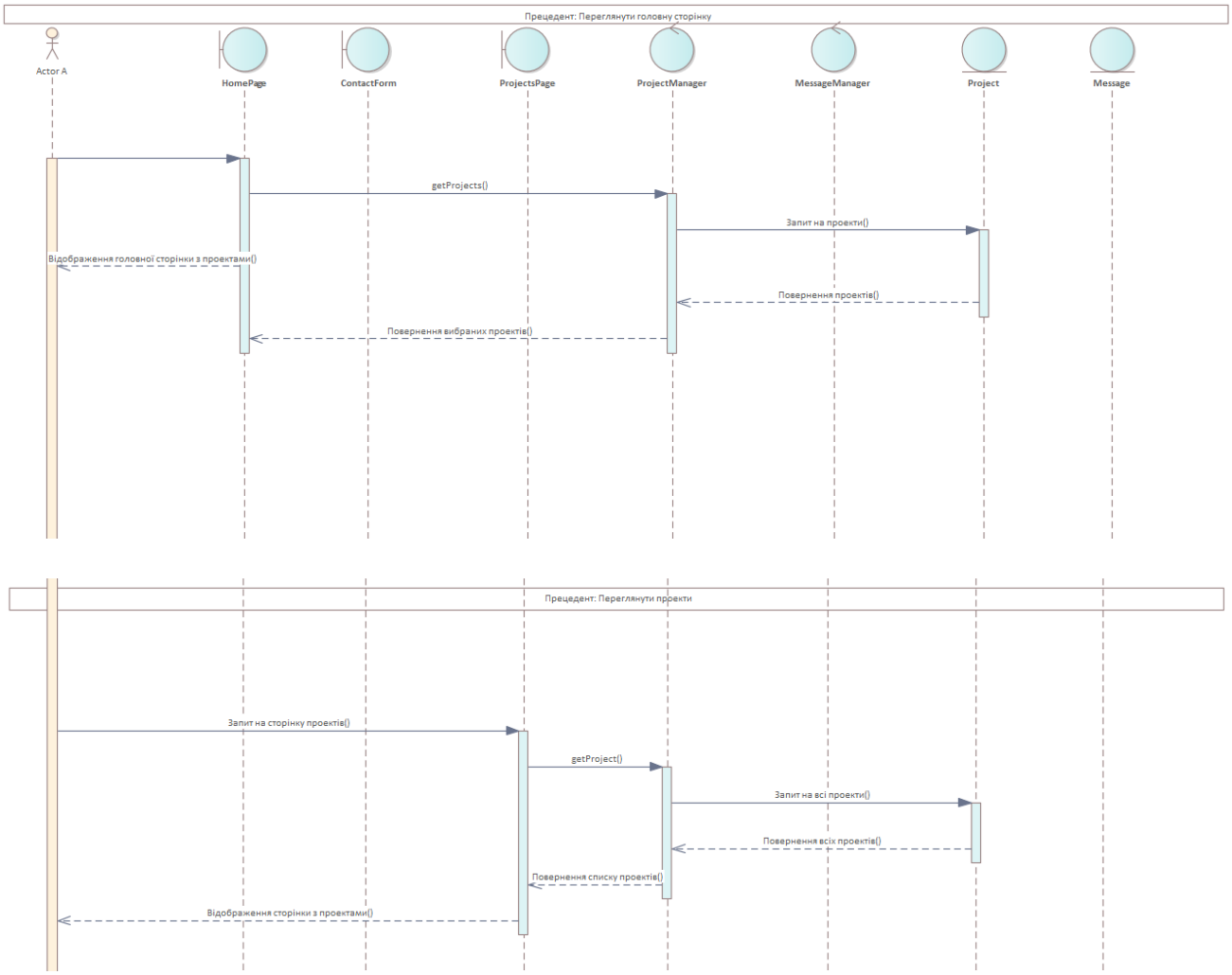

Рисунок 2.3.2.1 – Діаграма послідовності

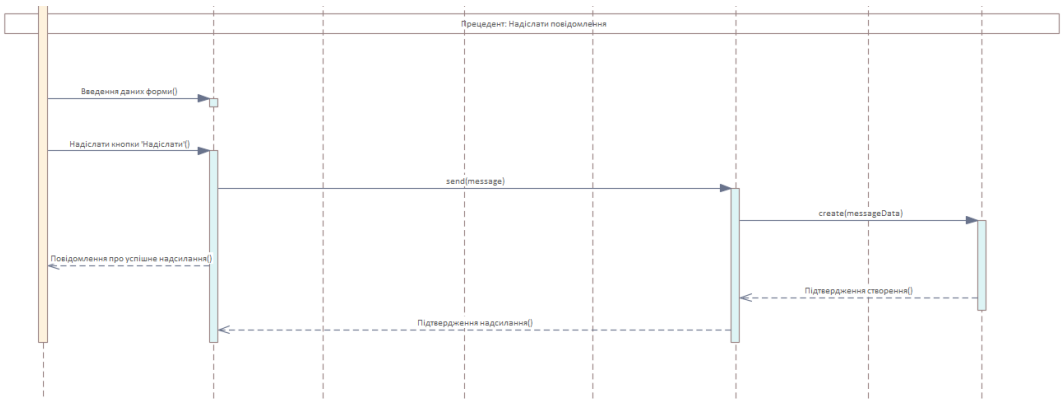

Рисунок 2.3.2.2 – Діаграма послідовності

Діаграма діяльності. У рамках загального процесу взаємодії з сайтом (від завантаження до відправки повідомлення) створено діаграму діяльності, що демонструє всі гілки поведінки користувача.

Рисунок 2.3.3 – Діаграма діяльності

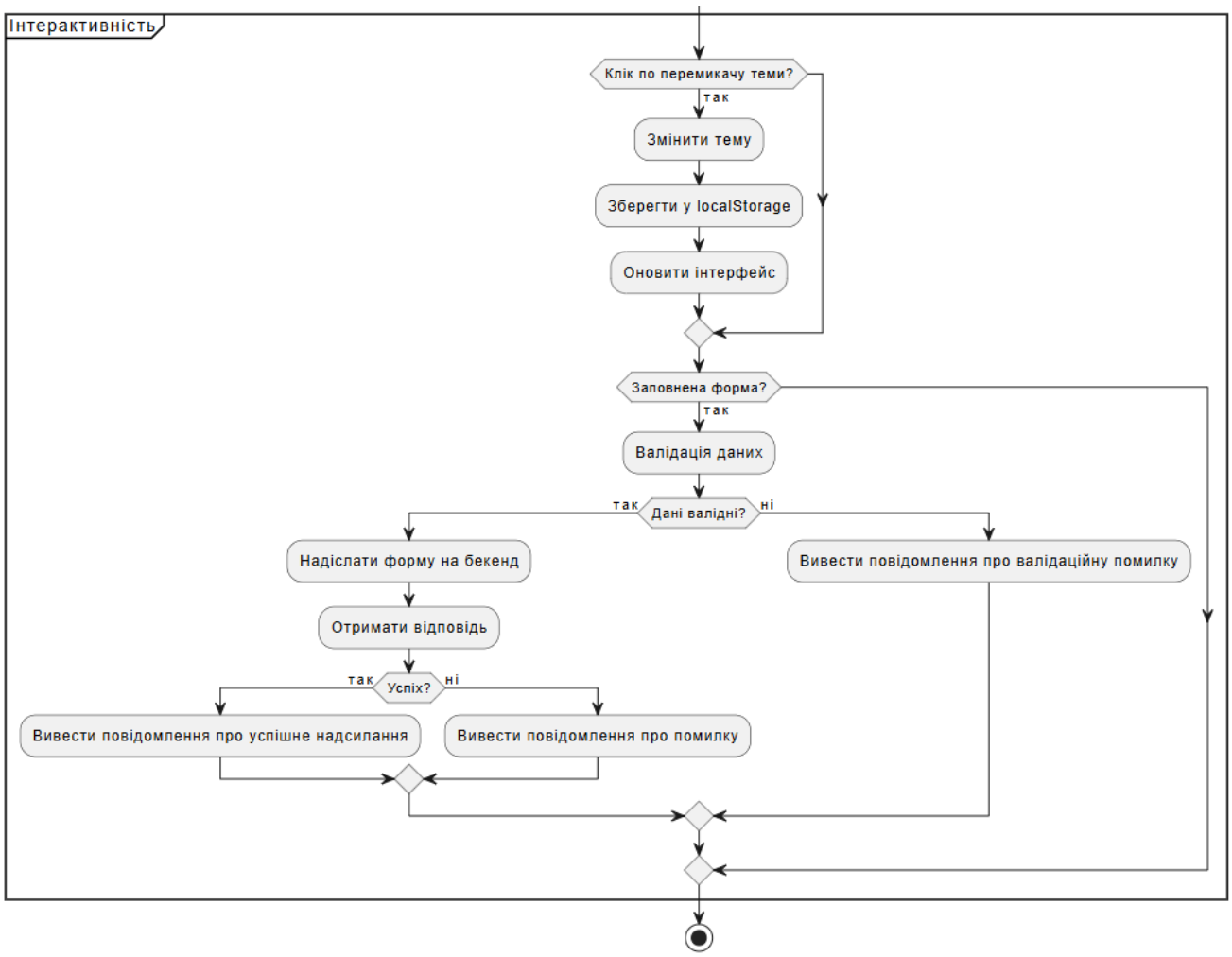

Рисунок 2.3.4 – Продовження діаграми діяльності

2.3.2. Моделювання структури системи

Модульна структура сайту реалізована за допомогою компонентів React. На діаграмі класів відображено:

- Класи-сутності: Project, Message, Experience, Skill
- Граничні класи (інтерфейс): ContactForm, AboutMePage, NavBar, ProjectPage, Footer, HomePage
- Класи-менеджери: MessageManager, UIController, ProjectManager

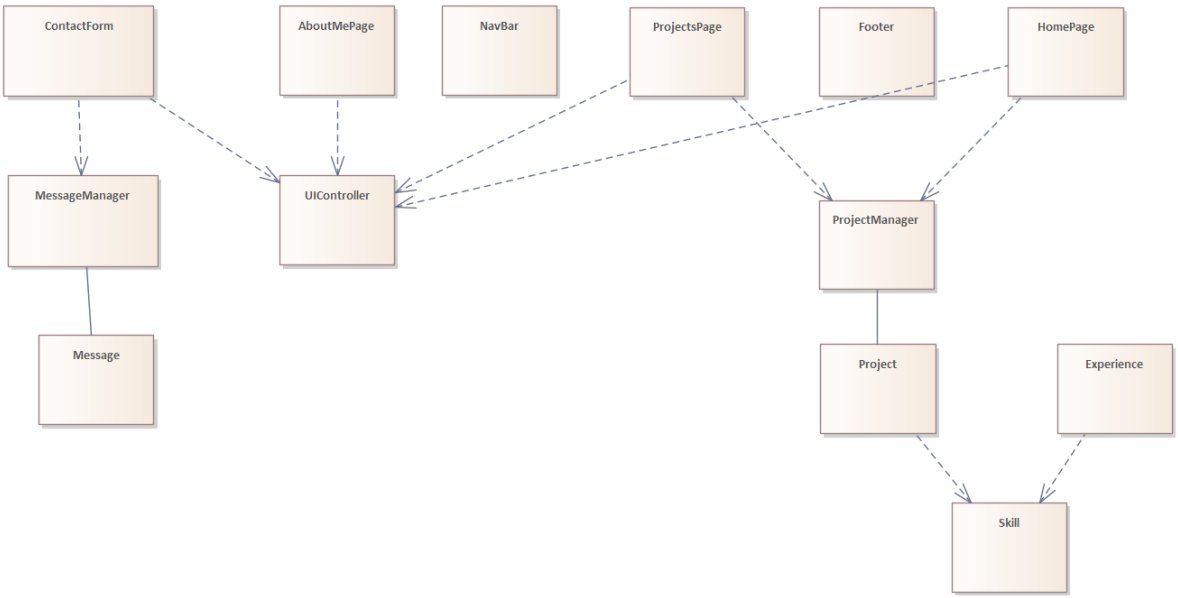

Рисунок 2.3.4 – Діаграма класів

2.3.3. Розподіл вимог за компонентами системи

На діаграмі трасування виконано зіставлення між бізнес-вимогами (BR1.1 – BR1.4), функціональними вимогами (FR1 – FR5) та компонентами реалізації:

- React-компоненти,
- менеджери,
- логіка форми,
- механізм теми).

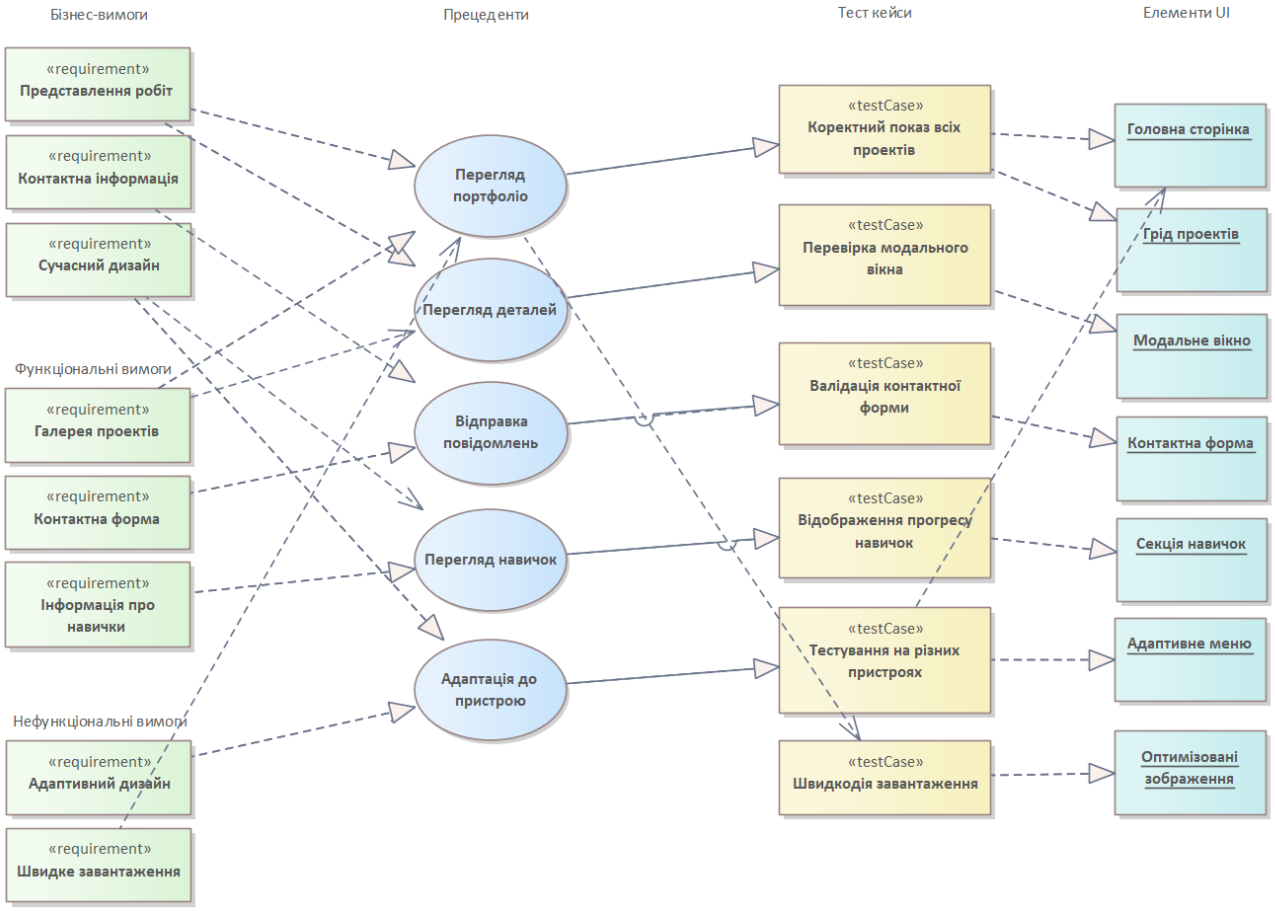

Рисунок 2.3.5 – Діаграма трасування вимог

Таблиця 2.3.1 – Таблиця відповідності вимог, прецедентів, тест-кейсів та елементів UI

Бізнес-вимоги	**Прецедент**	**Тест-кейс**	**Елемент UI**
Представлення робіт	Перегляд портфоліо	Коректний показ всіх проектів	Головна сторінка
Контактна інформація	Перегляд деталей	Перевірка контактного блоку	Футер проектів
Сумісний дизайн	Відгуки користувачів	Валідація адаптивної форми	Медіальний блок
Галерея проектів	Відгуки користувачів	Валідація адаптивної форми	Медіальний блок
Контактна форма	Перегляд навички	Відображення списку навичок	Блок навичок

Продовження таблиці 2.3.1

Бізнес-вимоги	**Прецедент**	**Тест-кейс**	**Елемент UI**
Інформація про навички	Перегляд навички	Відображення списку навичок	Блок навичок
Адаптація до пристроїв	Адаптація до пристроїв	Тестування на різних розмірах	Адаптивне меню
Швидкість завантаження	Адаптація до пристроїв	Тестування на різних розмірах	Адаптивне меню
-	Перегляд портфоліо	Коректний показ всіх проектів	Оптимізовані зображення
-	Перегляд деталей	Перевірка контактного блоку	-
-	-	Валідація адаптивної форми	Контактна форма
-	-	Відображення списку навичок	Список навичок
-	-	Відображення списку навичок	-

Таким чином, моделювання інформаційної підсистеми дозволило наочно представити логіку роботи сайту-портфоліо, визначити структуру та функціональні взаємозв’язки між його компонентами, що є важливим підґрунтям для наступного етапу – реалізації системи.

Матриці взаємозв'язків розширюють можливості діаграм трасування, створюючи організований формат для відображення зв'язків між компонентами проекту. Такі матриці сприяють упорядкуванню інформації та спрощують контроль за реалізацією вимог на всіх етапах розробки. Завдяки матрицям взаємозв'язків можна точно визначити шляхи верифікації вимог та забезпечити ефективне відстеження модифікацій у проекті.

Start Page | Requirements Traceability | Relationship Matrix

Source: Requirements Set Type: <All>

Target: Use Cases Set Type: <All>

Source \ Target	Use Cases::Адаптація до пристрою	Use Cases::Відправка повідомлень	Use Cases::Перегляд деталей	Use Cases::Перегляд навичок	Use Cases::Перегляд портфоліо
Requirements::Адаптивний дизайн	↑				
Requirements::Галерея проектів					↑
Requirements::Інформація про навички				↑	
Requirements::Контактна інформація		↑			
Requirements::Контактна форма		↑			
Requirements::Представлення робіт			↑		↑
Requirements::Сучасний дизайн	↑				
Requirements::Швидке завантаження					

Рисунок 2.3.6 – Матриця взаємозв'язків: Вимоги – Прецеденти

Start Page | Requirements Traceability | Relationship Matrix

Source: Use Cases Set Type: <All>

Target: Test Cases Set Type: <All>

Source \ Target	Test Cases::Валідація контактної форми	Test Cases::Відображення прогресу навичок	Test Cases::Коректний показ всіх проектів	Test Cases::Перевірка модального вікна	Test Cases::Тестування на різних пристроях	Test Cases::Швидкодія завантаження
Use Cases::Адаптація до пристрою					↑	
Use Cases::Відправка повідомлень	↑					
Use Cases::Перегляд деталей				↑		
Use Cases::Перегляд навичок		↑				
Use Cases::Перегляд портфоліо			↑			

Рисунок 2.3.7 – Матриця взаємозв'язків: Прецеденти - Тест-кейси

Start Page | Requirements Traceability | Relationship Matrix

Source: Test Cases | Set | Type: <All>

Target: Blocks | Set | Type: <All>

Source \ Target	Blocks::Адаптивне меню	Blocks::Головна сторінка	Blocks::Грід проектів	Blocks::Контактна форма	Blocks::Модальне вікно	Blocks::Оптимізовані зображення	Blocks::Секція навичок
Test Cases::Валідація контактної форми				↑			
Test Cases::Відображення прогресу навичок							↑
Test Cases::Коректний показ всіх проектів		↑	↑				
Test Cases::Перевірка модального вікна					↑		
Test Cases::Тестування на різних пристроях	↑						
Test Cases::Швидкодія завантаження						↑	

Рисунок 2.3.8 – Матриця взаємозв'язків: Тест-кейси - Елементи UI

РОЗДІЛ 3
ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ

3.1 Інформаційне забезпечення

Інформаційне забезпечення (ІЗ) системи охоплює структуровані вхідні та вихідні дані, їх зберігання, обробку та передачу в межах вебзастосунку «Сайт-портфоліо». ІЗ реалізовано з використанням локальних інформаційних масивів (JSON-структури, локальні state-змінні, об'єкти), а також клієнт-серверної взаємодії у форматі HTTP-запитів до backend-обробника.

Інформаційне забезпечення вебзастосунку розроблено з урахуванням кількох ключових характеристик. Його структура передбачає логічний поділ даних відповідно до функціональних блоків, таких як окремі секції сайту, теми оформлення та форми зворотного зв’язку. Уся інформація доступна безпосередньо з компонентів React, що забезпечує оперативність її обробки. Надійність функціонування гарантується через обов’язкову перевірку даних перед подальшими діями, зокрема здійснюється валідація полів, перевірка наявності обов’язкових значень і правильності формату, наприклад, для адрес електронної пошти (див. Таблицю 3.1.1).

Основним носієм користувацького введення в системі є електронна форма зворотного зв’язку, ескіз якої зображено на Рисунку 3.1.1. Вона виступає як первинний документ, створений під час роботи користувача із застосунком, і містить усю необхідну інформацію для подальшої обробки або надсилання.

Для зберігання інформації використовуються виключно електронні носії, серед яких – локальне сховище браузера (localStorage), внутрішні масиви даних (описані у Таблиці 3.1.2), а також програмно визначені точки доступу до API. У цьому застосунку відсутнє використання системи керування базами даних, оскільки інформація обробляється в оперативному режимі без потреби у

довготривалому збереженні.

Вхідні дані збираються в процесі взаємодії користувача із сайтом, зокрема під час заповнення форми зворотного зв’язку, зміни теми інтерфейсу або переходу між сторінками. Уся ця інформація надходить через інтерфейс і зберігається тимчасово у стані компонентів або у localStorage. Структуру масиву, що формується на основі цих даних, представлено у Таблиці 3.1.3. Для передавання даних на сервер використовується протокол HTTP, а саме POST-запити з тілами у форматі JSON.

Система класифікації інформації побудована відповідно до логіки застосунку. Дані кодуються умовними ключами, які описують стан або ідентифікатори елементів, наприклад, теми можуть позначатися як light або dark, статус форми – як valid, invalid, submitted, error, а розділи сайту – як home, about, projects, contact. Усі коди мають алфавітно-цифрову структуру довжиною не більше 20 символів.

Основним носієм користувацького введення в системі є електронна форма зворотного зв’язку. Вона виступає як первинний документ, створений під час роботи користувача із застосунком, і містить усю необхідну інформацію для подальшої обробки або надсилання.

Таблиця 3.1.1 - Опис форми зворотного зв’язку

Назва документа	Ідентифікатор	Користувачі	Періодичність	Термін обробки
Повідомлення з форми	FORM_MESSAGE	Адміністратор	У момент надсилання	Миттєво

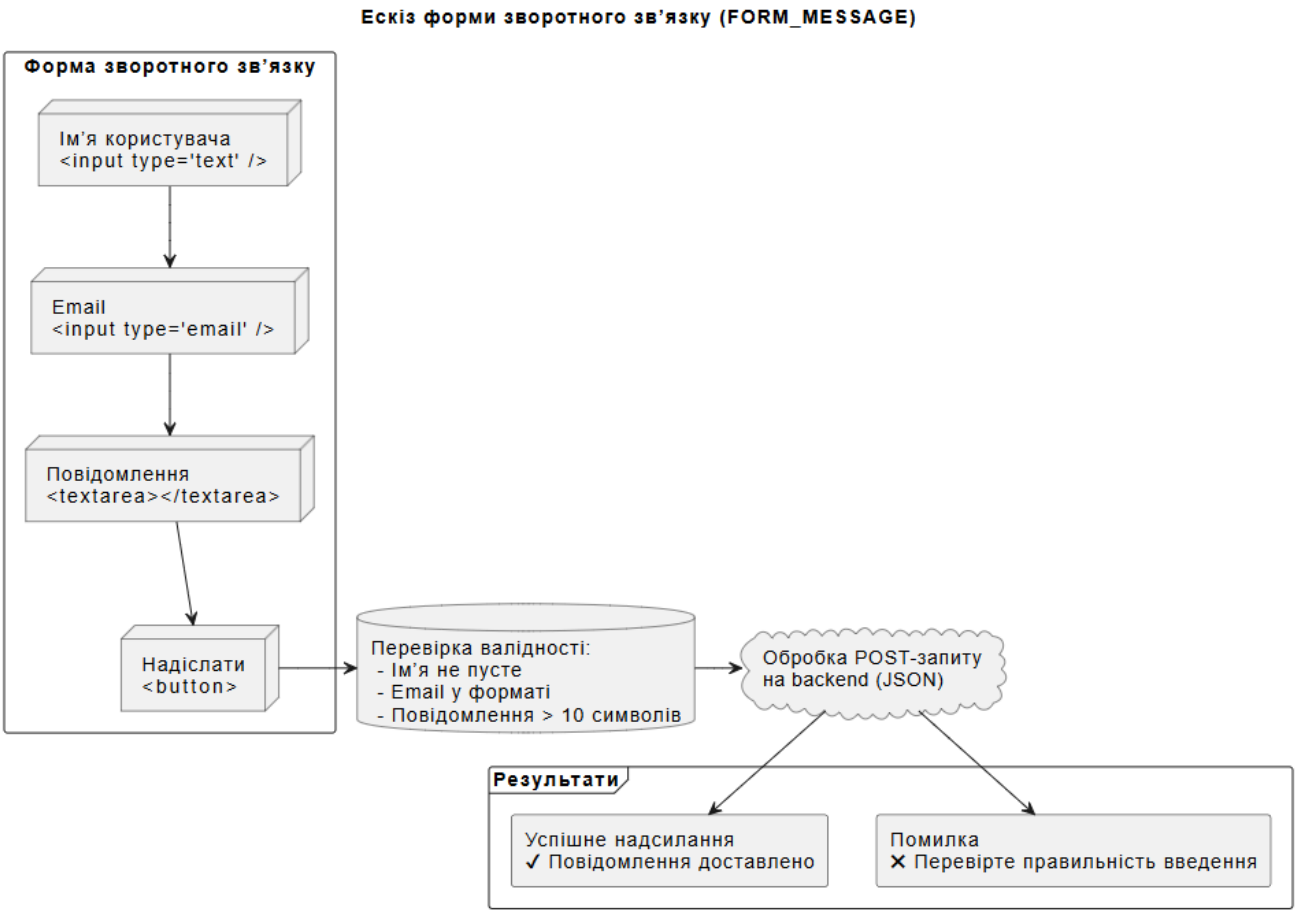

Рисунок 3.1.1 - Ескіз форми зворотного зв'язку

Структура інформаційних масивів

Таблиця 3.1.2 – Опис масиву

Найменування масиву	**Ідентифікатор**	**Носій інформації**	**Об'єм, записів**	**Довжина запису**	**Метод організації**
Повідомлення	MSG_LIST	Пам'ять React	До 100	До 500 байтів	Список (масив)

Таблиця 3.1.3 – Опис структури одного масиву (FORM_MESSAGE)

Найменування	**Ідентифікатор**	**Формат**	**Перв./втор. ключ**	**Обов. поле**	**Індекс**	**Умова значення**
Ім'я	name	string	–	так	–	≠ null
Email	email	string	–	так	ІНД	Valid Email
Повідомлення	message	string	–	так	–	≥ 10 символів
Статус	status	string	–	ні	–	"pending"/"sent"

У рамках даного проєкту реляційна СКБД не використовується, оскільки дані обробляються оперативно, не потребують довготривалого збереження та мають обмежений обсяг. Проміжне збереження теми здійснюється у localStorage, обробка форм – через REST API (Node.js backend), який може бути розширений з використанням MongoDB або Firebase у майбутньому.

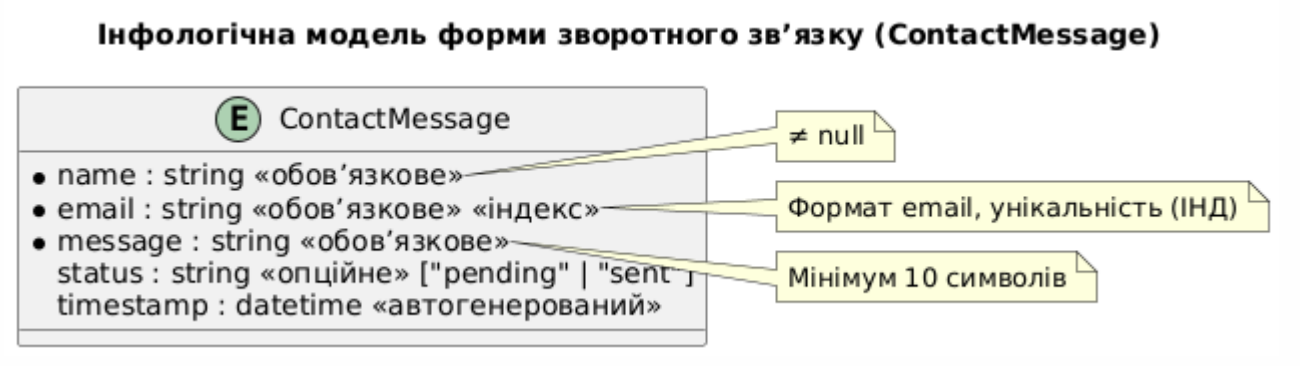

Рисунок 3.1.2 - Інфологічна модель бази (сховища) даних

Інфологічна модель відображає логічну структуру інформації, яка обробляється системою. У межах проєкту «Сайт-портфоліо» вона складається з однієї основної сутності – ContactMessage, що представляє повідомлення, надіслані користувачами через форму зворотного зв'язку.

Кожен об'єкт ContactMessage має наступні атрибути:

- name – ім'я користувача, який залишив повідомлення;
- email – електронна адреса користувача;
- message – текст повідомлення;
- timestamp – дата й час надсилання повідомлення;
- status – статус обробки повідомлення (pending, sent).

Модель відповідає принципам цілісності та нормалізації, що забезпечують уніфіковане представлення даних, їх однозначність і зручність у подальшій обробці. Інфологічна модель реалізована у вигляді структурованого JSON-об'єкта на стороні клієнта або серверної частини у випадку розширення до бази даних.

Рисунок 3.1.3 - Даталогічна модель бази (сховища) даних

У разі підключення бази даних (наприклад, MongoDB) структура може виглядати так:

```
{
  "name": "Frad Mayer",
  "email": "Fredmayer90@gmail.com",
  "message": "I’m interested in your work!",
  "timestamp": "2025-05-29T14:00:00Z",
  "status": "sent"
}
```

Даталогічна модель є фізичним представленням інфологічної моделі у вигляді таблиці для потенційного зберігання даних у реляційній базі (наприклад, MySQL або PostgreSQL). У межах цієї моделі таблиця ContactMessage має наступну структуру:

Таблиця 3.1.4 – Таблиця ContactMessage

Назва поля	**Тип даних**	**Обмеження**
id	INT	Первинний ключ, AUTO_INCREMENT
name	VARCHAR(100)	NOT NULL
email	VARCHAR(100)	NOT NULL, UNIQUE
message	TEXT	NOT NULL

Продовження таблиці 3.1.4

Назва поля	Тип даних	Обмеження
timestamp	DATETIME	NOT NULL, DEFAULT CURRENT_TIMESTAMP
status	ENUM('pending', 'sent')	DEFAULT 'pending'

Ця структура дозволяє зберігати повідомлення з форми у формалізованому вигляді для подальшого аналізу, фільтрації або інтеграції з адміністративною панеллю.

Дані можуть передаватися з клієнтської частини у форматі JSON через HTTP-запити на сервер, де вони можуть бути записані до бази даних за потреби. Поточна реалізація не вимагає використання СКБД, однак структура готова до масштабування за допомогою MongoDB, Firebase або SQL-рішень.

3.2. Технічне забезпечення

3.2.1 Загальні положення та схема автоматизації

Інформаційна підсистема «Сайт-портфоліо» функціонує як клієнт-серверна архітектура. Вона передбачає:

- використання віддаленого серверу (наприклад, хмарного сервісу або хостинг-провайдера, такого як Vercel або Render) для розміщення frontend і backend частин застосунку;
- доступ користувачів до сайту через браузер на ПК або мобільному пристрої з підключенням до мережі Інтернет.

Ядро системи (кодова база React-додатку) зберігається у віддаленому репозиторії (GitHub), а також задеплоєне на хостинг-платформі.

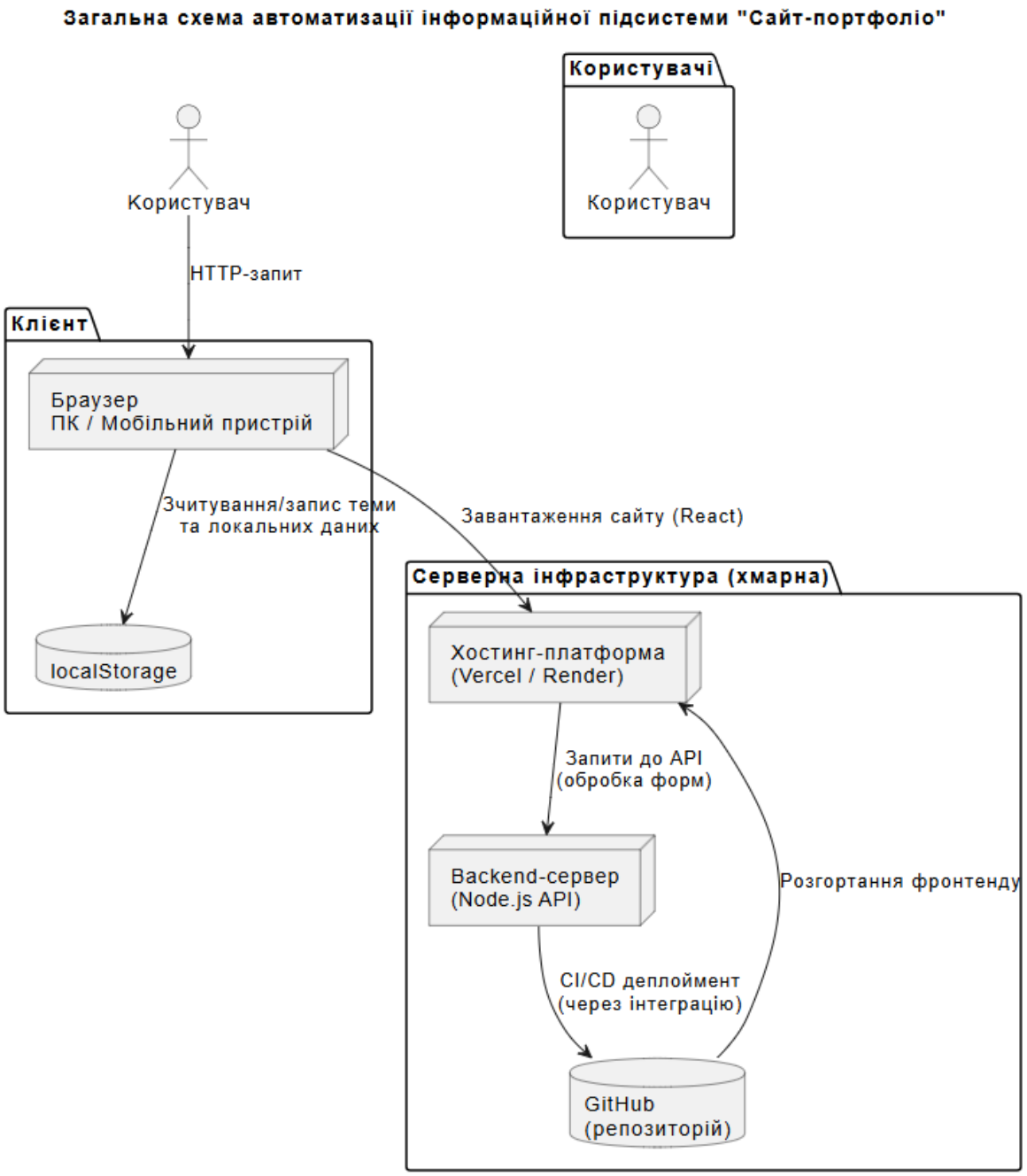

Рисунок 3.2.1 – Загальна схема автоматизації

3.2.2 Структура комплексу технічних засобів

Інформаційна система функціонує на стандартній інфраструктурі без потреби в спеціалізованих апаратних засобах:

- Серверна частина: Node.js сервер, запущений на хмарному середовищі (наприклад, Render або Railway) з мінімальними системними вимогами (1 CPU core, 512 МБ RAM, 1 ГБ SSD);
- Клієнтська частина: фронтенд (React SPA) доступний через браузери (Chrome, Firefox, Edge, Safari).

Оскільки система не потребує спеціального розміщення апаратних засобів, немає необхідності у фізичному обладнанні на об'єкті управління.

Захист реалізовано на програмному рівні: використання HTTPS-протоколу, обмеження запитів на сервер, перевірка форми з боку клієнта та сервера.

3.2.3 Опис автоматизованого робочого місця (АРМ)

Для взаємодії з системою користувачеві достатньо мати пристрій із доступом до Інтернету та сучасним браузером. Типове АРМ:

- Операційна система: Windows 10/11, Linux, macOS;
- Браузер: Google Chrome (рекомендований), Firefox, Edge;
- Процесор: Intel Core i3 або аналог;
- ОЗП: 4 ГБ;
- Дисковий простір: 500 МБ;
- Засоби введення/виведення: клавіатура, миша/тачпад, монітор;
- Засоби підключення: WiFi або Ethernet.

Для розробника: Visual Studio Code, Node.js, Git.

3.2.4 Схема мережі передачі даних

Система функціонує в умовах глобальної мережі. Доступ до сайту забезпечується через Інтернет-з’єднання.

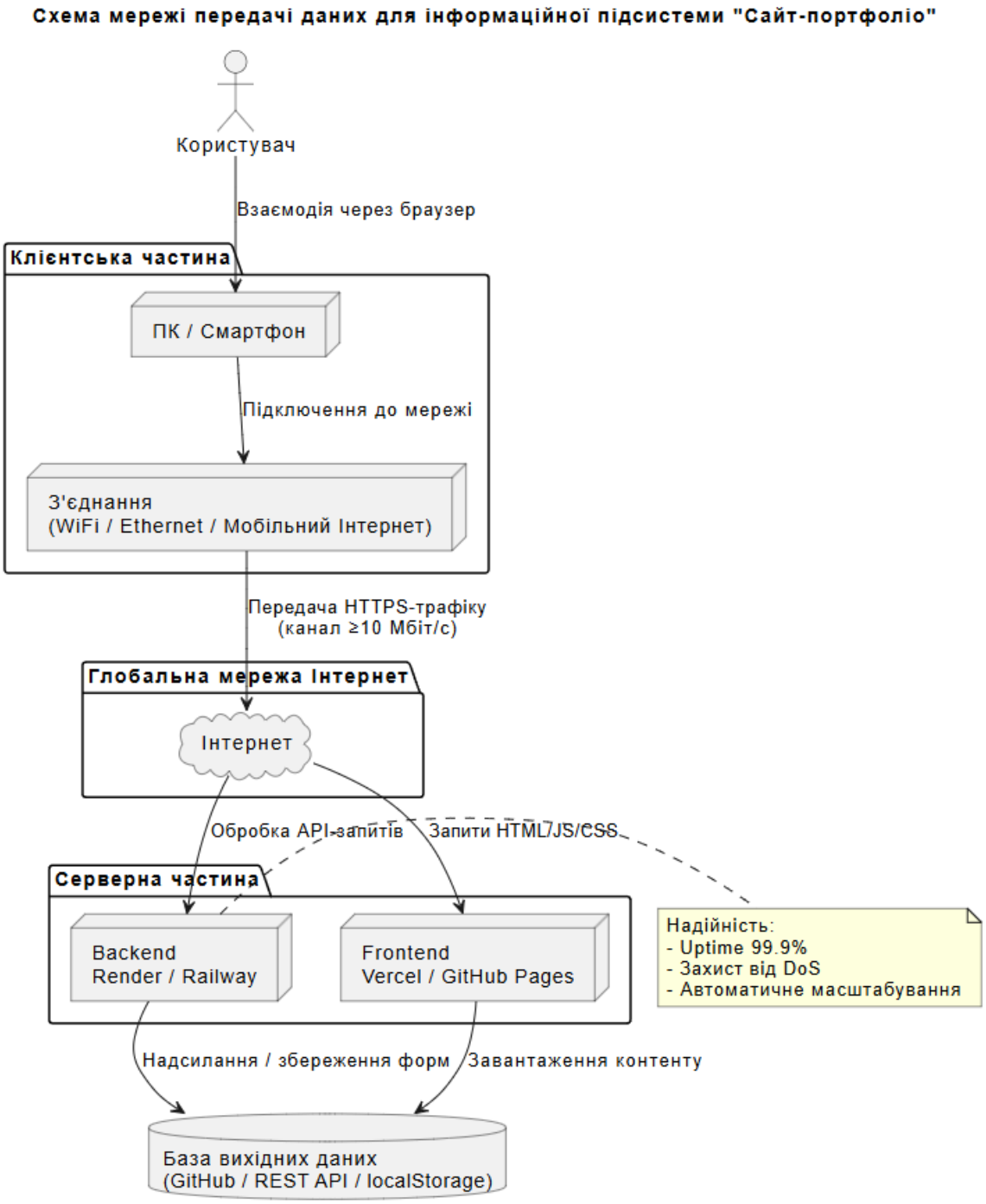

Рисунок 3.2.2 – Схема мережі передачі даних

Схема мережі передачі даних

Вибір засобів передачі:

- Підключення: WiFi / Ethernet / мобільний Інтернет;
- Протокол: HTTPS;
- Хостинг: Vercel / GitHub Pages для frontend; Render / Railway для backend;
- Канал зв’язку: мінімум 10 Мбіт/с (рекомендовано).

Надійність і підтримка:

- гарантія від провайдера хостингу (uptime 99.9%);
- мінімальне споживання ресурсів;
- автоматичне масштабування у разі підвищеного навантаження;

3.3. Програмне забезпечення

3.3.1 Структура програмного забезпечення

Програмне забезпечення сайту-портфоліо побудоване за принципом розділення на системне, прикладне та документальне. Його структура відображає всі компоненти, необхідні для створення, розгортання і підтримки інтерактивного вебзастосунку з можливістю перемикання теми (світлої/темної), а також формою зворотного зв’язку з передачею даних на сервер.

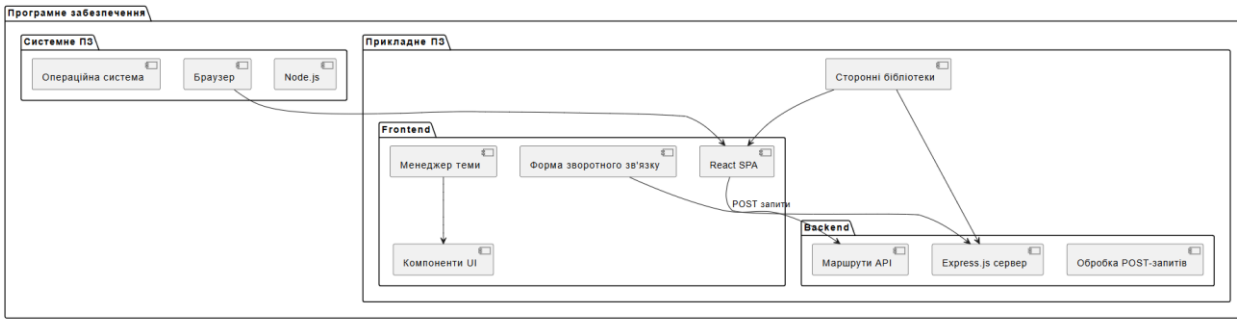

Рисунок 3.3.1 – Схема структури програмного забезпечення

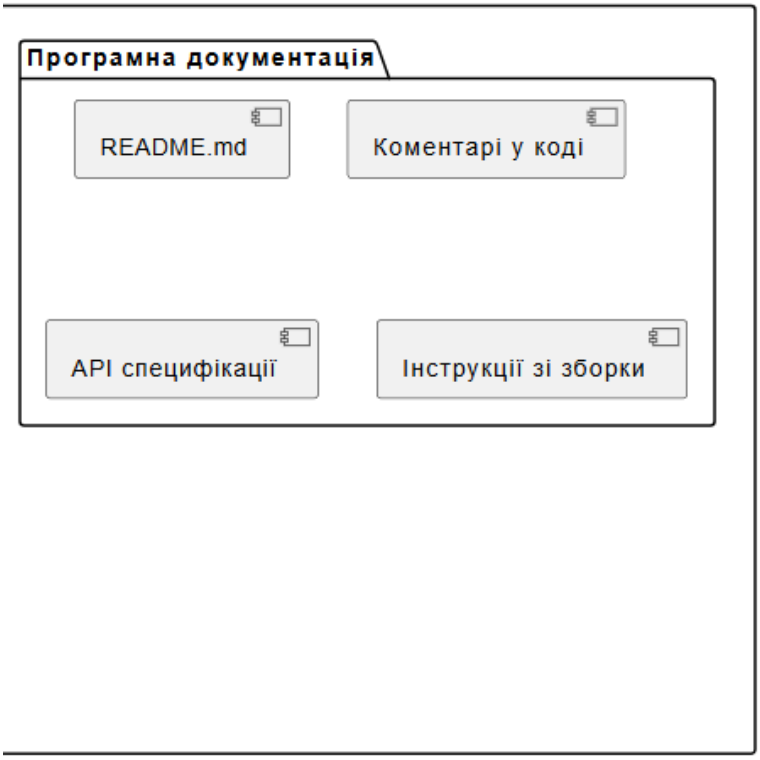

Рисунок 3.3.2 – Продовження схеми структури програмного забезпечення

Схема структури програмного забезпечення включає:

- Базове (системне) ПЗ: браузер, операційна система, Node.js;
- Прикладне ПЗ: frontend (React SPA), backend (сервер на Express.js), супровідні бібліотеки;
- Програмна документація: README, коментарі в коді, специфікації API, інструкції зі зборки та розгортання.

3.3.2 Системне програмне забезпечення.

У рамках реалізації системи використовувалося таке системне ПЗ:

- Операційна система: Windows 10+, Ubuntu 22.04 (на боці розробника та серверного хостингу);
- Браузери (клієнт): Google Chrome, Mozilla Firefox, Edge;
- Платформа розробки: Node.js (версія 18+);

- Пакетний менеджер: npm/yarn;
- CASE-засіб: draw.io для побудови UML-діаграм;
- Хостинг: Vercel (frontend) (див. [11]) та Render (backend);
- Системи контролю версій: Git + GitHub (див. [10]).

Цей стек було обрано через його відкритість, широке поширення в індустрії, наявність великої спільноти та підтримку компонентного підходу до розробки.

3.3.3 Прикладне програмне забезпечення

До прикладного ПЗ входять:

- React – JavaScript-бібліотека для побудови інтерфейсів, що дозволяє реалізувати SPA із динамічною навігацією (див. [9]);
- React Router – для маршрутизації між сторінками (розділами);
- Tailwind CSS – для реалізації адаптивного, мінімалістичного дизайну (див. [13]);
- EmailJS або власне API – для надсилання повідомлень з форми зворотного зв’язку;
- Formik + Yup – для обробки форм та валідації даних користувача;
- nodemailer або сторонній сервіс (наприклад, EmailJS) – для пересилання повідомлень з форми на email.

Функціональні можливості прикладного ПЗ охоплюють відображення основної інформації про користувача та його проєкти, перемикання між світлою та темною темами оформлення, інтерактивну форму зворотного зв’язку з повідомленням про результат надсилання, а також обробку відповідних запитів на серверній частині застосунку.

3.3.4 Програмна документація

До складу програмної документації проєкту входять матеріали, що забезпечують зрозуміле ознайомлення з функціональністю сайту та процесом його використання. У репозиторії на GitHub міститься файл README.md, який містить загальний опис функцій застосунку, вказівки щодо його встановлення та запуску, а також інформацію про необхідні системні вимоги.

Додатково підготовлено технічний формуляр, де стисло подано основні характеристики програмного продукту; цей документ розміщено в додатках до роботи. Для кінцевого користувача розроблено коротке керівництво, що пояснює, як користуватися сайтом: як його відкрити, перемикати тему оформлення та надсилати повідомлення через форму зворотного зв’язку.

Контрольний приклад подає зразок заповнення форми – повідомлення, надіслане під час тестування функціоналу з підтвердженням доставки. Для зручності навігації по кодовій базі надано структуру файлової системи проєкту з поясненням призначення основних каталогів, таких як /src/components, /src/pages, /src/styles тощо. Коментарі в коді супроводжують ключові частини логіки, включаючи функції, параметри компонентів (props), особливості рендерингу та маршрутизації, що сприяє легшому розумінню архітектури застосунку.

3.4. Результати реалізації інформаційної системи

В ході виконання бакалаврського дипломного проєкту було реалізовано повнофункціональний сайт-портфоліо розробника з використанням сучасних технологій JavaScript та React. В основу проєкту покладено оригінальні рішення, що вирізняють розробку серед типових реалізацій у цій галузі.

Однією з ключових особливостей стала підтримка перемикання між світлим і темним режимами оформлення, що дозволяє адаптувати інтерфейс під уподобання користувача та відповідає сучасним стандартам зручності та доступності в UX/UI-дизайні. Також було впроваджено інтерактивну форму зворотного зв'язку, яка напряму передає повідомлення на сервер, не вимагаючи використання сторонніх сервісів.

Сайт демонструє адаптивне компонування та логічну структуру розділів портфоліо, що забезпечує комфортне ознайомлення з матеріалами для потенційних роботодавців. Завдяки застосуванню React, оптимізації процесів рендерингу та впровадженню відкладеного завантаження (lazy loading), досягнуто високої продуктивності та швидкої роботи ресурсу.

Пілотне впровадження і тестування

Пілотне впровадження сайту було виконано шляхом публікації вебресурсу на платформі Vercel з прив'язкою до GitHub-репозиторію. Система була апробована шляхом тестового наповнення сторінки, надсилання тестових повідомлень через форму зворотного зв'язку, перемикання теми в різних браузерах та пристроях.

Контрольний приклад:

Користувач заповнив форму контактів, ввів ім'я, e-mail та повідомлення, після чого отримав підтвердження про успішне надсилання. Повідомлення оброблено backend-сервісом без помилок, а тема сайту була успішно збережена у localStorage.

Тестування показало, що функціонал працює коректно в браузерах Chrome, Firefox і Edge. Форма не відправляється без заповнення обов'язкових полів. Сайт адаптується до екранів із роздільною здатністю від 320px до 1920px. Надсилання повідомлень відбувається без збоїв із перевіркою правильності формату email.

Порівняно з іншими рішеннями, такими як шаблонні сайти-портфоліо на Wix чи WordPress, розроблений вебресурс вирізняється можливістю легкого масштабування та розширення функціоналу. Він повністю керується кодом

розробника без зовнішніх обмежень, не залежить від сторонніх бібліотек для основної логіки та дозволяє гнучко налаштовувати компоненти й інтеграції. Сайт є універсальним інструментом для особистого просування, а також може слугувати прикладом для демонстрації навичок при працевлаштуванні у сфері frontend-розробки.

Перспективи удосконалення

У майбутньому систему можна вдосконалити, додавши повноцінну базу даних, наприклад MongoDB, для зберігання повідомлень користувачів. Також можливе впровадження аналітики переглядів через інструменти на кшталт Google Analytics або Matomo. Крім того, розглядається інтеграція з Telegram-ботом для миттєвих сповіщень про нові повідомлення з сайту. Для оптимізації планується застосувати SSR або SSG, наприклад, за допомогою Next.js, а також додати підтримку кількох мов через i18n.

Готовність до реального використання

Сайт-портфоліо повністю готовий до використання як інструмент для:

- самопрезентації;
- пошуку роботи;
- демонстрації навичок розробки з використанням сучасних вебтехнологій.

Аналіз структури, логіки, тестування та особливостей реалізації підтверджує, що створена інформаційна підсистема є функціонально завершеною, з високим рівнем адаптивності та потенціалом для розширення.

ВИСНОВКИ

У процесі створення бакалаврської дипломної роботи було досягнуто поставленої мети – створено сучасний, адаптивний і функціональний сайт-портфоліо на основі JavaScript і бібліотеки React. Цей вебресурс відображає професійні навички розробника, забезпечує зручну навігацію та можливість зворотного зв’язку з користувачами.

На основі аналізу предметної галузі сформульовано вимоги до інформаційної підсистеми, визначено об’єкт і предмет дослідження, а також розроблено алгоритми для розв’язання поставленої задачі. Для реалізації обрано технології, які відповідають сучасним стандартам створення динамічних вебдодатків.

Сайт вирізняється низкою особливостей: можливістю перемикання між світлою та темною темами залежно від уподобань користувача, зручною формою зворотного зв’язку з обробкою повідомлень на сервері, адаптивним інтерфейсом, що коректно відображається на різних пристроях, а також структурованою подачею інформації через розділи «Про себе», «Навички», «Портфоліо» та «Контакти», орієнтовані на користувачів, рекрутерів і потенційних роботодавців.

У процесі розробки створено інформаційне, технічне та програмне забезпечення, а також проведено тестування системи за допомогою контрольних прикладів. Готовий проєкт розгорнуто на хостинг-платформі й підготовлено до використання в реальних умовах.

Розробка має високу практичну цінність, демонструє гнучкість і потенціал для масштабування, а також може слугувати основою чи прикладом для створення інших вебпортфоліо чи презентаційних сайтів.

Особистий внесок автора охоплює аналіз предметної галузі, формулювання задачі, створення структури сайту та його компонентів, програмну реалізацію функціоналу, розробку адаптивного дизайну з урахуванням принципів UX/UI, впровадження форми зворотного зв’язку з

серверною обробкою, а також функцію перемикання тем із збереженням налаштувань користувача.

У майбутньому систему можна вдосконалити шляхом додавання бази даних, підтримки кількох мов, аналітики відвідувань і розширення можливостей персоналізації інтерфейсу.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. ISO/IEC 19501:2005. Information technology – Open Distributed Processing – Unified Modeling Language (UML), Version 1.4. International Organization for Standardization, 2005.
2. ISO/IEC 19514:2017. Information technology – Object Management Group Systems Modeling Language (OMG SysML). International Organization for Standardization, 2017.
3. РД 50-34.698-90. Автоматизовані системи. Вимоги до змісту документів. – М.: Госстандарт, 1990.
4. Бублик М.І., Костюк А.Ф. Інформаційні системи і технології: навчальний посібник. – Київ: Центр учбової літератури, 2020. – 296 с.
5. Костромін О.Є. Розробка веб-додатків з використанням JavaScript та React.js: методичні вказівки до виконання курсових та дипломних робіт. – Київ: НАУ, 2022. – 82 с.
6. Гросс Дж., Максвелл А. React. Професійна розробка сучасних веб-застосунків. – Львів: Видавництво "База знань", 2021. – 448 с.
7. Wieruch R. The Road to React: Your journey to master React.js in JavaScript (2021 edition). – Independently published, 2021. – 250 p.
8. Mozilla Developer Network (MDN). JavaScript documentation. https://developer.mozilla.org/
9. ReactJS Documentation. Main official site. https://reactjs.org/
10. GitHub Documentation. Getting started with Git and GitHub. https://docs.github.com/
11. Vercel Documentation. Deployment of React apps. https://vercel.com/docs
12. EmailJS Documentation. Send email from JavaScript. https://www.emailjs.com/docs/
13. Tailwind CSS Documentation. https://tailwindcss.com/docs

14. Node.js Documentation. https://nodejs.org/en/docs/

15. Express.js Guide. https://expressjs.com/

16. OWASP Foundation. OWASP Top 10: Web Application Security Risks. https://owasp.org/www-project-top-ten/

17. Bootstrap Documentation. https://getbootstrap.com/docs/

18. Netlify Documentation. https://docs.netlify.com/

19. Render Documentation. https://render.com/docs

20. Railway Documentation. https://docs.railway.app/

21. Firebase Documentation. https://firebase.google.com/docs

22. MongoDB Manual. https://www.mongodb.com/docs/manual/

23. Form Validation in React. https://react-hook-form.com/

24. JSONPlaceholder – Free Fake API for Testing and Prototyping. https://jsonplaceholder.typicode.com/

25. UXPin Blog. Principles of Effective User Interface Design. https://www.uxpin.com/studio/blog/

26. Google Fonts. Web Typography Resources. https://fonts.google.com/

27. Lighthouse Performance Auditing Tool. https://developer.chrome.com/docs/lighthouse/

28. Can I use – Browser support tables for modern web technologies. https://caniuse.com/

29. Stack Overflow Developer Survey 2024. https://survey.stackoverflow.co/2024/

30. GitHub Pages Documentation. https://pages.github.com/

31. Create React App Documentation. https://create-react-app.dev/docs/getting-started/

ДОДАТКИ

Додаток А

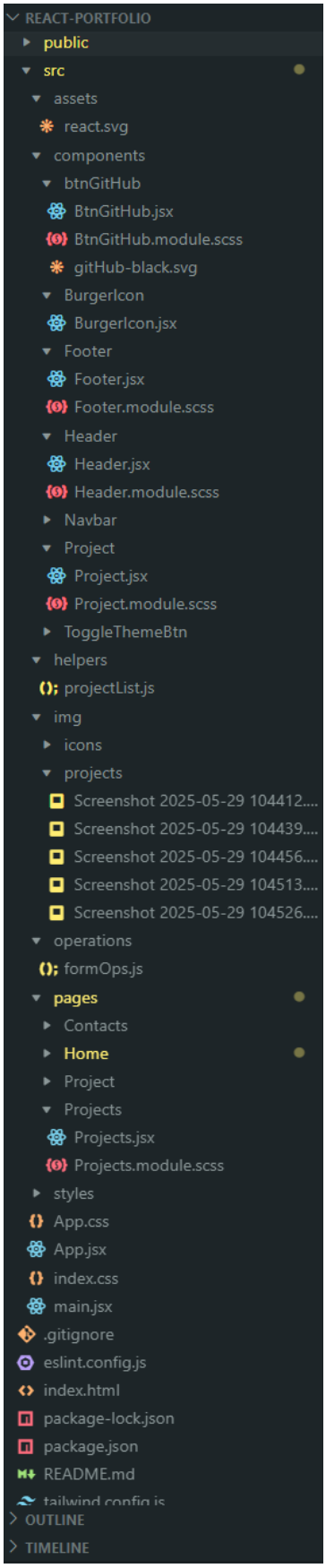

Рисунок А.1 – Структура проекту

src > components > btnGitHub > BtnGitHub.jsx > ...

```jsx
import gitHubIcon from "./gitHub-black.svg";
import s from "./BtnGitHub.module.scss";

const BtnGitHub = ({ link }) => {
  return (
    <a href={link} target="_blank" className={s.btnOutline}>
      <img src={gitHubIcon} alt="" />
      GitHub repo
    </a>
  );
};

export default BtnGitHub;
```

Рисунок А.2 – Компонент BtnGitHub

src > components > BurgerIcon > BurgerIcon.jsx > BurgerIcon

```
1   import clsx from "clsx";
2   import s from "../Navbar/Navbar.module.scss";
3   import { NavLink } from "react-router-dom";
4   const buildLinkClass = ({ isActive }) => {
5     return clsx(
6       s.navList__link,
7
8       isActive && s.navList__link_active
9     );
10  };
11  const BurgerIcon = () => {
12    return (
13      <div className="sm:hidden dropdown dropdown-center">
14        <div tabIndex={0} role="button" className="btn btn-ghost btn-circle">
15          <svg
16            xmlns="http://www.w3.org/2000/svg"
17            className="h-5 w-5 stroke-black dark:stroke-white"
18            fill="none"
19            viewBox="0 0 24 24"
20            stroke="currentColor"
21          >
22            {" "}
23            <path
24              strokeLinecap="round"
25              strokeLinejoin="round"
26              strokeWidth="2"
27              d="M4 6h16M4 12h16M4 18h7"
28            />{" "}
29          </svg>
30        </div>
31        <ul
32          tabIndex={0}
33          className="menu menu-sm dropdown-content  rounded-box z-1 mt-3 w-20 p-2 shadow bg-white d
34        >
35          <li>
36            <NavLink to="/" className={buildLinkClass}>
37              Home
38            </NavLink>
39          </li>
40          <li>
41            <NavLink to="/projects" className={buildLinkClass}>
42              Projects
43            </NavLink>
44          </li>
45          <li>
46            <NavLink to="/contacts" className={buildLinkClass}>
47              Contacts
48            </NavLink>
49          </li>
50        </ul>
51      </div>
52    );
53  };
54
55  export default BurgerIcon;
```

Рисунок А.3 – Компонент BurgerIcon

```
src > components > Footer > Footer.jsx > ...
1  import { FaGithub, FaLinkedin, FaTelegramPlane } from "react-icons/fa";
2
3  const Footer = () => {
4    return (
5      <footer className="bg-zinc-200 dark:bg-zinc-900 py-[60px] pb-[50px]">
6        <div className="text-zinc-900 dark:text-white max-w-[1200px] mx-auto px-4">
7          <div className="flex flex-col items-center gap-[27px]">
8            <ul className="flex items-center gap-[30px] max-[620px]:gap-[20px]">
9              <li className="w-[32px] max-[620px]:w-[28px]">
10               <a href="https://github.com/aleksey-pru">
11                 <FaGithub size={38} />
12               </a>
13             </li>
14             <li className="w-[32px] max-[620px]:w-[28px]">
15               <a href="https://www.linkedin.com/in/oleksii-pruzhanskyi-07231a205">
16                 <FaLinkedin size={38} />
17               </a>
18             </li>
19             <li className="w-[32px] max-[620px]:w-[28px]">
20               <a href="https://t.me/soovven">
21                 <FaTelegramPlane size={38} />
22               </a>
23             </li>
24           </ul>
25           <div className="text-[16px] text-center">
26             <p>© 2025 Oleksii. All rights reserved.</p>
27           </div>
28         </div>
29       </div>
30     </footer>
31   );
32 };
33
34 export default Footer;
```

Рисунок А.4 – Компонент Footer

src > components > Header > Header.jsx > Header

```jsx
const Header = () => {
  const el = useRef(null);
  const secondRef = useRef(null);

  useEffect(() => {
    const firstTyped = new Typed(el.current, {
      strings: ["Hi, my name is <i>Oleksii</i>"],
      typeSpeed: 50,
      showCursor: false,
      onComplete: () => {
        new Typed(secondRef.current, {
          strings: ["a frontend developer"],
          typeSpeed: 50,
          showCursor: true,
        });
      },
    });

    return () => {
      firstTyped.destroy();
    };
  }, []);
  return (
    <header className="min-h-screen flex items-center justify-center text-black bg-[radial-gradient(
      <div className="text-zinc-900 dark:text-white px-4 max-w-[660px] text-center">
        <div className="avatar">
          <div className="ring-black dark:ring-white ring-offset-base-100 w-34 rounded-full ring-
            <img src="/photo_2025-05-27_15-25-04.jpg" alt="" />
          </div>
        </div>
        <h1 className="mb-5 text-4xl leading-snug">
          <span ref={el}></span>
          <br />
          <span ref={secondRef}></span>
        </h1>
        <div className="mb-10 text-lg leading-[1.333]">
          <p>with passion for learning and creating.</p>
        </div>
        <div className="inline-block">
          {/* <button className="btn btn-neutral btn-outline">Download CV</button> */}
          <a href="/cv.pdf" download className="btn btn-neutral btn-outline">
            Download CV
          </a>
        </div>
        <div className="divider"></div>

        <section>
          <h1 className="mb-5 text-4xl leading-snug">About Me</h1>
          <p>
            Frontend Developer with a strong foundation in HTML, CSS,
            JavaScript, and modern web technologies. Experienced in creating
            responsive and adaptive web applications, with a focus on clean
            design and usability. Skilled in Git, REST APIs, and Vite, and
            actively expanding knowledge in React.js, TypeScript, and Node.js
            through the GoIT FullStack Developer program. Demonstrated ability
            to work in both team-based and independent projects, contributing to
            the development of landing pages and interactive UI components.
            Strong analytical skills, problem-solving mindset, and adaptability.
          </p>
        </section>
      </div>
    </header>
```

Рисунок А.5 – Компонент Header

src > components > Navbar > Navbar.jsx > buildLinkClass

```
import s from "./Navbar.module.scss";
import { NavLink } from "react-router-dom";
import clsx from "clsx";
import ToggleThemeBtn from "../ToggleThemeBtn/ToggleThemeBtn";
import BurgerIcon from "../BurgerIcon/BurgerIcon";

const buildLinkClass = ({ isActive }) => {
  return clsx(
    s.navList__link,
    "text-gray-900 dark:text-white",
    isActive && s.navList__link_active
  );
};

const Navbar = () => {
  return (
    <nav className=" bg-white dark:bg-zinc-900 border-nav-border text-nav-text tracking-normal">
      <div className=" mx-auto p-5 max-w-screen-xl">
        <div className="flex justify-end items-center gap-x-7 gap-y-5 flex-wrap">
          <NavLink
            to="/"
            className="mr-auto text-gray-900 dark:text-white text-2xl font-poppins"
          >
            <strong>Freelancer</strong> portfolio
          </NavLink>
          <BurgerIcon />
          <ToggleThemeBtn />

          <div className="max-sm:hidden">
            <ul className="flex flex-wrap items-center gap-y-[10px] gap-x-[40px] text-[16px] font-medi
              <li className=" text-gray-900 dark:text-white">
                <NavLink to="/" className={buildLinkClass}>
                  Home
                </NavLink>
              </li>
              <li className=" text-gray-900 dark:text-white">
                <NavLink to="/projects" className={buildLinkClass}>
                  Projects
                </NavLink>
              </li>
              <li className=" text-gray-900 dark:text-white">
                <NavLink to="/contacts" className={buildLinkClass}>
                  Contacts
                </NavLink>
              </li>
            </ul>
          </div>
        </div>
      </div>
    </nav>
  );
};

export default Navbar;
```

Рисунок А.6 – Компонент NavBar

src > components > Project > Project.jsx > Project

```
1   import { Link } from "react-router-dom";
2
3   const Project = ({ title, img, id }) => {
4     return (
5       <div className="relative w-96 overflow-hidden rounded-xl shadow-md group bg-zinc-200 dark:bg-
6         <div className="absolute inset-0 bg-gradient-to-r from-zinc-400 via-gray-200 to-pink-50 d
7         <Link to={`/project/${id}`} className="relative z-10 block">
8           <figure>
9             <img src={img} alt={title} className="w-full" />
10          </figure>
11          <div className="card-body">
12            <h2 className="card-title text-zinc-900 dark:text-white">{title}</h2>
13          </div>
14        </Link>
15      </div>
16    );
17  };
18
19  export default Project;
20
```

Рисунок А.7 – Компонент Project

```
src > components > ToggleThemeBtn > ToggleThemeBtn.jsx > ToggleThemeBtn
1  import { useEffect, useState } from "react";
2  const ToggleThemeBtn = () => {
3    const [isDarkMode, setIsDarkMode] = useState(() => {
4      if (typeof window !== "undefined") {
5        return (
6          localStorage.theme === "dark" ||
7          (!("theme" in localStorage) &&
8            window.matchMedia("(prefers-color-scheme: dark)").matches)
9        );
10     }
11     return false;
12   });
13
14   useEffect(() => {
15     const root = window.document.documentElement;
16
17     if (isDarkMode) {
18       root.classList.add("dark");
19       localStorage.setItem("theme", "dark");
20     } else {
21       root.classList.remove("dark");
22       localStorage.setItem("theme", "light");
23     }
24   }, [isDarkMode]);
25
26   const toggleDarkMode = () => setIsDarkMode((prev) => !prev);
27   return (
28     <div>
29       <label className="text-gray-900 dark:text-white flex items-center cursor-pointer gap-2">
30         <svg
31           xmlns="http://www.w3.org/2000/svg"
32           width="20"
33           height="20"
34           viewBox="0 0 24 24"
35           fill="none"
36           stroke="currentColor"
37           strokeWidth="2"
38           strokeLinecap="round"
39           strokeLinejoin="round"
40         >
41           <circle cx="12" cy="12" r="5" />
42           <path d="M12 1v2M12 21v2M4.2 4.2l1.4 1.4M18.4 18.4l1.4 1.4M1 12h2M21 12h2M4.2 19.8l1.4-1.4M1
43         </svg>
44         <input
45           onChange={toggleDarkMode}
46           checked={isDarkMode}
47           type="checkbox"
48           value="synthwave"
49           className="toggle theme-controller text-gray-900 dark:text-white"
50         />
51         <svg
52           xmlns="http://www.w3.org/2000/svg"
53           width="20"
54           height="20"
55           viewBox="0 0 24 24"
56           fill="none"
57           stroke="currentColor"
58           strokeWidth="2"
59           strokeLinecap="round"
```

Рисунок А.8 – Компонент ToggleThemeBtn

src > helpers > projectList.js > projects

```
1   import project01 from "./../img/projects/Screenshot 2025-05-29 104412.png";
2   import project01Big from "./../img/projects/Screenshot 2025-05-29 104412.png";
3   import project02 from "./../img/projects/Screenshot 2025-05-29 104439.png";
4   import project02Big from "./../img/projects/Screenshot 2025-05-29 104439.png";
5   import project03 from "./../img/projects/Screenshot 2025-05-29 104456.png";
6   import project03Big from "./../img/projects/Screenshot 2025-05-29 104456.png";
7   import project04 from "./../img/projects/Screenshot 2025-05-29 104513.png";
8   import project04Big from "./../img/projects/Screenshot 2025-05-29 104513.png";
9   import project05 from "./../img/projects/Screenshot 2025-05-29 104526.png";
10  import project05Big from "./../img/projects/Screenshot 2025-05-29 104526.png";
11  import { v4 as uuidv4 } from "uuid";
12
13  export const projects = [
14    {
15      id: uuidv4(),
16      title: "BOSE Project",
17      skills: "HTML,CSS",
18      img: project01,
19      imgBig: project01Big,
20      gitHubLink: "https://github.com/aleksey-pru/project",
21      desc: "Landing for music product presentation. Landing with responsive design that displays equally well on al
22    },
23    {
24      id: uuidv4(),
25      title: "Watchcharm",
26      img: project02,
27      imgBig: project02Big,
28      skills: "HTML,CSS",
29      gitHubLink: "https://github.com/Syvyj/team-04/tree/gh-pages",
30      desc: "Landing with a responsive layout. Team project. Role: Developer. Was responsible for Reviews section",
31    },
32    {
33      id: uuidv4(),
34      title: "WebStudio Landing",
35      img: project03,
36      imgBig: project03Big,
37      skills: "HTML,CSS",
38      gitHubLink: "https://github.com/aleksey-pru/goit-markup-hw-06",
39      desc: "Pet-project. Responsive layout project with a more complex structure.",
40    },
41    {
42      id: uuidv4(),
43      title: "Portfolio Project",
44      img: project04,
45      imgBig: project04Big,
46      skills: "HTML, CSS, JavaScript, REST API, Vite",
47      gitHubLink: "https://github.com/aleksey-pru/team-04-js",
48      desc: "Team project. Role: Team-lead & Developer. Led a team of 9 developers, managed workflows, and was respo
49    },
50    {
51      id: uuidv4(),
52      title: "Movie Search App",
53      img: project05,
54      imgBig: project05Big,
55      skills: "HTML, CSS, JavaScript, React, REST API, Vite",
56      desc: "Built a responsive movie search application that allows users to find films by keyword using a public A
57    },
58  ];
```

Рисунок А.9 – JSON-файл з даними про проекти

src > operations > formOps.js > addInfoFromForm

```
1   import axios from "axios";
2   import toast from "react-hot-toast";
3   axios.defaults.baseURL = "https://6814732b225ff1af1628de21.mockapi.io/";
4
5   export const addInfoFromForm = async ({ email, message }) => {
6     try {
7       const response = await axios.post("/form_feedback", { email, message });
8       toast.success("Form successfully submitted");
9       return response.data;
10    } catch (error) {
11      toast.error("Sorry, request error!");
12      console.log(error);
13    }
14  };
15
```

Рисунок А.10 – POST запит на базу даних з інформацією з форми

```
src > pages > Contacts > Contacts.jsx > Contacts
import { BsSendFill } from "react-icons/bs";
import { addInfoFromForm } from "../../operations/formOps";
const Contacts = () => {
  const handleSubmit = (e) => {
    e.preventDefault();
    const form = e.target;
    const { email, message } = form.elements;
    const newValues = {
      email: email.value,
      message: message.value,
    };
    addInfoFromForm(newValues);
    form.reset();
  };
  return (
    <main className=" bg-zinc-200 dark:bg-zinc-900 py-[70px] flex ">
      <div className=" text-zinc-900 dark:text-white w-md mx-auto px-4">
        <h2 className="mb-[60px] text-[50px] font-bold leading-[1.3] text-[var(--title-1)] text-center
          Contact Me
        </h2>

        <section className=" mb-20 sm:mb-28 w-[min(100%,38rem)] text-center">
          <form
            className="mt-10 flex flex-col dark:text-black"
            onSubmit={handleSubmit}
            action=""
          >
            <input
              type="email"
              name="email"
              placeholder="Your email"
              className="border-2 h-14 px-4 rounded-lg borderBlack dark:bg-white dark:bg-opacity-80
            />
            <textarea
              name="message"
              placeholder="Your message"
              className="border-2 h-52 my-3 rounded-lg borderBlack p-4 dark:bg-white dark:bg-opacity
              id=""
            ></textarea>
            <button
              type="submit"
              className="group flex items-center justify-center gap-2 h-[3rem] w-[8rem] bg-gray-900
            >
              Submit
              <BsSendFill className="text-m opacity-70 transition-all group-hover:translate-x-1 group-
            </button>
          </form>
        </section>
      </div>
    </main>
  );
};

export default Contacts;
```

Рисунок А.11 – Сторінка Contacts

src > pages > Home > Home.jsx > ...

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

const Home = () => {
  return (
    <>
      <Header />

      <main className=" bg-zinc-200 dark:bg-zinc-900 py-[70px]  flex flex-col items-center gap-[4
        <div className=" mb-[20px] text-[40px] font-bold leading-[1.3] text-zinc-900 dark:text-wh
          My Skills
        </div>
        <ul className="max-w-2xl flex flex-wrap justify-center gap-2 text-lg text-gray-800">
          <li className=" bg-white borderBlack rounded-xl px-5 py-3 dark:bg-white/10 dark:text-w
            HTML
          </li>
          <li className=" bg-white borderBlack rounded-xl px-5 py-3 dark:bg-white/10 dark:text-w
            CSS
          </li>
          <li className=" bg-white borderBlack rounded-xl px-5 py-3 dark:bg-white/10 dark:text-w
            SCSS
          </li>
          <li className=" bg-white borderBlack rounded-xl px-5 py-3 dark:bg-white/10 dark:text-w
            JavaScript
          </li>
          <li className=" bg-white borderBlack rounded-xl px-5 py-3 dark:bg-white/10 dark:text-w
            TypeScript
          </li>
          <li className=" bg-white borderBlack rounded-xl px-5 py-3 dark:bg-white/10 dark:text-w
            ReactJS
          </li>
          <li className=" bg-white borderBlack rounded-xl px-5 py-3 dark:bg-white/10 dark:text-w
            Redux
          </li>
          <li className=" bg-white borderBlack rounded-xl px-5 py-3 dark:bg-white/10 dark:text-w
            TailwindCSS
          </li>
          <li className=" bg-white borderBlack rounded-xl px-5 py-3 dark:bg-white/10 dark:text-w
            CSS Modules
          </li>
          <li className=" bg-white borderBlack rounded-xl px-5 py-3 dark:bg-white/10 dark:text-w
            Rest API
          </li>
          <li className=" bg-white borderBlack rounded-xl px-5 py-3 dark:bg-white/10 dark:text-w
            Vite
          </li>
          <li className=" bg-white borderBlack rounded-xl px-5 py-3 dark:bg-white/10 dark:text-w
            Git
          </li>
        </ul>
      </main>
    </>
  );
};

export default Home;
```

Рисунок А.12 – Сторінка Home

src > pages > Project > Project.jsx > ...

```
1   import { useParams } from "react-router-dom";
2   import BtnGitHub from "../../components/btnGitHub/BtnGitHub";
3   import { projects } from "../../helpers/projectList";
4
5   const Project = () => {
6     const { id } = useParams();
7     const project = projects.find((proj) => proj.id === id);
8
9     return (
10      <main className="bg-zinc-200 dark:bg-zinc-900 py-[70px]">
11        <div className="text-zinc-900 dark:text-white max-w-[1200px] mx-auto px-4">
12          <div className="max-w-[865px] mx-auto flex flex-col items-center text-center">
13            <h1 className="mb-[60px] text-[60px] font-bold leading-[1.3] text-[var(--title-1)]">
14              {project.title}
15            </h1>
16
17            <img
18              src={project.imgBig}
19              alt={project.title}
20              className="max-w-full mb-[40px] rounded-[10px] shadow-[var(--box-shadow)]"
21            />
22
23            <div className="mb-[30px] font-bold text-[24px] leading-[1.3]">
24              <p>{project.skills}</p>
25            </div>
26            <div className="mb-[30px] text-[24px] leading-[1.3]">
27              <p>{project.desc}</p>
28            </div>
29
30            <BtnGitHub link={project.gitHubLink} />
31          </div>
32        </div>
33      </main>
34    );
35  };
36
37  export default Project;
```

Рисунок А.13 – Сторінка Project

```
src > pages > Projects > Projects.jsx > ...
1   import Project from "../../components/Project/Project";
2
3   import { projects } from "../../helpers/projectList";
4
5   const Projects = () => {
6     return (
7       <main className="bg-zinc-200 dark:bg-zinc-900 py-[70px]">
8         <div className="text-zinc-900 max-w-[1200px] px-4 mx-auto">
9           <h2 className="text-zinc-900 dark:text-white mb-[60px] text-[60px] font-bold leading-[1.3
10            Projects
11          </h2>
12          <ul className="flex justify-center flex-wrap gap-[30px]">
13            {projects.map((project) => {
14              return (
15                <Project
16                  id={project.id}
17                  key={project.id}
18                  title={project.title}
19                  img={project.img}
20                />
21              );
22            })}
23          </ul>
24        </div>
25      </main>
26    );
27  };
28
29  export default Projects;
```

Рисунок А.14 – Сторінка Projects

Додаток Б

StrikePlagiarism.com

Дата звіту 6/9/2025
Дата редагування ---

Звіт не був оцінений

Звіт подібності

метадані

Назва організації
Kyiv National Economic University named after Vadym Hetman KNEU
Заголовок
РОЗРОБЛЕННЯ САЙТУ-ПОРТФОЛІО З ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ JAVASCRIPT І REACT
Автор
Пружанський Олексій Володимирович
Науковий керівник / Експерт
Лозовик Ю.М.
підрозділ
кафедра інформаційних систем в економіці

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.

КП 1	КП 2	КЦ
4.99% (4.99%)	2.46% (2.46%)	1.53% (1.53%)
25	**5687**	**45043**
Довжина фрази для коефіцієнта подібності 2	Кількість слів	Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв	ß	1
Інтервали	A→	1
Мікропробіли		0
Білі знаки		0
Парафрази (SmartMarks)	a	23

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз — Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://ir.kneu.edu.ua/bitstreams/a5c7419f-c089-4b0c-8e70-3923cf3566a2/download	55 0.97 %
2	https://ir.kneu.edu.ua/bitstreams/a5c7419f-c089-4b0c-8e70-3923cf3566a2/download	43 0.76 %
3	https://ir.kneu.edu.ua/bitstreams/066dbca0-6afa-4cf4-a181-c3e88a2aab6f/download	42 0.74 %
4	https://ir.kneu.edu.ua/bitstreams/a5c7419f-c089-4b0c-8e70-3923cf3566a2/download	19 0.33 %
5	https://ask.csdn.net/questions/8173328	17 0.30 %