МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВОЛИНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ЛЕСІ УКРАЇНКИ
Кафедра комп'ютерних наук та кібербезпеки

На правах рукопису

ВАСЮТА ДАРІЯ ВОЛОДИМИРІВНА

РОЗРОБКА ВЕБДОДАТКУ ДЛЯ СММ-СПЕЦІАЛІСТІВ ЗА ДОПОМОГОЮ ТЕХНОЛОГІЇ NEXT.JS

Спеціальність: 122 Комп'ютерні науки
Освітньо-професійна програма: Комп'ютерні науки та інформаційні технології
Кваліфікаційна робота на здобуття освітнього ступеня «бакалавр»

Науковий керівник:
ГЛИНЧУК ЛЮДМИЛА ЯРОСЛАВІВНА,
Кандидат фіз.-мат наук, доцент кафедри комп'ютерних наук та кібербезпеки,

РЕКОМЕНДОВАНО ДО ЗАХИСТУ
Протокол №_____
засідання кафедри комп'ютерних наук та кібербезпеки від
_____
2025 р.
Завідувач кафедри

(_____) Гришанович Т. О.

ЛУЦЬК 2025

ЗМІСТ

ВСТУП

Актуальність теми. У сучасному світі соціальні медіа стали невід'ємною частиною життя людей та бізнесу, маючи значний вплив на формування іміджу брендів та взаємодію з цільовою аудиторією. SMM-спеціалісти відіграють ключову роль у просуванні брендів, продуктів та послуг через різні соціальні платформи, таких як Instagram, Facebook, Twitter, TikTok та інші. Вони відповідальні за створення контенту, моніторинг ефективності кампаній, взаємодію з підписниками та підтримку високого рівня залученості.

Розроблений програмний засіб є достатньо актуальним, оскільки процес створення та редагування контенту для соціальних мереж часто вимагає значних зусиль і часу. Багато існуючих інструментів для редагування контенту орієнтовані на окремі задачі, що часто не задовольняє усі потреби СММ-спеціалістів. Для досягнення бажаного результату вони змушені використовувати кілька різних програм або платформ, що збільшує час на обробку контенту та ускладнює робочий процес. Це знижує ефективність робочих процесів та може призводити до втрати часу та ресурсів.

У той же час, автоматизація та інтеграція функцій редагування контенту в одному інструменті здатні значно спростити робочий процес та підвищити продуктивність спеціалістів. Розробка спеціалізованого вебдодатку, який би полегшив цей процес, є важливим кроком для оптимізації робочого процесу СММ-спеціалістів і забезпечення зручного та швидкого виконання завдань.

Метою роботи є розробка інструменту, що сприятиме оптимізації робочого часу СММ-спеціалістів, об'єднуючи в одному програмному засобі ключові функції, необхідні для створення, редагування та планування контенту. Інструмент має забезпечити інтуїтивно зрозумілий інтерфейс для роботи з публікаціями, формування шаблонів візуального стилю профілю, налаштування кольорових схем відповідно до брендових вимог або особистих уподобань.

Для досягнення цієї мети необхідно виконати такі **завдання**:

- провести аналіз предметної галузі та існуючих рішень;

- визначити оптимальні технології для реалізації;
- розробити архітектуру програмного засобу;
- вибрати інструментальні засоби для розробки;
- реалізувати функціонал вебдодатку;
- провести тестування програмної розробки;
- надати рекомендації по впровадженню та використанню продукту.

Об’єкт дослідження – процеси створення, редагування та планування контенту для соціальних мереж, які спрямовані на оптимізацію роботи SMM-спеціалістів.

Предмет дослідження – інструмент для створення, редагування та планування контенту, який об’єднує функціональні можливості в одному програмному засобі.

Апробація результатів роботи. Результати дослідження опубліковані у вигляді тез на тему «Розробка вебдодатку для SMM-спеціалістів на основі Next.js», які увійшли до збірника матеріалів II Міжнародної науково-практичної конференції «Проблеми комп’ютерних наук, програмного моделювання та безпеки цифрових систем». Конференція проходила 9-11 червня 2025 року на базі практик табору «Гарт» Волинського національного університету імені Лесі Українки (Шацький район, с. Світязь).

РОЗДІЛ 1
РОЗВИТОК ЦИФРОВИХ ІНСТРУМЕНТІВ ДЛЯ СТВОРЕННЯ КОНТЕНТУ ТА ЇХ ЗАСТОСУВАННЯ У МАРКЕТИНГУ СОЦІАЛЬНИХ МЕРЕЖ

1.1. Історія розвитку графічних редакторів та їх застосування у сфері SMM

Графічний редактор – це прикладна програма, призначена для створення й обробки графічних зображень на комп'ютері. Вона дозволяє створені зображення записувати у файл, а також посилати зображення на пристрій виведення. [1]

Перші графічні редактори були вкрай обмеженими за можливостями і мали лише базові інструменти для малювання та редагування простих зображень. Одним із перших прикладів був редактор Skethpad (рис. 1.1), створений в 1960-х роках, який дозволяв користувачам малювати на екрані комп'ютера за допомогою графічного планшета. Цей редактор мав великий вплив на подальший розвиток графічної комп'ютерної графіки, оскільки дозволяв створювати векторні зображення, що стало основою для багатьох майбутніх інструментів. [2]

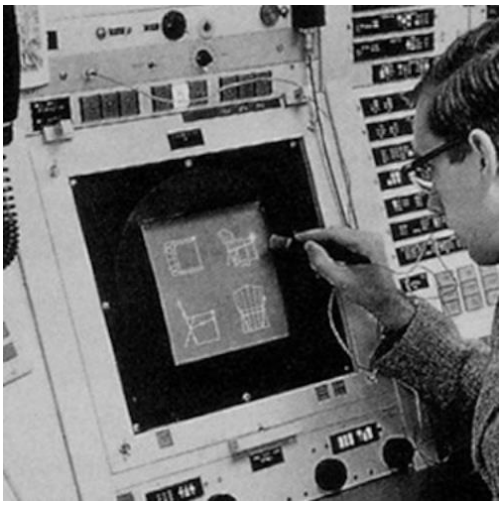


Рисунок 1.1 – Редактор Sketchpad

У 1970-1980-х роках технології розвивалися, і графічні редактори ставали все більш доступними для дизайнерів і художників. Основні розробки в цей період включали появу перших растрових графічних редакторів, таких як Paint на системі Microsoft Windows. [3] Microsoft Paint (рис.1.2) – це один із найвідоміших і найпростіших графічних редакторів, який входить до складу операційної системи Windows із 1985 року. Він розроблявся як базовий інструмент для роботи з растровими зображеннями і дозволяв користувачам створювати та редагувати малюнки за допомогою простих інструментів, таких як олівець, пензлі, лінії, фігури та заливка кольором.

Хоча Paint не мав просунутих функцій професійних графічних редакторів, він став популярним серед користувачів завдяки своїй доступності та простоті використання. Багато людей починали знайомство з цифровою графікою саме через цей редактор. Paint також підтримував збереження зображень у форматах BMP, JPG, PNG та GIF, що робило його універсальним для базових завдань редагування. [4]

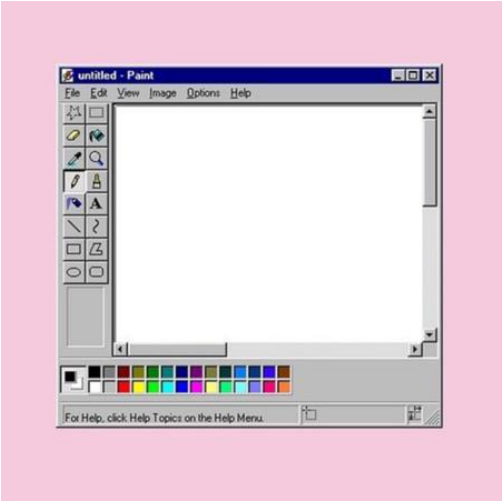


Рисунок 1.2 – Графічний редактор Paint

У 1990-х роках індустрія графічних редакторів пережила справжній бум завдяки розвитку персональних комп'ютерів. В цей час з'являються такі популярні програми, як Adobe Photoshop (рис.1.3), CorelDRAW, PaintShop Pro,

які дозволяли працювати як з растровою, так і з векторною графікою. З'являються нові можливості для редагування зображень, включаючи інструменти для корекції кольору, ретуші, змінюючи розмір і форму об'єктів. Цей період став основою для того, що ми зараз маємо як цифрові графічні редактори, які є невід'ємною частиною роботи дизайнерів, маркетологів та художників. [3]

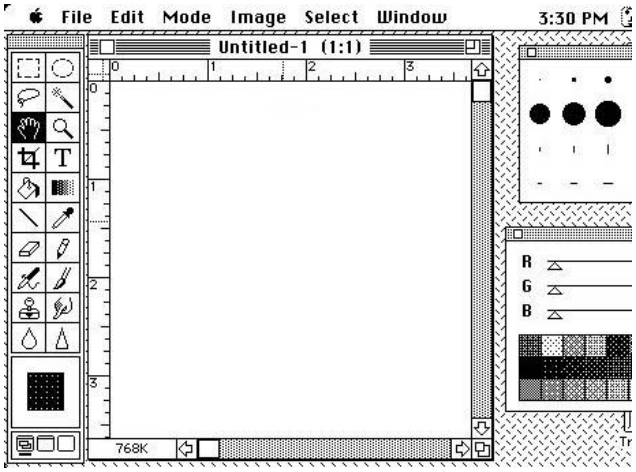


Рисунок 1.3 – Графічний редактор Adobe Photoshop

Розвиток інтернет-технологій сприяв появі онлайн-редакторів, які дозволяють працювати над дизайном безпосередньо в браузері, спрощуючи процес створення графіки та роблячи його доступним для широкого кола користувачів. Одним із таких інструментів є Figma (рис.1.4), розробка якої розпочалася у 2012 році Діланом Філдом та Еваном Воллесом під час їхнього навчання в Університеті Брауна. Figma була офіційно запущена для публічного використання у вересні 2016 року. [5]

Іншим популярним онлайн-редактором є Canva, заснована Мелані Перкінс у 2013 році в Сіднеї, Австралія. Цей сервіс надає користувачам доступ до вбудованої бібліотеки шаблонів, стокових фотографій, ілюстрацій та шрифтів, що дозволяє легко створювати графічний контент. Canva орієнтована як на звичайних користувачів, так і на професіоналів у сфері дизайну та цифрового маркетингу. [6]

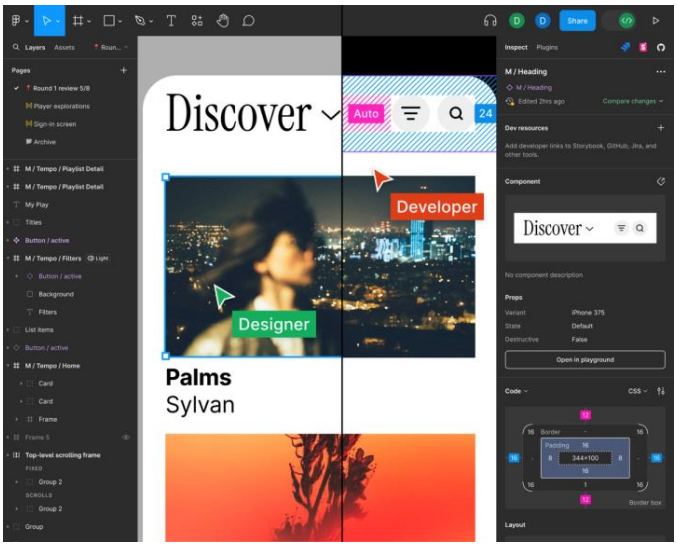


Рисунок 1.4 – Онлайн-редактор Figma

Сучасні графічні редактори інтегруються з іншими програмами та сервісами, надаючи можливість зберігати файли в хмарі, автоматично оновлюватися та працювати колективно над проєктами. Багато з них використовують штучний інтелект для автоматизації рутинних завдань, таких як ретушування або заміна фону. Наприклад, Figma пропонує функцію спільної роботи в реальному часі, що дозволяє дизайнерам одночасно редагувати проєкти та обмінюватися ідеями.

Таким чином, графічні редактори еволюціонували від простих інструментів для малювання до комплексних систем, що задовольняють потреби як початківців, так і професіоналів у різних сферах діяльності.

1.2. Графічний редактор як інструмент для роботи СММ-спеціаліста

У сфері маркетингу графічні редактори стали незамінними інструментами для створення візуального контенту, який привертає увагу споживачів. Вони дозволяють розробляти рекламні матеріали, банери, плакати, що підсилюють вплив маркетингових кампаній. [7]

Візуальний контент є ефективним способом донесення інформації, оскільки людський мозок обробляє зображення набагато швидше, ніж текст. Саме тому графічні матеріали допомагають підкреслити головні переваги

продукту або послуги, сформувати впізнаваний імідж бренду та викликати емоційний відгук у цільової аудиторії. Завдяки сучасним графічним редакторам, компанії можуть створювати якісний контент без необхідності залучати професійних дизайнерів, що значно економить час і бюджет. [7]

З розвитком соціальних мереж та появою професії SMM-спеціаліста, роль графічних редакторів у маркетингу значно зросла. Платформи, такі як Facebook, Instagram, Twitter, LinkedIn, TikTok та Pinterest, стали основними каналами для просування брендів, де візуальний контент відіграє ключову роль. Успішні компанії активно використовують різні типи графічного оформлення для залучення аудиторії, включаючи рекламні банери, інфографіку, анімовані GIF-файли, каруселі зображень та відеоконтент. Візуальні елементи не тільки допомагають виділитися серед конкурентів, а й сприяють підвищенню рівня залученості, покращенню комунікації з клієнтами та збільшенню конверсій. Особливо ефективним є адаптивний контент, який оптимізується під різні платформи та пристрої, забезпечуючи максимальну ефективність рекламних кампаній. [8]

Перспективи розвитку графічних редакторів у маркетингу соціальних мереж продовжують зростати завдяки штучному інтелекту та автоматизації процесів. Новітні інструменти дозволяють створювати контент швидше, використовуючи шаблони, генерацію дизайну на основі трендів та інтеграцію з платформами аналітики. Це дає можливість персоналізувати рекламу, адаптувати її під різні сегменти аудиторії та підвищувати ефективність просування бренду. [9]

1.3. Характеристика сучасних інструментів для роботи СММ-спеціаліста

Ефективне використання сучасних інструментів є ключовим аспектом успішного просування брендів у соціальних мережах. Використання спеціалізованих сервісів допомагає СММ-спеціалістам планувати, аналізувати та

вдосконалювати стратегії контент-маркетингу, що значно підвищує ефективність роботи.

Зараз розглянемо основні види платформ та сервісів, які допомагають створювати якісний контент, автоматизувати процеси та покращувати ефективність роботи СММ-спеціаліста:

Платформи для редагування відео дозволяють швидко обробляти відеоконтент без необхідності володіння професійними навичками. Вони забезпечують можливість обрізки відео, додавання спецефектів, корекції кольору, накладання музики, використання переходів та текстових вставок. Деякі сервіси мають функцію автоматичного створення відео на основі шаблонів, що значно економить час і дозволяє адаптувати контент для різних платформ

Календарі-планери для контенту допомагають SMM-спеціалістам організовувати розклад публікацій, підтримувати періодичність постингу та уникати хаотичного розміщення матеріалів. Планери дозволяють заздалегідь визначити теми постів, ключові повідомлення, часові рамки для публікацій та розподіляти завдання між членами команди. Інтеграція з платформами автопостингу забезпечує стабільний вихід контенту навіть у разі відсутності спеціаліста онлайн.

Сервіси для збору трендового контенту аналізують популярні теми, найкращі візуальні рішення та допомагають визначити, які пости викликають найбільший інтерес у користувачів. Такі інструменти дозволяють SMM-фахівцям швидко адаптувати успішні стратегії та створювати контент, що відповідає актуальним тенденціям.

Графічні редактори дозволяють редагувати зображення, змінювати кольорові параметри, додавати фільтри та створювати візуально привабливі публікації. Багато редакторів містять готові шаблони для соціальних мереж, що спрощує процес створення професійного контенту.

Застосунки для створення колажів дають можливість комбінувати кілька зображень в одну композицію, додаючи текстові елементи, графіку та ефекти. Це зручно для оформлення рекламних постів, анонсів та інтерактивного контенту.

Застосунки для автоматичного додавання субтитрів значно покращують доступність відеоконтенту, особливо для користувачів, які переглядають відео без звуку. Такі сервіси автоматично розпізнають мовлення, створюють текстову розшифровку та синхронізують її з відео.

Платформи для автоматизації постингу допомагають СММ-спеціалістам підтримувати регулярну активність у соціальних мережах без необхідності щоденного ручного розміщення постів. Ці сервіси дозволяють налаштувати розклад публікацій, аналізувати ефективність контенту та оптимізувати стратегію просування.

Інструменти аналітики дають змогу відстежувати ефективність контенту за такими показниками, як залученість, перегляди, кліки та конверсії. Використання таких сервісів допомагає швидко реагувати на зміни в поведінці аудиторії та вносити корективи в контент-стратегію.

Онлайн-платформи для управління проєктами забезпечують координацію всіх процесів, пов’язаних зі створенням контенту. Завдяки цим сервісам можна ставити завдання, відстежувати їхнє виконання, обговорювати правки та працювати над матеріалами в режимі реального часу. [10]

Таким чином, застосування сучасних інструментів у сфері СММ дозволяє не лише спрощувати роботу спеціалістів, а й значно покращувати кінцевий результат. Це сприяє ефективнішому просуванню бренду, підвищенню залученості аудиторії та досягненню маркетингових цілей. У сучасних умовах конкуренції впровадження таких рішень є необхідністю для успішного ведення соціальних мереж.

1.4. Архітектура вебдодатків для створення та планування контенту

Архітектура вебдодатків для створення та планування контенту має на меті забезпечити ефективний процес управління контентом та взаємодію з

соціальними мережами. Зазвичай такі додатки складаються з кількох основних компонентів: фронтенду, бекенду та бази даних. Фронтенд забезпечує інтерфейс для користувачів, де вони можуть створювати та редагувати контент, вибирати шаблони для публікацій та налаштовувати графік публікацій. Бекенд відповідає за обробку запитів, управління користувачами, інтеграцію з API соціальних мереж та зберігання даних у базах даних. База даних містить інформацію про пости, користувачів, розклад публікацій та історію взаємодій з платформами.

Особливістю таких вебдодатків є інтеграція з різними соціальними мережами та API, що дозволяє автоматизувати процес публікацій і планування контенту. Це може включати інтерфейси для управління акаунтами в Instagram, Facebook, Twitter, LinkedIn та інших платформах. Бекенд також забезпечує функціональність для відкладеного постингу, що дозволяє користувачам планувати публікації заздалегідь і автоматично публікувати їх у визначений час. Важливим елементом є інтеграція з календарними системами, такими як Google Calendar чи Outlook, для синхронізації публікацій із особистим або командним графіком.

Вимоги до продуктивності вебдодатків для SMM включають високий рівень масштабованості, безперервну доступність та швидку обробку даних. Оскільки такі додатки можуть обробляти велику кількість запитів одночасно (особливо в періоди активних кампаній), вони повинні підтримувати навантаження та забезпечувати швидку реакцію на дії користувачів. Крім того, безпека є важливим аспектом, оскільки вебдодатки для SMM часто взаємодіють з приватними акаунтами користувачів і конфіденційною інформацією.

Загальна архітектура вебдодатку має вигляд, показаний на діаграмі нижче(рис.1.5).

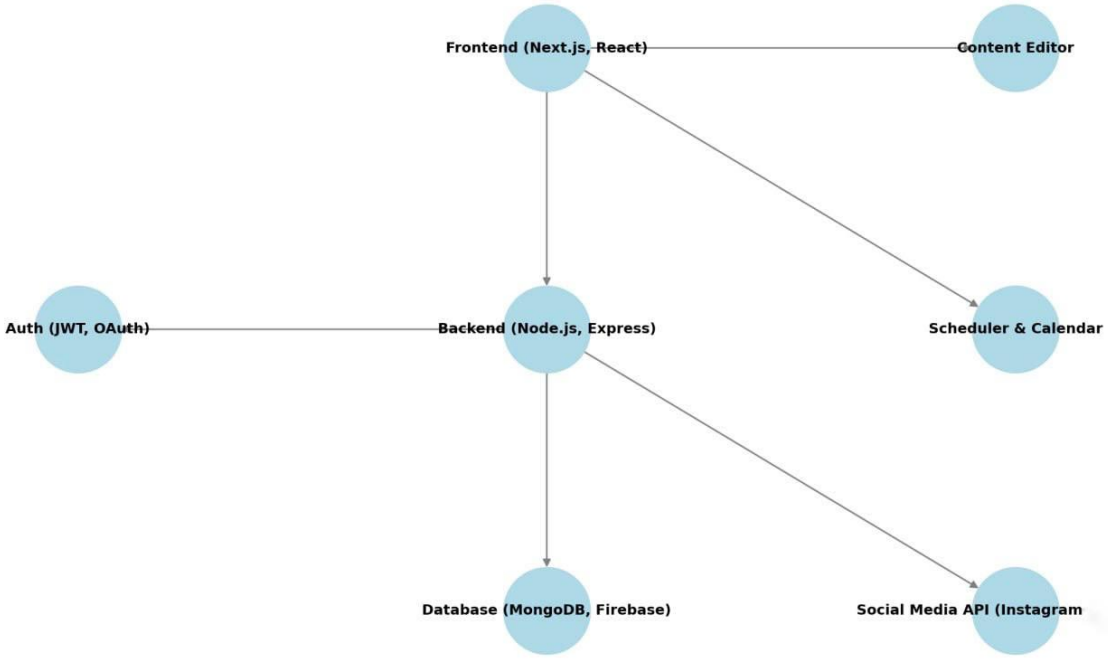


Рисунок 1.5 – Архітектура вебдодатку для СММ

1.5. Огляд і аналіз аналогічних програмних розробок

У сфері SMM існує широкий вибір програмних рішень, що допомагають спеціалістам оптимізувати робочі процеси, підвищити ефективність контент-стратегії. Аналіз аналогічних програмних розробок дозволяє оцінити їхні можливості, функціональність та відмінності, що допомагає обрати найкраще рішення визначення потреб спеціалістів у роботі з контентом та автоматизацією процесів.

Canva (рис.1.5) – це зручний онлайн-інструмент для графічного дизайну, що дозволяє створювати візуальні матеріали, такі як графіка для соціальних мереж, презентації, плакати тощо. Платформа має інтуїтивно зрозумілий інтерфейс з функцією перетягування елементів, шаблонами та можливістю редагування. Canva також підтримує командну роботу та інструменти для брендування. Це зручно як для початківців, так і для досвідчених дизайнерів.

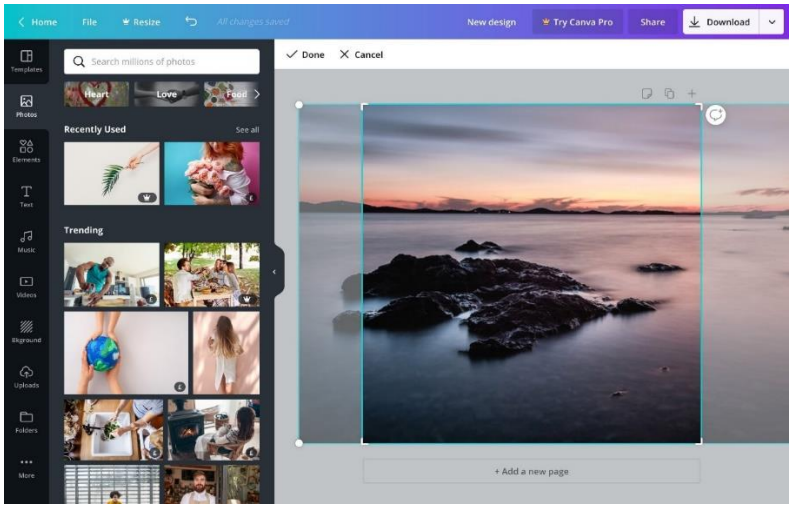


Рисунок 1.5 – Графічний редактор Canva

Розробник: Canva Inc.

Архітектура: Web application, mobile application (iOS and Android).

Мова реалізації: JavaScript, Python, Go.

Функції:

- багатий набір шаблонів для різних типів контенту (пости для соцмереж, презентації, плакати, листівки тощо);
- можливість працювати в команді та спільно редагувати проєкти;
- бібліотека векторних зображень, фотографій, шрифтів, іконок та інших елементів дизайну;
- інструменти для редагування фото, додавання фільтрів, накладання тексту та інших графічних ефектів;
- мобільні додатки для Android та iOS, що дають змогу працювати з дизайнами на ходу.

Переваги:

- безкоштовний доступ до основних інструментів з можливістю покупки преміум-підписки для розширених функцій;
- простота у використанні, підходить як для новачків, так і для досвідчених дизайнерів;
- доступність на різних платформах (веб, мобільний додаток для Android

та iOS). Широкий вибір шаблонів для швидкого старту.

Недоліки:

- обмежена гнучкість для складних або професійних дизайнів, порівняно з більш спеціалізованими програмами;
- преміум-функції та елементи доступні тільки за підпискою.

Джерело інформації: https://www.canva.com/.

My Feed (рис.1.6) – це мобільний додаток для планування та публікації контенту в Instagram. Він дозволяє користувачам візуалізувати пости, редагувати їх, а також планувати публікації за допомогою календаря. Додаток підтримує кілька акаунтів та дозволяє організувати пости на тижні вперед. Він також включає функції для управління профілями та покращення взаємодії з аудиторією.

Розробник: Не відомо.

Архітектура: Mobile application (iOS and Android).

Мова реалізації: Не відомо.

Функції:

- планування та автоматизація публікацій у соцмережах;
- аналітика ефективності контенту;
- інтеграція з популярними платформами (Instagram, Facebook, Twitter);
- оптимізація таймлайну публікацій;
- збереження груп хештегів для швидкого використання.

Переваги:

- інтуїтивний інтерфейс, який підходить для користувачів різного рівня досвіду;
- автоматизацію рутинних завдань, таких як публікації та планування історій;
- інтеграція з платформами, як-от Dropbox і Google Drive;
- інструменти для командної роботи, що полегшують спільне управління акаунтами.

Рисунок 1.6 – Мобільний додаток My Feed

Недоліки: багато функцій доступні лише у платній версії, безкоштовний план має обмеження на кількість постів, а також можуть виникати складнощі через обмеження API Instagram для особистих акаунтів, що впливає на доступність окремих функцій.

Джерело інформації: https://myfeed.app/.

MyFeed добре підходить для малого та середнього бізнесу а також для SMM-менеджерів, які потребують автоматизації роботи з соцмережами. Для більшого ефекту варто інтегрувати цей інструмент із програмами дизайну або аналітики.

Мобільний застосунок MyFeed і графічний редактор Canva є популярними інструментами для SMM, але їх використання окремо вимагає більше часу та ресурсів. MyFeed зосереджений на автоматизації публікацій і аналізі ефективності, тоді як Canva пропонує потужні інструменти для створення графічного контенту.

Об'єднання ключових функцій цих платформ у вебдодаток SMM-Helper зробить його оптимальним рішенням, яке забезпечить ефективність роботи. Це

дозволить SMM-спеціалістам створювати, редагувати й планувати контент швидше, що стане вагомою конкурентною перевагою нового програмного засобу.

РОЗДІЛ 2
РОЗРОБКА ВЕБДОДАТКУ ДЛЯ СММ-СПЕЦІАЛІСТІВ «SMM-HELPER»

2.1. Постановка задачі, призначення та вимоги до вебдодатку «SMM-helper»

Задача розробки вебдодатку «SMM-helper» полягає у створенні функціоналу, який дозволить SMM-спеціалістам зручно та ефективно виконувати щоденні завдання зі створення та планування контенту для соціальних мереж.

Вебдодаток призначений для створення графічного контенту (сторіс, постів), керування кольоровими схемами для оформлення Instagram-профілю, що дозволить користувачам виконувати свої завдання в єдиній системі, зменшити час на рутинні процеси та підвищити візуальну цілісність бренду.

Основними вимогами до розробки є:

- вебдодаток повинен мати широкий набір функцій, зокрема, з можливістю імпорту зображень, роботи з текстом, фігурами, стікерами, емодзі, а також створення власних шаблонів;

- інтерфейс має бути адаптивним і зручним для навігації на різних пристроях, з фокусом на UX-дизайн;

- система повинна містити функції збереження та редагування проєктів, а також імпорту/експорту даних (у форматі JSON);

- у додатку має бути реалізована функціональність створення та керування кольоровими схемами з використанням HEX-кодів та можливістю імпорту зображень у схему;

- забезпечення стабільної взаємодії з мережею Інтернет і базою даних для зберігання інформації про проєкти та користувачів;

- мобільна сумісність – важливо, щоб додаток був зручним для використання зі смартфонів та планшетів;

- наявність системи авторизації користувачів з можливістю входу та виходу з облікового запису.

2.2. Опис проєкту «SMM-helper»

Розробка вебдодатку «SMM-helper» здійснюється у кілька послідовних етапів, які охоплюють весь життєвий цикл створення програмного забезпечення. На початковому етапі проводиться детальне вивчення потреб цільової аудиторії, що дозволяє визначити основні запити користувачів. Це дослідження стало основою для формування функціональних вимог до продукту, серед яких – зручний графічний редактор, підбір кольорових схем, створення шаблонів для постів і стратегій просування, а також функціонал авторизації й збереження персональних налаштувань.

Після цього здійснюється проєктування архітектури вебдодатку, де визначаються основні модулі, способи їхньої взаємодії, формат обміну даними між клієнтською та серверною частинами. Відповідно до функціональних вимог обрано технологічний стек, який включає сучасні інструменти веброзробки: Next.js для клієнтської частини, Node.js для серверної логіки, а також MongoDB або Firebase як систему керування базами даних.

На рисунку В.8 подано архітектурну схему клієнтської частини програмного засобу «SMM-helper», яка відображає ключові елементи інтерфейсу, їхню внутрішню логіку та послідовність дій користувача. Схема починається з етапу авторизації або реєстрації користувача, після чого відбувається перенаправлення до основних модулів застосунку: візуального редактора, вибору шаблонів, генератора палітр тощо. Така логіка побудови інтерфейсу реалізує принцип короткого користувацького шляху (user journey) та мінімізує кількість кроків до досягнення мети. Це сприяє інтуїтивній навігації, зменшенню когнітивного навантаження і покращенню загального користувацького досвіду. Правильність обраного підходу підтверджена результатами опитування серед представників цільової аудиторії, які високо оцінили зручність і зрозумілість інтерфейсу.

Крім архітектурного планування, важливою складовою розробки є структура проєкту, що подана на рисунку 2.1. Вона демонструє логічну організацію каталогів і файлів у межах репозиторію застосунку. Основні функціональні частини (автентифікація, графічний редактор, генерація кольорових схем, інтерфейсні компоненти) розміщені у відокремлених модулях, що відповідає принципам модульності, повторного використання коду та масштабованості. Такий підхід дозволяє зручно розширювати проєкт у майбутньому, спрощує підтримку й забезпечує ефективну командну роботу. Завдяки цьому програмний засіб є стійким до змін, гнучким та зручним у розробці.

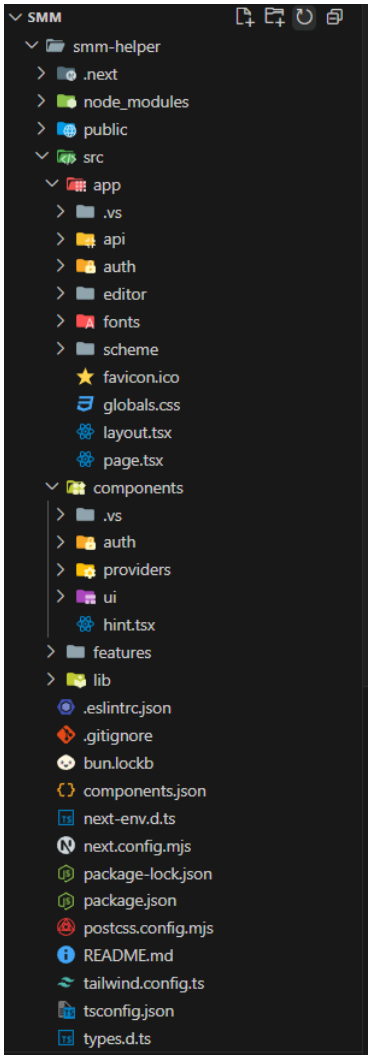


Рисунок 2.1 – Файлова структура проєкту «SMM-Helper»

2.3. Вибір моделі розробки

У розробці програмного засобу «SMM-Helper» було застосовано ітераційну модель розробки. Такий вибір був зумовлений потребою гнучко реагувати на зміну вимог, здійснювати поетапну реалізацію функціоналу та забезпечити ефективний контроль якості на всіх етапах створення системи.

Ітераційна модель передбачає поділ процесу розробки на серії послідовних циклів (ітерацій), кожен з яких включає повний набір етапів: аналіз вимог, проєктування, реалізацію, тестування та отримання зворотного зв’язку. Замість того щоб розробляти всю систему одразу, кожна ітерація зосереджується на створенні певного функціонального модуля або вдосконаленні вже реалізованої частини.

Під час розробки SMM-helper ітераційна модель забезпечила можливість спершу розробити базову архітектуру односторінкового застосунку (SPA) з використанням Next.js, Tailwind CSS та TypeScript. На першій ітерації реалізувати систему автентифікації користувачів, яка базується на JWT-токенах, зберігаючи дані автентифікації в HTTP-only cookies для підвищеної безпеки.

Під час другої ітерації було реалізовано основну логіку маршрутизації за допомогою App Router, яка забезпечила зручну навігацію між сторінками програми, організованими на основі файлової системи. Наступні ітерації зосереджувалися на реалізації ключових функцій: візуального редактора постів (з використанням Fabric.js), UI-компонентів (на базі shadcn/ui), а також підсистеми планування та збереження публікацій за допомогою бази даних MariaDB, інтегрованої через Drizzle ORM.

Кожна ітерація завершувалася:

- перевіркою відповідності реалізованого функціоналу технічним вимогам;
- аналізом помилок;
- плануванням змін на наступну ітерацію.

Завдяки ітераційній моделі вдалося мінімізувати ризики, пов’язані з невизначеністю на початкових етапах, а також оптимізувати час розробки. Наприклад, певні рішення щодо користувацького інтерфейсу змінювалися в процесі розробки. Крім того, поетапне тестування та перевірка функціональності дозволили своєчасно виявити та усунути критичні помилки ще до завершення повного циклу розробки.

Таким чином, використання ітераційної моделі забезпечило високу гнучкість, покращену якість продукту та ефективний розподіл ресурсів під час розробки вебдодатку «SMM-helper».

2.4. Обґрунтування вибору інструментальних засобів розробки «SMM-helper»

Під час розробки вебдодатку «SMM-helper» було обрано сучасний технологічний стек, який забезпечує продуктивність, масштабованість, гнучкість та зручність у подальшій підтримці додатку. Проєкт реалізований як повноцінний вебдодаток з клієнтською та серверною частинами, орієнтований на створення зручного інструменту для SMM-спеціалістів.

Базовою платформою для клієнтської частини став Next.js – фреймворк, побудований на основі React, що поєднує переваги клієнтського та серверного рендерингу. [11] Це дозволяє скоротити час першого завантаження сторінки, підвищити продуктивність, а також забезпечити кращу індексацію пошуковими системами, що особливо важливо для сервісів із відкритими сторінками або SEO-оптимізованим контентом.

Використання React дало змогу організувати архітектуру на базі незалежних компонентів, що спрощує підтримку та розширення додатку. Управління станом реалізовано через Hooks, що забезпечує централізоване зберігання логіки.

Додатково, у клієнтську частину було інтегровано TypeScript – мову програмування, яка розширює JavaScript типізацією. [12] Це дозволяє зменшити

кількість помилок ще на етапі компіляції та зробити код більш надійним і придатним до масштабування.

Інтерфейс користувача створено з використанням Tailwind CSS – утилітарного CSS-фреймворку, який дає змогу створювати адаптивний, швидкий та уніфікований дизайн. [13] Для забезпечення швидкого та доступного UX/UI додано shadcn/ui – набір готових React-компонентів, який забезпечує сучасний вигляд інтерфейсу без надлишкової кастомізації.

Особливу увагу було приділено створенню вбудованого графічного редактора. Для цього інтегровано Fabric.js – JavaScript-бібліотеку для роботи з HTML5 Canvas. [14] Fabric.js дозволяє користувачам малювати, редагувати, групувати та трансформувати об’єкти, а також створювати анімовані компоненти. Завдяки цьому функціоналу SMM-спеціалісти можуть створювати контент прямо в додатку без потреби у зовнішніх інструментах.

Серверна частина реалізована також у межах Next.js завдяки можливостям серверних маршрутів (API Routes). Це спрощує архітектуру проєкту, дозволяє зменшити залежності та забезпечити високу продуктивність.

Для роботи з базою даних було обрано MariaDB – реляційну СУБД, яка є надійною, продуктивною та відкритою альтернативою MySQL. [15] Вона добре масштабується, підтримує ACID-властивості та має активну спільноту.

Доступ до бази даних здійснюється через Drizzle ORM – сучасну безпечну ORM-бібліотеку, яка дозволяє писати SQL-запити у зручному синтаксисі з TypeScript-підтримкою. [16] Це забезпечує безпечну взаємодію з БД, підвищує читабельність коду та спрощує міграції.

Для зручної роботи з базою даних у процесі розробки використовувався інструмент HeidiSQL – графічний інтерфейс для адміністрування MariaDB. [17] Він дозволив швидко переглядати структуру бази, редагувати таблиці, перевіряти запити та експортувати дані.

Для реалізації автентифікації в додатку використано JWT (JSON Web Token) у поєднанні з HTTP-only cookies. Такий підхід забезпечує безпечне

зберігання токенів, які не доступні через JavaScript, що зменшує ризик XSS-атак. Це критично важливо для збереження конфіденційності даних користувачів. [18]

Таким чином, вибрані інструменти повністю відповідають потребам проєкту та сучасним стандартам веброзробки, а також дають змогу створити надійний, безпечний, зручний у використанні та готовий до розширення інструмент.

2.5. Особливості програмної реалізації та основі режими функціонування вебдодатку «SMM-helper»

Для того, щоб розпочати роботу з вебдодатком і отримати доступ до інструментів «Графічний редактор» та «Кольорова схема», в першу чергу користувач повинен пройти процес реєстрації. Нижче розглянемо ключові процеси, що забезпечують коректне функціонування системи на початковому етапі взаємодії з користувачем.

Після завантаження головної сторінки вебдодатку, користувач має можливість створити обліковий запис. Це реалізовано за допомогою API-роуту POST. Коли клієнт надсилає запит на реєстрацію, система приймає дані: email, пароль та ім’я. Далі виконується перевірка, чи існує користувач у базі даних. У випадку, коли користувач вже зареєстрований у системі, і при спробі повторної реєстрації передає на сервер свою електронну пошту, система виконує перевірку на унікальність. Завдяки функції user_get(email) відбувається звернення до бази даних, де здійснюється пошук відповідного запису. Якщо запис з таким email уже існує, сервер повертає JSON-відповідь із повідомленням "Користувач з такою електронною поштою вже існує" та статусом 400. Це дозволяє уникнути дублювання облікових записів і забезпечує цілісність даних у таблиці користувачів.

```
import bcrypt from 'bcrypt'; // crypto password

//server side
export async function POST( request ) {

    try {

    const { email, password, name } = await request.json();

    console.log('############ api/auth/signup server side: request from client POST signup: '+ email + password + name );

    // Check user already exists
    const userdb = await user_get( email );

    console.log('############### api/auth/signup user from database');  console.log (userdb);

     if( userdb != null )
     {
            return NextResponse.json(
              { message: "Користувач з такою електронною поштою вже існує" }, { status: 400 }
            );
     }

    const hashedPassword = await bcrypt.hash(password, 10);
    console.log("############# /api/auth/signup user_add " + email + " password:" + password +" hashedPassword:"+ hashedPassword  );

    const user = { user:email, password: hashedPassword, name: name || email.split("@")[0] };
    //add user to users database
    await user_add( user );

    // Set cookie or session (simplified)
    const response = NextResponse.json( user, { status: 201 });
    response.cookies.set("auth-token", "simulated-jwt-token", {
      httpOnly: true,
      secure: process.env.NODE_ENV === "production",
      maxAge: 60 * 60 * 24 * 7, // 1 week
      path: "/",
    });
```

Рисунок 2.2 – Файл «signup-route.js» для реєстрації нового користувача

У разі відсутності такого користувача, пароль хешується з використанням бібліотеки bcrypt, а дані нового користувача додаються в базу даних. Після цього створюється cookie-файл auth-token, який використовується для збереження сеансу користувача. Таким чином, користувач після реєстрації автоматично входить у систему, що покращує загальний досвід використання додатку.

Фрагмент коду на рисунку 2.2 ілюструє не лише перевірку користувача, а й демонструє налаштування cookie з обмеженим доступом та терміном дії в один тиждень, що сприяє підвищенню безпеки.

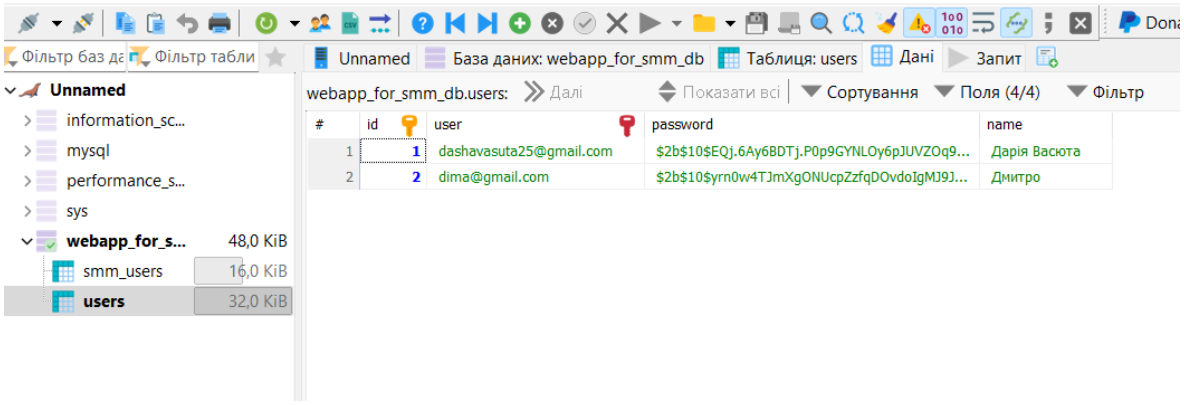


Рисунок 2.3 – База даних

На рисунку 2.3 представлено структуру бази даних, зокрема таблиці users, у якій зберігається інформація про зареєстрованих користувачів. Основними полями цієї таблиці є:

- id – унікальний ідентифікатор користувача (генерується автоматично);
- user – електронна пошта, що виступає в ролі логіна;
- password – захешований пароль;
- name – ім’я користувача, яке вводиться при реєстрації.

Така структура дозволяє ефективно зберігати та обробляти основні дані, необхідні для авторизації та подальшої персоналізації роботи в додатку. Поле password не зберігається у відкритому вигляді, що забезпечує безпеку користувачів.

На рисунку 2.4 зображено приклад відображення форми для реєстрації нового користувача, яка реалізована у вигляді простого, зручного інтерфейсу з перевіркою введених даних. Форма містить усі необхідні поля, валідацію email-адреси, перевірку мінімальної довжини пароля, а також повідомлення про помилки для покращення користувацького досвіду.

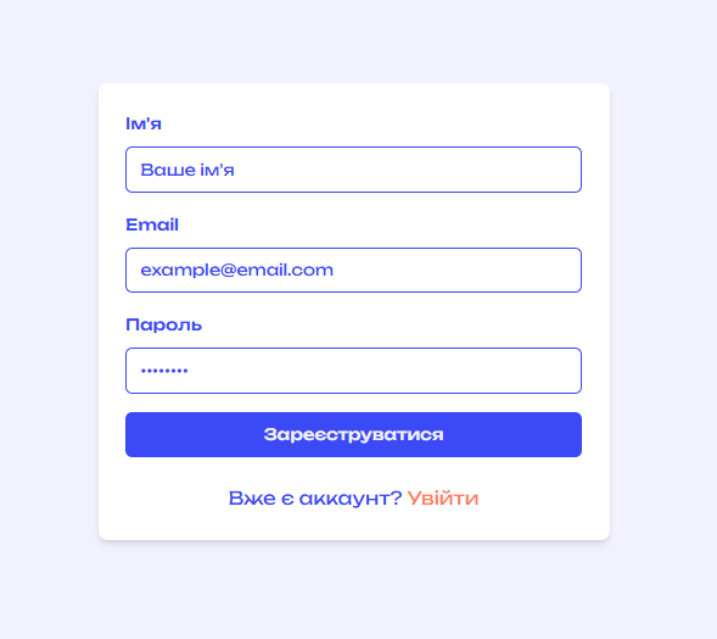


Рисунок 2.4 – Відображення форми реєстрації у браузері

Після успішної реєстрації користувач має змогу увійти до системи через відповідну форму авторизації. Обробка логінації реалізована у файлі login-route.js (рисунок В.1). У цьому файлі сервер приймає POST-запит, що містить електронну пошту та пароль користувача. Спочатку відбувається очищення попереднього стану авторизованого користувача за допомогою функції setUserLogged. Далі викликається функція user_get, яка отримує дані користувача з бази даних за переданою електронною поштою.

Якщо користувача з такою поштою не існує, повертається відповідь із повідомленням про помилку. Якщо користувача знайдено, відбувається перевірка пароля шляхом порівняння введеного значення з хешованим паролем, збереженим у базі даних. Для цього використовується функція bcrypt.compareSync. У разі успішної перевірки на сервері створюється cookie з токеном автентифікації, аналогічно до процесу під час реєстрації. Також зберігається інформація про авторизованого користувача у змінній userlogged, яка може використовуватись у подальших перевірках авторизації.

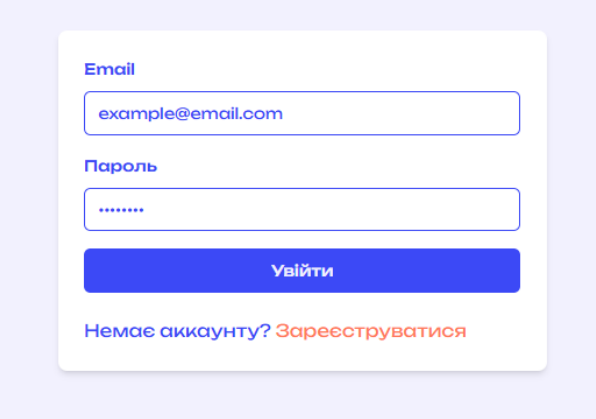


Рисунок 2.5 – Відображення форми авторизації у браузері

Результат відображення форми авторизації зображено на рисунку 2.5.

У разі спроби доступу до інструментів «Графічний редактор» або «Кольорова схема» неавторизованим користувачем, система автоматично блокує вхід та відображає відповідне повідомлення (рис. 2.6) про необхідність реєстрації або входу в обліковий запис.

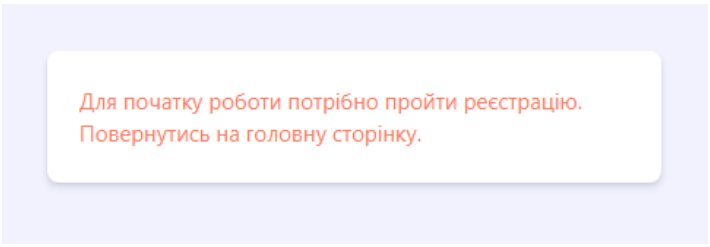


Рисунок 2.6 – Відображення повідомлення про обов’язкову реєстрацію

Переходимо до сторінки «Графічний редактор». Основну структуру цієї сторінки реалізовано у файлі editor.tsx (рис. В.2), який виступає центральним компонентом графічного редактора та відповідає за візуалізацію користувацького інтерфейсу для роботи з редагуванням. У цьому компоненті використовується стороння бібліотека fabric.js для створення інтерактивного полотна, на якому користувач може взаємодіяти з графічними елементами.

Крім того, у файлі підключені локальні компоненти, зокрема Navbar, Sidebar, ShapeSidebar, FillColorSidebar, StrokeColorSidebar, StrokeWidthSidebar, OpacitySidebar, TextSidebar, FontSidebar, ImageSidebar, StickerSidebar,

EmojiSidebar, FilterSidebar, DrawSidebar, SettingsSidebar, TemplateSidebar, Toolbar. Вони реалізують функціонал навігації, налаштувань, панелей інструментів та редагування. Полотно, створене з використанням fabric.js, дає змогу користувачам малювати, додавати текст і зображення, змінювати кольори, налаштовувати прозорість, а також застосовувати фільтри й наліпки. Результат роботи наведений на рис. 2.7.

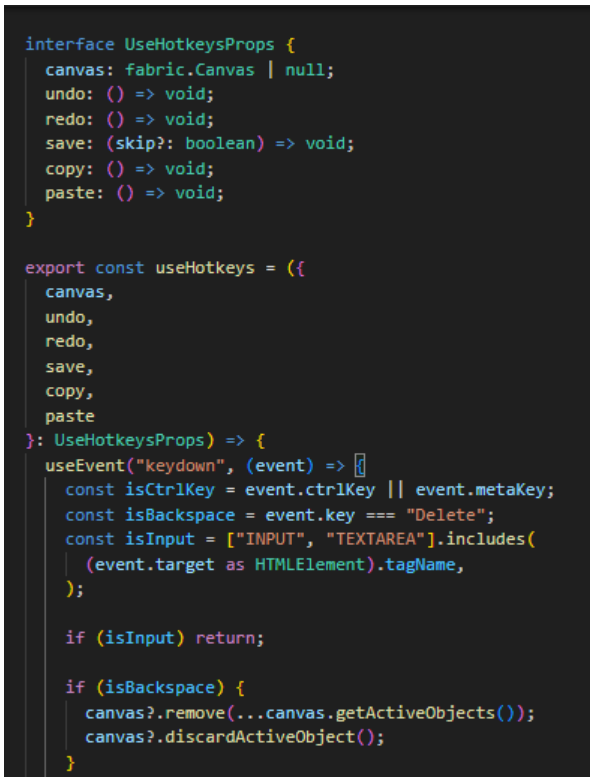
Рисунок 2.7 – Інтерфейс графічного редактору

У процесі вдосконалення проєкту до графічного редактора було додано важливий функціонал – ефект пам’яті (undo/redo), що забезпечує можливість повернення до попередніх станів полотна або відновлення змінених. Це значно покращує зручність користування редактором і дозволяє користувачам експериментувати без страху втратити попередній результат.

Реалізація цього функціоналу відбулася за допомогою створення власного хука useHistory. У ньому зберігаються послідовні стани полотна у вигляді JSON, які зберігаються в масиві canvasHistory. Кожна зміна автоматично фіксується методом save, тоді як методи undo і redo дозволяють переміщатися назад або вперед по історії змін. Для відтворення збережених станів використовується функція loadFromJSON, а сам запис стану виконується методом toJSON з ключами JSON_KEYS. Додатково передбачено оптимізацію, яка дозволяє

уникати зайвих збережень під час виконання відкату змін за допомогою прапорця skipSave.(Рисунок В.3)

Також для вдосконалення функціональності графічного редактора у проєкті «SMM- helper» було реалізовано підтримку гарячих клавіш за допомогою кастомного хука useHotkeys. Цей хук слухає подію натискання клавіш (keydown) на всій сторінці та дозволяє виконувати базові дії з полотном без використання миші. Зокрема, реалізовано можливість скасування та повторення останніх дій (Ctrl+Z, Ctrl+Y), копіювання та вставлення об’єктів (Ctrl+C, Ctrl+V), збереження проєкту (Ctrl+S), вибір усіх об’єктів (Ctrl+A) та видалення активного об'єкта (Delete). Додатково було передбачено обмеження: якщо фокус перебуває в текстовому полі або іншому елементі вводу, обробка гарячих клавіш вимикається, щоб уникнути конфліктів. (Рисунок 2.8)



```
interface UseHotkeysProps {
  canvas: fabric.Canvas | null;
  undo: () => void;
  redo: () => void;
  save: (skip?: boolean) => void;
  copy: () => void;
  paste: () => void;
}

export const useHotkeys = ({
  canvas,
  undo,
  redo,
  save,
  copy,
  paste
}: UseHotkeysProps) => {
  useEvent("keydown", (event) => {
    const isCtrlKey = event.ctrlKey || event.metaKey;
    const isBackspace = event.key === "Delete";
    const isInput = ["INPUT", "TEXTAREA"].includes(
      (event.target as HTMLElement).tagName,
    );

    if (isInput) return;

    if (isBackspace) {
      canvas?.remove(...canvas.getActiveObjects());
      canvas?.discardActiveObject();
    }
```

Рисунок 2.8 – Хук Use-hotkeys для гарячих клавіш

Ця реалізація значно підвищила зручність користування редактором, особливо для досвідчених користувачів, які звикли до подібних комбінацій

клавіш у професійних графічних інструментах, таких як Adobe Photoshop, Figma чи Sketch. Інтеграція гарячих клавіш дозволила реалізувати звичні дії – як-от копіювання, вставлення, скасування змін, масштабування – не відволікаючись на навігацію по інтерфейсу. Завдяки цьому взаємодія з полотном стала значно швидшою й ефективнішою, що в свою чергу позитивно впливає на загальну продуктивність під час створення візуального контенту, зокрема у випадках, коли користувач працює з великою кількістю об'єктів або повторюваними елементами.

Крім того, хук для обробки гарячих клавіш реалізовано у вигляді гнучкого та модульного рішення, що дозволяє легко масштабувати функціональність у майбутньому. За потреби його можна доповнити новими комбінаціями клавіш, налаштувати для різних типів елементів або адаптувати під інші функціональні модулі вебдодатку, не змінюючи основну архітектуру.

Також було реалізовано механізм буфера обміну за допомогою кастомного хука useClipboard. Цей функціонал забезпечує можливість копіювати вибрані об’єкти на полотні та вставляти їх повторно без втрати властивостей чи стилів. При копіюванні активний об’єкт автоматично клонується та зберігається у змінній clipboard, реалізованій через useRef. Під час вставлення, яке може відбуватись через гарячу клавішу або інші події, клонований об’єкт відображається на полотні зі зсувом (наприклад, на 10px правіше й нижче), що дозволяє користувачеві легко помітити дубль і, за потреби, одразу його відредагувати або перемістити.

Ця функція підтримує як окремі об’єкти, так і групи (тип activeSelection). У разі групи кожен об’єкт у складі клонованої групи додається окремо, зберігаючи відносне розташування. Після вставки нові об’єкти автоматично активуються та відображаються на полотні. Така реалізація не лише підвищує зручність редагування, але й надає користувачеві більшу гнучкість у створенні візуального контенту, роблячи вебдодаток ближчим до можливостей професійних графічних редакторів. (Рисунок В.4)

Тепер розглянемо перехід з головної сторінки до інструменту «Кольорова схема» (рис. 2.9). На цій сторінці відображається інтерфейс для підбору кольорових палітр. Основний компонент цієї сторінки реалізовано у файлі scheme.tsx (фрагмент подано на рисунку В.5). У ньому передбачено функціональність генерації кольорових схем на основі вибраного початкового кольору. Для цього використовуються сторонні бібліотеки hex-to-hsl та color-scheme, які дозволяють створювати різні типи палітр – монохроматичні, контрастні, тріадні, аналогові тощо. Такі палітри зручно використовувати для оформлення профілю в Instagram або створення візуального стилю.

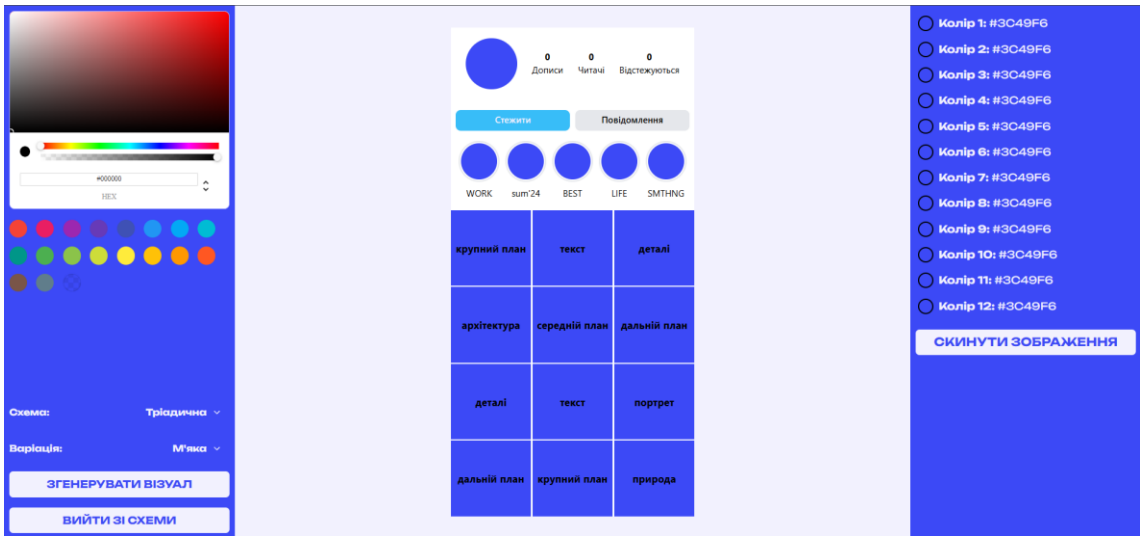


Рисунок 2.9 – Користувацький інтерфейс інструменту «Кольорова схема»

Інтерфейс дозволяє користувачеві обрати базовий колір, після чого палітра генерується автоматично. Для реалізації цієї функції застосовуються функції перетворення кольорів між форматами HEX, HSL та RGBA, а також використання рефів для динамічного оновлення елементів DOM.

Для забезпечення можливості вибору кольору інтерфейсу користувачем у вебдодатку було реалізовано компонент ColorPicker (рис. 2.10), який створено з використанням бібліотеки react-color. Цей компонент надає зручний інструментарій для взаємодії з кольоровими палітрами та дозволяє обрати відтінок через два альтернативних інтерфейси: ChromePicker та CirclePicker.

```
// Основний компонент ColorPicker
export const ColorPicker = ({
  value,   // поточне значення кольору
  onChange, // функція для зміни кольору
}: ColorPickerProps) => {
  return (
    <div className="w-full space-y-4">
      {/* ChromePicker для вибору кольору з повним спектром */}
      <ChromePicker
        color={value}
        onChange={(color) => {
          const formattedValue = rgbaObjectToString(color.rgb); // форматування значення кольору
          onChange(formattedValue); // передача зміненого кольору
        }}
        className="border rounded-lg"
      />

      {/* CirclePicker для вибору кольору з палітри */}
      <CirclePicker
        color={value}
        colors={colors} // набір попередньо визначених кольорів
        onChangeComplete={(color) => {
          const formattedValue = rgbaObjectToString(color.rgb); // форматування значення кольору
          onChange(formattedValue); // передача зміненого кольору
        }}
      />
    </div>
  );
};
```

Рисунок 2.10 – Компонент ColorPicker

ChromePicker забезпечує розширений функціонал, що дозволяє не лише вибирати колір з повного спектру, а й точно задавати значення у форматах RGB, HEX або HSL. Це особливо зручно для користувачів, які мають потребу у точному підборі кольорів, наприклад, відповідно до брендбуку або дизайну візуального стилю.

CirclePicker, натомість, є спрощеним інтерфейсом, який демонструє набір кольорових мініатюр у формі кругів. Він орієнтований на швидкий вибір одного з попередньо визначених кольорів і є ідеальним варіантом для тих, хто не потребує детального налаштування.

Завдяки такому поєднанню варіантів, компонент ColorPicker охоплює потреби як досвідчених користувачів, так і новачків, забезпечуючи інтуїтивно зрозумілий та гнучкий інтерфейс для персоналізації кольорової теми сторінки. Результат зображено на рисунку 2.11.

Рисунок 2.11 – Вибір кольору на палітрі кольорів

Для реалізації вибору типу кольорової схеми (монохроматична, контрастна, тріадна, аналогова) та її варіації (стандартна, пастельна, м’яка, світла, насичена, бліда), було створено функцію newPalette (рис. В.6), яка є ключовим елементом логіки генерації кольорової палітри у редакторі. Ця функція виконує кілька послідовних кроків для забезпечення коректного та зручного для користувача формування кольорових рішень.

Перш за все, відбувається перевірка правильності введеного кольору у форматі HEX за допомогою регулярного виразу. Це дозволяє гарантувати, що подальші розрахунки не спричинять помилок через некоректний формат. У випадку позитивного результату, введений колір конвертується з HEX у формат HSL за допомогою функції hexToHsl. Це перетворення потрібне для подальшої роботи з бібліотекою color-scheme, яка оперує значеннями кольору саме у HSL-просторі.

На основі значення тону (hue), отриманого з HSL, та обраного користувачем типу схеми – mono, contrast, triade, analogic – формується відповідний набір кольорів. Кожен тип схеми має власну логіку генерації: наприклад, для тріадної палітри створюються три кольори, розміщені

рівновіддалено на колірному колі, а для аналогової – кілька відтінків, близьких до базового кольору. Додатково застосовуються варіації яскравості й насиченості для досягнення стилістичних ефектів, як м’якість палітри.

Генеровані кольори передаються у відповідні стани компонента через функції setColor1, setColor2 і далі до setColor12, що дозволяє зберігати повну палітру у реактивному стані. Це дає можливість легко відображати кольори у візуальному інтерфейсі користувача, наприклад, у вигляді блоків вибору або для автоматичного застосування до елементів дизайну.

Таким чином, функція newPalette виконує не лише технічну задачу генерації кольорів, а й сприяє створенню візуально привабливого та інтуїтивного інтерфейсу, що дозволяє навіть недосвідченим користувачам легко працювати з професійними колірними схемами. Це розширює функціональність редактора та покращує досвід взаємодії з додатком.

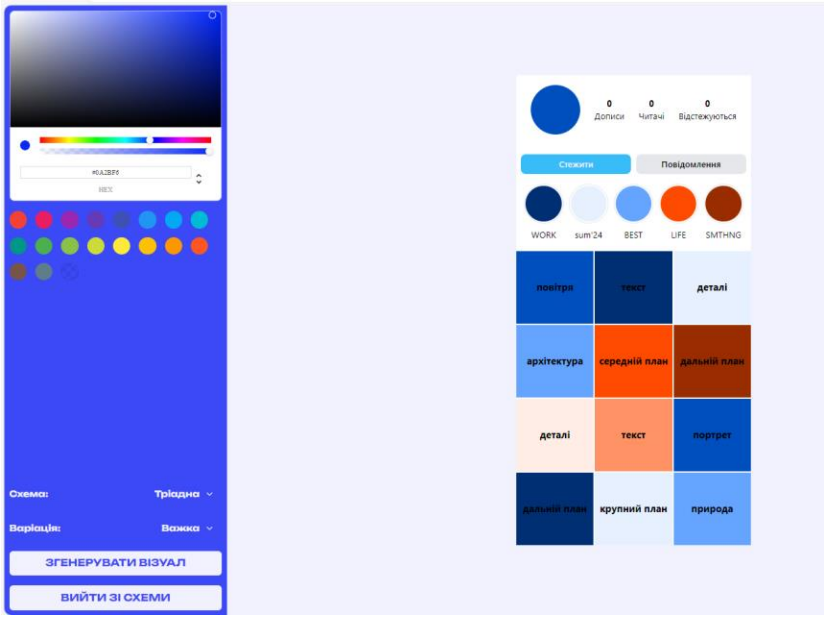


Рисунок 2.12 – Згенерована кольорова схема

У випадку введення некоректного значення кольору або його відсутності, користувач отримує повідомлення з відповідним попередженням. Після натискання кнопки «ЗГЕНЕРУВАТИ ВІЗУАЛ» система створює палітру, що

враховує початковий колір, обрану варіацію та тип схеми, як показано на рисунку 2.12.

Окрім генерації палітри, реалізовано можливість вставки зображення у шаблон посту, сторіс чи аватарки. При натисканні на відповідний шаблон створюється прихований елемент введення, який дозволяє завантажити лише зображення. Подія onchange, що спрацьовує після вибору файлу, активує зчитування зображення через об'єкт FileReader. Зображення перетворюється на Data URL, що дає змогу використовувати його як фонове зображення для елемента div, на який здійснено клік. Це фонове зображення вставляється автоматично за допомогою CSS-стилів.

Увесь функціонал завантаження і вставки зображення реалізований у функції handleDivClick (рис. В.7), а приклад результату з використанням відповідних зображень і палітри кольорів продемонстровано на рисунку 2.13.

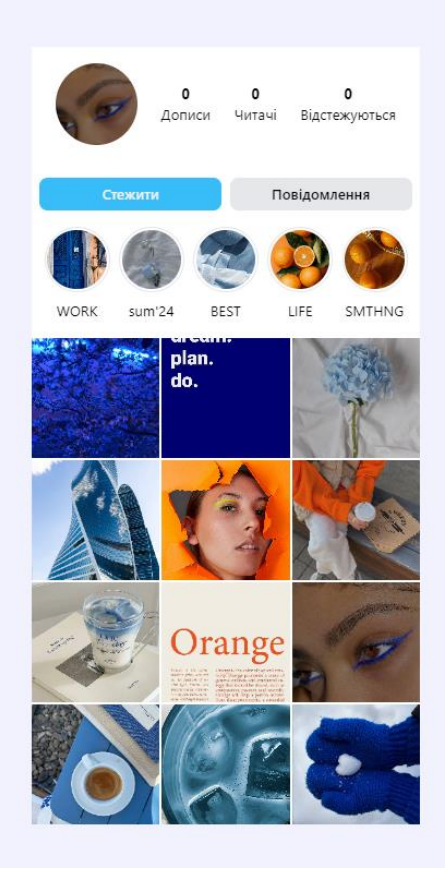


Рисунок 2.13 – Шаблон профілю Instagram на основі згенерованої схеми

2.6. Тестування та налагодження вебдодатку «SMM-helper»

У процесі тестування вебдодатку «SMM-helper» основну увагу було приділено перевірці функціональності ключових модулів, що забезпечують взаємодію користувача з інтерфейсом та основними інструментами. Зокрема, було протестовано функції створення та редагування графічного контенту, перевірено збереження й відновлення стану полотна, правильну роботу механізмів масштабування, копіювання, вставки та видалення об’єктів. Також проводилося тестування функції автозавантаження даних при старті сесії, що відновлює попередній стан проєкту.

Особливу увагу приділено реакції інтерфейсу на дії користувача, зокрема роботі гарячих клавіш, що відповідають за швидкий доступ до функцій редагування, збереження та вибору об’єктів. Усі перевірки проводилися вручну шляхом моделювання дій користувача в середовищі браузера. Автоматизоване тестування не здійснювалося, однак ручна перевірка засвідчила стабільну роботу додатку, правильне відображення графічних елементів і відповідність функціоналу заявленим вимогам.

2.7. Рекомендації по використанню та впровадженню вебдодатку «SMM-helper»

Вебдодаток «SMM-helper» має значний потенціал для практичного застосування фахівцями у сфері соціального медіа-маркетингу, дизайну та реклами, які займаються створенням контенту для соціальних мереж. Завдяки поєднанню кількох ключових інструментів у межах одного середовища, «SMM-helper» істотно полегшує процеси планування, редагування та управління контентом, дозволяючи економити час і підвищувати продуктивність роботи.

Можливості практичного використання:

- інструмент «Кольорова схема» надає можливість створювати шаблонний візуальний стиль Instagram-профілю без потреби у сторонньому програмному забезпеченні;

- завдяки цьому ж інструменту користувачі можуть формувати власні кольорові схеми та шаблони, адаптовані до фірмового стилю бренду або персональних вподобань;
- інструмент «Редактор графіки» дозволяє створювати графічний контент як з нуля, так і на основі заздалегідь підготовлених шаблонів, що розширює можливості для дизайну без потреби в складних редакторах.

Умови експлуатації:

Для забезпечення коректної роботи програмного засобу необхідно дотримуватися наступних умов:

Операційна система: від Windows 7, від macOS X, Ubuntu 20.04 чи вище.

Веббраузер: Google Chrome, Mozilla Firefox, Microsoft Edge (будь-якої версії).

Технічні характеристики пристрою:

- Процесор: Intel Core i3 (або аналогічний) та вище.
- Оперативна пам'ять: не менше 4 ГБ (рекомендовано 8 ГБ).
- Місце на диску: щонайменше 1 ГБ для локального зберігання даних.

Програмне забезпечення: Node.js (версія 16 або вище) для запуску локального сервера, NPM для встановлення залежностей.

Мережеві умови: стабільне підключення до Інтернету зі швидкістю не менше 5 Мбіт/с.

ВИСНОВКИ

Результатом виконання кваліфікаційної роботи є розробка інноваційного вебдодатку «SMM-helper», призначеного для фахівців у сфері SMM, маркетингу та дизайну. У ході реалізації проєкту було проведено ретельний аналіз аналогічних цифрових рішень, який дозволив виявити низку функціональних недоліків, зокрема відсутність комплексного підходу до створення, редагування та організації контенту для соціальних мереж. З огляду на ці проблеми було розроблено власне рішення, яке поєднує декілька ключових інструментів в єдиному середовищі, що оптимізує повсякденну роботу SMM-фахівців.

Основний функціонал вебдодатку охоплює графічний редактор, який підтримує повноцінну взаємодію з векторними та растровими об’єктами. Користувач має змогу працювати з текстом, формами, кольорами, а також створювати унікальні дизайни як з нуля, так і на основі наданих шаблонів. Додатково реалізовано окремий інструмент для формування кольорових схем, що дозволяє легко адаптувати візуальний стиль під потреби бренду чи особисті уподобання, з орієнтацією на платформу Instagram. Реалізовано базову систему авторизації, що забезпечує безпечний доступ до персоналізованого середовища користувача.

У ході тестування було здійснено ручну перевірку працездатності усіх функціональних компонентів. Це дозволило виявити й усунути ключові помилки, оптимізувати продуктивність та забезпечити стабільну роботу інтерфейсу. Проведені перевірки підтвердили, що інструмент успішно виконує покладені на нього завдання, а також відповідає вимогам зручності, гнучкості та інтуїтивності для цільової аудиторії. Таким чином, розроблений додаток повністю задовольняє поставлені завдання у кваліфікаційній роботі.

Попри успішну реалізацію основного функціоналу, у процесі створення виникли певні труднощі. Зокрема, не вдалося реалізувати модуль контент-календаря, який планувався як інструмент для управління графіком публікацій. Основними причинами стали обмеження в часі та складність технічної реалізації.

На ранніх етапах також спостерігалися труднощі з вибором кольору – користувачу необхідно було вручну вводити HEX-коди. Проте цю проблему вдалося вирішити шляхом впровадження інтерактивної палітри. У процесі розробки також було проведено редизайн стартової сторінки задля покращення користувацького досвіду, що позитивно позначилося на загальному сприйнятті додатку.

Перспективи подальшого розвитку вебдодатку є досить широкими. Планується реалізація підтримки кількох ключових кольорів у схемах для розширення гнучкості створення дизайну. Важливим напрямом є впровадження інтеграції з Instagram API, що дозволить користувачеві авторизуватися через акаунт соцмережі та автоматично імпортувати публікації, сторіс, рілс, аватар і опис профілю безпосередньо у додатку. Крім того, користувач зможе додавати новий контент і редагувати існуючий просто з інтерфейсу «SMM-Helper». У майбутньому також заплановано впровадження функції редагування зображень у наявних публікаціях, підтримку кількох акаунтів у рамках преміум-версії, що дасть змогу управляти контентом з різних профілів одночасно. Ще одним пріоритетом стане впровадження функціоналу відкладеного постингу, що значно підвищить зручність планування та автоматизації публікацій.

Таким чином, «SMM-helper» вже на етапі кваліфікаційної роботи продемонстрував високу функціональність та потенціал до масштабування, що робить його перспективним інструментом для впровадження в реальні робочі процеси SMM-фахівців, фрилансерів та малих брендів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Графічний редактор: поняття, класифікація та призначення. Графічний редактор: поняття, класифікація та призначення. URL: https://vseosvita.ua/blogs/hrafichnyi-redaktor-poniattia-klasyfikatsiia-ta-pryznachennia-68427.html (дата звернення: 20.02.2025).
2. Sketchpad (jan 1, 1963 – jan 1, 1964). time.graphics. URL: https://time.graphics/period/146328 (date of access: 20.02.2025).
3. Основи комп'ютерної графіки. *Основи комп'ютерної графіки. Граф. редактор Paint.* URL: https://vseosvita.ua/library/embed/0038c4-63b3.doc.html (дата звернення: 21.02.2025).
4. Головна | Elib LNTU. URL: https://elib.lntu.edu.ua/sites/default/files/elib_upload/федік%202/page6.html (дата звернення: 21.02.2025).
5. ГРАФІЧНИЙ РЕДАКТОР MICROSOFT PAINT. Stud. URL: https://stud.com.ua/43445/informatika/grafichniy_redaktor_microsoft_paint (дата звернення: 21.02.2025).
6. shelest. Figma за 20 млрд. Історія про непросту реальність побудови стартапу та міф про успіх за одну ніч. *UXPUB UA Дизайн-спільнота*. URL: https://ux.pub/shelest/figma-za-20-mlrd-istoriia-pro-nieprostu-riealnist-pobudovi-startapu-ta-mif-pro-uspikh-za-odnu-nich-528 (дата звернення: 09.03.2025).
7. Canva: історія створення і успіху компанії Канва . *Інформаційна цифрова платформа*. URL: https://worldbank.org.ua/4666-canva.html#google_vignette (дата звернення: 20.02.2025).
8. Вплив візуального контенту на ефективність маркетингу: тенденції та практичні поради. Рекламне digital-агенство «Mas Agency». URL: https://mas-agency.com.ua/blog/article/?id=37&srsltid=AfmBOor9rDrqadTXtDRRbP

vIEtPZXMaVN_evdQeLPe6hM2S0P0hmQtKh&(дата звернення: 20.02.2025).

9. Дизайн соціальних медіа та його роль у маркетингових кампаніях: вплив на ефективність та конверсію. PerfectPR. URL: https://perfect-pr.com.ua/dyzajn-soczialnyh-media-ta-jogo-rol-u-marketyngovyh-kampaniyah-vplyv-na-efektyvnist-ta-konversiyu/ (дата звернення: 20.02.2025).
10. Інструменти, які спростять роботу SMM-спеціаліста. Інструменти, які спростять роботу SMM-спеціаліста. URL: https://www.newlook.ua/blog/top-app (дата звернення: 20.02.2025).
11. Vercel. Introduction | Next.js. Next.js by Vercel - The React Framework. URL: https://nextjs.org/docs (date of access: 12.04.2025).
12. The starting point for learning TypeScript. TypeScript: JavaScript With Syntax For Types. URL: https://www.typescriptlang.org/docs/ (date of access: 20.04.2025).
13. Documentation - Tailwind CSS. Tailwind CSS - Rapidly build modern websites without ever leaving your HTML. URL: https://v2.tailwindcss.com/docs (date of access: 12.04.2025).
14. Introduction to Fabric.js. Docs and Guides. URL: https://fabricjs.com/docs/ (date of access: 20.04.2025).
15. Documentation - MariaDB.org. MariaDB.org. URL: https://mariadb.org/documentation (date of access: 20.04.2025).
16. Drizzle ORM - Why Drizzle?. Drizzle ORM - next gen TypeScript ORM. URL: https://orm.drizzle.team/docs/overview (date of access: 26.04.2025).
17. HeidiSQL - MariaDB/MySQL, MSSQL, PostgreSQL, SQLite and Interbase/Firebird made easy. HeidiSQL - MariaDB/MySQL, MSSQL, PostgreSQL, SQLite and Interbase/Firebird made easy. URL: https://www.heidisql.com (date of access: 25.045.2025).
18. JSON Web Token Introduction - jwt.io. JSON Web Tokens - jwt.io. URL: https://jwt.io/introduction (date of access: 25.04.2025).

ДОДАТКИ

ДОДАТОК А

Технічне завдання

1. Вступ

1.1. Найменування програмного продукту

«SMM-helper».

2. Підстави для розробки

Затвердив – Волинський національний університет імені Лесі Українки. Тема – «Розробка вебдодатку для СММ-спеціалістів за допомогою технології Next.js».

3. Призначення розробки

Вебдодаток призначений для об’єднання в одному програмному засобі кількох функціональних можливостей, що використовуються у роботі SMM-спеціалістів.

4. Вимоги до продукту

4.1. Вимоги до функціоналу

Вебдодаток повинен забезпечувати наступний функціонал для ефективної роботи користувача:

- мати інтуїтивно зрозумілий інтерфейс, який дозволяє легко орієнтуватися у можливостях редагування контенту;
- реалізувати механізм авторизації та автентифікації користувача для доступу до інструментів вебдодатку;
- забезпечити візуалізацію кольорових схем у реальному часі;
- забезпечити можливість створення, редагування та збереження проєктів у графічному редакторі;

- підтримувати імпорт та експорт проєктів у форматі JSON для зручного обміну між пристроями або збереження локальних копій;
- дозволяти імпорт зображень з локального пристрою в редактор або інструмент «кольорова схема»;
- надавати можливість налаштування розміру полотна у графічному редакторі для адаптації до різних форматів (наприклад, сторіс, пост, банер тощо);
- містити бібліотеку готових шаблонів для сторіс та постів з можливістю їх редагування;
- забезпечувати можливість додавання геометричних фігур, емодзі, стікерів та текстових блоків на полотно;
- дозволяти гнучке налаштування тексту, включаючи шрифт, розмір, колір та вирівнювання;
- підтримувати зміну кольору фону та розміру всього полотна;
- мати режим малювання з можливістю його вмикання та вимикання;
- містити інструмент для створення власних кольорових схем візуалу instagram-сторінки на основі hex-кодів;
- забезпечувати можливість імпорту зображення у «кольорову схему» для автоматичного підбору кольорів та функцію його видалення при потребі.

4.2. Вимоги до надійності

Вебдодаток повинен забезпечувати надійне функціонування в наступних умовах:

- забезпечення коректної взаємодії з інтернет-мережею;
- підтримка функціонування на пристроях із різними технічними параметрами;

Система повинна коректно обробляти можливі помилки, зокрема неправильні запити, втрату з’єднання чи некоректні дані. Важливі дані, зокрема проєкти користувачів, повинні бути захищені та відновлювані у випадку збою.

4.3. Умови експлуатації

Вебдодаток має коректно працювати на пристроях, що відповідають таким технічним вимогам:

- операційні системи: Windows 10/11, macOS, Android 10.0+ або iOS 14.0+;
- оперативна пам’ять – щонайменше 4 ГБ;
- вбудована пам’ять – не менше 5 ГБ.

4.4. Вимоги до складу і параметрів технічних засобів

Вебдодаток не вимагає використання додаткових технічних засобів.

4.5. Вимоги до інформаційної і програмної сумісності

Інформаційна сумісність вебдодатку «SMM-helper» забезпечується за рахунок підтримки стандартних форматів даних та інтеграції з популярними сервісами. Передача даних між клієнтською та серверною частиною відбувається у форматі JSON через REST API, що дозволяє ефективно взаємодіяти з іншими системами. Додаток підтримує кодування UTF-8 для коректного відображення тексту.

4.6. Вимоги до маркування і упаковки

Вебдодаток повинен бути маркований наступними даними:

- найменування програми;

4.7. Вимоги до транспортування і збереження

Вебдодаток не потребує фізичного транспортування чи спеціального зберігання, оскільки працює в онлайн-середовищі. Всі дані та налаштування

користувачів зберігаються на сервері, забезпечуючи безперервний доступ до функціоналу з будь-якого пристрою, що відповідає технічним вимогам.

5. Вимоги до програмної документації

Документація повинна містити опис функціональності вебдодатку, архітектури, вимог до системи, інструкцій для користувачів та розробників. Також важливим є використання стандартизованих форматів, актуальність даних та легкість оновлення. Вона має забезпечувати зрозумілість для всіх зацікавлених сторін та містити приклади використання.

6. Техніко-економічні показники

Техніко-економічні показники вебдодатку включають продуктивність та експлуатацію, а також ефективність використання. Продуктивність визначається швидкістю завантаження сторінок, обробкою запитів та взаємодією з базою даних. Економічна ефективність проявляється у зниженні витрат часу SMM-спеціалістів, оптимізації контент-планування та підвищенні залученості аудиторії, що сприяє розвитку бізнесу та зростанню прибутку.

7. Стадії і етапи розробки

- Аналіз вимог користувачів
- Розробка архітектури додатку, проектування інтерфейсу та бази даних
- Розробка функціональних вимог
- Створення інтерфейсу користувача (UI)
- Програмування, реалізація основних функцій
- Оптимізація інтерфейсу та UX
- Тестування
- Запуск і впровадження

8. Порядок контролю і приймання

Контроль і приймання користувацького досвіду будуть здійснюватися розробником вебдодатку. Це дозволить виявити та усунути недоліки, а також забезпечити відповідність вимогам користувачів. В процесі тестування буде враховано зворотний зв'язок від кінцевих користувачів для подальшого вдосконалення взаємодії з додатком.

ДОДАТОК Б

Інструкція користувачу

SMM-helper – це вебдодаток для створення, планування та редагування контенту для соціальних мереж, що дозволяє налаштовувати кольорові схеми профілю, а також працювати з публікаціями та сторіс.

Вебдодаток призначений для полегшення роботи SMM-спеціалістів, зокрема:

- налаштування кольорових схем для візуалу профілю;
- створення візуального контенту у графічному редакторі.

Для роботи вебдодатку необхідно:

- персональний комп'ютер або ноутбук(будь-яка ОС);
- програмне забезпечення: будь-який сучасний браузер.

Для запуску вебдодатку «SMM-helper» користувач повинен мати локальну копію проєкту на своєму комп’ютері. Спочатку необхідно відкрити термінал у кореневій директорії проєкту та виконати команду npm install, яка встановить усі необхідні залежності. Після цього слід запустити сервер розробки за допомогою команди npm run dev. Після запуску у браузері потрібно перейти за адресою http://localhost:3000, де відкриється головна сторінка застосунку (рис. Б.1).

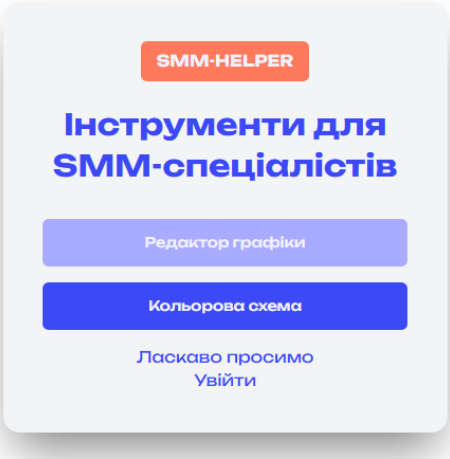


Рисунок Б.1 – Головна сторінка

Для того щоб отримати доступ до функціоналу вебдодатку «SMM-helper», користувачу необхідно пройти обов’язкову процедуру реєстрації. Під час реєстрації потрібно вказати електронну пошту та пароль, після чого система зберігає ці дані у базі. Цей процес забезпечує індивідуальний доступ до інструментів, захищаючи персональні дані та прогрес користувача у роботі з додатком.(Рисунок Б.2 та Б.3)

Після успішної реєстрації користувач повинен авторизуватись, тобто увійти до свого облікового запису, використовуючи раніше вказану електронну пошту та пароль. Лише після входу в систему відкривається доступ до таких інструментів, як «Графічний редактор» та «Кольорова схема». У разі спроби потрапити на відповідні сторінки без авторизації, користувач отримає повідомлення з попередженням про необхідність входу в систему.

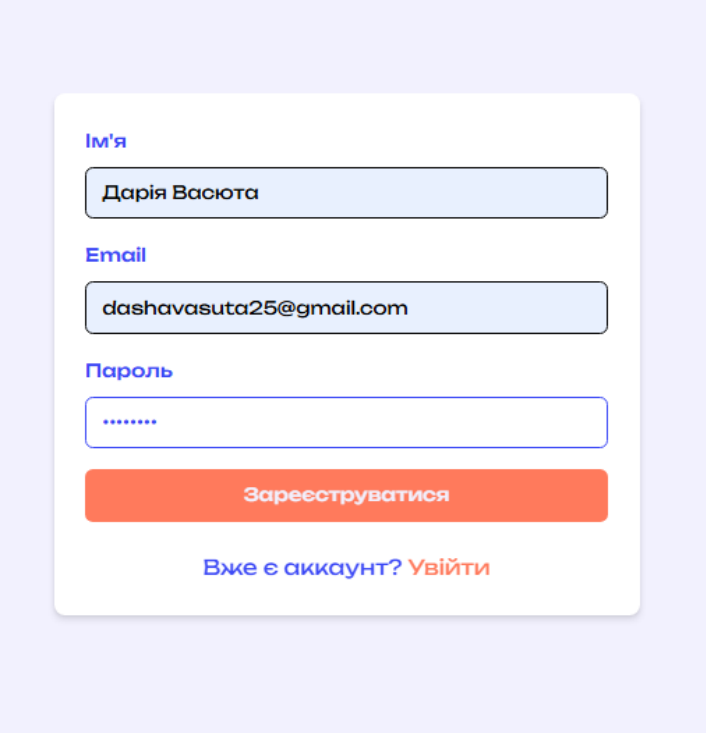


Рисунок Б.2 – Форма реєстрації

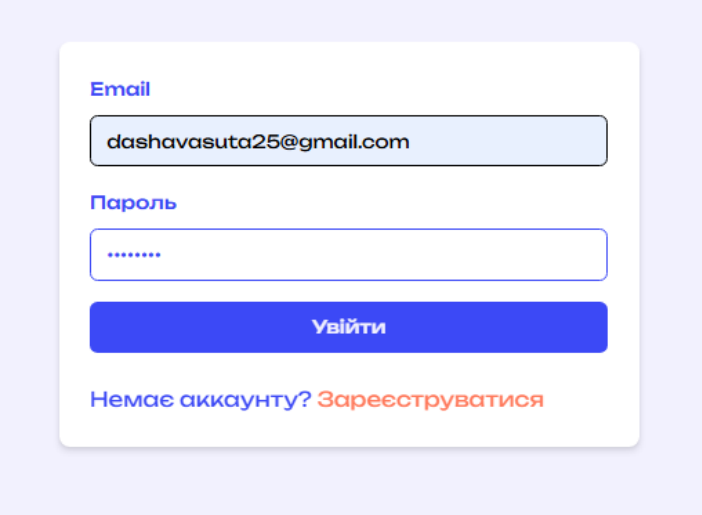


Рисунок Б.3 – Форма авторизації

Після запуску застосунку користувач має можливість обрати необхідний інструмент – «Графічний редактор» або «Кольорова схема». Розглянемо перехід до графічного редактора. При натисканні на кнопку «Редактор графіки» користувач автоматично перенаправляється на сторінку http://localhost:3000/editor, де відображається інтерфейс редактора. Вигляд цієї сторінки продемонстровано на рисунку Б.4.

Рисунок Б.4 – Інтерфейс графічного редактору

У правій частині інтерфейсу розміщується полотно для редагування, ліворуч – сайдбар із набором інструментів графічного редактора, а у верхній частині – елементи керування файлами. Нижче розташована панель налаштувань

вибраного інструмента з сайдбару. Приклад додавання готового шаблону на полотно зображено на рисунку Б.5.

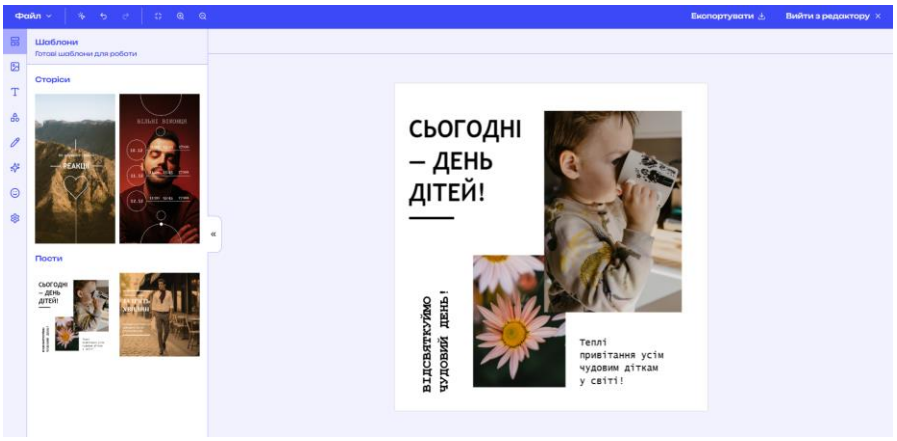


Рисунок Б.5 – Додавання шаблону до полотна

Після цього користувач отримує повний контроль над елементами редактора: може змінювати текст, масштабувати зображення, додавати емодзі або стікери, малювати на полотні, змінювати його колір, вставляти геометричні форми, а також експортувати готовий макет на свій пристрій. Результат виконаних дій у графічному редакторі показано на рисунку Б.6 та Б.7.

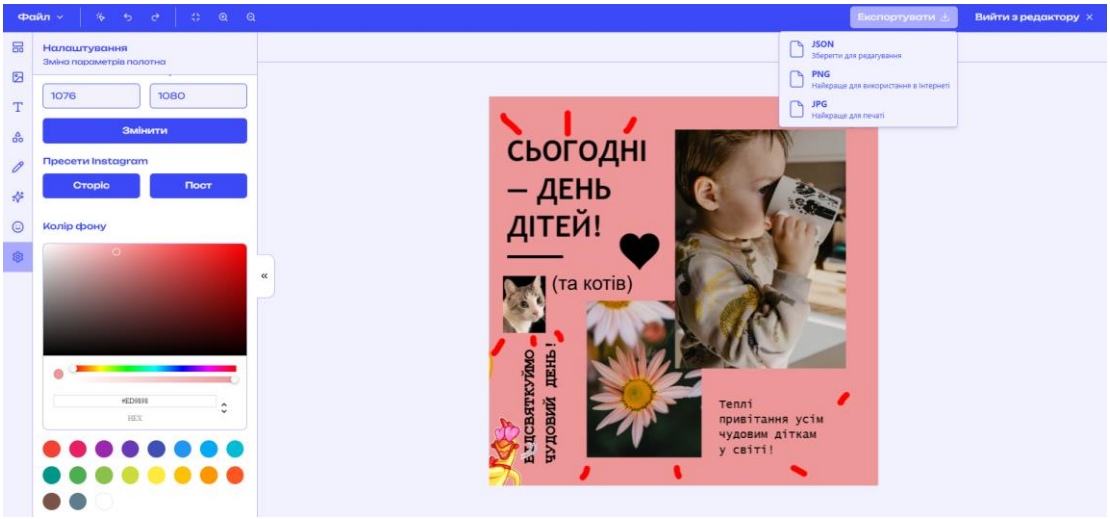


Рисунок Б.6 – Відкриття меню експортування

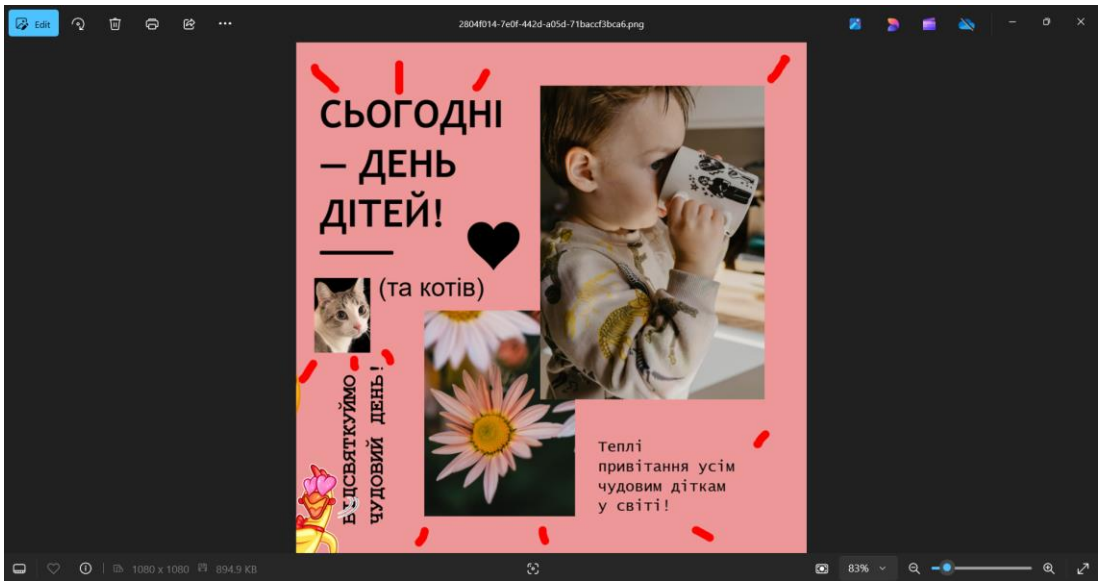


Рисунок Б.7 – Результат експортування у формат PNG

Після роботи з графічним редактором користувач може перейти до іншого інструменту – створення кольорової схеми. Для цього достатньо натиснути на кнопку «Кольорова схема» на головній сторінці, як показано на рисунку Б.8.

Рисунок Б.8 – Користувацький інтерфейс інструменту «Кольорова схема»

Інтерфейс інструменту створення кольорової схеми організовано зручно та логічно. Зліва розташований сайдбар, який містить назву інструменту, палітру для вибору базового кольору, випадаючі списки для вибору типу схеми

(наприклад, монохроматична, контрастна) та варіації (стандартна, пастельна тощо), а також кнопки для генерації кольорової палітри та виходу з інструменту.

Справа знаходиться додатковий сайдбар, де відображається список згенерованих кольорів у форматі HEX, а також розміщена кнопка для скидання завантажених зображень.

У центральній частині екрана розміщується попередньо підготовлений шаблон Instagram-профілю з сіткою 3x4, аватаром користувача та п’ятьма сторіс.

Щоб згенерувати кольорову схему, користувачу потрібно вибрати базовий колір у палітрі, визначити тип схеми та варіацію, після чого натиснути кнопку «ЗГЕНЕРУВАТИ ВІЗУАЛ». На основі обраних параметрів буде створено палітру, яка автоматично застосовується до шаблону профілю. Приклад результату наведено на рисунку Б.9.

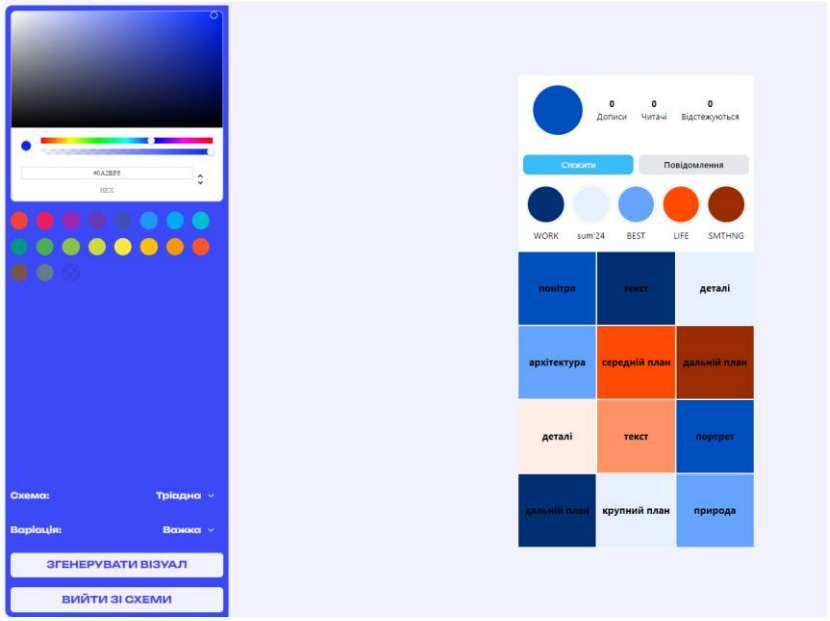


Рисунок Б.9 – Результат генерації кольорової схеми

Для додавання зображень до шаблону кольорової схеми користувачу потрібно натиснути на один із елементів Instagram-шаблону (пости, сторіс або аватар). Після цього відкриється вікно вибору файлу, де можна обрати зображення зі свого пристрою. Завантажене зображення автоматично вставляється у відповідну комірку шаблону. У разі потреби видалити всі

зображення, достатньо натиснути кнопку «СКИНУТИ ЗОБРАЖЕННЯ», після чого шаблон повернеться до початкового вигляду без зображень. Результат виконання цих дій зображено на рисунку Б.10.

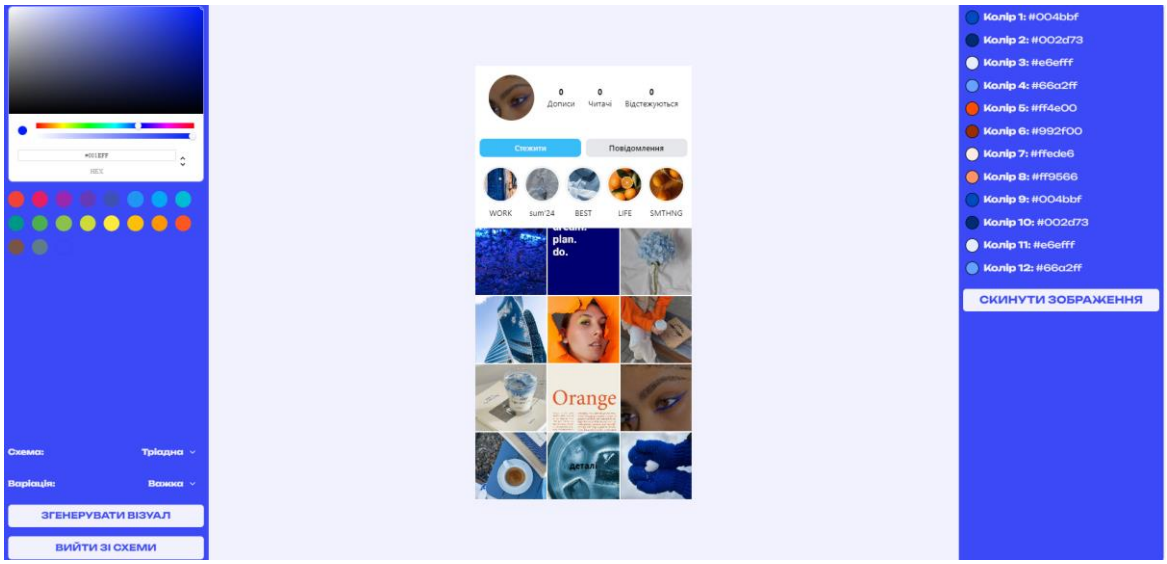


Рисунок Б.10 – Результат додавання зображень

ДОДАТОК В

Код до програмної реалізації

```
console.log(" ############## api/auth/login check password : " + password + "  " + user_password );
const passwordMatches = bcrypt.compareSync( password, user_password );

  if( passwordMatches == false )
      return NextResponse.json( { message: "Невірний пароль" }, { status: 401 });

  console.log("############## api/auth/login user ok " );

  // Set cookie or session (simplified)
  const response = NextResponse.json(user);
  response.cookies.set("auth-token", "simulated-jwt-token", {
    httpOnly: true,
    secure: process.env.NODE_ENV === "production",
    maxAge: 60 * 60 * 24 * 7, // 1 week
    path: "/",
  });

  setUserLogged(  user );
  return response;

 } catch (error) {
  console.error("Login error:", error);
  return NextResponse.json(
    { message: "Помилка при вході" }, { status: 500 }
  );
 }
}
```

Рисунок В.1 – Файл «login-route.js» для авторизації користувача

```
64    // Створюємо посилання на HTML-елементи для полотна та контейнера
65    const canvasRef = useRef(null);
66    const containerRef = useRef<HTMLDivElement>(null);
67
68    useEffect(() => {
69      // Ініціалізуємо полотно Fabric.js з основними налаштуваннями
70      const canvas = new fabric.Canvas(canvasRef.current, {
71        controlsAboveOverlay: true,
72        preserveObjectStacking: true, // Зберігаємо порядок шарів при переміщенні об'єктів
73      });
74    // Передаємо початкове полотно та контейнер у функцію ініціалізації редактора
75      init({
76        initialCanvas: canvas,
77        initialContainer: containerRef.current!,
78      });
79   // Очищуємо ресурси (знищуємо полотно) при видаленні компонента
80      return () => {
81        canvas.dispose();
82      };
83    }, [init]);
84
85    //Рендеринг React-компонентів
86    return (
87      <div className={unbounded.className}>
88        <div className="h-full flex flex-col bg-primary">
89          <Navbar
90            editor={editor}
91            activeTool={activeTool}
92            onChangeActiveTool={onChangeActiveTool}
93          />
94          <div className="absolute h-[calc(100%-48px)] w-full top-[48px] flex bg-primary">
95            <Sidebar
96              activeTool={activeTool}
97              onChangeActiveTool={onChangeActiveTool}
98            />
99
100           <ShapeSidebar
101             editor={editor}
```

Рисунок В.2 – Основний файл editor.tsx графічного редактору

```ts
interface UseHistoryProps {
  canvas: fabric.Canvas | null;
  saveCallback?: (values: {
    json: string;
    height: number;
    width: number;
  }) => void;
};

export const useHistory = ({ canvas, saveCallback }: UseHistoryProps) => {
  const [historyIndex, setHistoryIndex] = useState(0);
  const canvasHistory = useRef<string[]>([]);
  const skipSave = useRef(false);

  const canUndo = useCallback(() => {
    return historyIndex > 0;
  }, [historyIndex]);

  const canRedo = useCallback(() => {
    return historyIndex < canvasHistory.current.length - 1;
  }, [historyIndex]);

  const save = useCallback((skip = false) => {
    if (!canvas) return;

    const currentState = canvas.toJSON(JSON_KEYS);
    const json = JSON.stringify(currentState);

    if (!skip && !skipSave.current) {
      canvasHistory.current.push(json);
      setHistoryIndex(canvasHistory.current.length - 1);
    }
```

Рисунок В.3 – Хук Use-history для ефекту пам’яті

use-clipboard.ts

src > features > editor > hooks > use-clipboard.ts > useClipboard > paste > useCallba

```ts
import { fabric } from "fabric";
import { useCallback, useRef } from "react";

interface UseClipboardProps {
  canvas: fabric.Canvas | null;
};

export const useClipboard = ({
  canvas
}: UseClipboardProps) => {
  const clipboard = useRef<any>(null);

  const copy = useCallback(() => {
    canvas?.getActiveObject()?.clone((cloned: any) => {
      clipboard.current = cloned;
    });
  }, [canvas]);

  const paste = useCallback(() => {
    if (!clipboard.current) return;

    clipboard.current.clone((clonedObj: any) => {
      canvas?.discardActiveObject();
      clonedObj.set({
        left: clonedObj.left + 10,
        top: clonedObj.top + 10,
        evented: true,
      });
```

Рисунок В.4 – Хук Use-clipboard для дублювання елементів

```
export const Scheme = () => {
  const divColorRef7 = useRef<HTMLDivElement>(null);
  const divColorRef8 = useRef<HTMLDivElement>(null);
  const divColorRef9 = useRef<HTMLDivElement>(null);
  const divColorRef10 = useRef<HTMLDivElement>(null);
  const divColorRef11 = useRef<HTMLDivElement>(null);
  const divColorRef12 = useRef<HTMLDivElement>(null);

  const [color1, setColor1] = useState("3C49F6");
  const [color2, setColor2] = useState("3C49F6");
  const [color3, setColor3] = useState("3C49F6");
  const [color4, setColor4] = useState("3C49F6");
  const [color5, setColor5] = useState("3C49F6");
  const [color6, setColor6] = useState("3C49F6");
  const [color7, setColor7] = useState("3C49F6");
  const [color8, setColor8] = useState("3C49F6");
  const [color9, setColor9] = useState("3C49F6");
  const [color10, setColor10] = useState("3C49F6");
  const [color11, setColor11] = useState("3C49F6");
  const [color12, setColor12] = useState("3C49F6");

  var hexToHsl = require("hex-to-hsl");

  useEffect(() => {
    if (schemeType === "mono") {
      setSchemeTypeString("Монохроматична");
    }
    if (schemeType === "contrast") {
      setSchemeTypeString("Контрастна");
    }
    if (schemeType === "triade") {
      setSchemeTypeString("Тріадна");
    }
    if (schemeType === "analogic") {
      setSchemeTypeString("Аналогова");
    }
  }, [schemeType]);

  useEffect(() => {
    if (schemeVariation === "default") {
      setSchemeVariationString("Стандартна");
    }
    if (schemeVariation === "pastel") {
      setSchemeVariationString("Пастельна");
    }
    if (schemeVariation === "soft") {
      setSchemeVariationString("М'яка");
    }
    if (schemeVariation === "light") {
      setSchemeVariationString("Світла");
```

Рисунок В.5 – Основний файл scheme.tsx кольорової схеми

```
const newPalette = () => {
  // Логування кольору користувача для дебагу
  console.log(userColor);
  // Перевірка на вибраний тип схеми кольорів "mono" (монохроматична)
  if (schemeType === "mono") {
      // Перевірка, чи є введений колір
      if (userColor) {

          // Регулярний вираз для перевірки коректності кольору в форматі HEX
          const hexColorRegex = /^#([A-Fa-f0-9]{6}|[A-Fa-f0-9]{3})$/;

          // Якщо колір вірний, продовжуємо обробку
          if (hexColorRegex.test(userColor)) {
              // Отримуємо відтінок кольору (hue) з формату HEX в HSL
              const hue = hexToHsl(userColor)[0];

              // Підключення бібліотеки для генерації палітри кольорів
              var ColorScheme = require("color-scheme");
              var scheme = new ColorScheme();
              // Генерація палітри на основі відтінку, типу та варіації схеми
              scheme.from_hue(hue).scheme(schemeType).variation(schemeVariation);
              // Встановлення кольорів в стейти для використання в інтерфейсі
              setColor1(scheme.colors()[0]);
              setColor2(scheme.colors()[1]);
              setColor3(scheme.colors()[2]);
              setColor4(scheme.colors()[3]);
              setColor5(scheme.colors()[0]);
              setColor6(scheme.colors()[1]);
              setColor7(scheme.colors()[2]);
              setColor8(scheme.colors()[3]);
              setColor9(scheme.colors()[0]);
              setColor10(scheme.colors()[1]);
              setColor11(scheme.colors()[2]);
              setColor12(scheme.colors()[3]);
```

Рисунок В.6 – Функція newPalette

```
    // Створення інпуту для вибору файлу
    const divRef = event.currentTarget;
    const input = document.createElement("input");
    input.type = "file";
    input.accept = "image/*";

    // Перетворення кольору в CSS фільтр
    const filterFromColor = CssFilterConverter.hexToFilter("#" + colorString);
    console.log(filterFromColor);

    // Якщо є колір, застосовуємо hue-rotate
    if (filterFromColor.color) {
      const hueRotate = filterFromColor.color.match(
        /hue-rotate\((\d+)deg\)/
      )?.[1];

      // Обробник зміни файлу
      input.onchange = (e) => {
        const fileInput = e.target as HTMLInputElement;
        if (fileInput.files && fileInput.files.length > 0) {
          const file = fileInput.files[0];
          const reader = new FileReader();

          reader.onload = () => {
            // Вставка зображення та застосування стилів
            divRef.style.backgroundImage = `url(${reader.result})`;
            divRef.style.backgroundPosition = "center";
            divRef.style.backgroundSize = "cover";
            divRef.style.backgroundRepeat = "none";
            divRef.style.filter = `hue-rotate(${hueRotate}deg)`;
            divRef.style.mixBlendMode = "multiply";
            divRef.innerHTML = "";
          };
          reader.readAsDataURL(file);
```

Рисунок В.7 – Функція handleDivClick для вставки зображень на схему

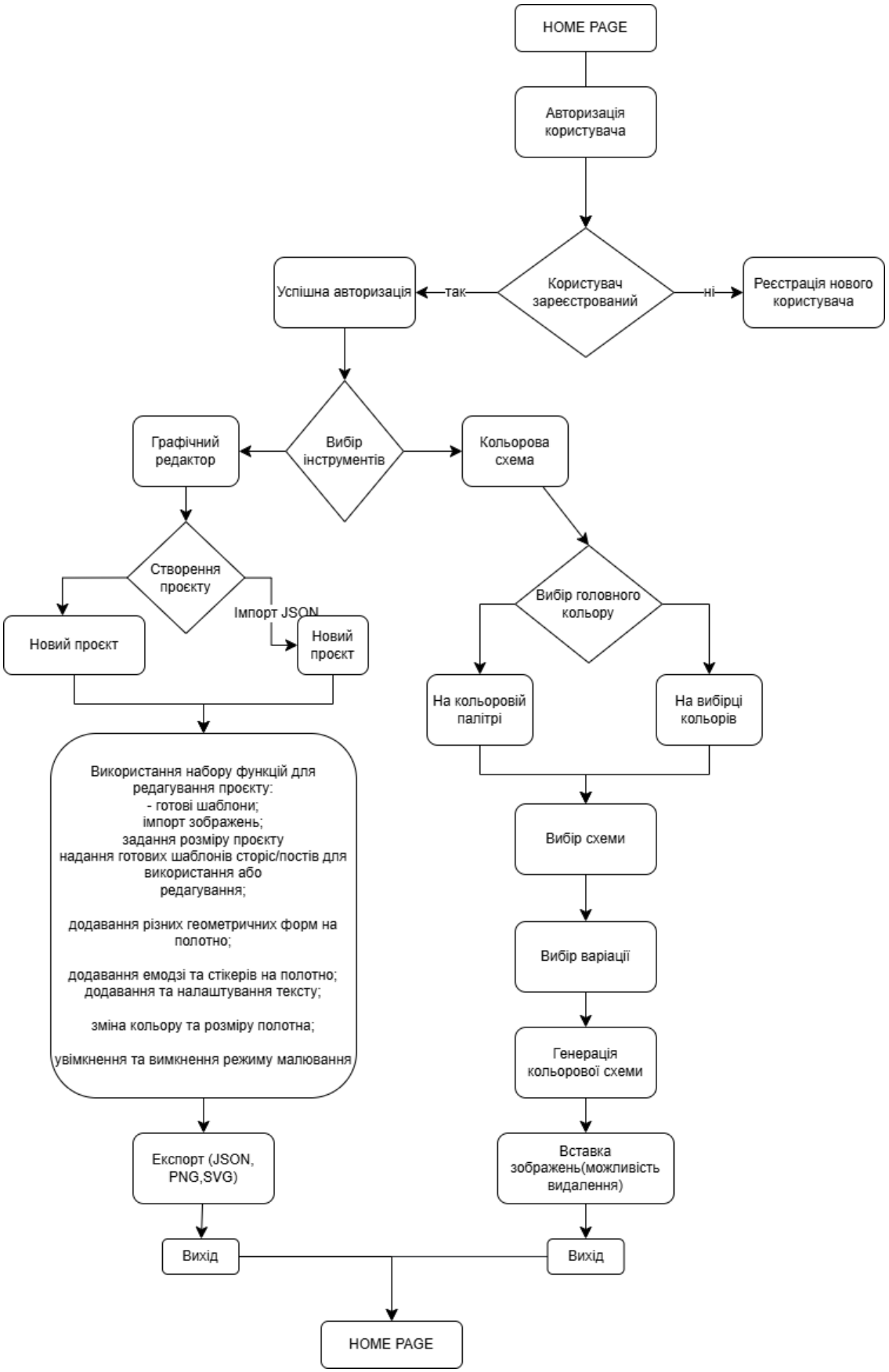


Рисунок В.8 – Схема архітектури проєкту

Анотація

Васюта Д.В. Розробка вебдодатку для SMM-спеціалістів на основі Next.js.

Кваліфікаційна робота на здобуття освітнього ступеня «бакалавр» за спеціальністю 122 Комп'ютерні науки. Волинський національний університет імені Лесі Українки, Луцьк, 2025 р.

Наукова робота містить інформацію про дослідження та розробку вебдодатку «SMM-helper», призначеного для оптимізації щоденних завдань спеціалістів у сфері соціального медіа-маркетингу. Застосунок створено з використанням фреймворку Next.js, що поєднує переваги React та серверного рендерингу, що забезпечує високу продуктивність і швидке завантаження контенту. Основна мета розробки – об'єднання ключових функцій (створення візуального контенту, керування стилістикою Instagram-профілю, планування публікацій) в одному інтерфейсі без потреби у використанні сторонніх сервісів.

У процесі роботи здійснено глибоке вивчення потреб цільової аудиторії, зокрема SMM-спеціалістів, які працюють із графікою, текстами та календарями контенту. Було проведено аналіз конкурентних платформ, який виявив фрагментарність існуючих рішень та низький рівень інтеграції. Для побудови зручного й адаптивного інтерфейсу застосовано Tailwind CSS та бібліотеку компонентів shadcn/ui, що дозволило створити кросплатформений дизайн. До складу додатку також входить графічний редактор, реалізований на основі Fabric.js, що забезпечує можливість редагування зображень, розміщення елементів дизайну та створення кастомного візуалу.

У результаті реалізовано комплексний програмний продукт, який дозволяє SMM-фахівцям зменшити час на виконання рутинних завдань, підвищити ефективність роботи та зосередитися на творчих аспектах контент-маркетингу.

Ключові слова: SMM, вебдодаток, Next.js, React, графічний редактор, планування публікацій, кольорова схема, JWT-автентифікація, MariaDB, Tailwind CSS, Fabric.js.