Міністерство освіти і науки України НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
«КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра інформатики факультету інформатики
Бакалаврська програма

Розробка веб-додатку для автоматизованого генерування кольорових палітр та оптимізації 2D-графіки для веб-дизайну

Текстова частина до курсової роботи
за спеціальністю “Інженерія програмного забезпечення”

Керівник курсової роботи
доцент Афонін А. О.

_____
(підпис)
“___” _____ 2025 р.
Виконала студентка БП ІПЗ-3
Сліпушкіна О. Г.
“___” _____ 2025 р.

Київ 2025

Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра інформатики технологій факультету інформатики

Затверджую

Зав.кафедри мультимедійних систем,

доцент, кандидат наук

_____ Жежерун О. П.

“____” _____ 2025 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на курсову роботу

студентці Сліпушкіній Олександрі Григорівні 3-го курсу факультету інформатики

ТЕМА: Розробка веб-додатку для автоматизованого генерування кольорових палітр та оптимізації 2D-графіки для веб-дизайну

Вихідні дані:

Зміст ТЧ до курсової роботи:

Індивідуальне завдання

Зміст

Анотація

Вступ

1. Теоретичні основи генерації кольорових палітр та оптимізації 2D-графіки

2. Аналіз аналогів та потреб цільової аудиторії

3. Проєктування та реалізація веб-додатку

Висновки

Список використаної літератури

Дата видачі “___” _____ 2025 р.

Керівник _____ Завдання отримано _____

Календарний план виконання курсової роботи

Тема: Розробка веб-додатку для автоматизованого генерування кольорових палітр та оптимізації 2D-графіки для веб-дизайну

№ п/п	Назва етапу курсової роботи	Термін виконання етапу	Примітка
1	Процес вибору теми	10.10.2024-16.10.2024	
2	Узгодження теми з керівником	29.10.2024	
3	Пошук матеріалів на обрану тему та створення мети на основі цього	Листопад 2024 – грудень 2024	
4	Проведення дослідницького аналізу	Січень 2025	
5	Розробка веб-додатку на основі обраної теми	Кінець січня 2025 – середина лютого 2025	
6	Тестування веб-додатку для виявлення можливих помилок	Кінець лютого 2025	
7	Написання теоретичної частини	Березень 2025 – квітень 2025	
8	Перевірка практичної та теоретичної частин	Кінець квітня 2025	
9	Подання роботи на кафедру для перевірки відповідності вимогам академічної доброчесності	05.05.2025	
10	Захист курсової роботи	12.05.2025 – 14.05.2025	

Студент Сліпушкіна О. Г. _____

Керівник Афонін А. О. _____

“_____” _____ 2025 р.

ЗМІСТ

АНОТАЦІЯ

Мета – розробка веб-додатку, який дозволяє дизайнерам автоматизовано генерувати гармонійні кольорові палітри та оптимізувати 2D-графіку для використання у веб-дизайні. Щоб досягти поставленої мети було використано такі методи дослідження: аналіз наукових джерел про теорії кольору та принципів UX/UI-дизайну; порівняльний аналіз вже існуючих сервісів для генерації палітр; моделювання алгоритмів гармонізації кольорів; а також експериментальна перевірка якості оптимізації зображень з використанням обраних методів стискання. Комбінування цих підходів дозволить обґрунтовано реалізувати всі функціональні компоненти веб-додатку.

Отже, досягнення цієї мети передбачає створення алгоритмів для формування палітр на основі теорії кольору та сучасних підходів у веб-дизайні, аналіз та вибір найбільш відповідних комбінацій кольорів відповідно до заданих критеріїв, а також розробку функцій для автоматичного стиснення та оптимізації 2D-зображень. Такі дії дозволять покращити продуктивність веб-додатків. До того ж, важливим аспектом є забезпечення зручного та зрозумілого інтерфейсу для взаємодії користувачів із системою. Крім того, невід’ємним є впровадження технологій веб-розробки, які забезпечать швидку обробку та генерацію даних. Результатом роботи стане функціональний веб-додаток, що спростить процес підбору кольорових рішень для веб-дизайну та підвищить ефективність роботи з графікою в онлайн-середовищі.

ВСТУП

Мета дослідження – створення веб-додатку, який допомагає дизайнерам автоматично генерувати гармонійні кольорові палітри на основі обраного базового кольору та оптимізувати 2D-графіку для використання у веб-додатках.

Об’єкт дослідження – процес розробки веб-дизайну з урахуванням естетичних і технічних чинників.

Предмет дослідження – використання кольорових палітр, інструментів оптимізації графіки та засобів автоматизації в сучасному веб-дизайні.

- Роль кольорових палітр у веб-дизайні

Роль кольорових палітр у веб-дизайні є досить значною. Кольорова палітра є однією з ключових складових веб-дизайну, яка впливає не лише на естетичну привабливість інтерфейсу, а й на ефективність взаємодії користувача з веб-ресурсом. Вона представляє собою комплексну систему візуального сприйняття, що створює атмосферу, передає емоції, виділяє акценти та підсилює основне повідомлення сайту або веб-додатка. “Колір допомагає виразити світло – не фізичне явище, а єдине світло, яке насправді існує, яке знаходиться в мозку художника” – Анрі Матісс, французький художник і скульптор. Ця цитата наголошує на тому, що колір — це не просто фізичне явище, а засіб вираження ідеї, емоції та інтерпретації світу. У контексті веб-дизайну це означає, що колірна палітра здатна транслювати цінності, настрій і характер ресурсу ще до того, як користувач почне читати тексти або взаємодіяти з елементами.

Веб-дизайн – це процес планування, розробки, створення візуального та функціонального вигляду веб-сторінок і веб-додатків. Це міждисциплінарна галузь, що поєднує естетичні засади графічного дизайну з технічними особливостями веб-розробки.

Колір є одним із найпотужніших засобів візуальної комунікації у веб-дизайні. Вдало підібрана кольорова палітра не лише формує естетичну привабливість інтерфейсу, але й безпосередньо впливає на емоційне сприйняття сайту користувачем, його зручність, ефективність навігації та

загальну довіру до ресурсу. Кольорова палітра – це набір кольорів, які застосовуються разом у дизайні для отримання гармонійного та функціонального візуального ефекту. У контексті веб-дизайну палітра визначає загальну кольорову гаму інтерфейсу і задає візуальний стиль сайту або веб-застосунку. Вперше термін «палітра» було зафіксовано в англійській мові у 1662 році. А саме у веб-дизайні концепція кольорових палітр почала активно розвиватися у 1990-х роках, коли з'явилися перші графічні браузери. Графічні браузери – це веб-браузери, що можуть відображати не тільки текст, а й зображення, відео, анімації.

Зараз неможливо уявити наше життя без інтернету, а отже й без веб-сайтів та веб-додатків, що забезпечують швидкий доступ до інформації, сервісів і засобів комунікації. У цьому цифровому середовищі саме візуальне оформлення відіграє ключову роль у формуванні першого враження про ресурс. Кольорова палітра – один із найважливіших елементів цього оформлення, адже саме вона створює емоційний настрій інтерфейсу, формує психологічне сприйняття контенту та значною мірою впливає на поведінкові реакції користувача.

І тому ми розуміємо, що одна з найперших речей у створенні будь-якого дизайну – це визначитися з кольоровою палітрою. Бо саме від цього буде залежати посил розробника та емоційне сприйняття користувача.

- Необхідність оптимізації графіки для швидкого завантаження сторінок

У сучасному веб-середовищі швидкість завантаження веб-сторінки є одним із критичних факторів, що визначають ефективність сайту.

Дослідження показують, що понад 50% користувачів очікують, що сторінка завантажиться менш ніж за 3 секунди, і майже 40% залишають сайт, якщо завантаження триває довше. Одним із основних чинників, що впливають на цей показник, є візуальний контент, зокрема – графіка. Зображення на сайті роблять контент привабливим для користувача. Вони ж серйозно

сповільнюють завантаження сторінок. Вихід з такої ситуації один – оптимізувати зображення. А оптимізація графіки – це ключ до продуктивності.

Оптимізована графіка забезпечує швидке завантаження сторінки, що особливо важливо для мобільних пристроїв з обмеженою пропускною здатністю мережі. Крім того, велика вага графіки створює надмірне навантаження на сервер і зменшує загальну ефективність сайту при високому трафіку. Пошукові системи також враховують швидкість завантаження при формуванні рейтингу сторінок. Це означає, що неоптимізовані зображення можуть погіршити не лише користувацький досвід, але й позиції сайту в результатах пошуку, що прямо впливає на видимість та охоплення аудиторії. У результаті зменшується трафік і знижуються шанси на комерційний або інформаційний успіх ресурсу.

Таким чином, оптимізація дозволяє підтримувати баланс між естетикою й технічною ефективністю, забезпечуючи комфортне, швидке та привабливе користування веб-сайтом.

- Автоматизація дизайнерських процесів

У сучасному веб-дизайні важливо не лише створити естетично привабливий інтерфейс, але й забезпечити його ефективність і зручність у використанні. Роль кольорових палітр та необхідність оптимізації графіки є ключовими складовими цього процесу. Вдало підібрана кольорова палітра не лише формує візуальний стиль і настрій ресурсу, а й впливає на емоційне сприйняття, довіру користувача та загальний комфорт взаємодії. Разом із цим, оптимізація графіки забезпечує швидке завантаження сторінок, покращує користувацький досвід, підвищує позиції у пошуковій видачі та зменшує навантаження на сервер. Ці аспекти тісно пов’язані між собою, і для досягнення найкращих результатів дизайнери все частіше звертаються до автоматизації процесів.

Автоматизація дизайнерський процесів – це процес заміни ручної роботи технічними засобами або програмним забезпеченням. Тобто автоматизація дозволяє використовувати інструменти, сервери або алгоритми для

полегшення дизайнерських завдань.

У сфері веб-дизайну автоматизація дозволяє значно скоротити час на виконання типових задач і підвищити загальну ефективність роботи. Зокрема, завдяки спеціалізованим інструментам дизайнери можуть, наприклад, автоматично створювати сітки, генерувати кольорові палітри, адаптувати графіку до різних екранів і пристроїв, перевіряти контрастність кольорів для забезпечення доступності та багато іншого.

Таким чином, автоматизація не тільки полегшує технічні аспекти створення інтерфейсів, а й підвищує загальну якість продукту, залишаючи більше простору для творчості, інновацій та стратегічного мислення. У результаті, автоматизовані дизайнерські процеси стають не допоміжним, а стратегічним інструментом у реалізації успішних веб-проєктів.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ ГЕНЕРАЦІЇ КОЛЬОРОВИХ ПАЛІТР ТА ОПТИМІЗАЦІЇ 2-D ГРАФІКИ

1.1. Основи колірної теорії

- Кольорове коло (RGB, RYB, HSL)

Кольорове коло – це основа теорії кольору в веб-дизайні. Це базова модель, яка використовується для візуалізації зв'язків між кольорами. Воно дозволяє зрозуміти принципи поєднання відтінків, створення контрастів і гармонійних комбінацій.

Кольорова модель RGB називається адитивною, тому що вона створює кольори шляхом додавання світла трьох основних кольорів – червоного, зеленого та синього. Уявіть собі три ліхтарі, які світять на білу стіну: один червоним, другий зеленим, третій синім. Якщо червоне світло змішається із зеленим, на стіні з'явиться жовтий колір. Якщо змінити яскравість одного з них – наприклад, зробити зелений слабшим або червоний сильнішим – колір зміниться, наприклад, на оранжевий. А якщо всі три кольори світять одночасно, то на стіні буде біле світло.

Колірна модель RGB – це структурована система, яка використовується в цифрових пристроях і світлових носіях для створення гами кольорів із невеликого набору основних кольорів – у цьому випадку червоного, зеленого та синього. Назва цієї колірної моделі походить від першої літери назви кожного основного кольору – red, green, blue.

Вперше такі експерименти зі світлом провів у 1665-1666 роках англійський фізик і математик Ісаак Ньютон. Він підніс скляну призму до променя світла, коли той входив у затемнену кімнату. Пізніше Ісаак Ньютон описав як біле світло розпадається на червоне, оранжеве, жовте, зелене, синє, блакитне та фіолетове. Він дійшов висновку, що біле світло є поєднанням усіх кольорів, і став першою людиною, яка натякнула на те, як колір сприймається людьми.

Існують й інші колірні моделі, серед них – RYB. Відомо, що традиційне кольорове коло художників є одним із прикладів субтрактивної системи

кольорів. Субтрактивна система кольорів – це спосіб створення кольорів шляхом віднімання (поглинання) світла, яке відбивається від поверхні. Основними кольорами є червоний, жовтий і синій. Тому колірну модель RYB також за першою літерою кожного основного кольору – red, yellow, blue. Кольори називаються основними, оскільки вони не можуть бути створені комбінуванням інших відтінків. Будь-які два з трьох основних кольорів можна змішувати, щоб отримати вторинні кольори: зелений (утворений поєднанням жовтого та синього), оранжевий (жовтий і червоний) і фіолетовий (синій і червоний). Змішування основного кольору з сусіднім вторинним кольором створює проміжний колір.

Якщо поєднати всі кольори колірної моделі RYB, теоретично вони створять чорний колір. Це тому, що барвники вибірково поглинають і відбивають світло для створення кольору. Наприклад, жовтий пігмент поглинає сині та фіолетові довжини хвиль, водночас відбиваючи жовті, зелені та червоні. Синій пігмент поглинає переважно жовті, оранжеві та червоні довжини хвиль. Якщо змішати жовтий і синій пігменти, утвориться зелений, оскільки це єдиний спектральний компонент, який не сильно поглинається жодним пігментом. У якомусь сенсі жовтий і синій пігменти забирають колір один від одного, залишаючи лише зелений колір.

У той час як колірна модель RGB базується на тому, як світло різних довжин хвиль додається разом, колірна модель HSL базується на іншому механізмі компонування кольорів. Hue, Saturation and Lightness – це три основні характеристики кольору, які допомагають точніше описати, як саме ми його бачимо, а також від цього і походить назва цієї колірної моделі. Відтінок – це, власне, назва кольору, яку ми сприймаємо, наприклад: червоний, жовтий, помаранчевий, синій тощо. Це те, що ми найперше називаємо, коли бачимо колір. Насиченість описує, наскільки яскравий або приглушений виглядає цей колір. Чим вища насиченість (наприклад, 100%), тим чистішим і «живішим» виглядає колір – без домішок сірого. Якщо насиченість зменшити до 0%, колір повністю втрачає свій відтінок і перетворюється на сірий. Тобто, насиченість

– це, по суті, присутність або відсутність сірого. Світліть, або яскравість, визначає, наскільки колір наближений до білого чи чорного. Коли значення світлості 100% – це практично білий колір, а при 0% – повністю чорний. Світліть дозволяє отримувати як дуже темні, так і дуже світлі варіанти одного й того ж відтінку.

Отже, кольорові моделі RGB, RYB і HSL мають різне призначення та особливості, але всі вони відіграють важливу роль у розумінні й створенні кольорів. RGB – це адитивна модель, яка створює кольори шляхом додавання світла трьох основних кольорів: червоного, зеленого і синього. Вона використовується в електронних пристроях – моніторах, телевізорах, смартфонах – де кольори формуються за допомогою світла. Основною перевагою RGB є широкий діапазон яскравих, насичених кольорів, проте її складно сприймати інтуїтивно, особливо людям без технічної підготовки. На відміну від неї, модель RYB – це субтрактивна система, яка базується на змішуванні фарб і пігментів. Вона знайома кожному, хто працює з реальними матеріалами – художникам, дизайнерам інтер'єрів, декораторам. Основними кольорами тут є червоний, жовтий і синій. Перевага цієї моделі полягає в її простоті й зрозумілості для візуального мистецтва, хоча в цифрових технологіях вона не настільки точна, як RGB. HSL, своєю чергою, є зручною моделлю для дизайну, оскільки дозволяє керувати кольором не лише за відтінком, а й за насиченістю та світлістю. Вона не ґрунтується ні на світлі, ні на пігментах, а на сприйнятті кольору людиною. Саме тому її зручно використовувати у веб-дизайні, де важливо легко добирати гармонійні кольори та змінювати їхні характеристики без складних розрахунків.

- Типи гармоній: аналогова, комплементарна, тріадна

Гармонія кольорів у дизайні є одним із ключових елементів, що визначає ефективність візуального сприйняття. Вона передбачає створення збалансованих, естетично приємних поєднань кольорів, які викликають позитивні емоції та полегшують взаємодію користувача з продуктом. Дизайнери активно використовують принципи гармонії кольорів для

формування візуальної ідентичності бренду, створення атмосфери, керування увагою користувача та покращення загальної доступності інтерфейсу. Варто зазначити, що гармонія – це не лише про «гарні» кольори. Вона охоплює глибше розуміння того, як саме кольори впливають на композицію, емоційне сприйняття й функціональність дизайну загалом.

Існує багато різних типів гармоній, а найпопулярніші з них – це аналогова, комплементарна, тріадна.

Аналогова колірна схема є однією з найпоширеніших у дизайні та мистецтві завдяки своїй природній гармонійності. Вона складається з трьох кольорів, розташованих поряд один з одним на колірному колі, наприклад, синій, синьо-зелений і зелений. Ці кольори мають спільні елементи, тому легко поєднуються між собою, створюючи м’які, ненав’язливі комбінації. Часто аналогові схеми використовуються для передачі спокійного, природного настрою або плавного переходу кольорів.

Подібні кольорові поєднання нерідко зустрічаються в природі – наприклад, на заході сонця, у листі дерев, у воді, що відбиває небо. Така природність сприймається людиною інтуїтивно як приємна та збалансована. Однією з варіацій цієї схеми є аналогова схема «високого тону», коли аналогічні кольори змішуються з білим. Це створює м’який, світлий ефект, який додає витонченості й легкості. Такі рішення широко використовуються в пастельних палітрах, дизайні інтер’єрів, моді, косметичних продуктах.

Художники-імпресіоністи активно застосовували цю схему для передачі атмосферності та освітлення у своїх роботах. М’які переходи між кольорами створюють ефект мерехтіння й розмитості, дозволяючи глядачеві сприймати зображення цілісно, без чітких меж. З відстані аналогова схема у високому тоні може навіть здаватися одним кольором, хоча насправді це складна комбінація відтінків. Такий ефект зорового злиття також активно використовується в сучасному графічному дизайні для створення глибини, градієнтів або ніжних фонів.

А у веб-дизайні аналогові схеми допомагають створити комфортний візуальний досвід, м’які градієнти, естетично приємні фони або інтерфейси, що не перевантажують користувача. Вони ідеальні для сайтів, які прагнуть передати спокій, довіру або ніжність.

Комплементарні кольори – це кольори, що розташовані навпроти один одного на колірному колі, наприклад, синій та оранжевий, червоний і зелений, жовтий та фіолетовий. Вони створюють найвищий можливий контраст, що робить таку кольорову гармонію візуально динамічною та емоційно насиченою. Це поєднання притягує погляд і часто використовується для виділення ключових елементів, оскільки протилежність кольорів створює яскраве візуальне зіткнення.

Однак саме через свій інтенсивний ефект комплементарні кольори можуть бути складними у використанні, особливо для новачків у дизайні. Якщо обидва кольори застосовуються з максимальною насиченістю, це може перенавантажити зір і зробити інтерфейс візуально агресивним. Щоб пом’якшити ефект, дизайнери часто додають до кольорів чорний або білий – це дозволяє приглушити контраст, зберігши при цьому ефективність гармонії.

У веб-дизайні комплементарні кольори широко використовуються для створення акцентів – наприклад, кнопок із закликом до дії, банерів або важливих повідомлень. Вони допомагають швидко скеровувати увагу користувача на ключові зони інтерфейсу, що особливо важливо в e-commerce, лендінгах та мобільних застосунках. Головне – використовувати їх дозовано та з урахуванням загального стилю платформи, аби не зруйнувати візуальну цілісність.

Тріадна колірна гармонія включає три кольори, які рівномірно розташовані по колірному колу, утворюючи гармонійну комбінацію. Така схема може створювати яскравий, але більш збалансований вигляд порівняно з комплементарними кольорами, де контраст інколи буває дуже різким. У веб-дизайні це дозволяє створити насичений і привабливий візуальний ефект, без перенасиченості кольорами.

Щоб забезпечити гармонію в дизайні з тріадною схемою, зазвичай один колір виступає домінуючим, а два інших – як акцентні. Це нагадує підхід розділеної комплементарної гармонії, де основний колір використовується для створення фокусу, а допоміжні кольори служать для підкреслення. Для кращого балансу дизайну часто використовують один темний відтінок, а для двох інших кольорів – більш світлі відтінки. Використання відтінків та тіней може створити спокійну атмосферу, тоді як яскраві, насичені кольори підходять для молодіжних, енергійних дизайнів, що потребують динаміки.

Ця колірна схема є дуже універсальною і може використовуватися в різноманітних веб-проєктах, створюючи ефект гармонії та динамізму одночасно, що робить її популярною серед дизайнерів, які прагнуть досягти балансу між яскравістю та елегантністю.

У дизайні гармонія кольорів відіграє вирішальну роль у формуванні естетики та функціональності інтерфейсів, створюючи комфортне сприйняття та покращуючи взаємодію користувача з продуктом. Вибір колірних схем – аналогічних, комплементарних чи тріадних – має суттєвий вплив на візуальне сприйняття та емоційний відгук. Аналогічні кольори, завдяки своїй природній гармонійності, створюють м’які та заспокійливі ефекти, що ідеально підходять для дизайнів, які прагнуть передати спокій та довіру. Комплементарні кольори ж, завдяки високому контрасту, здатні привертати увагу до важливих елементів інтерфейсу, однак потребують обережного використання, щоб уникнути перенасичення. Тріадні колірні схеми надають можливість створювати яскраві, але збалансовані поєднання, що дозволяє додати динамічності без зайвого візуального навантаження. Залежно від контексту та цілей проєкту, кожна з цих схем може бути використана для досягнення різних візуальних ефектів, підкреслюючи бренд або сприяючи кращій взаємодії з користувачем.

• Психологія кольору у дизайні

Колір відіграє життєво важливу роль у світі, в якому ми живемо. Він впливає на все: від настрою та поведінки до підсвідомого прийняття рішень щодо дизайну.

Психологія кольору є потужним інструментом, що дозволяє дизайнерам, маркетологам і брендам свідомо впливати на поведінку та емоції людей. Колір – це не просто естетичний вибір, а стратегічний засіб комунікації, який допомагає формувати потрібне враження, підсилювати функціональність продукту або викликати певні асоціації. Залежно від контексту, той самий відтінок може бути джерелом спокою або тривоги, викликати довіру або настороженість, асоціюватися з радістю або скорботою. Саме тому дизайнери мають враховувати не лише загальні психологічні реакції, але й культурні та соціальні контексти.

На прикладі медицини ми бачимо, що колір здатен змінювати не лише настрій, а й фізіологічну реакцію – синій, наприклад, знижує збудженість, а червоний стимулює. У сфері брендингу колір може стати тією невидимою мовою, яка промовляє до підсвідомості споживача. Саме тому візуальний стиль компаній ретельно підбирається з урахуванням цільової аудиторії, ринку та культурних норм.

Як влучно зауважив Пабло Пікассо: «Кольори, як риси, слідують за змінами емоцій». Ця фраза чудово підкреслює, що колір є не лише фоном, а активним учасником комунікації.

Зрештою, психологія кольору – це міст між візуальним мистецтвом і людською природою. Вона дозволяє дизайнерам не просто створювати гарне, а впливати, викликати реакції та формувати враження. Успішний дизайнер – це той, хто розуміє цю мову і використовує її усвідомлено, з повагою до контексту та з урахуванням культурної чутливості.

1.2. Основи 2D-графіки в інтерфейсах

- Растрова і векторна графіка

Растрова графіка – тип цифрового зображення, яке використовує крихітні прямокутні пікселі або елементи зображення, розташовані у вигляді сітки, для представлення зображення. Оскільки формат може підтримувати широкий діапазон кольорів та відображати тонкі градієнтні тони, він добре підходить

для відображення зображень із безперервним тоном, таких як фотографії або затінені малюнки, а також інших детальних зображень.

Растрова графіка бере свій початок із технологій телебачення, де зображення формувалися за принципом сканування екрана рядок за рядком. В основі растрової графіки лежить сітка з мікроскопічних елементів — пікселів, що мають однаковий розмір і розташовані у двох вимірах: по горизонталі (стовпцях) і вертикалі (рядках). Кожен окремий піксель зберігає певну кількість інформації, яка визначає його колір та яскравість, а обсяг цієї інформації залежить від бажаної деталізації зображення.

Наприклад, для чорно-білих зображень використовується мінімум даних – лише один біт на піксель, що дозволяє фіксувати два стани: чорний або білий. Більш складні зображення з відтінками сірого або кольорові зображення вимагають більше інформації: зазвичай 24 біти на кожен піксель. Такий формат називають "справжнім кольором", оскільки він здатен відтворювати понад 16 мільйонів кольорових відтінків, що робить його надзвичайно реалістичним для візуалізації.

Глибина кольору – це термін, який описує кількість бітів, що використовуються для представлення кольору одного пікселя. Чим більше бітів припадає на піксель, тим багатшими й точнішими можуть бути кольорові переходи. Іншою важливою характеристикою є роздільна здатність – кількість пікселів, що утворюють зображення на ширину та висоту. Вона зазвичай позначається у вигляді добутку кількості пікселів у рядку на кількість у стовпці, наприклад, 800 × 600. Вища роздільна здатність забезпечує більшу чіткість і деталізацію візуального контенту.

Отже, растрова графіка – це фундаментальна технологія відображення зображень, яка поєднує простоту структури з високою гнучкістю відтворення кольорів і відтінків.

На відміну від растрової графіки, векторна графіка – це комп'ютерні зображення, створені за допомогою послідовності команд або математичних операторів, які розміщують лінії та фігури у двовимірному або тривимірному

просторі. Файл векторної графіки містить набір векторних операторів – інструкцій, які вказують, які саме геометричні елементи необхідно з’єднати й як саме їх потрібно обробити. Інакше кажучи, замість збереження кожного окремого пікселя, як це робиться у растровій графіці, векторні файли описують послідовність точок і кривих, що формують об'єкти. Саме тому такі файли іноді називають геометричними файлами.

За допомогою векторної графіки художники створюють гнучкі й масштабовані зображення, які можна змінювати без втрати якості. Це можливо тому, що векторне зображення не залежить від роздільної здатності: воно залишається чітким і ясним незалежно від того, наскільки його збільшити чи зменшити.

Спрощено векторну графіку можна уявити як малюнок, створений шляхом з'єднання точок. Кожна лінія або крива – це математично виведена форма, а не фізичний піксель. Саме завдяки цьому векторні зображення ідеально підходять для логотипів, іконок, технічних ілюстрацій та будь-якої графіки, яка потребує масштабування без втрати чіткості.

Отже, векторна графіка забезпечує точність, гнучкість і компактність файлів, що робить її незамінною в багатьох сферах професійного дизайну.

У підсумку хочу зазначити, що растрова та векторна графіка є двома основними підходами до створення й збереження цифрових зображень, кожен із яких має свої унікальні особливості та сфери застосування. Таким чином, растрова графіка краще підходить для фотореалістичних або деталізованих зображень із багатими переходами кольорів, тоді як векторна графіка забезпечує високу точність, універсальність і економічність у створенні масштабованих об'єктів. Вибір між цими двома типами графіки залежить від завдань і вимог конкретного проєкту.

- Поширені формати: PNG, JPG, SVG, WebP

У сучасному цифровому світі для зберігання й обміну зображеннями використовуються різні формати файлів. Кожен із них має свої особливості,

переваги та сфери застосування. Найпопулярнішими серед них є PNG, JPG, SVG і WebP.

PNG (Portable Network Graphics) – формат растрової графіки, створений як альтернатива GIF. Він підтримує прозорість і глибоку кольорову палітру (до 48 біт), забезпечує стиснення без втрати якості, що робить його ідеальним для веб-графіки, логотипів і зображень із прозорим фоном.

JPG або JPEG (Joint Photographic Experts Group) – один із найпоширеніших форматів для збереження фотографій і складних кольорових зображень. Він застосовує стиснення з втратою якості, завдяки чому суттєво зменшує розмір файлів, проте багаторазове редагування може призводити до поступового погіршення якості зображення.

SVG (Scalable Vector Graphics) – формат векторної графіки, який базується на XML-описах фігур, ліній та кольорів. Завдяки можливості масштабування без втрати якості SVG широко використовується для створення іконок, логотипів, технічних креслень і графіки для веб-сайтів.

WebP – сучасний формат, розроблений компанією Google для ефективного використання в інтернеті. Він підтримує як стиснення без втрат, так і з втратами, забезпечує прозорість і анімацію. WebP дозволяє створювати зображення високої якості при суттєво меншому розмірі файлів порівняно з PNG та JPG.

Отже, кожен формат має свої переваги залежно від поставлених завдань. PNG ідеально підходить для зображень із прозорістю і високою якістю, JPG – для фотографій та реалістичних зображень із плавними переходами кольорів. SVG – найкращий вибір для масштабованої векторної графіки, а WebP – сучасне рішення для швидкого завантаження графіки на веб-сайтах. Обираючи формат, важливо враховувати потреби конкретного проєкту: якість, розмір файлу та особливості використання.

• Вимоги до графіки у веб-додатках

Графіка у веб-додатках відіграє важливу роль у створенні враження користувача про продукт. Вона має бути якісною, швидко завантажуватися та

правильно відображатися на різних пристроях, тому існують певні вимоги, яких варто дотримуватися.

Однією з основних вимог є оптимізація розміру файлів. Зображення повинні бути мінімізовані за обсягом для прискорення завантаження сторінок, при цьому сучасні формати, як-от WebP, допомагають зменшити вагу без помітної втрати якості. Важливо також забезпечити адаптивність графіки – зображення мають коректно виглядати на смартфонах, планшетах і комп’ютерах.

Ще одна важлива вимога – висока якість відображення на пристроях із високою щільністю пікселів. Для таких екранів застосовують або зображення з підвищеною роздільною здатністю, або векторну графіку, наприклад, SVG, яка не втрачає чіткості при масштабуванні. Якщо потрібна прозорість фону, слід обирати формати, що підтримують альфа-канал, наприклад, PNG або WebP. Альфа-канал – це спеціальний канал в графічному файлі, який зберігає інформацію про прозорість кожного пікселя зображення.

Також варто подбати про сумісність графіки з різними браузерами. Для нових форматів, таких як WebP, бажано передбачити резервні варіанти на випадок, якщо браузер користувача їх не підтримує. Крім того, кожне зображення має супроводжуватися описовим атрибутом, що покращує доступність для людей із вадами зору і сприяє оптимізації сайту для пошукових систем. У випадку використання анімацій потрібно дбати про їхню легкість, надаючи перевагу CSS-анімаціям або анімованим SVG замість важких GIF-файлів.

Підсумовуючи зазначу, що правильна організація графіки у веб-додатках – це баланс між якістю, швидкістю завантаження і зручністю для користувача. Дотримання цих вимог забезпечує позитивний користувацький досвід і сприяє успіху веб-проєкту.

1.3. Методи оптимізації 2D-зображень

- Стиснення зображень

Стиснення зображень – це процес, який зменшує розмір файлів зображень. Воно найчастіше працює або шляхом видалення байтів інформації з зображення, або за допомогою алгоритму стиснення зображень для перезапису файлу зображення таким чином, щоб він займав менше місця для зберігання. Це важлива частина оптимізації зображень.

Існують два основні типи стиснення зображень: стиснення з втратами (lossy) та стиснення без втрат (lossless). Кожен з цих методів має свої особливості і використовується в залежності від вимог до якості зображення і розміру файлу.

Стиснення з втратами передбачає видалення деяких частин зображення, які вважаються менш важливими для сприйняття. Це дозволяє значно зменшити розмір файлу, але також може призвести до втрати частини деталей зображення. Цей тип стиснення є незворотним, що означає, що після застосування такого стиснення відновити оригінальне зображення неможливо. Найчастіше стиснення з втратами застосовується до фотографій, де деяке зниження якості, наприклад, при зменшенні кольорів або різкості, не помітне для користувача.

Поширеним форматом стиснення з втратами є JPEG (Joint Photographic Experts Group). Він використовує складні алгоритми для аналізу зображення і видалення тих даних, які, ймовірно, не будуть помічені людьми при перегляді. Наприклад, JPEG ефективно стискає фотографії, де присутні великі однотонні ділянки, і зменшує інформацію про колір або дрібні деталі в тінях. JPEG дозволяє регулювати рівень стиснення, що дає можливість користувачу вибрати баланс між якістю зображення і розміром файлу.

Однією з переваг стиснення з втратами є те, що воно значно зменшує розмір файлів, що важливо для збереження швидкості завантаження на веб-сайтах та економії місця на дисках. Однак, повторне стиснення одного і того ж зображення з втратами може призвести до його поступового погіршення і сильного спотворення, тому важливо зберігати оригінальні файли для редагування.

На відміну від стиснення з втратами (lossy), стиснення без втрат не призводить до втрати жодної інформації з зображення, що дозволяє відновити його до початкового стану після стиснення. Цей метод зберігає всі деталі зображення, тому результати виглядають абсолютно так само, як і оригінали, що робить його ідеальним для зображень, де важлива точність і збереження якості. Однак стиснення без втрат не зменшує розмір файлу так сильно, як стиснення з втратами, і часто використовується в тих випадках, коли критично важливо зберегти високу якість зображення.

Популярним форматом стиснення без втрат є PNG (Portable Network Graphics), який застосовується, зокрема, для графіки з прозорими фонами, логотипів, іконок та елементів інтерфейсу. PNG використовує методи стиснення, що знаходять повторювані патерни в зображенні і стискають їх, не видаляючи важливої інформації. Це дозволяє значно зменшити розмір файлу без втрати якості. Оскільки PNG підтримує прозорість, він є ідеальним для використання в графіці для веб-сайтів та інтерфейсів, де важлива можливість накладення зображень один на один.

Також одним із важливих є CDN (Content Delivery Network). Це система серверів, розподілених по різних регіонах світу, які зберігають та доставляють контент, такий як зображення, відео, аудіофайли та інші ресурси, кінцевим користувачам. Головна мета CDN – забезпечити швидке завантаження контенту шляхом розміщення копій даних на серверах, що знаходяться фізично ближче до користувача.

Однією з головних переваг CDN є зменшення затримок при завантаженні контенту. Завдяки географічному розподілу серверів контент доставляється з найближчого до користувача сервера, що значно скорочує час, необхідний для завантаження. Це особливо важливо для великих мультимедійних файлів, таких як високоякісні зображення, відео або контент веб-сайтів, де час відгуку є критично важливим для користувацького досвіду.

Коли стиснення зображень поєднується з використанням CDN, це забезпечує надзвичайно ефективну доставку контенту. Оптимізуючи

зображення і доставляючи їх з серверів, розташованих ближче до кінцевих користувачів, можна досягти значної економії часу завантаження, зменшити навантаження на основні сервери і підвищити загальну продуктивність веб-сайтів і мобільних додатків.

Стиснення зображень є важливим інструментом у забезпеченні ефективної роботи веб-сайтів та програм. Вибір методу стиснення залежить від того, чи важливі для вас швидкість завантаження і економія місця, чи збереження максимальної якості. Стиснення з втратами, хоча і призводить до втрати деякої частини деталей, є ефективним для багатьох застосувань, де помірне зниження якості непомітне для користувачів. У свою чергу, стиснення без втрат гарантує збереження оригінальної якості зображення, що робить його ідеальним для специфічних випадків, де важлива кожна деталь.

РОЗДІЛ 2. АНАЛІЗ АНАЛОГІВ ТА ПОТРЕБ ЦІЛЬОВОЇ АУДИТОРІЇ

2.1. Огляд існуючих рішень

- Онлайн генератори кольорових палітр

З появою Інтернету почався й розвиток спрощених рішень для веб-дизайнерів. З кожним роком з’являється все більше і більше нових онлайн генераторів кольорових палітр. Та чи кожний з них є якісним?

Одними з найвідоміших онлайн генераторів є Coolors, Adobe Color, Paletton. Coolors – онлайн-інструмент для швидкої генерації кольорових палітр, орієнтований на дизайнерів і веб-розробників. Він дозволяє створювати набори з п’яти кольорів на основі випадкової генерації або вибраного основного кольору. Користувач може заблокувати певні кольори, щоб вони не змінювались при повторній генерації, а також вручну змінювати кожен колір, використовуючи різні колірні моделі – HEX, RGB, HSB, CMYK та LAB. Інструмент також надає функцію перевірки контрастності згідно зі стандартами WCAG, що є особливо важливим для забезпечення доступності веб-інтерфейсів. Палітри можна експортувати у різних форматах (PNG, PDF, SCSS, SVG) або зберігати в обліковому записі. Додаткова перевага – наявність мобільного застосунку, що дозволяє працювати з палітрами будь-де.

Іншим популярним рішенням є Adobe Color – професійний інструмент для створення гармонійних кольорових схем, розроблений Adobe для дизайнерів, що працюють з Adobe Creative Cloud. Цей сервіс орієнтований на професійних дизайнерів і дозволяє створювати кольорові схеми на основі колірної гармонії: аналогічної, комплементарної, тріадної, монохромної тощо. Особливістю Adobe Color є можливість витягування кольорів з зображення, яке користувач завантажує самостійно. Крім того, доступний режим перевірки кольорової доступності для людей з порушеннями зору, що дозволяє підвищити інклюзивність дизайну. Палітри, створені в цьому сервісі, можна безпосередньо інтегрувати у Photoshop, Illustrator та інші продукти Adobe, що робить його незамінним для професійної роботи.

Третій популярний продукт – це Paletton. Він базується на побудові кольорових схем на основі колірного кола. Користувач може створити палітру вручну, обираючи тип схеми: монохромну, аналогову, тріадну, тетрадну або власну комбінацію. Інтерфейс дозволяє в реальному часі змінювати розташування кольорів на колі та переглядати, як ці кольори виглядають у типових елементах веб-дизайну – кнопках, тексті, фонах. Інструмент також генерує тіні, відтінки та тони кожного кольору, що дає змогу створити цілісну систему кольорового оформлення. Хоча Paletton виглядає дещо застарілим у порівнянні з сучасними сервісами, він залишається надійним інструментом для тих, хто хоче глибше зрозуміти основи колірної гармонії.

- Порівняння функціональності

Інструмент	Автоматичне генерування	Створена палітра із зображення	Теорія гармоній	Експорт палітр	Перевірка доступності
Coolors	Наявне автоматичне генерування	Немає. Тільки ручний вибір кольору	Частково. Є базові варіанти: аналогічна, комплементарна	Наявний (PNG, PDF, SVG, SCSS тощо)	Наявна перевірка контрастності тексту
Adobe Color	Наявне напівавтоматичне генерування, залежить від режиму	Є. Можна витягнути палітру з зображення	Підтримуються всі основні схеми. тріадна, тетрада, монохром,	Наявний (інтеграція з Creative Cloud)	Наявне моделювання порушень зору

	вибору гармонії		аналогічна тощо		
Paletton	Наявне лише ручне обертання колірного кола	Немає. Відсутня підтримка імпорту зображень	Має широкий спектр схем та ручне налаштування	Наявний (CSS, HTML, текстовий формат)	Відсутня перевірка контрасту

Отже, провівши аналіз, можна зрозуміти, що жоден із онлайн-генераторів не охоплює повного спектра необхідних функцій. Адже деякі зосереджені на зручності та швидкості, інші — на професійних інструментах і теорії кольору. Також спостерігається нестача комплексної перевірки доступності, гнучкого експорту та роботи з зображеннями в одному рішенні. Все це відкриває можливості для створення нового веб-додатку, який об’єднає ключові переваги існуючих сервісів і забезпечить універсальний інструмент для дизайнерів.

- Інструменти стиснення графіки (TinyPNG, Squoosh, ImageOptim)

У сучасному веб-дизайні важливо не лише створити естетично привабливу графіку, а й забезпечити її ефективне завантаження без втрати якості. Оптимізація зображень – ключ до швидкої роботи веб-додатків і кращого користувацького досвіду. На ринку існує багато інструментів для стиснення графіки, серед яких особливо виділяються TinyPNG, Squoosh та ImageOptim.

TinyPNG – це онлайн-сервіс, який дозволяє зменшувати вагу зображень з форматом PNG та JPG. Він використовує алгоритми стиснення, що видаляють надлишкові дані зображення без помітної втрати якості. Сервіс особливо зручний завдяки простому інтерфейсу. Крім того, розробники можуть скористатися API для автоматизації обробки.

Squoosh – веб-інструмент від Google, який орієнтований на більш досвідчених користувачів. Його особливість полягає у роботі безпосередньо в браузері завдяки WebAssembly. Користувачі можуть вручну налаштовувати

параметри стиснення – обирати кодеки, знижувати якість, змінювати роздільну здатність і кількість кольорів. Інтерфейс надає порівняння “до/після” у реальному часі. Squoosh може працювати навіть без інтернету після завантаження сторінки. Все це робить його зручним для роботи з графікою в автономному режимі.

Щодо ImageOptim – це десктопний застосунок для macOS, який призначений для професійної оптимізації зображень. Програма об'єднує кілька алгоритмів стиснення, наприклад, OptiPNG, JPEGoptim, Gifsicle і дозволяє очищати файли від непотрібних метаданих, зокрема EXIF-інформації. EXIF-інформація (Exchangeable Image File Format) – це метадані (дата та час зйомки, модель камери, GPS-координати місця зйомки), які автоматично зберігаються у фотографіях і зображеннях, зазвичай зроблених на цифрові камери або смартфони. Завдяки простому інтерфейсу, ImageOptim активно використовується дизайнерами, які готують графіку до публікації. Він також підтримує інтеграцію з автоматизованими CI/CD-процесами, що є перевагою в командній розробці.

- Порівняння функціональності

Інструмент	Тип	Платформи	Формати	Можливість налаштування	Робота офлайн	Додаткові функції
TinyPNG	Онлайн	Браузер	PNG, JPG	Відсутнє. Автоматично	Ні	API, пакетне завантаження
Squoosh	Онлайн	Браузер	WebP, AVIF, JPG, PNG	Наявне. Кодеки, якість,	Так	Порівняння до/після, підтримка офлайн

				розмір, палітра		
ImageOptim	Десктоп	macOS	PNG, JPG, GIF	Відсутнє. Використовує зовнішні інструменти	Так	Видалення метаданих, інтеграція в CI/CD

Отже, розглянуті інструменти демонструють різний підхід до стиснення зображень: від простоти й швидкості (TinyPNG), до гнучких налаштувань (Squoosh) і професійної обробки (ImageOptim). Варто зазначити, що при розробці власного веб-додатку доцільно поєднати їх сильні сторони. А це – автоматичне стиснення з можливістю ручного налаштування, підтримка сучасних форматів, офлайн-режим і видалення метаданих. Такі дії дозволять підвищити продуктивність веб-сайтів без втрати візуальної якості.

2.2. Аналіз потреб цільової аудиторії

Цільовою аудиторією веб-додатку є веб-дизайнери, UI/UX-спеціалісти, фронтенд-розробники та творці цифрового контенту, які працюють із кольоровими палітрами та 2D-графікою в інтерфейсах. Основною потребою таких користувачів є автоматизація рутинних завдань, зокрема генерації гармонійних колірних рішень та оптимізації зображень для веб-середовища. Дизайнерам важливо швидко отримати привабливі палітри на основі заданого кольору, не заглиблюючись у складні принципи колірної теорії. При цьому значення має можливість вибору типу гармонії та гнучке налаштування кольорів відповідно до стилістики конкретного проєкту.

Окрім цього, значущою є потреба у якісному стисненні зображень без втрати візуальної якості. Це необхідно для підвищення швидкості завантаження веб-сайтів та економії трафіку. Зокрема, користувачі очікують підтримку сучасних форматів і автоматичне видалення зайвих метаданих,

таких як EXIF-інформація. Також важливою є функція попереднього перегляду – це можливість одразу бачити результат генерації палітри або стиснення зображення для швидшого прийняття рішень.

Щодо інтерфейсу, то додаток повинен бути інтуїтивно зрозумілим, не перевантаженим зайвими елементами, оскільки значна частина цільової аудиторії працює у високому темпі або має базовий досвід. Важливо також, щоб додаток дозволяв експортувати результати у зручних форматах, наприклад, CSS, PNG, JSON для подальшого використання у верстці або дизайні. Онлайн-доступність і робота без встановлення на будь-якому пристрої – ще одна критична потреба, тому що фахівці часто працюють на різних платформах.

Отже, дії, які будуть спрямовані на потреби користувачів, зможуть допомогти розробити зручний веб-додаток, що автоматизує роботу веб-дизайнерів.

РОЗДІЛ 3. ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ ВЕБ-ДОДАТКУ

Основною мовою для написання веб-додатку було взято JavaScript. Цей веб-додаток було реалізовано в середовищі WebStorm. Як основу для створення веб-сайту було використано набір стандартних файлів – HTML5 Boilerplate. Додаток є повністю клієнтським (client-side).

3.1. Архітектура системи

- Frontend і Backend

Frontend: HTML-структура має лише один файл index.html, який містить розмітку, стилі, підключення бібліотек і React-компоненти. Це все робить додаток легким для розгортання.

Для побудови інтерфейсу користувача (UI) було використано React. React – це бібліотека, яка допомагає продуктивно і масштабовано створювати якісні веб-застосунки. Одним з ключових понять в React JS є компоненти. Серед компонентів у даному веб-додатку є:

App – головний компонент, який відповідає за управління станом додатку.

Palette – компонент для відображення палітри кольорів.

PaletteGenerator – компонент для генерації палітр.

ImageCompressor – компонент для стиснення зображень.

Усі дії користувача, наприклад, зміна базового кольору або вибір типу палітри, зберігаються у стані через useState. Це дозволяє динамічно змінювати відображення палітри без перезавантаження сторінки. Також було використано React CDN для рендерингу компонентів, а Babel – для трансформування JSX у JavaScript у реальному часі.

Крім того, для стилізації було використано:

Tailwind CSS CDN – для швидкого оформлення сучасного інтерфейсу;

Google Fonts – для забезпечення гарного, читабельного шрифту;

Font Awesome CDN – для створення іконок, наприклад, пензель та палітра.

А для обробки зображень, а саме масштабування та стиснення, було використано Canvas API у компоненті ImageCompressor. Його було обрано через відсутність необхідності у зовнішніх бібліотеках.

Backend: Backend відсутній, адже як було зазначено вище, цей додаток працює виключно на клієнтській стороні. Уся обробка даних виконується в браузері, що знижує вимоги до серверної інфраструктури. Але такі дії обмежують можливості глибокої оптимізації зображень. Для збереження результатів (експорт зображень) використовується браузерний API (navigator.clipboard для копіювання HEX-кодів, <a> для завантаження файлів).

Така архітектура має кілька переваг. По-перше, вона відзначається простотою розгортання. Це значить, що для початку роботи достатньо лише одного HTML-файлу. По-друге, вона має низьку залежність від серверів, що дозволяє значно скоротити витрати на інфраструктуру. Крім того, використання CDN та бібліотеки CSS Tailwind дозволяє швидко створювати прототипи інтерфейсів. У той же час, ця архітектура має певні недоліки. Через відсутність серверних бібліотек можливості оптимізації зображень обмежені. Крім того, залежність від CDN може негативно вплинути на продуктивність у разі поганого інтернет-з'єднання. Ще одним недоліком є відсутність механізмів збереження даних між сеансами, таких як localStorage.

3.2. Структура додатку

Додаток має досить просту та зрозумілу структуру. Він складається з одного HTML-файлу - index.html, який містить React-компоненти, стилі та логіку. Хоча фізично це один файл через використання CDN, проте логічно додаток поділено на декілька модулів.

- Модуль генерації кольорів

Цей модуль відповідає за створення гармонійних палітр на основі базового кольору, мова йдеться про комплементарну, аналогову, тріадну палітри. А також модуль генерації кольорів відповідає за генерацію випадкових палітр: пастельної, темної, яскравої.

Компоненти, які є в цьому модулі:

PaletteGenerator – це головний компонент для генерації палітр. Він керує станами, такими як baseColor, paletteType, pastelPalette, darkPalette, brightPalette. Також цей компонент містить UI для вибору кольору (<input type="color">), типу палітри (кнопки) та генерації випадкових палітр.

Palette – це компонент для відображення палітри. Він приймає масив кольорів (colors) і заголовок (title) та відображає кольорові блоки з HEX-кодами та копіюванням у буфер обміну.

Функції, які є в цьому модулі:

hexToHsl – конвертує HEX-код у HSL для обчислень;

hslToHex – конвертує HSL назад у HEX для відображення;

generatePalette – генерує палітру на основі базового кольору та типу;

generateRandomPalette – створює випадкові палітри з різними стилями (пастельний, темний, яскравий).

- Модуль оптимізації зображень

Цей модуль відповідає за масштабування та стиснення зображень для зменшення розміру файлу, а тим самим оптимізує зображення.

Компоненти, які є в цьому модулі:

ImageCompressor – це головний компонент для обробки зображень. Він керує станами: originalImage, compressedImage, originalSize, compressedSize. Також цей компонет містить UI для завантаження зображення (<input type="file">) та завантаження результату (кнопка).

Функції, які є в цьому модулі:

handleImageUpload – обробляє завантаження зображення, масштабує його та стискає у JPEG;

downloadCompressedImage – завантажує стиснене зображення як файл.

- Модуль збереження та експорту

Цей модуль дозволяє зберігати результати роботи. Наприклад, збереження кольорів та зображень.

Для кольорів:

Копіювання HEX-кодів у буфер обміну відбувається через navigator.clipboard.writeText у компоненті Palette.

Для зображення:

Завантаження стисненого зображення як файлу відбувається через створення <a> з атрибутом download у downloadCompressedImage.

3.3. UI/UX-дизайн

- Основні компоненти інтерфейсу

Основним компонентам інтерфейсу є головне меню. Воно має заголовок, короткий опис та кнопки, які відповідають за вибір: «Генератор палітр» чи «Стиснення зображень».

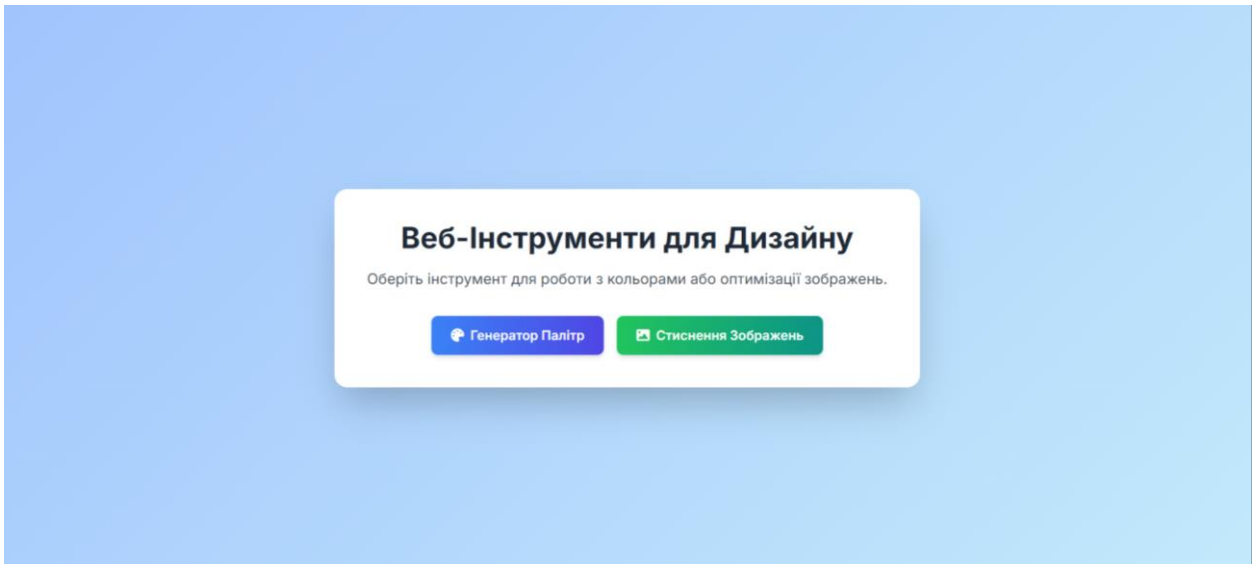

Рис. 3.3.1 – Головне меню

Одним з компонентів інтерфейсу є генератор палітр. Він відповідає за вибір основного кольору, з яким потім буде працювати користувач; за вибір типу палітри: компліментарна, аналогова, тріадна; за створення випадкових палітр: пастельна темна, яскрава; за відображення палітр.

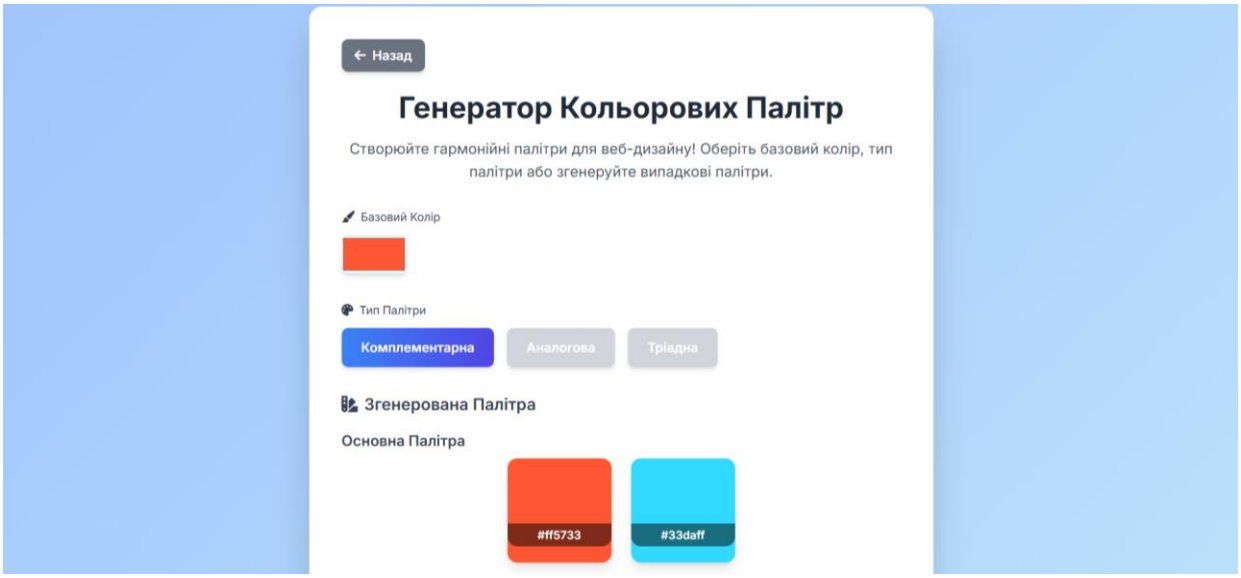

Рис. 3.3.2 – Генератор кольорових палітр: комплементарна, аналогова, тріадна

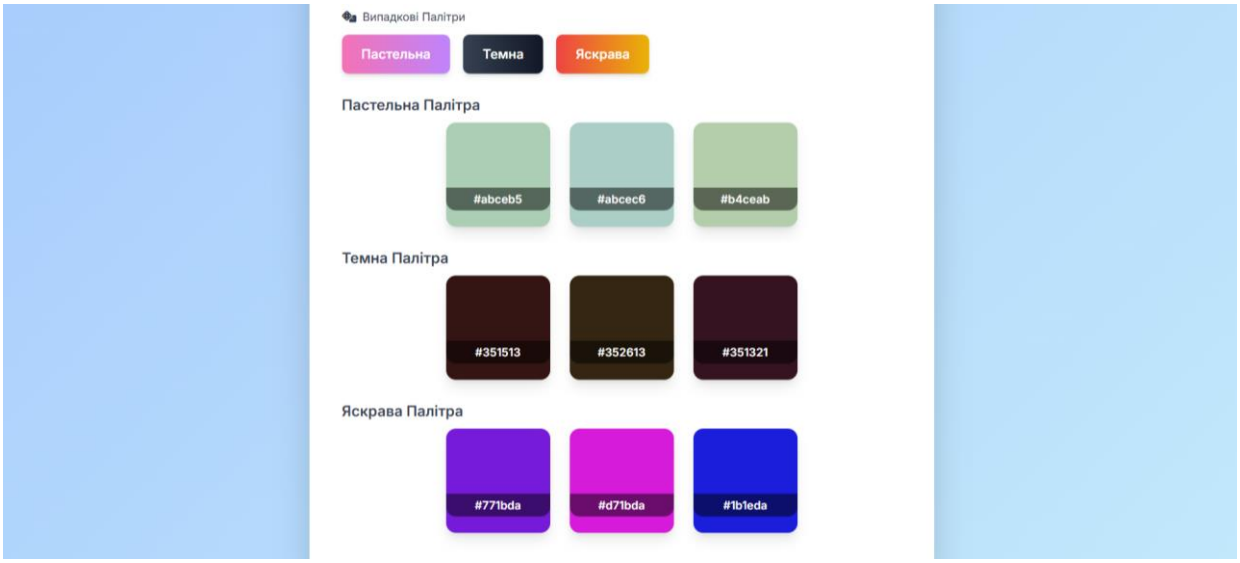

Рис. 3.3.3 – Генератор кольорових палітр: пастельна, темна, яскрава

А щоб повернутися у головне меню з будь-якої сторінки, було створено кнопку «Назад», яка знаходиться у верхньому лівому куті.

Ще одним компонентом є стиснення зображень. Воно відповідає за завантаження зображення для оптимізації; за відображення початкового та оптимізованого зображення; за експорт оптимізованого зображення.

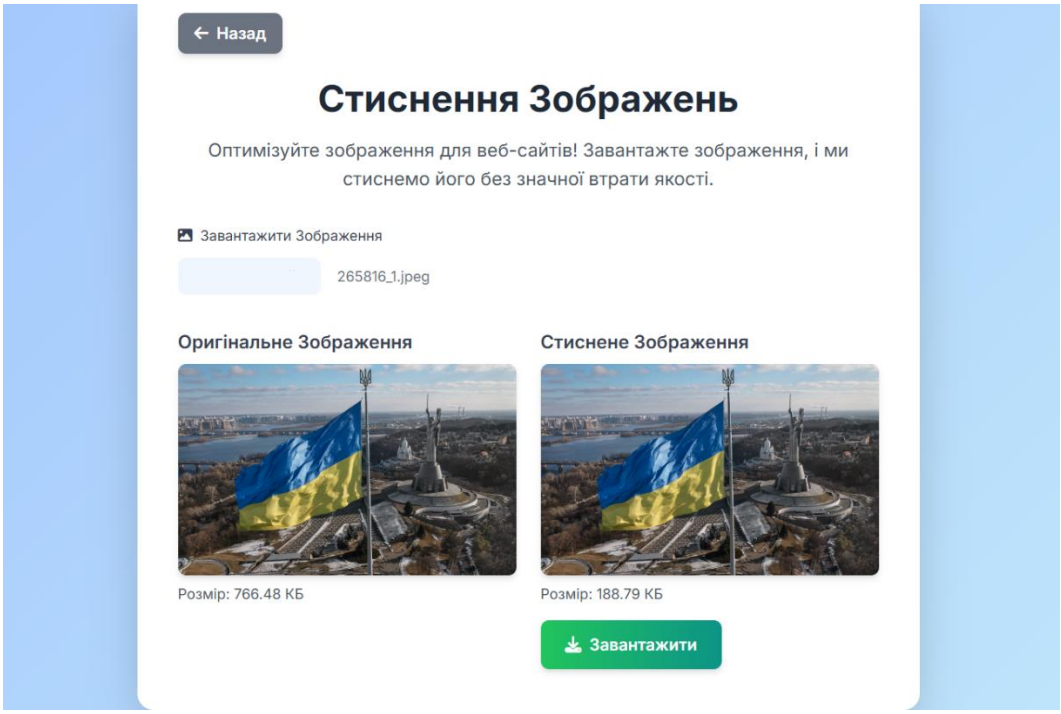

Рис. 3.3.4 – Стиснення зображень

Щодо дизайну, то у веб-додатку використано приємні для зору градієнтні кольори. Вони створюють м’який та сучасний вигляд. Розмір шрифту підібрано так, щоб користувач чітко все бачив. А також дизайн кнопок виконаний градієнтом з додаванням тіней та закругленням кутів. Все це забезпечує доступність та зручність для користувача.

- Забезпечення доступності та зручності

У цьому застосунку особлива увага приділена доступності (Accessibility) та зручності використання (Usability). Для покращення доступності використовується правильна семантика. Тобто структура сторінки будується на тегах <h1>, <h2>, <h3>, що допомагають користувачам краще орієнтуватися на сторінці. Кожне поле вводу (<input>) супроводжується підписом (<label>) із текстом та іконками, що чітко пояснюють та відображають його призначення. Також при наведенні на кнопку застосовуються певні підказки – атрибут title і CSS-клас .tooltip. Крім того, була приділена увага контрастності, адже темний текст на білому фоні забезпечує високу читабельність навіть для користувачів із вадами зору. А адаптивність інтерфейсу реалізована через можливості Tailwind CSS. Це

значить, що завдяки класам flex-wrap, grid-cols-1 і md:grid-cols-2 сторінка добре виглядає як на мобільних пристроях, так і на великих екранах.

Щодо зручності використання (Usability), інтерфейс інтуїтивно зрозумілий і достатньо простий. Іконки пензля, палітри та кубика візуально допомагають швидко розпізнати для чого призначена та чи інша кнопка або дія. Також інтерактивність підвищують анімації – це ефекти при наведенні. Вони роблять елементи "живими" і приємними у використанні. Структура сторінки побудована чітко, бо кожна велика секція, наприклад, вибір базового кольору, тип палітри, генерація випадкових палітр – розділена за допомогою відступів, що покращує візуальне сприйняття. Також передбачено зворотній зв'язок із користувачем: при копіюванні кольору з'являються сповіщення, а також відображається інформація про розміри згенерованих зображень, що дозволяє краще контролювати результат.

3.4. Алгоритм генерації палітр

Алгоритм для генерації палітр здійснюється на основі HSL. HSL (Hue, Saturation, Lightness) – це колірна модель та представлення кольору у веб-розробці. Вона дозволяє легко обчислювати гармонійні кольори шляхом зміни відтінку (Hue).

Для того, щоб алгоритм почав працювати найперше, що треба зробити – це конвертувати HEX у HSL. HSL – колірна модель, яка визначається як рядок із шести шістнадцяткових символів, що представляють кількість червоного, зеленого та синього відповідно. Конвертація робиться для того, щоб полегшити створення гармонійних кольорових палітр, адже колірна модель HSL є більш зручною для маніпуляцій із кольорами порівняно з HEX. Ось приклад коду, в якому наведено як саме відбувається конвертація:

```
function hexToHsl(hex) {  Show usages
  let r = 0, g = 0, b = 0;
  // Якщо формат "#rrggbb", парсимо значення
  if (hex.length === 7) {
    r = parseInt(hex.slice(1, 3), 16) / 255;
    g = parseInt(hex.slice(3, 5), 16) / 255;
    b = parseInt(hex.slice(5, 7), 16) / 255;
  }
  // Знаходимо максимум і мінімум серед RGB
  const max = Math.max(r, g, b), min = Math.min(r, g, b);
  let h, s, l = (max + min) / 2;

  if (max === min) {
    h = s = 0;
  } else {
    const d = max - min;
    s = l > 0.5 ? d / (2 - max - min) : d / (max + min);
    // Визначаємо відтінок на основі того, яка складова максимальна
    switch (max) {
      case r: h = (g - b) / d + (g < b ? 6 : 0); break;
      case g: h = (b - r) / d + 2; break;
      case b: h = (r - g) / d + 4; break;
    }
    h /= 6;
  }
  return [Math.round(h * 360), Math.round(s * 100), Math.round(l * 100)];
}
```

Рис. 3.4.1 – Конвертація HEX у HSL

Наступне, що відбувається – це генерація палітр. За це відповідає компонент generatePalette. Генерація палітр поділяється на 3 типи: компліментарна (кольори, які створюють найбільший контраст), аналогова (кольори, які знаходяться поруч на колірному колі), тріадна (кольори, які створюють збалансований контраст). Ось приклад коду, який показує як саме відбувається обчислення:

```
// Генерація палітри кольорів на основі типу: complementary, analogous, triadic
function generatePalette(baseColor, type) {  Show usages
  const [h, s, l] = hexToHsl(baseColor);
  let colors = [baseColor]; // Масив палітри починається з базового кольору

  if (type === 'complementary') {
    // Додаємо колір, який знаходиться напроти на колірному колі
    colors.push(hslToHex((h + 180) % 360, s, l));
  } else if (type === 'analogous') {
    // Додаємо кольори, близькі за відтінком
    colors.push(hslToHex((h + 30) % 360, s, l));
    colors.push(hslToHex((h - 30 + 360) % 360, s, l));
  } else if (type === 'triadic') {
    // Додаємо кольори, що утворюють рівносторонній трикутник на колірному колі
    colors.push(hslToHex((h + 120) % 360, s, l));
    colors.push(hslToHex((h - 120 + 360) % 360, s, l));
  }

  return colors;
}
```

Рис. 3.4.2 – Генерація палітр

Наступний етап – це генерація випадкових різних палітр. За це відповідає компонент generateRandomPalette. Він генерує випадкові кольорові

палітри: пастельна, темна, яскрава. Таких генерацій можна зробити безліч. Ось приклад коду, що демонструє алгоритм виконання:

```
// Генерація випадкових гармонійних палітр різних стилів (pastel, dark, bright)
function generateRandomPalette(type) {  Show usages
  const randomHue = Math.floor(Math.random() * 360);
  let s, l;

  // Встановлюємо діапазон насиченості і яскравості залежно від типу
  if (type === 'pastel') {
    s = Math.floor(Math.random() * 20) + 20; // 20-40%
    l = Math.floor(Math.random() * 20) + 70; // 70-90%
  } else if (type === 'dark') {
    s = Math.floor(Math.random() * 20) + 40; // 40-60%
    l = Math.floor(Math.random() * 20) + 10; // 10-30%
  } else if (type === 'bright') {
    s = Math.floor(Math.random() * 20) + 70; // 70-90%
    l = Math.floor(Math.random() * 20) + 40; // 40-60%
  }

  // Створюємо три кольори з трохи зсунутим відтінком
  return [
    hslToHex(randomHue, s, l),
    hslToHex((randomHue + 30) % 360, s, l),
    hslToHex((randomHue - 30 + 360) % 360, s, l)
  ];
}
```

Рис. 3.4.3 – Генерація випадкових різних палітр

Однак, після кожної генерації палітр відбувається конвертація з HSL у HEX. Це робиться для того, щоб результати обчислень гармонійних кольорових палітр, виконаних у моделі HSL, можна було використати у веб-інтерфейсі та передати користувачу у зрозумілому та стандартному форматі. Ось приклад коду, що демонструє логіку цієї функції:

```
// Конвертація HSL у HEX - це зворотній процес до hexToHsl
function hslToHex(h, s, l) {  Show usages
  s /= 100;
  l /= 100;
  let c = (1 - Math.abs(2 * l - 1)) * s;
  let x = c * (1 - Math.abs((h / 60) % 2 - 1));
  let m = l - c / 2;
  let r = 0, g = 0, b = 0;

  // Вибираємо сегмент кола для визначення RGB
  if (0 <= h && h < 60) { r = c; g = x; b = 0; }
  else if (60 <= h && h < 120) { r = x; g = c; b = 0; }
  else if (120 <= h && h < 180) { r = 0; g = c; b = x; }
  else if (180 <= h && h < 240) { r = 0; g = x; b = c; }
  else if (240 <= h && h < 300) { r = x; g = 0; b = c; }
  else if (300 <= h && h < 360) { r = c; g = 0; b = x; }

  // Перетворення RGB в HEX
  r = Math.round((r + m) * 255).toString(16).padStart(2, '0');
  g = Math.round((g + m) * 255).toString(16).padStart(2, '0');
  b = Math.round((b + m) * 255).toString(16).padStart(2, '0');
  return `#${r}${g}${b}`;
}
```

Рис. 3.4.4 – Конвертація з HSL у HEX

3.5. Алгоритм оптимізації зображень

Користувач завантажує зображення через <input type="file">. У цьому веб-додатки підтримуються PNG та JPEG формати. Файл зчитується як Data URL за допомогою FileReader. Data URL — це спеціальний текстовий формат, що містить саме зображення в кодованому вигляді. Потім створюється об'єкт Image для отримання його розмірів.

Якщо ширина зображення більша за висоту та перевищує 1920 пікселів, або висота перевищує 1080 пікселів, зображення масштабується пропорційно. Потім воно малюється на canvas, яке змінює розмір до нового масштабу.

Для стиснення canvas перетворюється у формат JPEG з якістю 80% (canvas.toDataURL('image/jpeg', 0.8)). Оригінальне та стиснуте зображення зберігаються, їх розміри відображаються в КБ. Для завантаження створюється посилання <a download>, яке дозволяє зберегти стиснуте зображення.

Ось як це відбувається в коді:

```
// Компонент стиснення зображень
function ImageCompressor() {  Show usages
  const [originalImage, setOriginalImage] = useState(null);
  const [compressedImage, setCompressedImage] = useState(null);
  const [originalSize, setOriginalSize] = useState(0);
  const [compressedSize, setCompressedSize] = useState(0);

  const handleImageUpload = (e) => {  Show usages
    const file = e.target.files[0];
    if (file) {
      const reader = new FileReader();
      reader.onload = (event) => {
        const img = new Image();
        img.onload = () => {
          // Обмеження розміру зображення
          const MAX_WIDTH = 1920;
          const MAX_HEIGHT = 1080;
          let width = img.width;
          let height = img.height;

          if (width > height) {
            if (width > MAX_WIDTH) {
              height = Math.round((height * MAX_WIDTH) / width);
              width = MAX_WIDTH;
            }
          } else {
            if (height > MAX_HEIGHT) {
```

Рис. 3.5.1 – Стиснення зображень (1)

```
        if (height > MAX_HEIGHT) {
          width = Math.round((width * MAX_HEIGHT) / height);
          height = MAX_HEIGHT;
        }
      }

      // Створення Canvas для стиснення
      const canvas = document.createElement('canvas');
      canvas.width = width;
      canvas.height = height;
      const ctx = canvas.getContext('2d');
      ctx.drawImage(img, 0, 0, width, height);

      // Стиснення з якістю 0.8
      const compressedDataUrl = canvas.toDataURL('image/jpeg', 0.8);
      setOriginalImage(event.target.result);
      setCompressedImage(compressedDataUrl);
      setOriginalSize(file.size);
      setCompressedSize(Math.round(compressedDataUrl.length * 0.75)); // Приблизний розмір у байтах
    };
    img.src = event.target.result;
  };
  reader.readAsDataURL(file);
}
```

Рис. 3.5.2 – Стиснення зображень (2)

```
const downloadCompressedImage = () => {  Show usages
  const link = document.createElement('a');
  link.href = compressedImage;
  link.download = 'compressed_image.jpg';
  link.click();
};
```

Рис. 3.5.3 – Стиснення зображень (3)

Бібліотеки, які були використані для оптимізації зображень: Canvas API та FileReader API. Canvas API – це вбудована в браузер технологія, яка не потребує додаткових бібліотек або залежностей. Вона дозволяє малювати зображення, виконувати їх масштабування за допомогою методу drawImage, а також конвертувати вміст у різні формати, наприклад, у базовий рядок через toDataURL. Завдяки цьому можна змінювати розміри зображення прямо у браузері та створювати стиснені копії без звернення до сервера. А FileReader API використовується для читання завантаженого користувачем файлу. Він дозволяє перетворити файл у Data URL – спеціальний текстовий формат, який можна відразу використати в коді для перегляду або обробки зображення в браузері.

ВИСНОВКИ

У ході дослідження було розглянуто ключові теоретичні та практичні аспекти генерації кольорових палітр та оптимізації 2D-графіки у веб-дизайні. Аналіз основ колірної теорії дозволив встановити, що грамотний підбір кольорів – це не лише питання естетики, а й ефективний інструмент впливу на емоційне сприйняття користувача. Типи гармоній кольорів, знання особливостей кольорового кола та психології кольору стали фундаментом для формування палітр, що гармонійно інтегруються у структуру інтерфейсів.

До того ж дослідження форматів графіки, типів зображень та технік оптимізації продемонструвало важливість раціонального підходу до вибору та обробки графічного контенту. Стиснення зображень, вибір відповідного формату та підтримка адаптивності стали необхідними умовами для забезпечення швидкого завантаження сторінок і якісного користувацького досвіду.

А ось практична реалізація веб-додатку об’єднала теоретичні засади з інженерною практикою. Було реалізовано модулі генерації кольорових палітр, оптимізації зображень, збереження та експорту, що дозволило забезпечити інтерактивну взаємодію користувача з дизайном. Особлива увага приділялася архітектурі системи, зручності інтерфейсу (UI/UX), доступності та ефективності роботи з графікою.

Таким чином, дослідження підтвердило, що кольорова палітра і графіка – це не другорядні елементи, а ключові складові веб-дизайну. Їх генерація, адаптація та оптимізація можуть і повинні бути автоматизованими – це не лише підвищує якість і продуктивність, а й відкриває нові горизонти для творчості у цифровому середовищі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Color wheel. Encyclopaedia Britannica. [Електронний ресурс]. – Режим доступу: https://www.britannica.com/science/color-wheel – Дата звернення: 12.11.2024.
2. Tomal R. Design Value: Introduction. [Електронний ресурс]. – Режим доступу: https://rafaltomal.com/class/intro/design-value/ – Дата звернення: 14.11.2024.
3. Color Psychology of Design – Medium [Електронний ресурс]. – Режим доступу: https://medium.com/design-bootcamp/color-psychology-of-design-3087ace48435 – Дата звернення: 10.01.2025.
4. Psychology of Color in Graphic Design – Platt College [Електронний ресурс]. – Режим доступу: https://platt.edu/blog/psychology-color-graphic-design/ – Дата звернення: 29.01.2025.
5. TechTarget – Network Encyclopedia [Електронний ресурс]. – Режим доступу: https://www.techtarget.com/network/ – Дата звернення: 17.02.2025.