DOI: https://doi.org/10.36910/6775-2524-0560-2025-60-32
УДК 004.514+ 004.4'24
Соколова Наталя Олегівна[1], к.т.н., доцент,
https://orcid.org/0000-0003-2493-3553
Лисун Юлія Ростиславівна[1], бакалавр,
https://orcid.org/0009-0005-7029-5371
Гаркуша Ігор Миколайович[1], к.т.н., доцент,
https://orcid.org/0000-0003-1190-1501
Балалаєва Олена Юріївна[2], к.т.н., доцент,
https://orcid.org/0000-0003-1461-4399
[1]Національний технічний університет «Дніпровська політехніка», м. Дніпро, Україна
[2]Державний вищий навчальний заклад «Приазовський державний технічний університет», м. Дніпро / Маріуполь, Україна

РОЗРОБКА АДАПТИВНОГО ВЕБДИЗАЙНУ НА ОСНОВІ ПАТЕРНІВ ПРОЄКТУВАННЯ

Соколова Н. О., Лисун Ю. Р., Гаркуша І. М., Балалаєва О. Ю. Розробка адаптивного вебдизайну на основі патернів проєктування. Стаття досліджує застосування патернів проєктування у розробці адаптивного вебдизайну корпоративного лендінгу – ефективного інструменту для стислого й зрозумілого представлення компанії в інтернеті. Лендінг поєднує статичні й інтерактивні елементи, що підвищує зацікавленість користувачів і покращує враження про компанію. Адаптивний дизайн забезпечує коректне відображення сторінки на різних пристроях (смартфонах, планшетах, ПК), що сприяє розширенню клієнтської бази. Патерни проєктування підвищують структурність, організованість і зручність підтримки сайтів, сприяють повторному використанню компонентів і реалізації складної поведінки. Досліджено використання 8 патернів: Фасад створює спрощений інтерфейс для взаємодії зі складними підсистемами сайту, приховуючи технічні деталі; Спостерігач відповідає за реакцію інтерфейсу на події користувача або зміну даних, що підвищує динамічність сторінки; Фабричний метод дозволяє гнучко створювати елементи інтерфейсу, що забезпечує розширюваність та адаптивність; Команда інкапсулює користувацькі дії, полегшуючи їх обробку та повторне використання; Стан управляє режимами роботи сайту, відповідаючи за зміну поведінки інтерфейсу залежно від контексту; Обгортка створює абстракцію навколо складних об'єктів, приховуючи деталі реалізації й спрощуючи роботу з компонентами на різних пристроях; Декоратор дозволяє динамічно додавати функції чи стилі елементам інтерфейсу без зміни основного коду, що корисно для адаптації під різні екрани; Стратегія забезпечує вибір і динамічну зміну відображення та взаємодії з користувачем залежно від контексту. Такий підхід забезпечує високу якість архітектури, оптимальну продуктивність і гнучкість роботи сайту.
Ключові слова: лендінг, вебдизайн, патерни проєктування, адаптивний дизайн, інтерфейс, події користувача, інтерактивність.

Sokolova N., Lysun Y., Garkusha I., Balalaieva O. Developing responsive web design based on Design Patterns. The article explores the use of design patterns in the development of adaptive web design for corporate landing pages. Landing pages are an effective tool for a concise and understandable presentation of a company on the Internet. Landing combines static and interactive elements, which increases user interest and improves the impression of the company. Adaptive design ensures correct display of the page on different devices (smartphones, tablets, PCs), which helps to expand the customer base. Design patterns increase the structure, organization and ease of maintenance of sites, promote the reuse of components and the implementation of complex behavior. The use of 8 patterns is researched: the facade creates a simplified interface for interacting with complex subsystems of the site, hiding technical details; the observer is responsible for the interface's reaction to user events or data changes, which increases the dynamism of the page; the factory method allows you to flexibly create interface elements, which ensures extensibility and adaptability; the command encapsulates user actions, facilitating their processing and reuse; the state controls the site's operating modes, responsible for changing the interface behavior depending on the context; The wrapper creates an abstraction around complex objects, hiding implementation details and simplifying work with components on different devices; the decorator allows you to dynamically add functions or styles to interface elements without changing the main code, which is useful for adapting the appearance to different screens; the strategy provides selection and dynamic change of display and interaction with the user depending on the context. This approach ensures high quality architecture, optimal performance and flexibility of the site.
Keywords: landing page, Web Design, Design Patterns, responsive design, interface, user events, interactivity.

Постановка проблеми. Сучасний бізнес не може ефективно функціонувати без потужної онлайн-присутності. Корпоративний сайт сьогодні – це не просто візитівка компанії, а стратегічний актив, важливий для залучення клієнтів, зміцнення бренду і генерації лідів. Він іміджево відображає місію, цінності, продукти чи послуги компанії та виступає платформою для взаємодії з клієнтами, партнерами та інвесторами. В умовах жорсткої конкуренції й швидких змін ринку впровадження новітніх технологій у веброзробку підвищує конкурентоздатність бізнесу, ефективність, продуктивність і безпеку онлайн-платформ. Ринок послуг із розробки корпоративних сайтів в Україні у 2025 році характеризується широким діапазоном цін, від базових рішень до складних порталів із високим рівнем кастомізації і інтеграцій. Вартість базового корпоративного сайту в Україні стартує приблизно від 18 000 до 48 000 гривень, середній рівень розробки становить від 24

500 до 48 000 грн, а преміум-рішення складають до 15 000 доларів та більше. Така доступність при високій якості та дотриманні міжнародних стандартів робить український ринок веб-розробки привабливим не лише для локального бізнесу, а й для іноземних клієнтів [1-3].

Односторінкові коропоративні сайти (лендінги) залишаються ефективним інструментом для швидких продажів і залучення потенційних покупців, особливо в нішах із складними продуктами. Лендінги нового покоління активно використовують інтерактивний контент, анімації, мікроанімації, темні режими, а також персоналізацію на основі штучного інтелекту, що підвищує залучення аудиторії і конверсію. Зі зростанням використання мобільних пристроїв важливо забезпечити безперебійну і комфортну роботу сайту на будь-якому розмірі екрана. Адаптивний дизайн – ключовий тренд вебдизайну. Він гарантує гнучкість макетів і зручність навігації, що суттєво підвищує користувацький досвід і зменшує показники відмов. Такі сайти краще ранжуються в пошукових системах і відповідають сучасним стандартам доступності [4-6]. Важливою особливістю є Mobile-first підхід, оскільки більше 60% трафіку надходить з мобільних пристроїв [7].

Отже, розробка корпоративних сторінок, з урахуванням трендів адаптивного дизайну і новітніх технологій, залишається актуальною. Ринок пропонує широкий вибір послуг від базових до складних рішень з додатковими функціями, а лендінги з адаптивним дизайном і персоналізацією залишаються потужним інструментом для маркетингу і продажів, тому патерни проєктування у веброзробці є незамінним інструментом для структурування коду, реалізації складної функціональності та підтримки довгострокової масштабованості додатків. Вони стандартизують підходи та оптимізують процес розробки, сприяючи підвищенню якості програмного продукту.

Аналіз останніх досліджень та публікацій. Дослідження [8] розглядає еволюцію та застосування патернів проєктування у сучасній розробці на JavaScript, включно з популярними фреймворками React, Vue, Angular. Акцент зроблено на тому, як породжувальні, структурні та поведінкові патерни підвищують ефективність, масштабованість і підтримуваність проєктів. Наведені приклади з великих компаній (Netflix, Airbnb, PayPal) демонструють практичний вплив патернів на складні системи. Також обговорено майбутні тренди, зокрема асинхронні патерни та інтеграцію WebAssembly.

Оглядова стаття [9] систематизує найпопулярніші патерни у веброзробці, пояснюючи їх роль у створенні підтримуваного, ефективного і масштабованого коду. Наводяться різні патерни, зокрема MVC, Стратегія (Strategy), Команда (Command), Спостерігач (Observer), Фасад (Facade), Одинак (Singleton), які широко використовуються у сучасних вебзастосунках. Стаття пояснює принципи трьох груп патернів: породжувальні (creational), структурні (structural) і поведінкові (behavioral), і дає невеличкі приклади застосування кожного з них на практиці.

Робота [10] присвячена оптимізації вебзастосунків із застосуванням патернів проєктування і статичного аналізу коду. Вона підкреслює, що патерни допомагають вирішувати повторювані об'єктноорієнтовані проблеми, покращують внутрішню структуру програм, посилюють поліморфізм та інкапсуляцію.

Недавнє дослідження [11] аналізує три основні типи патернів проєктування – породжувальні, структурні та поведінкові з акцентом на їх практичне застосування у програмуванні, зокрема у веброзробці. Робота демонструє, як впровадження патернів допомагає створювати масштабовані, гнучкі рішення, підвищувати продуктивність коду і знижувати витрати на розробку. Дослідження підкреслює, що патерни проєктування не є готовими фрагментами коду, а рекомендаціями щодо структурування архітектури, що сприяє легшому додаванню функціональності та підтримці систем.

Автори [12] фокусуються на практичному застосуванні патернів в розробці вебзастосунку для автоматизації створення розкладу. У дослідженні описано, як різні патерни дозволяють вирішувати конкретні технічні проблеми веброзробки, покращувати структуру коду та забезпечувати більш керовану реалізацію складних функцій.

У роботі [13] представлено п'ять патернів проєктування, які використовувалися для розробки функцій управління даними у кількох вебінформаційних системах для обробки даних підприємства студентами Німецько-Йорданського університету. Запропоновані рішення спрямовані на розв'язання таких завдань як гнучкий користувацький інтерфейс управління даними, багаторазовий модуль для залежних випадаючих фільтрів, відкладене завантаження даних таблиць, уніфіковані модулі для додавання та редагування даних, а також відновлення стану сторінок при навігації між пов'язаними сторінками. Наведені патерни було застосовано під час реалізації шести інформаційних систем університету, причому один з патернів використовувався понад 700 разів.

Оглядові систематичні праці показують, що патерни корисні для повторного використання і дизайну, а також є багато інструментів та рекомендацій для їх вибору [14, 15]; проте більшість досліджень – концептуальні або аналітичні, і не розглядають конкретну реалізацію патернів в адаптивному вебдизайні.

Таким чином, дослідження всебічно підкреслюють важливість патернів проєктування у веброзробці як засобу для забезпечення структурованості, модульності та повторного використання коду, покращення гнучкості та підтримуваності проєктів, стандартизації розробки та полегшення командної роботи, оптимізації продуктивності та масштабування вебзастосунків. Особливий акцент робиться на тому, що патерни не є готовими шаблонами коду, а допомагають керувати складністю проєкту через абстракції і узгоджені підходи.

Метою дослідження є вивчення ефективності використання патернів проєктування у сучасному адаптивному вебдизайні.

Виклад основного матеріалу й обґрунтування отриманих результатів дослідження. Об'єктом даного дослідження є розробка корпоративного вебсайту для умовної компанії CNST, що спеціалізується на наданні високоякісних будівельних послуг, з акцентом на комплексне супроводження будівельних проєктів від етапів планування та проєктування до реалізації та передачі об'єктів «під ключ». Визначено, що компанія CNST поєднує у своїй діяльності застосування сучасних технологій з професійним підходом, маючи при цьому близько десяти років практичного досвіду роботи у відповідній галузі. Враховуючи сучасні тенденції веброзробки, була обрана концепція розробки односторінкового сайту (лендінгу), головним завданням якого є стисле та ефективне донесення до користувача найважливішої інформації щодо діяльності компанії та спектру її послуг. Також вебресурс має бути реалізований із застосуванням сучасних підходів до адаптивного дизайну для забезпечення комфортного перегляду на різноманітних типах пристроїв, включно зі смартфонами, планшетами та десктопними комп'ютерами, а також бути оптимізованим для швидкого завантаження. Адаптив реалізовано підходом Mobile-first (проєктування інтерфейсу сайту, орієнтованого в першу чергу на мобільні пристрої).

Структура сайту передбачає ознайомлення користувача з основними послугами компанії, ключовими перевагами, етапами виконання робіт, відгуками задоволених клієнтів, а також містить відповіді на типові запитання. Завдяки продуманій та логічній організації контенту користувач має змогу отримати комплексне уявлення про діяльність компанії без необхідності переходити на інші сторінки. Кожен розділ послідовно доповнює попередній, що сприяє поетапному підштовхуванню користувача до цільової дії – заповнення форми зворотного зв'язку, яка забезпечує можливість швидко залишити запит на консультацію чи інший вид звернення. Введені у форму дані автоматично зберігаються у базі даних і надсилаються адміністратору через месенджер Telegram, що забезпечує оперативну обробку звернень.

На рисунку 1 подано мапу сайту, яка наочно представляє його структуру та ієрархію, демонструючи основні секції та функціональне навантаження кожної з них. Така візуалізація є корисною як для розробників, так і для замовників, оскільки дозволяє ефективно оцінити логіку побудови вебресурсу, а також виявити потенційні недоліки або напрями для подальшого вдосконалення.

Мапа сайту компанії відображає його логічну структуру, розділену на основні тематичні блоки. Центральним елементом структури є головна сторінка, яка забезпечує доступ до ключових розділів: «Про нас», «Сервіс», «Портфоліо», «Блог» та «Зв'язок», а також до додаткових секцій, таких як «Команда», «Відгуки», «F.A.Q.», «Як ми працюємо» та «Ціни». Кожна секція містить інформацію, що деталізує діяльність компанії. Наприклад, у розділі «Сервіс» представлені конкретні напрямки діяльності, такі як «Реконструкція будівлі» та інші спеціалізації. Розділ «Портфоліо» призначений для демонстрації прикладів реалізованих проєктів, а у секції «Зв'язатись» розміщено карту розташування офісу та форму зворотного зв'язку для оперативного контакту. Подібна організація структури сайту сприяє зручній навігації та швидкому пошуку необхідної інформації користувачами.

Розробка шаблону дизайну не входила до завдань цього дослідження. Після консультацій із професійними веброзробниками у спеціалізованих спільнотах було знайдено відповідний макет у Telegram-групі «FIGMA Templates», який відповідає всім вимогам чіткого, інформативного та сучасного представлення будівельної компанії. Прототип дизайну включає дві версії: десктопну та мобільну. Сайт передбачає меню навігації та різноманітні тематичні секції з інформаційним наповненням, як-от розділи про компанію, сервіс, портфоліо, новини, контактні дані тощо. Сайт

також повинен містити функціональні елементи, такі як відео, слайдери, елементи взаємодії при наведенні курсору миші (наприклад, зміна кольорів слайду), акордеон для розкривання додаткової інформації, мапу місця розташування і форму зворотного зв'язку

Мобільна версія сайту повинна включати мобільне меню для зручної навігації на смартфонах та планшетах. Проте у обраному шаблоні дизайн такого меню відсутній, тому однією з додаткових задач дослідження була розробка мобільного меню, яке наслідує стилю десктопної версії сайту і забезпечує його коректне функціонування у мобільній версії.

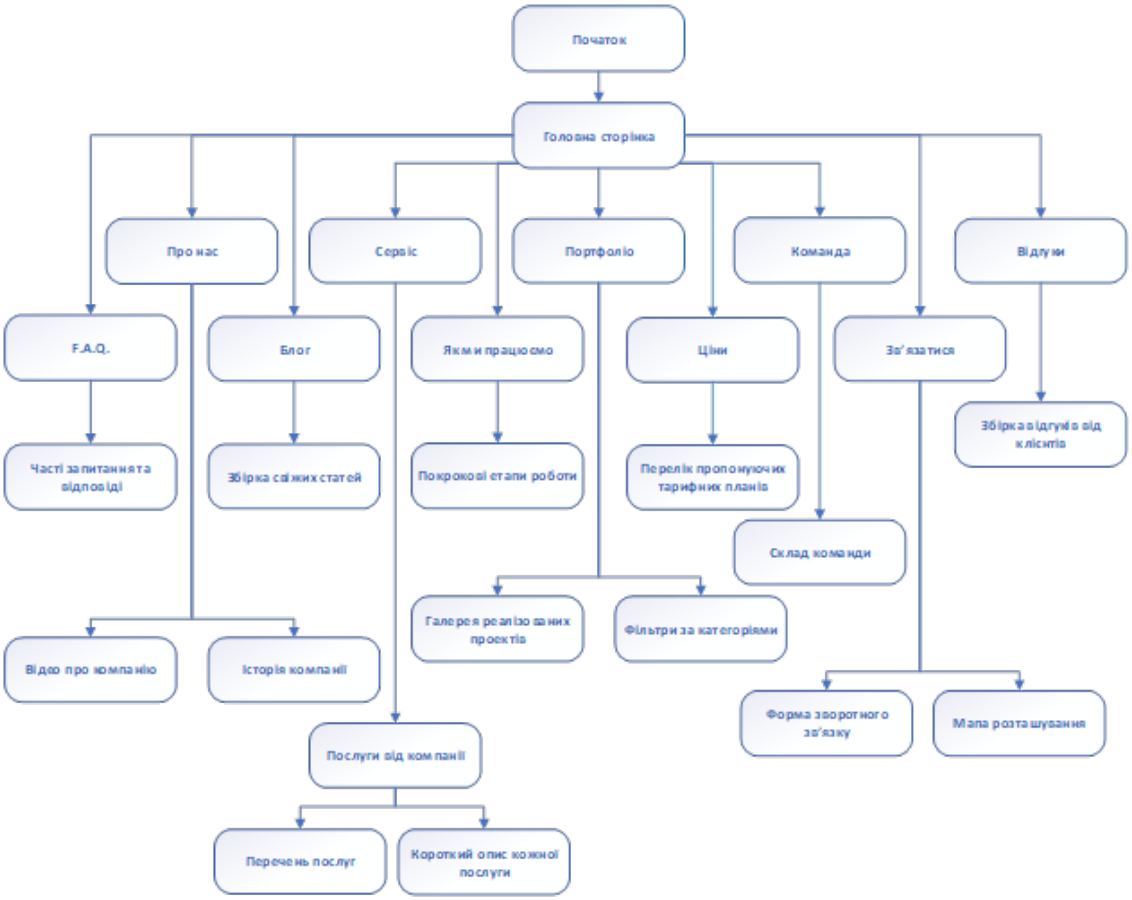


Рис. 1. Мапа сайту

Основні налаштування та налаштування розширень (Live Sass Compiler) визначені у конфігураційному файлі settings.json (рис.2).

```
"liveSassCompile.settings.formats": [
  {
    "format": "compressed",
    "extensionName": ".min.css",
    "savePath": "/css"
  }
```

Рис.2. Налаштування Live Sass Compiler

У файлі зі SASS-змінними (_variables.scss), який розміщується в окремому каталозі scss/utils для файлів-утиліт, створено кореневі змінні для кольорів, які використані в макеті, а також змінні із значеннями переломних точок (breakpoints) для реалізації адаптиву для мобільних пристроїв, планшетів та комп'ютерів. Breakpoints – це точки, при яких змінюється вигляд сайту в залежності від ширини екрану. Для застосування breakpoints створені медіа-запити (@media), які застосовуються до блоку/елементу, де потрібно додати адаптив. Для реалізації адаптивного дизайну використана технологія Flexbox, за допомогою якої зручно робити адаптивні блоки, будувати сітки або центрувати вміст. Основа цієї технології – це контейнер div, в якого значення

властивості display дорівнює flex, і елементи цього контейнера автоматично вирівнюються в залежності від властивостей, які будуть встановлені.

SASS-код далі конвертується у CSS, створюється мінімізований (стислий) CSS-файл main.min.css, в який переноситься переведений файл з кодом та зберігається у каталозі CSS.

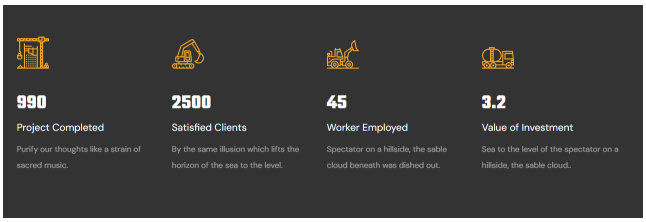
В дослідження використана методологія БЕМ (Блок, Елемент, Модифікатор), яка є зручною системою іменування CSS-класів, що дозволяє створювати більш структурований, зрозумілий та легкий у підтримці код. Її основний принцип полягає у поділі інтерфейсу на автономні блоки, які можуть містити елементи та модифікатори. Застосування БЕМ полегшує розуміння структури коду без необхідності перегляду HTML розмітки, а також сприяє повторному використанню стилів і зменшенню ймовірності конфліктів між класами. Однією з переваг методології є її сумісність із препроцесорами, такими як SCSS, які підтримують вкладеність. Така вкладена організація стилів дозволяє наочно відслідковувати взаємозв'язки між блоками, елементами та модифікаторами, що значно полегшує підтримку коду і покращує ефективність командної роботи над проєктом.

Для скидання стилів браузера за замовчуванням створено окремий файл _base.scss, який також містить базову типографіку та допоміжні стилі, що застосовуватимуться по всьому проєкту. Таке структурування дає змогу уникнути дублювання коду, підвищити його читабельність та спростити подальше обслуговування. Завдяки продуманій організації цього файлу розробка наступних модулів значною мірою полегшується, оскільки найпоширеніші стилі вже враховані й не вимагають повторного оголошення.

Стилістика сайту реалізована на основі CSS. Файли зображень, у тому числі фавікон, а також файли стилів бібліотек Owl Carousel і Swiper розміщені у відповідних окремих каталогах. Файли Owl Carousel завантажено локально і винесено в окрему папку owlcarousel, що забезпечує підвищену надійність роботи. Стилі Swiper підключено через CDN-посилання, що дає змогу уникнути завантаження файлу на сервер, забезпечуючи зручність і скорочуючи час підключення. Користувацькі стилі сайту формуються із згенерованого CSS-файлу, отриманого за допомогою компілятора SASS, причому підключається мінімізований (стислий) файл main.min.css, що позитивно впливає на швидкість завантаження та оптимізацію сторінки. Прообразом файлу main.min.css є файл main.scss, який слугує об'єднувачем усіх SCSS-файлів для зручної та оптимізованої компіляції і трансформації стилів у CSS. У цьому файлі через директиву @import підключаються стилі з інших модулів, таких як базові стилі, компоненти, змінні, макети секцій тощо. Такий підхід дозволяє грамотно організувати код, полегшуючи його керування в процесі розробки і є реалізацією патерна проєктування Фасад (Facade), оскільки main.scss виконує роль єдиної точки доступу до всієї системи стилів. Фасад приховує складність внутрішньої структури проєкту і забезпечує зручний і простий інтерфейс для подальшої компіляції та підтримки стилів.

Файли _bar-section.scss, _column-section.scss, _row-section.scss та _section.scss створені для зручності та прискорення процесу стилізації секцій, оскільки в проєкті застосовуються схожі за стилем секції, що мають певні відмінності (рис.3). Для уникнення багаторазового дублювання однакових стилів у кожному блоці створюється базовий клас із відповідними вкладеними класами, які потім повторно використовуються у необхідних місцях. Такий метод відповідає патерну Адаптер (Adapter), оскільки повторювані елементи «адаптуються» під різні контексти застосування без зміни їхньої базової структури.

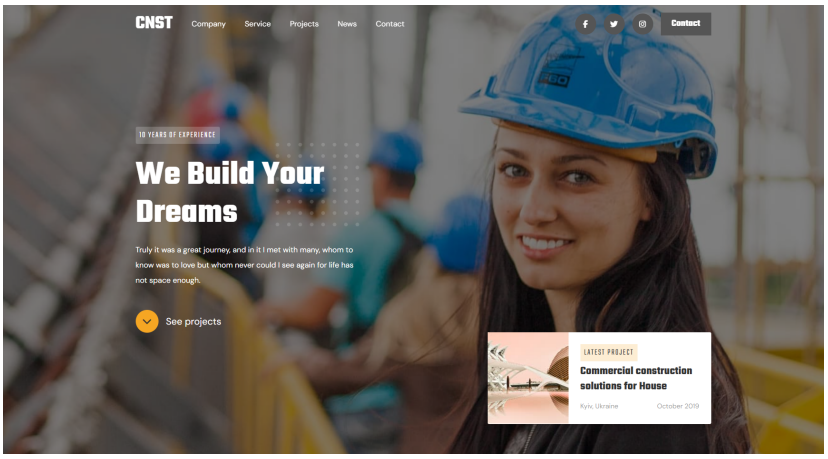
а)



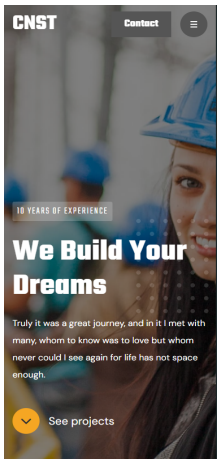
б)

Рис.3. Схожі між собою секції, стилізовані класами файлу _bar-section.scss: а) «як ми працюємо»; б) статистика

Для успішної реалізації адаптивної верстки важливо створювати та використовувати технічні класи. Прикладом такого класу є wrapper, який слугує обгорткою сторінки. В межах блоку з цим класом пов´язується вся структура сторінки, що гарантує коректне вертикальне розташування її частин в адаптивному дизайні і підвищує надійність розробки (рис. 4). Реалізація такої обгортки зручно здійснюється за допомогою патерна Декоратор (Decorator), оскільки він дозволяє додавати стилі й впливати на вміст всередині класу без прямої зміни його структури.



а)



б)

Рис.4. Контент шапки та головної секції обмежені по ширині: а) у десктопній версії контент не розтягується на всю ширину екрану; б) у мобільній версії використовуються відступи зліва та справа

У випадках, коли два елементи мають подібну структуру або вигляд, доцільно уникати дублювання класів, винісши спільні стилі у окремі блоки. Якщо ж цей елемент передбачено використовувати багаторазово, його логічно оформити як компонент. Стилі таких повторно застосовуваних блоків розміщені в окремому каталозі components (рис. 5). Це відповідає принципам патерна Фасад (Facade), оскільки загальні стилі компонентів абстрагують складну внутрішню структуру та забезпечують єдиний, зручний інтерфейс для роботи з ними.

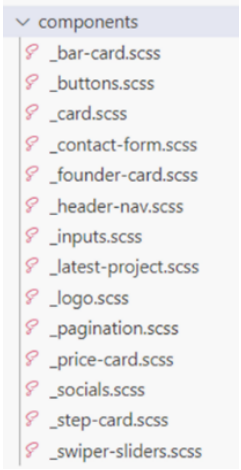
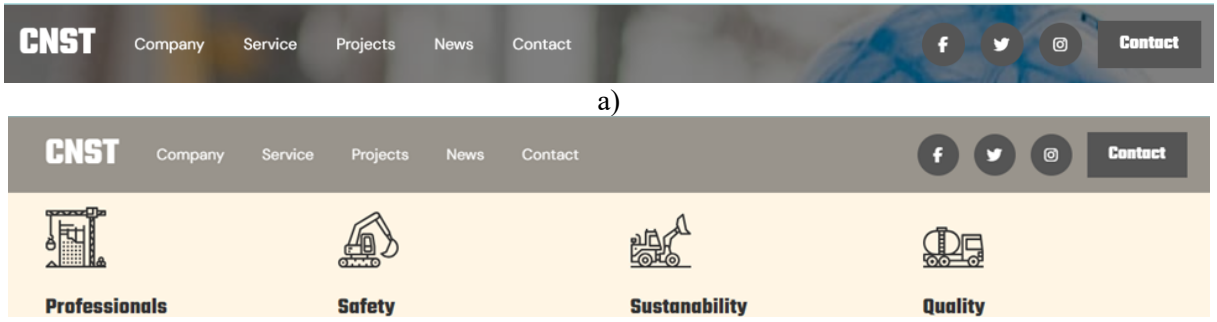


Рис.5. Каталог components

При розробці меню навігації (горизонтальний маркований список з 5 пунктами) було виявлено проблему у шаблоні: при скролі вниз, шапка зливається із самим сайтом. Було винайдено рішення використанням патерна Декоратор (Decorator) – надати кольорове тло шапці при скролі на деяку відстань униз (рис. 6). Для цього додається прослуховувач події scroll. Функція-обробник за умови, що вікно браузера скролиться вверх на більше ніж 100 пікселів, до шапки додає клас scrolled, який надає колір тлу, інакше видаляє цей клас. Також при цьому використовується патерн Спостерігач (Observer): вікно браузера виступає у ролі суб'єкта, що генерує подію scroll, а функція-обробник є спостерігачем, який реагує на цю подію та змінює DOM. Такий підхід дозволяє реалізувати гнучке реагування на дії користувача без жорсткого зв'язування між компонентами.

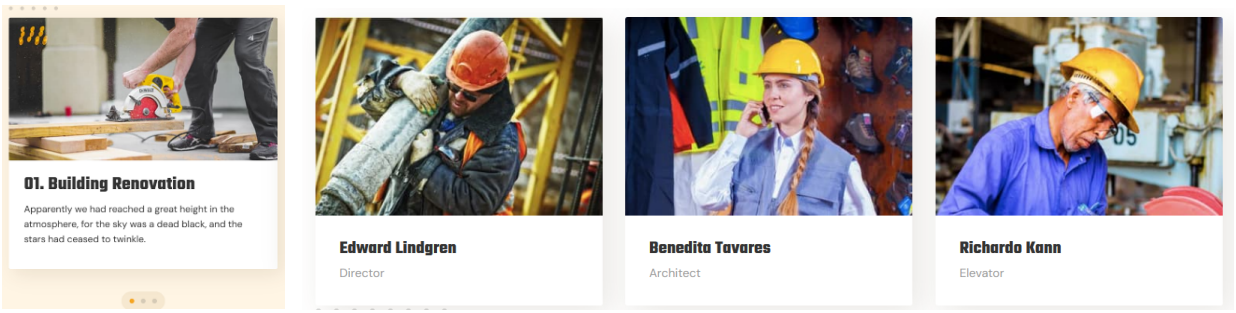


а)

б)

Рис.6. Шапка сайту а) без кольорового тла; б) з кольоровим тлом

Бібліотеки для створення слайдерів мають готові класи, що визначають блоки та їх елементи, а також забезпечують базову стилізацію. Для початкового оформлення слайдера з Owl Carousel достатньо присвоїти основному контейнеру класи .owl-carousel (активує слайдер) та .owl-theme (додає базові стилі). Окрім них, використовується користувацький клас row-slider для додаткової стилізації через SCSS. Слайдер «Сервіс» (рис.7, а) складається з кількох div-блоків з класом card, які є окремими слайдами. Кожен блок містить зображення (img) та текстову частину card__content з елементами-параграфами p класів card__title (з модифікатором card__title--service для заголовків) та card__text. Така структура дозволяє стилізувати заголовки секції окремо і уникати дублювання стилів для інших частин сайту. Оскільки всі слайди мають однакову структуру, бібліотека коректно їх обробляє, забезпечуючи адаптивне автоматичне переключення. Зображення, текст і заголовки налаштовані для гармонійного відображення на різних розмірах екранів. Однак для роботи слайдера необхідно активувати його через JavaScript, викликавши метод owlCarousel() з об'єктом параметрів, що відповідають за налаштування поведінки. Цей підхід реалізує патерн проєктування Стратегія (Strategy), коли конфігураційний об'єкт змінює функціональність слайдера без зміни коду плагіна.



а) б)

Рис.7. Слайдери: а) «Сервіс»; б) «Команда»

Реалізація слайдера «Команда» (рис. 7, б) є більш складною. Слайдер «Команда» автоматично прокручується що 3 секунди у циклі. За замовчуванням на екрані видно один слайд; при ширині від 600 пікселів – два слайди, а від 992 пікселів – три. Відстань між слайдами становить 30 пікселів. Реалізована також клікабельна пагінація (розбивка величезного масиву даних (зазвичай однотипного) на невеликі за обсягом сторінки з відображенням нумерованої навігації по ним), яка керується елементом з класом .swiper-pagination. У цій реалізації застосовано патерн проєктування Фабричний метод (Factory Method), оскільки створення об'єкта слайдера здійснюється через виклик конструктора бібліотеки Swiper із заданими параметрами. Такий підхід відокремлює логіку створення об'єкта від клієнтського коду, а конфігурація поведінки слайдера передається через аргументи, що забезпечує гнучкість та можливість повторного використання.

Бібліотека Swiper використовує основні класи, які визначають головний контейнер, кожен слайд (обгортку та сам елемент слайда), а також додаткові компоненти, такі як пагінація і навігація. Головний контейнер містить класи team-section__carousel та swiper. Усередині нього розташована обгортка swiper-wrapper, що включає окремі слайди з класом swiper-slide. Кожен слайд є карткою (card), яка містить зображення учасника з класом card__image, заголовок з класом card__title та опис посади з класом card__type. Після списку слайдів розміщено елемент пагінації з класом swiper-pagination. Для активації слайдера створюється змінна, якій присвоюється новий об'єкт класу Swiper (рис. 8), передаючи два аргументи: HTML-елемент-контейнер та об'єкт із налаштуваннями слайдера.

Секція «Портфоліо» є важливим елементом сайту будівельної компанії, оскільки дозволяє потенційним клієнтам ознайомитися із реалізованими проєктами та оцінити професійні можливості компанії. Саме портфоліо часто є основою при прийнятті рішення клієнтом про співпрацю. У секції «Портфоліо» представлені яскраві натискні картки з зображеннями та описами проєктів компанії. Кожен проєкт віднесений до однієї з категорій: будівництво, будівельні роботи або проєктне планування. Для швидкого фільтрування передбачені кнопки, які дозволяють переключатися між категоріями (рис.9).

```
const swiper = new Swiper(".swiper", {
  loop: true,
  slidesPerView: 1,
  spaceBetween: 30,
  breakpoints: {
    600: {
      slidesPerView: 2,
    },
    992: {
      slidesPerView: 3,
    },
  },
  autoplay: {
    delay: 3000,
  },
  pagination: {
    el: ".swiper-pagination",
    clickable: true,
  },
});
```

Рис.8. Ініціалізація слайдера «Команда»

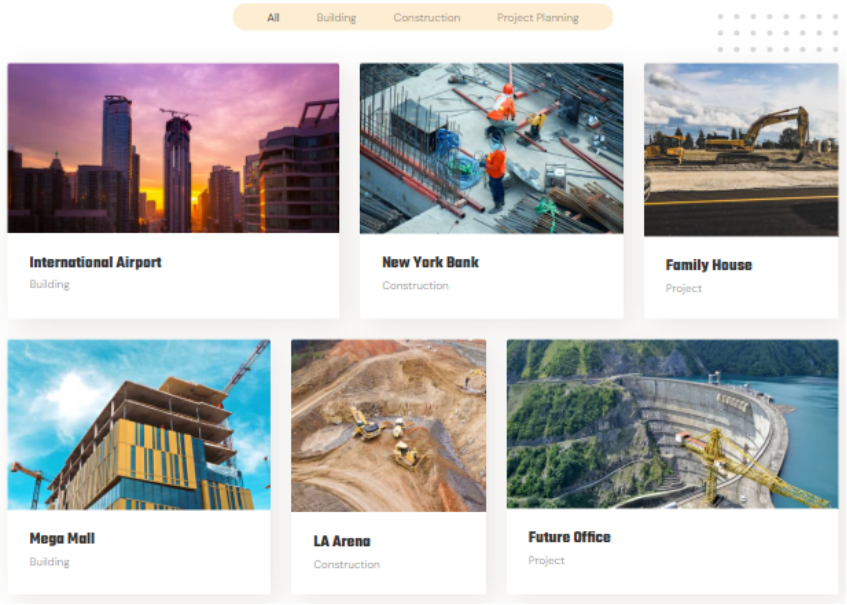


Рис.9. Секція «Портфоліо»

При натисканні на кнопку запускається скрипт, який зчитує значення дата-атрибуту filter натиснутої кнопки. Всі кнопки позбавляються класу button-active, що змінює колір тексту, а цей клас додається лише обраній кнопці. Якщо вибрано фільтр «all», клас hidden (що приховує картку) видаляється з усіх карток. Якщо обрано конкретну категорію, відбувається перебір усіх карток: ті, що відповідають класу з назвою дата-атрибуту, залишаються видимими, решті додається клас hidden. Тут застосовано патерн проєктування Стратегія (Strategy) з прослуховувачем події click для кнопок фільтру, коли обрана фільтрація змінює логіку відображення елементів без змін у структурі обробника (рис.10). Це робить інтерфейс більш зручним та інтуїтивним, дозволяючи користувачам швидко знаходити потрібні проєкти та ознайомлюватися з виконаними роботами компанії.

```
$(".portfolio-section__button").on("click", function () {
  const filter = $(this).data("filter");

  $(".portfolio-section__button").removeClass("button-active");
  $(this).addClass("button-active");

  if (filter === "all") {
    $(".card").removeClass("hidden");
  } else {
    $(".card").each(function () {
      $(this).toggleClass("hidden", !$(this).hasClass(filter));
    });
  }
});
```

Рис.10. Прослуховувач події click для кнопок фільтру

Акордеон – це інтерактивний інтерфейсний елемент, який забезпечує можливість відкривати та закривати текстові блоки. У цьому проєкті акордеон застосовано у секції «F.A.Q.», де кожне питання відкриває приховану відповідь при кліку на заголовок. Таке рішення дозволяє ефективно економити простір на сторінці, демонструючи лише ті відповіді, які зацікавили користувача. Подібна реалізація є широко розповсюдженою на сучасних вебресурсах завдяки своїй простоті, зручності та ефективності. Акордеон складається з кількох однакових «спойлерів» (елемент, що йде безпосередньо після заголовка), кожен із яких містить заголовок – активну зону для кліку, та прихований текст із відповіддю. Заголовок супроводжується невеликою SVG-іконкою-стрілкою, яка при відкритті спойлера плавно повертається на 90 градусів вправо (рис. 11).

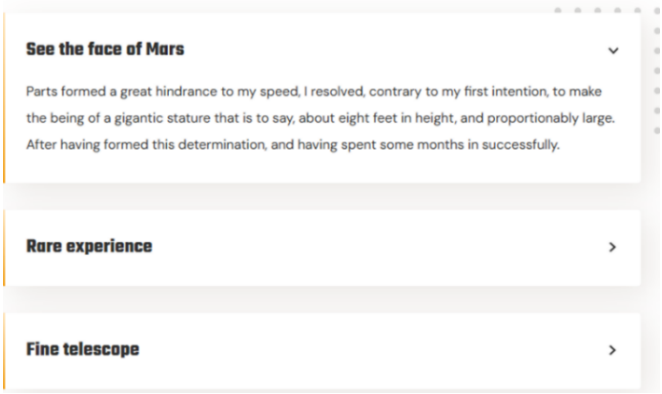


Рис.11. Акордеон «F.A.Q.»

При натисканні на заголовок виконується скрипт, що додає або видаляє клас rotate у блоку зі стрілкою, який відповідає за анімацію повороту. Плавність руху забезпечується CSS-властивістю transition із тривалістю 400 мс. Далі скрипт знаходить текстову частину спойлера і за допомогою методу slideToggle() плавно показує або ховає його, використовуючи тривалість анімації за замовчуванням у 300 мс. У цьому випадку застосовується патерн проєктування Стан (State), оскільки поведінка і вигляд елемента змінюються залежно від його поточного стану – відкритий або закритий. Клас rotate та функція slideToggle() динамічно відображають ці зміни, забезпечуючи інтуїтивну і зручну взаємодію користувача з інтерфейсом.

Динамічні анімації на сайті реалізовані за допомогою API Intersection Observer у поєднанні з бібліотекою jQuery. Це дозволяє відслідковувати появу елементів у видимій області екрану користувача. Приклад – анімація появи контейнерів у секціях та чисел у секції «Статистика» (рис. 3, б). Функція animateCounters() шукає елементи з класом .count і анімує їх від 0 до значення в атрибуті data-target, поступово оновлюючи текст. Intersection Observer запускає цю анімацію, коли хоча б 10% елемента видимі, додає клас visible, викликає метод animateCounters() і припиняє спостереження за цим елементом, щоб анімація не повторювалась. При готовності DOM до контейнера .container додається клас animate, що позначає його для анімації. За винятком контейнера головної секції, якому клас animate додається із затримкою 300 мс, відбувається спостереження за всіма контейнерами, окрім container-hero і секції «Статистика». Цей підхід реалізує патерн Спостерігач (Observer): Intersection Observer реагує на появу елемента в зоні видимості і запускає відповідну функцію анімації, що відокремлює логіку визначення видимості від власне анімації, підвищуючи гнучкість і модульність коду.

Для декоративної анімації у секції «Як ми працюємо» (рис. 3, а) використовується асинхронний метод setInterval(). Функція changeStepColor() з інтервалом 2 секунди послідовно додає клас active до поточного елемента і вилучає його з інших, створюючи «ефект підсвічування» для позначення етапів роботи. Цей механізм реалізує патерн Ітератор (Iterator), де кожен крок обробляється циклічно, а setInterval() керує послідовністю змін станів.

Планшетна та мобільна версії сайту оснащені мобільним меню. Оскільки дизайн для нього відсутній у макеті, реалізація виконана інтуїтивно, акцентуючи увагу на важливості цього елемента у мобільній версії. Для вирішення проблеми злипанню пунктів меню у шапці (рис. 6, а) при зменшенні екрану застосовано патерн Команда (Command): пункти меню приховуються, натомість з'являється кнопка «бургер» (рис. 12). Кожна дія – відкриття або закриття меню – оформлена як окрема команда, яку можна викликати у відповідь на подію (натискання кнопки). Це дозволяє централізовано керувати поведінкою інтерфейсу через визначені дії.

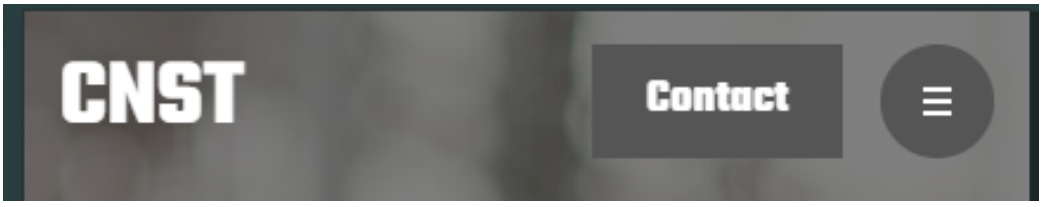


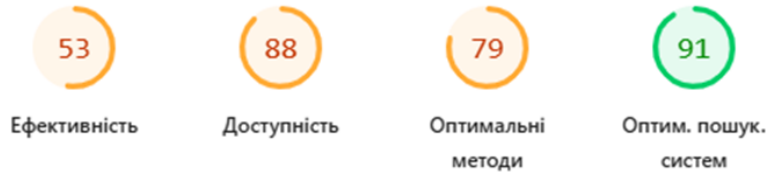
Рис.12. Шапка сайту(мобільна версія)

При кліку на кнопку «бургер» до меню додається клас is-open, що відкриває його, а для блокування прокручування сторінки в HTML-документ додається overflow: hidden, який забороняє скрол вертикально і горизонтально. За замовчуванням меню зміщене по осі X на 100% і приховане, а при додаванні класу is-open повертається на екран. Закриття меню відбувається при кліку на кнопку закриття або пункт меню – клас is-open знімається, а властивість overflow видаляється з тіла документа, повертаючи скрол до нормального стану (рис.13).

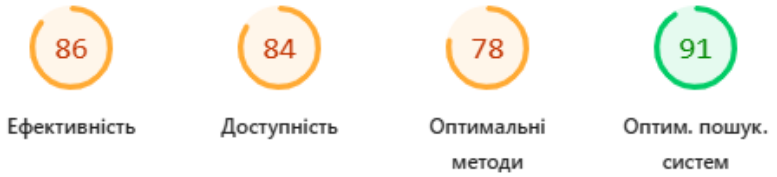
```
$(".header__burger-btn").on("click", function () {
  $(".mobile-menu").addClass("is-open");
  $("body").css("overflow", "hidden");
});

$(".mobile-menu__btn-close, .header-nav__link").on("click", function () {
  $(".mobile-menu").removeClass("is-open");
  $("body").css("overflow", "");
});
```

Рис.13. Відкриття та закриття мобільного меню

Було проведено тестування розробленого лендінгу сервісами Google, які оцінюють завантаження сторінки за такими метриками: ефективність, доступність, оптимальні методи, оптимальний пошук. На рисунку 14 наведено результати завантаження сторінки на мобільних пристроях та комп'ютерах відповідно – вони є у межах норми, а це свідчить про те, про проблеми із завантаженням сторінки не виникнуть, але ефективність мобільної версії можна ще покращувати.



а)



б)

Рис. 14. Оцінка завантаження сторінки; а) на мобільних пристроях; б) на комп'ютерах

Ознайомитися з розробленим за допомогою патернів корпоративного лендінгу можна за посиланням https://julls.firemoretest.space/CNST/.

Висновки та перспективи подальшого дослідження. У статті проаналізовано важливість використання патернів проєктування у адаптивному дизайні корпоративних лендінгів. Результатом є повноцінна адаптивна сторінка з інтерактивними елементами, включно з формами зворотного зв'язку, анімаціями, слайдерами, інтеграцією з Telegram-ботом та збереженням даних у базу даних, яка пройшла перевірку працездатності в реальному середовищі після його розміщення на хостинг.

Сайт успішно проходить тестування на адаптивність та має прийнятні показники швидкості завантаження, що підтверджує правильність обраних рішень. Таким чином, створений лендінг не тільки виконує поставлені функції, але й відповідає сучасним вимогам як до дизайну, так і до технічної реалізації.

Розроблений лендінг можна використовувати як макет для конструювання корпоративних сторінок інших галузей. У подальшому сайт можна доповнити додатковими сторінками, багатомовною підтримкою або системою керування контентом (CMS), що розширить його функціонал і зробить зручнішим для адміністрування, а також покращити ефективність завантаження мобільної версії

Список бібліографічного опису:

1.Паталяк Є. Скільки коштує зробити сайт у 2025. URL: https://surl.li/jqngms (дата звернення 05.08.2025).
2.Дімура М. Скільки коштує створення корпоративного сайту у 2025 році? URL: https://surl.li/rwbaix (дата звернення 05.08.2025).
3.Шмичков М. Просування корпоративного сайту:Посібник для бізнесу у 2025 році. URL: https://surl.li/fcrtov (дата звернення 05.08.2025).
4.Read M. 25 Top Web Design Trends 2025. URL: https://surl.li/abdzgi (дата звернення 05.08.2025).
5.Bratslavsky P.. 15 Web Development Trends for 2025. URL: https://surl.lu/jjctbl (дата звернення 05.08.2025).
6.Loy M. Top 15 Web Development Trends To Expect In 2025. URL: https://surl.li/kjwmht (дата звернення 05.08.2025).
7.Digital 2025: Global Overview Report. DataReportal – Global Digital Insights. URL: https://surl.lt/xkmjcd (дата звернення 05.08.2025).
8.Awais M. JavaScript Design Patterns: A Comprehensive Analysis of Their Evolution, Usage, and Impact in Modern Web Development. Muhammad Awais. 2024; URL: https://surl.li/oxfapu (дата звернення 05.08.2025).
9.Latest Design Patterns for Web Development. Last Updated. 2025. URL: https://surl.lu/tvzbpa (дата звернення 05.08.2025).
10. Fyama B., Kadiata F. Optimization by design patterns and static analysis of web applications for a sharp adaptation of e-business start-ups in the city of Lubumbashi in DR Congo (Nesher). Revue Internationale du Chercheur. Août 2021. Volume 2: Numéro 3. p. 1349-1373.
11. Григорчук Г.В., Григорчук Л.І. Застосування патернів проєктування для підвищення ефективності програмного коду. Вчені записки ТНУ імені В.І. Вернадського, 2025, Том 36 (75) № 1. с. 59-64.
12. Борозенний С.О., Мисько Ю. М. Аналіз патернів проєктування у веб розробці та їх застосування у розробці вебзастосунку для автоматизації створення розкладу навчального закладу. Теоретичні та прикладні аспекти побудови програмних систем: праці 15 міжнародної науково-практичної конференції, Київ, 23-24 грудня 2024 р. с. 71-73.
13. Feras Al-Hawari, Software design patterns for data management features in web-based information systems, Journal of King Saud University - Computer and Information Sciences, 2022, Volume 34, Issue 10, Part B. p. 10028-10043. https://doi.org/10.1016/j.jksuci.2022.10.003
14. Asghar M. Z., Alam K. A., Javed S.. Software Design Patterns Recommendation : A Systematic Literature Review. International Conference on Frontiers of Information Technology (FIT), Islamabad, Pakistan, 2019, pp. 167-172, doi: 10.1109/FIT47737.2019.00040.
15. Almadi Sara H. S., Hooshyar D., Ahmad R. B. Bad Smells of Gang of Four Design Patterns: A Decade Systematic Literature Review. 2021, Sustainability 13, no. 18: 10256. https://doi.org/10.3390/su131810256

References:

1.Pataliak Ye. Skilky koshtuie zrobyty sait u 2025.. URL: https://surl.li/jqngms (access 10.08.2025).
2.Dimura M. Skilky koshtuie stvorennia korporatyvnoho saitu u 2025 rotsi? URL: https://surl.li/rwbaix (access 10.08.2025).
3.Shmychkov M. Prosuvannia korporatyvnoho saitu:Posibnyk dlia biznesu u 2025 rotsi. URL: https://surl.li/fcrtov (access 10.08.2025).
4.Read M. 25 Top Web Design Trends 2025. URL: https://surl.li/abdzgi (access 10.08.2025).
5.Bratslavsky P. 15 Web Development Trends for 2025. URL: https://surl.lu/jjctbl (access 05.08.2025).
6.Loy M. Top 15 Web Development Trends To Expect In 2025. URL: https://surl.li/kjwmht (access 10.08.2025).
7.Digital 2025: Global Overview Report. DataReportal – Global Digital Insights. URL: https://surl.lt/xkmjcd (access 10.08.2025).
8.Awais M. JavaScript Design Patterns: A Comprehensive Analysis of Their Evolution, Usage, and Impact in Modern Web Development. Muhammad Awais. (2024). URL: https://surl.li/oxfapu (access 10.08.2025).
9.Latest Design Patterns for Web Development. Last Updated. (2025). URL: https://surl.lu/tvzbpa (access 10.08.2025).
10. Fyama B., Kadiata F. (2021). Optimization by design patterns and static analysis of web applications for a sharp adaptation of e-business start-ups in the city of Lubumbashi in DR Congo (Nesher). Revue Internationale du Chercheur. Volume 2: Numéro 3. p. 1349-1373.
11. Hryhorchuk H.V., Hryhorchuk L.I. Zastosuvannia paterniv proiektuvannia dlia pidvyshchennia efektyvnosti prohramnoho kodu. (2025). Vcheni zapysky TNU imeni V.I. Vernadskoho, Том 36 (75) № 1. с. 59-64.
12. Borozennyi S.O., Mysko Yu. M. Analiz paterniv proiektuvannia u veb rozrobtsi ta yikh zastosuvannia u rozrobtsi vebzastosunku dlia avtomatyzatsii stvorennia rozkladu navchalnoho zakladu. Teoretychni ta prykladni aspekty pobudovy prohramnykh system: pratsi 15 mizhnarodnoi naukovo-praktychnoi konferentsii, Kyiv, 23-24 hrudnia 2024 p. c. 71-73.
13. Feras Al-Hawari. (2022). Software design patterns for data management features in web-based information systems, Journal of King Saud University - Computer and Information Sciences, Volume 34, Issue 10, Part B. p. 10028-10043. https://doi.org/10.1016/j.jksuci.2022.10.003
14. Asghar, M. Z., Alam, K. A. & Javed, S. (2019). Software Design Patterns Recommendation : A Systematic Literature Review, International Conference on Frontiers of Information Technology (FIT), Islamabad, Pakistan, 2019, pp. 167-172, doi: 10.1109/FIT47737.2019.00040.
15. Almadi, S. H. S., Hooshyar, D., & Ahmad, R. B. (2021). Bad Smells of Gang of Four Design Patterns: A Decade Systematic Literature Review. Sustainability, 13(18), 10256. https://doi.org/10.3390/su131810256