Вінницький національний технічний університет
(повне найменування вищого навчального закладу)

Факультет інтелектуальних інформаційних технологій та автоматики
(повне найменування інституту, назва факультету (відділення))

Кафедра комп'ютерних наук
(повна назва кафедри (предметної, циклової комісії))

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ З ПРОЕКТУВАННЯ WEB-РЕСУРСІВ. ЧАСТИНА 1. КЛІЄНТСЬКА ЧАСТИНА»

Виконав: студент 2 курсу, групи 2КН-22м
спеціальності 122 – Комп'ютерні науки
(шифр і назва напряму підготовки, спеціальності)
Сторожук А. С.
(прізвище та ініціали)

Керівник: к.т.н., доцент кафедри КН
Сілагін О. В.
(прізвище та ініціали)
«07» 12 2023 р.

Опонент: к.т.н., доцент кафедри САІТ
Козачко О. М.
(прізвище та ініціали)
«07» 12 2023 р.

Допущено до захисту
Завідувач кафедри КН
д.т.н., проф. Яровий А.А.
(прізвище та ініціали)
«08» 12 2023 р.

Вінниця ВНТУ – 2023 рік

Вінницький національний технічний університет

Факультет інтелектуальних інформаційних технологій та автоматизації

Кафедра Комп'ютерних наук

Рівень вищої освіти – другий (магістерський)

Галузь знань – 12 «Інформаційні технології»

Спеціальність – 122 «Комп'ютерні науки»

Освітньо-професійна програма – «Системи штучного інтелекту»

ЗАТВЕРДЖУЮ

Завідувач кафедри КН

проф., д.т.н. Яровий А. А.

«29» 03 2023р.

ЗАВДАННЯ

НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Сторожуку Антону Сергійовичу

1. Тема роботи: Інформаційна технологія з проектування WEB-ресурсів. Частина 1. Клієнтська частина.

керівник роботи: Сілагін О.В. к.т.н., доц. каф. КН
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу "18" 09 2023року №247

2. Термін подання студентом роботи: 13. 11. 2023р.

3. Вихідні дані до роботи: розробити клієнтську частину плагіна і налаштування, які доступні тільки адміністратору; час відгуку не повинен перевищувати 0,5 секунд.

4. Зміст текстової частини (перелік питань, які потрібно розкрити): Аналіз відомих інформаційних технологій з проектування веб-ресурсів; Покращення існуючої технології проектування Front-end; Моделювання засобами UML роботи клієнтської частини плагіна для Wordpress з функціональними можливостями управління рекламою на WEB-ресурсі; Проектування клієнтської

частини плагіна для wordpress з функціональними можливостями управління рекламою на WEB-ресурсі; Обґрунтування вибору інструментів розробки.

5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень): Схема покращеної технології проектування клієнтської частини WEB-ресурсів з використанням плагіна для управління рекламою, діаграма класів, діаграма активностей, діаграма послідовності, діаграма станів, скріншоти налаштувань адмін-панелі, результати роботи плагіна.

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1, 2, 3	Сілагін О.В., к.т.н., доц. каф.КН	29.08.23	30.10.23
4	Ратушняк О.Г., к.т.н., доц. каф. ЕПВМ	08.10.23	27.10.23

7. Дата видачі завдання 29.08.2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Теоретичні основи та інформаційна технологія проектування WEB-ресурсів	01.09.23 - 05.09.23	
2	Проектування клієнтської частини плагіна	06.09.23 – 16.09.23	
3	Реалізація клієнтської частини плагіна	17.09.23 – 02.10.23	
4	Розробка економічної частини клієнтської частини плагіна	03.10.23 – 23.10.23	
5	Оформлення пояснювальної записки і графічного матеріалу	24.10.23 – 01.11.23	
6	Попередній захист	15.11.23	
7	Остаточний захист	12.12.23	

Студент _____ (підпис) Сторожук А.С.

Керівник роботи _____ (підпис) Сілагін О.В.

АНОТАЦІЯ

УДК 004.8

Сторожук А. С. Інформаційна технологія з проектування WEB-ресурсів: Front-end частина. Магістерська кваліфікаційна робота зі спеціальності 122 – «Комп'ютерні науки», освітня програма – «Системи штучного інтелекту». Вінниця: ВНТУ, 2022. 121с.

На укр. мові. Бібліогр.: 37 назв; рис.: 25; табл. 8.

Магістерська кваліфікаційна робота входить до складу комплексної магістерської роботи і присвячена використанню інформаційних технологій у процесі створення веб-ресурсів. Зокрема, це включає розробку плагіна для WordPress, який забезпечує функціональне управління рекламою на WEB-ресурсі. У рамках даного дослідження були створені компоненти плагіна для WordPress з можливістю ефективного керування рекламою на веб-ресурсі. Це виконано на основі системи управління вмістом WordPress з використанням Open Server Panel, і використані мови програмування PHP та JavaScript, а також каскадні таблиці стилів CSS та мова розмітки гіпертексту HTML.

Графічна частина складається з 6 плакатів.

У економічній чатині розраховано витрати на розробку нового програмного продукту, а також основні економічні показники, за допомогою яких доводиться ефективність розробленого програмного продукту та доцільність використання розробки.

Ключові слова: веб-ресурс, плагін, Wordpress, клієнтська частина, система управління вмістом, реклама.

ABSTRACT

Storozhuk A.S. Information technology for designing WEB resources. Part 2. The client-side part. Master's thesis in the specialty 122 «Computer sciences», educational program «Artificial intelligence systems». Vinnytsia: VNTU, 2022. 121 p.

In Ukrainian language. Bibliographer: 37 titles; fig.: 25; table 8.

The Master's qualification work is part of the comprehensive master's project and is dedicated to the utilization of information technologies in the process of creating web resources. Specifically, this includes the development of a plugin for WordPress that ensures functional management of advertising on the web resource. Within the framework of this research, plugin components for WordPress were created with the ability to effectively manage advertising on the web resource. This was done based on the WordPress content management system using the Open Server Panel, and the programming languages PHP and JavaScript were employed, along with Cascading Style Sheets (CSS) for styling and Hypertext Markup Language (HTML) for markup.

The graphic part consists of 6 posters.

In the economic aspect, the costs for developing the new software product are calculated, along with key economic indicators that demonstrate the effectiveness of the developed software product and the feasibility of its implementation.

Keywords: web resource, plugin, WordPress, client-side, front-end, content management system, advertising.

ЗМІСТ

ВСТУП

Актуальність теми.

Актуальність створення плагіна для управління рекламою на WEB-ресурсі у контексті інформаційних технологій при проектуванні WEB-ресурсів обумовлена постійним ростом і важливістю рекламного середовища в Інтернеті. У сучасному світі, де веб-ресурси стають необхідною частиною бізнесу, успішна реклама стає ключовою для досягнення успіху в онлайн середовищі. Реклама на WEB-ресурсах не лише привертає цільову аудиторію, але й є джерелом доходу для власників веб-сайтів.

Зростання популярності платформи WordPress, яка використовується для створення WEB-ресурсів, робить створення плагіна для управління рекламою на WordPress-сайтах особливо актуальним. Це дозволяє WEB-розробникам і власникам сайтів розширювати функціональність своїх ресурсів і забезпечувати ефективну рекламну стратегію безпосередньо на своєму сайті.

Розробка такого плагіна передбачає розуміння основних принципів інформаційних технологій при проектуванні WEB-ресурсів, включаючи створення інтерфейсу користувача, оптимізацію швидкості завантаження сторінок та забезпечення безпеки.

Отже, дослідження та розробка плагіна для управління рекламою на WordPress-сайтах у контексті інформаційних технологій при проектуванні WEB-ресурсів є актуальною та значущою темою, що сприяє поліпшенню функціональності веб-ресурсів та оптимізації їх рекламних зусиль.

Зв’язок роботи з науковими програмами, планами, темами.

Магістерська кваліфікаційна робота виконана відповідно до напряму наукових досліджень кафедри комп’ютерних наук Вінницького національного технічного університету 22 К1 «Моделі, методи, технології та пристрої інтелектуальних інформаційних систем управління, економіки, навчання та комунікацій» та плану наукової та навчально-методичної роботи кафедри.

Мета дослідження.

Метою дослідження є покращення технології проектування клієнтської частини WEB-ресурсів за рахунок використання інтегрованого плагіну для розширення функціоналу по управлінню рекламою та автоматизації процесів статистичної обробки.

Задачі дослідження.

Для досягнення поставленої мети необхідно розв’язати такі задачі подальшого дослідження:

- виконати аналіз різновидів веб-ресурсів;
- розробити покращену технологію Front-end частини плагіна;
- спроектувати складові частини плагіна, що відповідають за Front-end частину;
- розробити Front-end частину плагіна у вибраних мовних середовищах;
- розробити економічну частину Front-end частини плагіна.

Об’єкт дослідження.

Об’єктом дослідження є технологічний процес розробки клієнтської частини WEB-ресурсів.

Предмет дослідження.

Предметом дослідження є технології та програмне забезпечення для створення клієнтської частини плагіна для WordPress з функціональними можливостями управління рекламою на веб-ресурсі.

Наукова новизна одержаних результатів.

Наукова новизна одержаних результатів заключається в удосконаленні інформаційної технології з проектування WEB-ресурсів, яка на відміну від існуючих використовує нову інформаційну модель управління вмістом даних, що дозволяє розширити функціональні можливості акумуляції і взаємодії баз даних рекламних кампаній і користувачів.

Практичне значення отриманих результатів.

Практичне значення отриманих результатів полягає у розробленні клієнтської частини плагіна для WordPress, що надає розширені функціональні можливості по управлінню рекламою на веб-ресурсі.

Використання інформаційної технології у проектуванні WEB-ресурсів сприяє покращенню якості розробки сайтів, відповідності сучасним стандартам та технологіям, поліпшенню користувацького досвіду та ефективному функціонуванню веб-ресурсів.

Розробка плагіна для управління рекламою дозволяє гнучко налаштовувати параметри рекламних кампаній, відстежувати їх ефективність та спрямовувати рекламні матеріали до цільової аудиторії, що сприяє оптимізації витрат на рекламу та підвищує прибутковість WEB-ресурсу.

Впровадження.

Результати дослідження впроваджені в створеному сайті стартап школи «Sikorsky challenge», який експлуатується ВНТУ [1]. Довідка про впровадження розміщена в додатку Д.

Достовірність теоретичних положень.

Достовірність теоретичних положень магістерської кваліфікаційної роботи підтверджується коректністю постановки завдання та експериментальними дослідженнями тестування програмної реалізації клієнтської частини плагіна для Wordpress з функціональними можливостями управління рекламою на WEB-ресурсі.

Особистий внесок здобувача.

Усі результати, що наведені у магістерській кваліфікаційній роботі, отримані самостійно. У працях, які написано у співавторстві, здобувачу належать: UML-діаграми Front-end частини плагіна.

Апробація результатів роботи.

Результат роботи були апробовані на LI і LII науково-технічних конференціях «Молодь в науці» ВНТУ в 2022 та 2023 роках [2, 3].

Публікації.

За результатами роботи опубліковані тези доповідей на науково-технічних конференціях «LI Науково-технічна конференція Молодь в науці» (м. Вінниця, Україна) [2] та «LII Науково-технічна конференція Молодь в науці» (м. Вінниця, Україна) [3] у 2022 та 2023 роках.

По теоретичним результатам дослідження подана стаття до публікації у фаховому журналі «Інформаційні технології та комп’ютерна інженерія».

1 АНАЛІЗ ІСНУЮЧИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ОРІЄНТОВАНИХ НА FRONT-END РОЗРОБКУ

1.1 Різновиди веб-ресурсів

Веб-ресурси - це сторінки або набори сторінок в Інтернеті, які мають специфічну адресу (URL) і можуть містити різноманітний контент, такий як текст, зображення, відео, звук і інше. Існує безліч різновидів веб-ресурсів, що відображають їх різноманітність та корисність для різних потреб користувачів.

Ось декілька основних різновидів веб-ресурсів:

- Сайти візитки - це невеликі веб-сайти, які представляють окрему особу, компанію чи організацію. Вони мають основну інформацію про себе, таку як контактна інформація, послуги або продукти, які надаються.
- Блоги - це веб-сайти, на яких автор (або автори) публікують регулярні записи про різні теми. Блоги можуть бути особистими чи професійними і можуть містити текст, фотографії, відео та інші медіа-елементи. Приклад блогу зі статтями наведений на рисунку 1.1.
- Новинні портали - це великі веб-сайти, які публікують новини та статті з різних галузей, таких як політика, економіка, спорт, наука тощо. Новинні портали можуть містити текстовий контент, фотографії, відео та аудіо-матеріали.
- Соціальні мережі - це веб-сайти, які дозволяють користувачам спілкуватися, обмінюватися інформацією, заводити друзів, відслідковувати новини тощо. Приклади соціальних мереж включають Facebook, Twitter, Instagram та інші.
- Електронні магазини - це веб-сайти, які дозволяють користувачам придбати товари або послуги в Інтернеті. Вони можуть містити каталоги товарів, корзини для покупок, системи оплати та інші елементи електронної торгівлі.
- Форуми та спільноти - це веб-сайти, де користувачі можуть обговорювати різні теми, ділитися досвідом, задавати питання та

надавати поради. Форуми можуть бути загальні або спеціалізовані за тематикою.

- Віртуальні бібліотеки та навчальні ресурси - це веб-сайти, які містять навчальний матеріал, книги, статті, відео-уроки та інші ресурси для навчання та самонавчання.
- Портали розваг - це веб-сайти, які містять розважальний контент, такий як ігри, відео, музика, анекдоти тощо.
- Бази даних та відкриті джерела - це веб-ресурси, які містять велику кількість інформації з різних галузей, яка може бути доступна для загального використання або певних категорій користувачів.
- Портали робочого спілкування - це веб-сайти, які використовуються компаніями для внутрішньої комунікації, спільної роботи над проектами, обміну документами та іншими корпоративними завданнями.
- Віртуальні тури та картографічні сервіси - це веб-сайти, які дозволяють користувачам відкривати світ не покидаючи дому. Вони можуть містити віртуальні тури до визначних місць, аерофотознімки, 360-градусні відео та інші інтерактивні можливості.
- Портали для саморозвитку - це веб-сайти, які надають користувачам можливість вивчати нові навички, мови, учитися програмуванню та іншим навичкам через онлайн-курси, відео-уроки та інтерактивні завдання.
- Фінансові портали - це веб-ресурси, які надають користувачам інформацію про фінансові новини, курси валют, біржові індекси та інші фінансові показники. Вони можуть також містити поради щодо інвестування та управління фінансами [4].

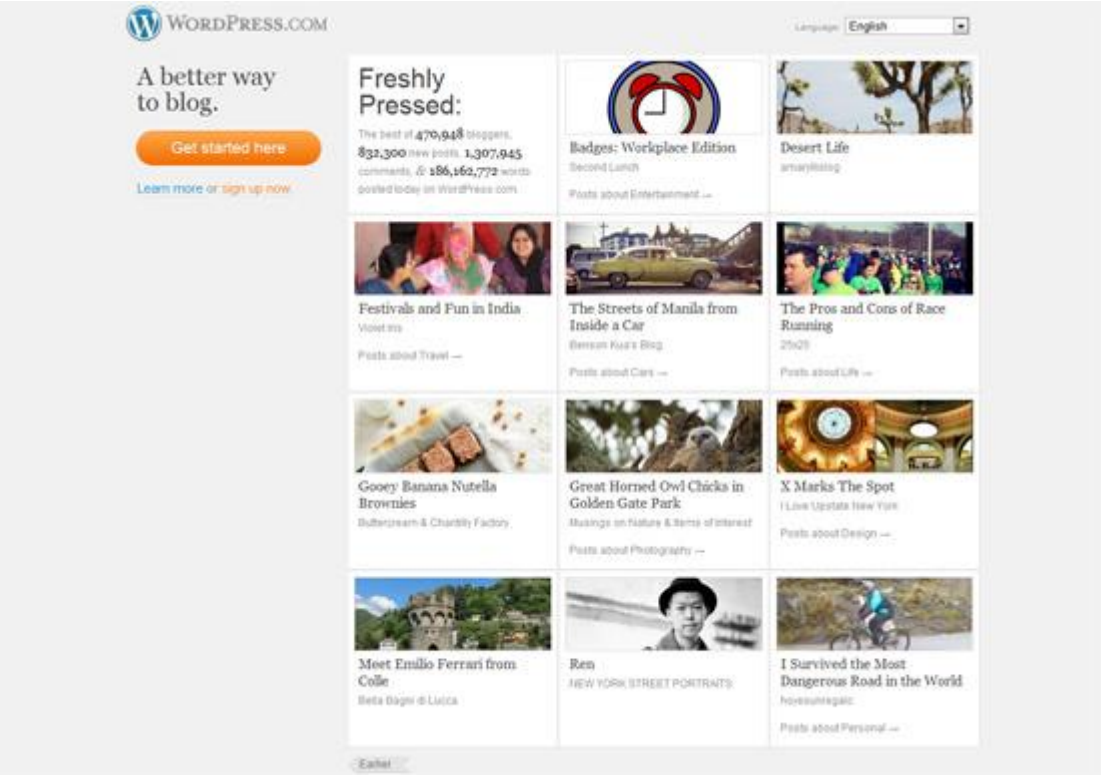

Рисунок 1.1 – Приклад блогу зі статтями

Ці різновиди веб-ресурсів створюють різноманітність інтернет-середовища та дозволяють користувачам знаходити інформацію та здійснювати різні операції в Інтернеті залежно від їхніх потреб і інтересів.

1.2 Аналіз відомих інформаційних технологій з проектування веб-ресурсів

У сучасному проектуванні веб-ресурсів використовуються різноманітні інформаційні технології для забезпечення ефективної функціональності, безпеки та зручності користування. HTML, CSS і JavaScript використовуються для створення структури веб-сторінок та надання їм вигляду та інтерактивності. Адаптивний веб-дизайн забезпечує коректне відображення вмісту на різних пристроях та розмірах екранів [5].

У розробці веб-додатків використовуються різні фреймворки, такі як Bootstrap, React і Angular, які спрощують процес розробки та надають готові компоненти для інтерфейсу користувача. На серверній стороні

використовуються технології, такі як Node.js, Django і Ruby on Rails, для обробки запитів та взаємодії з базами даних [6].

З погляду безпеки, широко використовуються технології шифрування SSL/TLS для забезпечення захищеного з'єднання між користувачем і сервером. Техніки захисту від вразливостей, такі як Cross-Site Scripting (XSS) Prevention та Cross-Site Request Forgery (CSRF) Protection, використовуються для запобігання атакам на безпеку [7].

Content Management Systems (CMS) як WordPress і Drupal допомагають легко керувати та публікувати вміст на веб-сайті, навіть без глибоких знань програмування. APIs, такі як REST і GraphQL, використовуються для обміну даними між різними компонентами програмного забезпечення через Інтернет [8].

Також розвиваються нові концепції, такі як Progressive Web Apps (PWA), які забезпечують зручний та швидкий досвід користувача, як у звичайних веб-додатках, навіть при відсутності Інтернет-з'єднання. Важливо враховувати ці технології під час розробки веб-ресурсів, щоб забезпечити їхню ефективність, безпеку та зручність для користувачів [9].

WordPress відзначається простотою використання та інтуїтивним інтерфейсом, що робить його відмінним вибором для початківців у веб-розробці. Також ця платформа має величезну кількість розширень, або плагінів, та тем, які значно розширюють його можливості.

Одна з важливих переваг - це велика активна спільнота користувачів та розробників, яка завжди готова допомогти і відповісти на питання. Це надає впевненість та підтримку для тих, хто використовує цю платформу.

WordPress також добре оптимізований для пошукових систем, що полегшує SEO-продвиження веб-сайтів. Багато тем і плагінів вже адаптовані для мобільних пристроїв, що забезпечує коректне відображення на різних пристроях.

Ще одна перевага - це можливість вибору між безкоштовною версією на WordPress.com та самостійним хостингом з WordPress.org. Це дозволяє користувачам вибирати оптимальний варіант для своїх потреб та бюджету.

І, хоча жодна платформа не є абсолютно безпечною, WordPress надає базові засоби для захисту від зловмисників і може бути безпечною з правильними налаштуваннями та використанням плагінів безпеки.

Усі ці фактори роблять WordPress дуже популярним та ефективним інструментом для великої кількості користувачів у всьому світі [10].

1.3 Роль реклами на веб-ресурсах та її управління

Реклама на веб-ресурсах відіграє важливу роль у сучасному інтернет-середовищі. Вона дозволяє веб-сайтам генерувати дохід, залучати нових користувачів і збільшувати свою популярність. Реклама може бути в різних форматах, таких як банери, текстові оголошення, відеоролики, поп-ап вікна та інші.

Роль реклами на веб-ресурсах полягає в наступних аспектах:

1. Генерація доходу: Один з головних метою реклами на веб-ресурсах є заробіток грошей. Веб-сайти можуть отримувати плату за показ рекламних матеріалів або за кожне натискання на них (платний трафік). Це дозволяє веб-сайтам отримувати фінансову підтримку та продовжувати забезпечувати якісний контент користувачам безкоштовно або за невелику плату.
2. Привертання уваги та залучення аудиторії: Реклама допомагає веб-ресурсам привернути увагу потенційних користувачів і залучити їх на свій веб-сайт. Вона може використовуватись для просування нових продуктів або послуг, розширення бренду, приваблення цільової аудиторії та підвищення свідомості про веб-ресурс.
3. Партнерські програми та співпраця: Реклама може бути використана для співпраці з іншими компаніями або партнерами. Це може включати партнерські програми, де веб-ресурс рекламує продукти або послуги інших компаній та отримує комісійні з продажів або дій користувачів.

4. Контроль над рекламним вмістом: Управління рекламою на веб-ресурсах відіграє важливу роль у забезпеченні якості та відповідності вмісту сайту. Власники веб-ресурсів можуть контролювати типи рекламних матеріалів, які вони показують, та забезпечувати, що вони відповідають їх цільовій аудиторії та сприяють позитивному користувацькому досвіду [11].

Приклад реклами на веб-ресурсі зображено на рисунку 1.2.

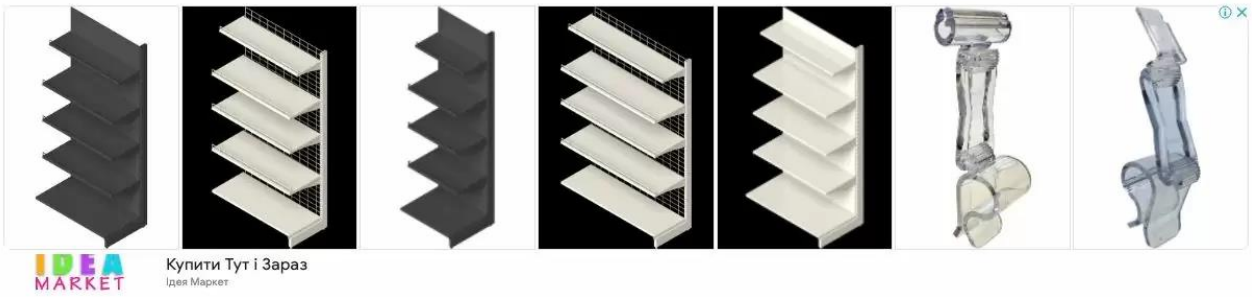

Рисунок 1.2 – Приклад реклами на веб-ресурсі

Управління рекламою на веб-ресурсах включає в себе вибір стратегій розміщення реклами, встановлення цілей та метрик для вимірювання її ефективності, налаштування рекламних кампаній, відстеження результатів та оптимізацію їх для досягнення кращих результатів.

Одним із важливих аспектів управління рекламою на веб-ресурсах є аналіз результатів рекламних кампаній. За допомогою спеціальних аналітичних інструментів, таких як Google Analytics, власники веб-сайтів можуть вимірювати ефективність своїх рекламних зусиль. Вони можуть отримувати дані про кількість переглядів рекламних матеріалів, кількість натискань на них, конверсійні показники та іншу корисну інформацію.

Управління рекламою на веб-сайтах включає в себе кілька ключових аспектів.

По-перше, важливо розробити чітку стратегію, вивчаючи аудиторію, визначаючи цілі та вибираючи відповідні рекламні платформи, такі як Google

Ads або соціальні мережі. Зміст реклами повинен бути цікавим та відповідати інтересам аудиторії.

Другий аспект - це аналітика та оптимізація. Рекламні кампанії повинні бути виміряні, і їхні результати повинні аналізуватися для зрозуміння ефективності. A/B тестування та збор даних про ROI допомагають у визначенні успішних стратегій та кампаній.

Третій аспект - це управління рекламним бюджетом. Розподілення бюджету між різними кампаніями та каналами має бути обґрунтованим для досягнення максимального впливу.

Взаємодія з рекламодавцями та партнерами також грає важливу роль. Пошук та вибір надійних партнерів для співпраці та плідної реклами взаємовигідний для всіх сторін [12].

Усі ці аспекти вимагають систематичної роботи, аналізу ринку та здатності адаптуватися до змінних умов та вимог аудиторії. Ретельне планування та аналіз є ключами до успіху в управлінні рекламою на веб-ресурсах.

Загалом, реклама на веб-ресурсах відіграє важливу роль у генерації доходу, залученні аудиторії та підтримці веб-сайтів. Ефективне управління рекламою дозволяє досягати кращих результатів і забезпечувати успішну присутність в онлайн-середовищі.

1.4 Висновок

Веб-ресурси мають різноманітні ролі та можливості. Вони можуть бути веб-сайтами, які надають інформацію про різні теми, від особистих блогів до корпоративних порталів. Також, це можуть бути соціальні мережі, форуми чи електронні магазини, які забезпечують спілкування, обмін інформацією та комерційні транзакції відповідно.

Щодо інформаційних технологій, які застосовуються в розробці веб-ресурсів, HTML, CSS, JavaScript, PHP та інші мови програмування є важливими для створення функціональних та естетично приємних веб-сайтів. Також,

фреймворки та системи управління контентом (CMS) спрощують процес створення та оновлення вмісту веб-сайтів.

У контексті реклами важливо враховувати потреби та вподобання аудиторії. Рекламні кампанії повинні бути направлені на привертання уваги, підвищення усвідомленості бренду та залучення нових користувачів. Управління рекламою включає в себе аналіз даних, вибір платформ, створення привабливого та цільового контенту, а також постійну оптимізацію рекламних стратегій на основі вимірюваних результатів.

Загалом, успіх веб-ресурсу залежить від збалансованості між технічними можливостями, розумінням аудиторії та ефективним управлінням рекламою. Лише зіставлення та інтеграція цих аспектів дозволяють створити не тільки функціональний веб-ресурс, але й ефективний засіб комунікації з аудиторією в онлайн-середовищі.

2 ПРОЕКТУВАННЯ FRONT-END ЧАСТИНИ З ВИКОРИСТАННЯМ ПОКРАЩЕНОЇ ТЕХНОЛОГІЇ

2.1 Покращення існуючої технології проектування Front-end

Покращення технології проектування Front-end у веб-розробці, зокрема на платформі WordPress, включає в себе комплексний підхід для оптимізації якості та продуктивності веб-сайтів. Цей підхід враховує адаптивний та відзивний дизайн, який забезпечує оптимальний вигляд та функціональність на різних пристроях. Оптимізація швидкості завантаження включає мінімізацію розміру файлів, зменшення запитів до сервера, використання кешування та компресії ресурсів для швидкого завантаження сторінок. Оптимізація зображень перед їхнім завантаженням, використання стиснених форматів та зменшення якості без втрати видимої якості допомагають покращити час завантаження графічних та відеоелементів. Використання Content Delivery Network (CDN) полегшує доступ до ресурсів, забезпечуючи швидке завантаження контенту. Локальні запити до сервера дозволяють мінімізувати навантаження та зменшити час відповіді на запити. Важливо також впевнитися, що веб-сайт виглядає однаково на різних веб-переглядачах, таких як Chrome, Firefox, Safari, Internet Explorer, через регулярне тестування та усунення можливих проблем. Безпека включає в себе усунення можливих уразливостей та застосування кращих практик безпеки. Оптимізація для пошукових систем (SEO) включає в себе вибір правильних ключових слів, оптимізацію метатегів та зображень для покращення видимості сайту у пошукових результатах [13]. Регулярне вивчення нових технологій та відгуків користувачів, а також регулярні оновлення контенту, допомагають підтримувати сучасність та конкурентоспроможність веб-сайту на платформі WordPress.

Важливим аспектом вдосконалення Front-end розробки є підтримка останніх стандартів мов програмування та технологій. Регулярні оновлення та адаптація кодової бази до нових версій мов програмування, бібліотек та

фреймворків дозволяють забезпечити високу ефективність та зручність розробки.

У контексті WordPress, важливо також вивчати та використовувати нові можливості та плагіни, які можуть полегшити та розширити можливості Front-end розробки. Постійне оновлення та вдосконалення плагінів, які використовуються на веб-сайті, сприяє безперервному підтриманню високого рівня функціональності.

Схему технології проектування Front-end частини плагіна управління рекламою на веб-ресурсі зображено на рисунку 2.1.

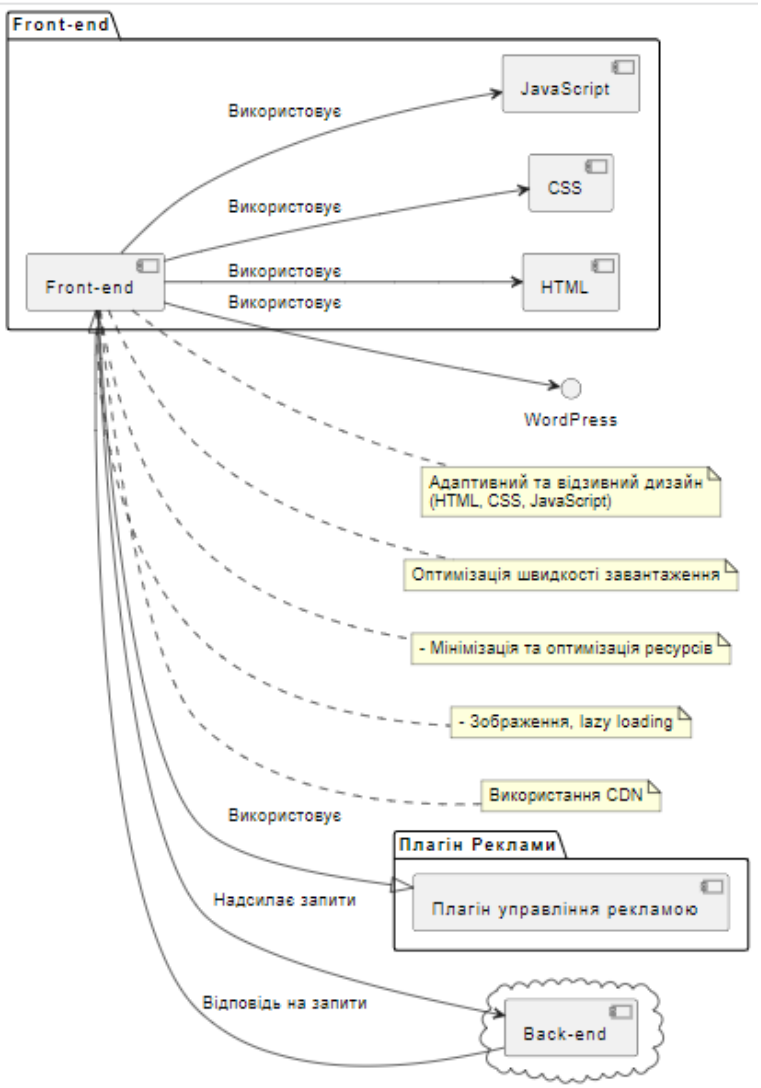

Рисунок 2.1 – Схема покращеної технології проектування клієнтської частини WEB-ресурсів з використанням плагіна для управління рекламою

Ця схема представляє архітектурну структуру Front-end частини плагіна для управління рекламою на платформі WordPress. Давайте розглянемо основні компоненти та їх взаємодію:

- Плагін управління рекламою (AdsPlugin): це центральний компонент плагіну, відповідальний за управління рекламними матеріалами та їх відображення.
- Front-end частина: представляє фронтенд частину плагіну, яка відповідає за взаємодію з користувачем та відображення рекламних елементів.
- HTML, CSS, JavaScript: ці три компоненти представляють технології, які використовуються для створення адаптивного та відзивного дизайну. HTML відповідає за структуру, CSS - за стилізацію, а JavaScript - за динамічну взаємодію.
- WordPress: це основний сайт, на якому розгорнутий плагін. Frontend взаємодіє з WordPress, використовуючи зазначені технології.
- Back-end (Backend): це хмарний сервер, який обробляє запити від Frontend та взаємодіє з базою даних та іншими бекенд сервісами.
- Використання технологій: схема показує використання HTML, CSS, та JavaScript для створення адаптивного та відзивного інтерфейсу. Зображення, lazy loading, та використання CDN підкреслюють оптимізацію швидкості завантаження.
- Взаємодія з AdsPlugin: Frontend взаємодіє з AdPlugin, щоб отримати та відображати рекламні матеріали. Це включає відправку запитів та отримання відповідей.

Додаткові примітки надають контекст щодо основних функцій та переваг плагіну, таких як адаптивність, швидкість завантаження, та взаємодія із зовнішніми сервісами.

Ця схема служить інструментом для розуміння взаємодії різних компонентів вашого плагіну та може використовуватися як посібник для розробників, щоб вони могли краще розуміти структуру та взаємодію системи.

Забезпечення безпеки є однією з найважливіших складових покращення

Front-end. Регулярні аудити коду, виявлення та усунення можливих уразливостей, а також використання заходів безпеки на рівні клієнта допомагають запобігати можливим атакам та зберігати конфіденційні дані користувачів у безпеці.

Для підтримки SEO-оптимізації важливо розробляти та впроваджувати стратегії ключових слів, а також оптимізувати метатеги та зображення. Регулярне оновлення контенту та взаємодія з інструментами аналітики допомагають відстежувати ефективність та вносити корективи в стратегії SEO.

Загалом, постійне вдосконалення Front-end розробки вимагає системного та комплексного підходу, щоб забезпечити не лише оптимальну продуктивність та безпеку, але й відмінний користувацький досвід для відвідувачів веб-сайту.

2.2 Моделювання засобами UML роботи Front-end частини плагіна для Wordpress з функціональними можливостями управління рекламою на WEB-ресурсі

Моделювання засобами UML (Unified Modeling Language) є ефективним способом проектування програмного забезпечення, включаючи розробку front-end частини плагіна для WordPress з функціональностями управління рекламою на веб-ресурсі. У цьому контексті UML може використовуватися для створення різних видів діаграм, що допомагають зрозуміти і визначити структуру та поведінку системи. Ось кілька типів діаграм UML, які можуть бути корисні при моделюванні front-end частини плагіна для управління рекламою:

1. Use Case Diagrams (Діаграми використання). Use Case діаграми допомагають ідентифікувати функціональність системи з точки зору користувача. У цьому випадку, вони можуть відобразити взаємодію користувачів з рекламним плагіном, такі як створення рекламних кампаній, редагування реклами та інші дії.
2. Activity Diagrams (Діаграми активності). Activity діаграми можуть бути

використані для моделювання послідовності операцій в системі. Наприклад, діаграма активності може показати процес створення нової рекламної кампанії, включаючи вибір рекламного матеріалу, встановлення параметрів та підтвердження операції.

3. Sequence Diagrams (Діаграми послідовності). Sequence діаграми можуть бути використані для моделювання взаємодії різних об'єктів у системі впродовж часу. Вони показують порядок викликів методів та повідомлень між об'єктами. Це може бути корисно при розробці функціоналу, який взаємодіє з сервером рекламних служб або базою даних.

4. Component Diagrams (Діаграми компонентів). Component діаграми допомагають візуалізувати архітектуру системи та компоненти, з яких вона складається. Це може включати компоненти front-end плагіна, такі як інтерфейс користувача, контролери та в'язку з back-end частиною.

5. State Machine Diagrams (Діаграми станів). State Machine діаграми можуть використовуватися для моделювання різних станів, через які проходить система відповідно до взаємодії з користувачем або змін зовнішніх умов. Наприклад, можна відобразити різні стани рекламних кампаній, такі як активна, призупинена або завершена.

6. Class Diagrams (Діаграми класів). Class діаграми використовуються для показування класів системи та їх взаємозв'язків. Вони можуть відображати класи, пов'язані з front-end реалізацією плагіна, такі як класи інтерфейсів, компоненти відображення і контролери [14].

На Front-end частині плагіна створено 5 класів, діаграма класів зі встановленими зв'язками зображена на рисунку 2.2.

Клас AdvertisementCampaign представляє рекламну кампанію і має атрибути для збереження назви, матеріалу, бюджету, часу початку та закінчення кампанії, а також її статусу. Методи цього класу дозволяють створювати та редагувати кампанії, а також змінювати їх статус.

Клас AdvertisementMaterial відповідає за матеріал реклами, і містить інформацію про тип матеріалу та його вміст. Цей клас має метод для створення нового матеріалу.

Перелічення CampaignStatus визначає можливі статуси для рекламних кампаній, такі як активна, призупинена або завершена.

Клас AdvertisementManager має методи для створення нових кампаній, редагування існуючих кампаній та зміни їх статусу. Цей клас взаємодіє з класом AdvertisementCampaign для управління кампаніями.

Клас UserInterface відповідає за інтерфейс користувача і має методи для відображення деталей кампанії та повідомлень про помилки. Цей клас взаємодіє з користувачем, показуючи інформацію та отримуючи від нього вхідні дані.

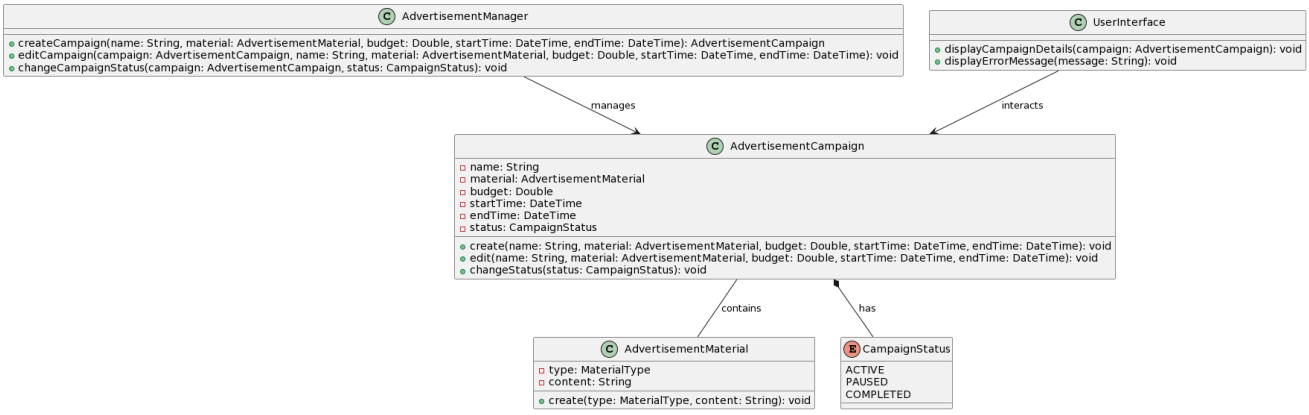

Рисунок 2.2 – Діаграма класів

Діаграма послідовності показана на рисунку 2.3.

1. Користувач взаємодіє з інтерфейсом, обираючи опцію "Створити кампанію".
2. Front-end частина системи перевіряє права користувача, щоб переконатися, що користувач має відповідні дозволи на створення кампанії.
3. Front-end відображає форму створення кампанії: якщо у користувача є відповідні права, front-end відображає форму, де користувач може ввести дані для нової кампанії.

4. Користувач вводить необхідні дані про кампанію, такі як назву, бюджет, матеріали тощо.
5. Front-end валідує дані: front-end перевіряє введені користувачем дані на валідність, щоб переконатися, що вони відповідають вимогам системи.
6. Front-end надсилає запит на створення кампанії до back-end: після успішної валідації, front-end відправляє запит на створення нової кампанії до back-end частини системи.
7. Back-end обробляє запит на створення кампанії: back-end частина системи приймає запит від front-end, обробляє його, створює нову кампанію та зберігає її в базі даних.
8. Back-end повертає підтвердження створення кампанії до Front-end: після успішного створення кампанії back-end повертає підтвердження успішної операції до front-end.
9. Front-end відображає повідомлення про успіх або помилку: на основі відповіді від back-end, front-end відображає користувачеві повідомлення, які повідомляють користувача про успішне створення кампанії або показують повідомлення про помилку, якщо операція не вдалася.

Рисунок 2.3 – Діаграма послідовності

Діаграма активностей показана на рисунку 2.4.

1. Користувач розпочинає процес, обираючи опцію "Створити кампанію".
2. Система перевіряє, чи користувач має адміністраторські права. Якщо так, відображається форма створення кампанії. Якщо ні, користувач отримує повідомлення про відмову в доступі.
3. Користувач вводить дані для нової кампанії.
4. Валідація деталей кампанії: система перевіряє введені дані на валідність. Якщо дані валідні, відправляється запит на створення кампанії на back-end. Якщо дані не валідні, користувач отримує повідомлення про помилки валідації.
5. Відображення повідомлення про успіх або помилку: після відповіді від back-end, система відображає повідомлення користувачеві про успішне створення кампанії або показує повідомлення про помилку, якщо операція не вдалася.

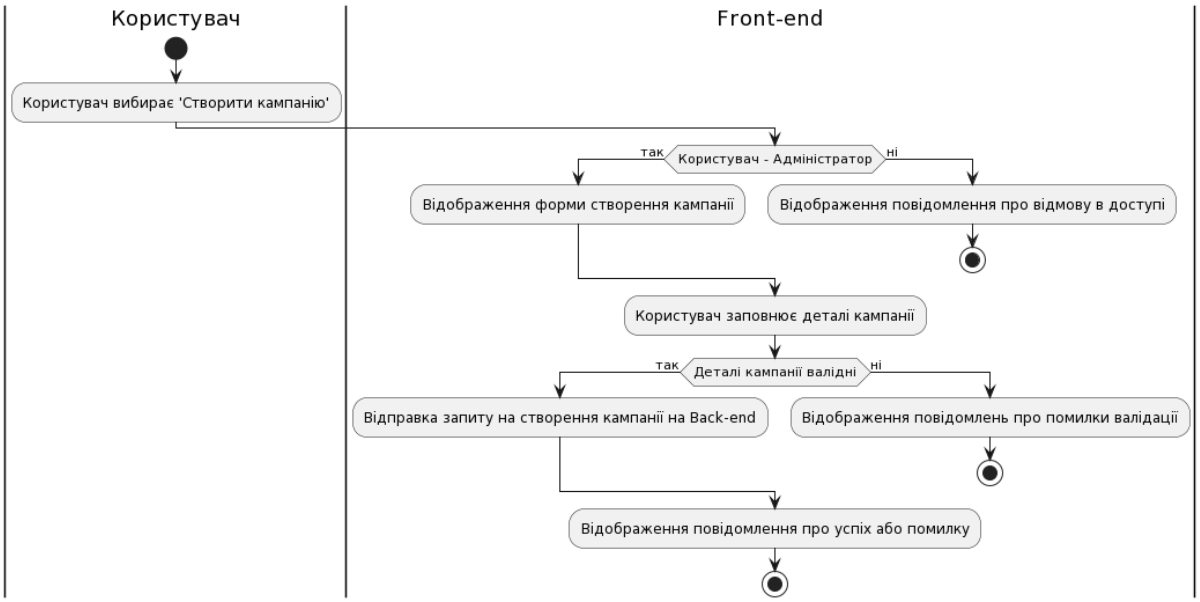

Рисунок 2.4 – Діаграма активностей

Діаграма станів для об'єкта "Рекламна кампанія" показана на рисунку 2.5.

1. Неактивний: це початковий стан, коли система неактивна і чекає вибору користувача для створення кампанії.
2. Активний: після того як користувач обрав створити кампанію, система переходить у стан "Активний". Звідси користувач може ввести дані для кампанії або скасувати створення.
3. Заповнення даних: якщо користувач починає вводити дані, система переходить у стан "Заповнення даних". З цього стану користувач може виправляти введені дані або надсилати їх на валідацію.
4. Валідація: в системі проводиться валідація введених даних. Якщо дані невірні, система повертається у стан "Активний" та відображає помилки валідації. Якщо дані валідні, система переходить у стан "Створення кампанії".
5. Створення кампанії: у цьому стані система створює кампанію на основі введених та валідних даних. Після створення кампанії, система переходить у стан "Неактивний".

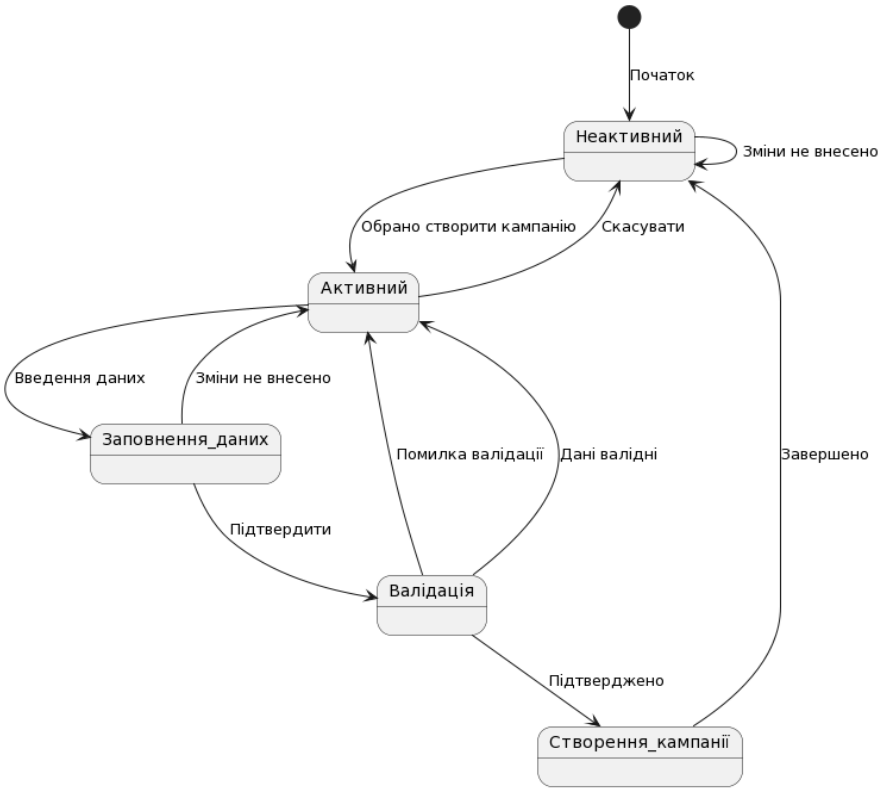

Рисунок 2.5 – Діаграма станів

Ці діаграми UML можна використовувати як частину документації проекту для забезпечення зрозуміння вимог, дизайну та взаємодії різних компонентів системи управління рекламою на веб-сайті, що розробляється.

2.3 Проектування Front-end частини плагіна для Wordpress з функціональними можливостями управління рекламою на WEB-ресурсі

Проектування front-end частини плагіна для WordPress з функціоналом управління рекламою є складним та важливим завданням. Цей процес передбачає створення інтерактивного та зручного інтерфейсу для адміністрування рекламних кампаній. Ключові кроки включають в себе ретельний аналіз функціональних вимог, розуміння потреб користувачів та їхньої цільової аудиторії.

У першу чергу, необхідно зрозуміти функціональність плагіна. Це включає в себе можливості створення, редагування та видалення рекламних кампаній, а також перегляд статистики. Розуміння потреб цільової аудиторії допомагає створити інтерфейс, який буде зрозумілий та простий у використанні.

Для досягнення оптимальної взаємодії з користувачем, важливо розробити макети інтерфейсу, що відображають розташування різних елементів. Це можна здійснити через використання wireframing та прототипування, що допомагають перевірити коректність розміщення та взаємодій елементів.

У технічному плані, розробка включає в себе використання HTML, CSS та JavaScript для створення структури інтерфейсу, його візуального оформлення та динамічних ефектів відповідно. Також важливо забезпечити адаптивність інтерфейсу для різних пристроїв та перевірити його на мобільних платформах.

З погляду взаємодії з back-end, ефективна робота плагіна забезпечується через зручний API, використання AJAX та Fetch API для взаємодії з сервером або базою даних.

Тестування грає важливу роль у вдосконаленні плагіна. Одиницеве

тестування компонентів та користувацьке тестування в реальних умовах допомагають виявити та виправити помилки, а також забезпечити користувачам зручний та безперервний досвід взаємодії з плагіном.

Нарешті, оптимізація коду забезпечує швидке завантаження сторінок та зручність розробки. Після завершення проектування, плагін можна публікувати в WordPress Plugin Repository або інших платформах для спрощення установки та оновлення. Усі ці аспекти допомагають забезпечити успішну front-end частину плагіна, яка не лише естетично виглядає, а й надає користувачам зручний та ефективний інструмент для управління рекламними кампаніями на веб-ресурсі [15].

Етапи проектування front-end частини плагіна для WordPress:

1. Аналіз вимог: необхідно розпочати з докладного вивчення вимог до функціоналу та зовнішнього вигляду плагіна. Це включає в себе функціональні вимоги і очікувані можливості, а також цільову аудиторію.
2. Дизайн інтерфейсу: потрібно створити макети інтерфейсу, зокрема розташування елементів, колірну палітру, типографіку і графіку. В нагоді стане робота над користувацьким досвідом та ергономікою.
3. Розробка HTML, CSS і JavaScript: далі обов’язково потрібно створити структуру HTML для ваших сторінок та елементів плагіна. Зазвичай використовується CSS для стилізації та JavaScript для додавання динамічної поведінки, наприклад, асинхронної взаємодії з сервером чи анімацій.
4. Інтеграція з WordPress: найкраще використовувати WordPress API для забезпечення взаємодії фронтенду з бекендом плагіна. Це може включати в себе роботу з користувацькими типами даних, користувацькими поліми, запитами до бази даних і т. д.
5. Тестування та оптимізація: авжеж потрібно переконатися, що front-end частина плагіна працює коректно у різних браузерах та на різних пристроях. Бажано також оптимізувати завантаження сторінок,

враховуючи продуктивність та швидкість реакції.

6. Документація та підтримка: на майбутнє важливо надати докладну документацію користувачам щодо використання front-end частини плагіна. Це забезпечує підтримку користувачів, відповідаючи на їх питання та вирішуючи проблеми.
7. Реліз та оновлення: останнім кроком є розгортання плагіну на WordPress і відслідкування відгуків користувачів. Здійснення оновлення, враховуючи отримані відгуки, для поліпшення функціоналу та виправлення помилок [16].

2.4 Висновок

Існують певні способи покращення технології проектування Front-end у веб-розробці, зокрема на платформі WordPress. Для оптимізації якості та продуктивності веб-сайтів, використовуються адаптивний та відзивний дизайн, оптимізація швидкості завантаження, використання Content Delivery Network (CDN), усунення можливих проблем у веб-переглядачах, застосування заходів безпеки та оптимізація для пошукових систем (SEO).

Корисним є моделювання Front-end частини плагіна для WordPress з функціональними можливостями управління рекламою на веб-ресурсі за допомогою UML (Unified Modeling Language). Застосування різних видів UML-діаграм (Use Case, Activity, Sequence, Component, State Machine, Class) допомагає зрозуміти та визначити структуру та поведінку системи.

Важливі ключові кроки проектування Front-end частини плагіна для WordPress з функціональністю управління рекламою включають аналіз функціональних вимог, розуміння потреб користувачів, розробку інтерактивного та зручного інтерфейсу, використання HTML, CSS, JavaScript, забезпечення адаптивності інтерфейсу, взаємодію з back-end, тестування та оптимізацію коду.

3 РЕАЛІЗАЦІЯ FRONT-END ЧАСТИНИ ПЛАГІНА ДЛЯ WORDPRESS З ФУНКЦІОНАЛЬНИМИ МОЖЛИВОСТЯМИ УПРАВЛІННЯ РЕКЛАМОЮ НА ВЕБ-РЕСУРСІ

3.1 Обґрунтування вибору інструментів розробки

У процесі створення Front-End частини плагіна для WordPress з функціональними можливостями управління рекламою на веб-ресурсі, важливо було ретельно вибрати інструменти розробки, які відповідали б вимогам завдання і забезпечили ефективну та стабільну роботу продукту.

Одним з ключових кроків у створенні ефективної Front-End частини плагіна був вибір середовища розробки. Для досягнення оптимальної продуктивності та забезпечення зручності у роботі було вирішено використовувати Visual Studio Code (VSCode). На рисунку 3.1 зображено візуальний вигляд середовища VSCode.

Рисунок 3.1 – Візуальний вигляд середовища Visual Studio Code

Visual Studio Code є відкритим інтегрованим середовищем розробки, розробленим компанією Microsoft. Основні переваги цього середовища включають:

- Легкість використання: VSCode має інтуїтивно зрозумілий інтерфейс та широкий набір функцій, що полегшує роботу розробників.
- Розширюваність: Широкий вибір розширень (extensions) дозволяє налаштовувати середовище розробки з урахуванням конкретних потреб проекту.
- Підтримка JavaScript і PHP: Оскільки проект використовує JavaScript та PHP, VSCode надає розширену підтримку для цих мов програмування, включаючи автодоповнення, перевірку помилок та інші корисні функції.
- Інтеграція з системами контролю версій: VSCode легко інтегрується з популярними системами контролю версій, такими як Git, що сприяє зручній роботі в команді [17].

У процесі створення Front-End частини плагіна, Visual Studio Code використовувався для:

- Редагування та відлагодження коду: Зручний редактор коду та інструменти для відлагодження значно полегшили процес розробки.
- Управління пакетами та залежностями: Вбудовані інструменти для управління пакетами спростили встановлення та оновлення необхідних бібліотек.

У реалізації Front-End частини плагіна використовувалися мову програмування JavaScript, мову гіпертекстової розмітки HTML та каскадні таблиці CSS.

JavaScript був обраний як основна мова програмування для Front-End розробки через його широкі можливості та велику підтримку у веб-розробці. З використанням сучасних стандартів ECMAScript та розширень для роботи з DOM, вдалося забезпечити динамічні та високоефективні функції управління рекламою.

JavaScript є важливою мовою програмування для веб-розробки. Розроблена компанією Netscape, вона швидко стала стандартом для створення динамічного веб-змісту. Основна особливість JavaScript - це те, що вона виконується на стороні клієнта, що дозволяє реалізовувати інтерактивність без перезавантаження сторінки [18].

Плюси JavaScript:

1. Універсальність - JavaScript використовується як на стороні клієнта, так і на стороні сервера (Node.js), що дозволяє розробникам використовувати одну мову для обох сторін веб-додатку.
2. Простота вивчення - JavaScript є високорівневою, легкою для вивчення мовою, особливо для тих, хто вже знайомий із HTML та CSS.
3. Динамічність - можливість змінювати та взаємодіяти з елементами сторінки без перезавантаження дозволяє створювати динамічні та інтерактивні веб-додатки.
4. Велике спільнота та Екосистема - JavaScript має широку спільноту розробників і багато бібліотек, фреймворків та інструментів, таких як React, Angular, Vue.js, що сприяє швидкому та ефективному розвитку.
5. Асинхронність - модель подій та асинхронний підхід дозволяють створювати швидкодіючі програми та запобігають блокуванню інтерфейсу користувача [19].

Мінуси JavaScript:

1. Браузерна залежність - виконання на стороні клієнта означає, що код може залежати від різних браузерів і їх імплементацій, що може викликати проблеми з сумісністю.
2. Безпека - використання JavaScript на стороні клієнта створює певні ризики щодо безпеки, такі як вразливості Cross-Site Scripting (XSS).
3. Неявна типізація - хоча це може зручно для деяких розробників, неявна типізація може призвести до неочікуваного поведінки програми.

4. Швидкодія- хоча існують інструменти для оптимізації, JavaScript може бути менш ефективним з точки зору швидкодії порівняно з іншими компільованими мовами [20].

У варіанті використання CMS WordPress усі перелічені недоліки мови програмування JavaScript є або не важливими або нівелюються за рахунок рішень у самій системі керування вмістом.

Переваги над іншими мовами:

1. Широке Використання - JavaScript широко використовується у веб-розробці, що робить його ключовим інструментом для розробників.
2. Асинхронність - у порівнянні з багатьма іншими мовами, JavaScript ідеально підходить для асинхронного програмування, що є ключовим для веб-розробки.
3. Сучасні Фреймворки - існує велика кількість сучасних фреймворків та бібліотек для розробки високопродуктивних веб-додатків.
4. Легко Інтегровано - JavaScript можна легко інтегрувати в багато середовищ розробки та платформ.

Уроки, які вивчено з плюсів та мінусів JavaScript, дозволяють розробникам ефективно використовувати цю мову для досягнення своїх цілей, усунення потенційних проблем та створення високоякісних веб-додатків [19].

JavaScript, основна мова для Front-End розробки, має неявну типізацію, що дозволяє змінювати типи змінних під час виконання коду. Це забезпечує гнучкість, але може призводити до непередбачуваної поведінки та помилок. JavaScript використовується з численними фреймворками, такими як React, Angular, та Vue.js, для створення динамічних та інтерактивних веб-додатків.

TypeScript, як суперсет JavaScript, додає статичну типізацію, дозволяючи визначати типи змінних, функцій та об'єктів на етапі компіляції. Це полегшує виявлення помилок під час розробки та забезпечує більшу безпеку. TypeScript є розширенням JavaScript, тобто весь валідний JavaScript також є валідним TypeScript [21].

CoffeeScript - це ще один варіант, який компілюється в JavaScript, пропонуючи більш простий та зрозумілий синтаксис. Це полегшує розробку, адже CoffeeScript дозволяє писати менше коду та використовує більш "природні" конструкції. Втім, JavaScript залишається стандартом для Front-End розробки [22].

При виборі між цими мовами важливо враховувати конкретні потреби проекту та вибір команди. JavaScript залишається невід'ємною частиною веб-розробки, в той час як TypeScript та CoffeeScript можуть бути використані як альтернативи для поліпшення безпеки, зрозумілості коду чи простоти розробки.

Для розробки сайтів використовується велика кількість мов програмування та фреймворків в залежності від задач сайту та для зручності програміста. Більшість фреймворків для розробки Front-end частини створені на базі мови програмування JavaScript, в той час як для розробки Back-end частини сайту існує багато різних мов програмування та фреймворків на базі різних мов програмування.

Кожна з мова програмування має свої переваги та недоліки, це наведено в таблиці 3.1.

Таблиця 3.1 – Порівняння мов програмування

Мова	Опис	Переваги	Обмеження
HTML	Мова розмітки для визначення структури веб-сторінки	Простота використання, основа для інших мов	Обмежені можливості для програмування
CSS	Мова стилів для оформлення зовнішнього вигляду веб-сторінки	Керування зовнішнім виглядом і стилями	Недостатня функціональність для логіки та взаємодії

Продовження таблиці 3.1

JavaScript	Мова програмування для додавання інтерактивності	Динамічна функціональність, взаємодія	Залежність від браузера, без JavaScript сайт статичний
Python	Універсальна мова програмування з широкими можливостями	Читабельний і легкий синтаксис, велика кількість бібліотек	Можливі проблеми з продуктивністю для складних проектів
PHP	Спеціалізована мова програмування для динамічних сторінок	Хороша підтримка баз даних, обробка форм	Уразливості без належного захисту, дещо складніші у вивченні
Ruby	Проста та елегантна мова програмування	Читабельний і зрозумілий код, Ruby on Rails фреймворк	Нішева мова, обмежена підтримка бібліотек та інструментів

Мови розмітки HTML та стилізації CSS були використані для створення структури та зовнішнього вигляду елементів інтерфейсу плагіна. Використання каскадних таблиць стилів дозволило легко управляти виглядом рекламних блоків та інших елементів [23].

HTML, або Hypertext Markup Language, є мовою розмітки, яка використовується для створення структури та представлення вмісту веб-сторінок. Основною функцією HTML є організація інформації на веб-сайті та надання браузерам інструкцій щодо відображення цієї інформації.

Основний принцип HTML - це використання тегів для визначення різних елементів на сторінці. Теги визначаються у кутових дужках і зазвичай мають пару, яка вказує на початок та кінець елемента. Наприклад, та вказують на початок і кінець абзацу тексту.

HTML-документ зазвичай має таку структуру:

1. <!DOCTYPE html>: цей тег визначає тип документа (HTML5) та надає браузеру інформацію про версію HTML.

2. <html>: основний елемент, який оточує всю структуру HTML-документа.
3. <head>: ця частина документа містить мета-інформацію, таку як заголовок сторінки, підключені стилі, скрипти і т.д.
4. <title>: елемент, який визначає заголовок сторінки, що відображається у вікні браузера або на вкладці.
5. <body>: ця частина документа містить весь вміст сторінки, такий як текст, зображення, посилання та інші елементи.

Основні теги та їх атрибути використовуються для визначення структури документа і відображення різних елементів. Наприклад:

- Текстові елементи: <p> для абзаців, <h1> до <h6> для заголовків різного рівня, <a> для посилань.
- Мультимедійні елементи: <img> для зображень, <audio> та <video> для аудіо і відео.
- Таблиці: <table> для визначення таблиці, <tr> для рядка, <th> для заголовків і <td> для комірок.
- Списки: <ul> та <ol> для ненумерованого та нумерованого списку, а <li> для елементів списку.

HTML є основою веб-розробки, і розуміння його основ є обов'язковим для будь-якого веб-розробника. Використовуючи правильну розмітку, можна створювати веб-сторінки з різними елементами та функціоналом [24].

CSS, або Cascading Style Sheets, є мовою стилів, що використовується для оформлення та стилізації веб-документів, написаних на HTML чи інших мовах розмітки. Ця мова дозволяє розробникам контролювати вигляд елементів на веб-сторінці, надаючи їм властивості, такі як кольори, шрифти, розміри та розташування.

Селектори та Стилізація: основна ідея CSS полягає в тому, щоб вибрати елементи на сторінці за допомогою селекторів та надати їм відповідні стилі. Селектори можуть бути елементами, класами, ідентифікаторами чи псевдокласами [25].

Бокс-модель: CSS використовує бокс-модель для визначення та керування розміром та розташування елементів. Кожен елемент має внутрішній вміст, відступи, рамку та поле для зовнішнього вмісту. Це дозволяє точно контролювати простір навколо та всередині елементів.

Позиціонування та Флексбокс/Грід: CSS надає засоби для позиціонування елементів на сторінці. Зокрема, флексбокс та грід-система надають зручні інструменти для організації та розміщення елементів в контейнері. Це особливо важливо для створення адаптивного дизайну та вирівнювання елементів відповідно до різних розмірів екрану [26].

Робота з Шрифтами та Кольорами: CSS дозволяє визначати різноманітні стилі для тексту, такі як розмір шрифту, колір, жирність чи курсив. Також можна використовувати властивості для задання фонового кольору та зображення, що застосовуються до елементів [25].

Адаптивний дизайн та Зміна стилів при Взаємодії: CSS дозволяє створювати адаптивний дизайн, забезпечуючи можливість адекватного відображення сторінки на різних пристроях та розмірах екранів. Також можна визначати стилі, які застосовуються при певних взаємодій, наприклад, при наведенні миші чи фокусу на елемент [27].

Вигоди використання CSS:

1. Відокремлення Змісту та Стилів: CSS дозволяє відокремити опис структури сторінки від її стилізації, що полегшує розробку та підтримку коду.
2. Многоразове Використання Стилів: застосування класів та ідентифікаторів дозволяє використовувати однакові стилі для різних частин сторінки, що забезпечує їх многоразове використання.
3. Адаптивний Дизайн: CSS дає можливість створювати адаптивний дизайн, що забезпечує оптимальне відображення сторінки на різних пристроях.
4. Виразність та Кроссбраузерність: CSS дозволяє виразно визначати стилі та гнучко керувати виглядом елементів. Більшість браузерів

підтримують стандарти CSS, що робить стилі однаково виглядовими на різних платформах.

CSS є критично важливою технологією для веб-розробників, оскільки вона забезпечує визначення зовнішнього вигляду веб-сторінок. Розуміння основних концепцій та методів стилізації за допомогою CSS є ключовим для успішної роботи в галузі веб-розробки [25].

Обрані інструменти розробки, такі як Visual Studio Code, та мови програмування, такі як JavaScript, HTML і CSS, вдало поєдналися для створення ефективної та функціональної Front-End частини плагіна для WordPress. Ці вибори дозволили забезпечити зручну розробку, високу продуктивність та гнучкість у майбутньому розвитку продукту.

Git є розподіленою системою керування версіями, яка надає засоби для ефективного відстеження змін в коді програм та спільної роботи над проектами. Використовуючи Git, розробники можуть ефективно співпрацювати, вносити зміни, відстежувати розвиток кодової бази та керувати версіями свого програмного продукту. Ось деякі ключові концепції та можливості Git:

Ключові концепції Git:

1. Репозиторій (Repository):
   - Локальний Репозиторій: Копія проекту, розташована на локальному комп'ютері розробника.
   - Віддалений Репозиторій: Централізована версія проекту, розташована на сервері, яка може бути використана для спільної роботи з іншими розробниками.
2. Коміт (Commit): основний спосіб збереження змін у Git. Коміт фіксує стан проекту на конкретний момент часу, зберігаючи зміни у внутрішній базі даних Git.
3. Гілки (Branches): логічні відгалуження від основної гілки роботи (зазвичай називається main чи master), які дозволяють розробникам вільно експериментувати та вносити зміни, не впливаючи на головний код.

4. Об'єднання (Merge): процес об'єднання змін із однієї гілки в іншу. Може бути використаний для інтеграції нового функціоналу чи виправлень у головний код.
5. Вилучення (Pull): отримання змін з віддаленого репозиторію і їх об'єднання з локальним кодом.
6. Вітки (Tags): спосіб відмітити конкретний коміт, щоб вказати на важливі точки в історії проекту, такі як релізи чи стабільні версії.
7. Конфлікти (Conflicts): ситуації, коли Git не може автоматично об'єднати зміни, що призводять до конфліктів. Розробник повинен вирішити конфлікт вручну.

Вигоди використання Git:

1. Історія та Відстеження Змін: Git зберігає повну історію змін, що дозволяє відстежувати розвиток проекту та знаходити причину конкретної зміни.
2. Гнучкість та Гілкована Розробка: гілкована розробка у Git дозволяє розробникам вільно працювати над функціями чи виправленнями без впливу на головний код.
3. Співпраця та Віддалене Спільне Використання: віддалені репозиторії дозволяють розробникам спільно працювати, обмінюватися змінами та вносити свої внески в проект.
4. Інструмент для Розвитку Коду: Git надає інструменти для управління розробкою коду, відстеження проблем, аналізу внесків та автоматизації процесів розробки.

Git є незамінним інструментом для будь-якого розробника, і використання його у комплексних проектах, таких як магістерські роботи, сприяє більш ефективній та організованій роботі над кодовою базою [28].

Використання Git у комплексній магістерській роботі має численні переваги, які роблять його класним інструментом для розробників та дослідників. Ось кілька ключових причин:

1. Версійний контроль: Git забезпечує повний версійний контроль над кодом та документацією. Кожен коміт фіксує стан проекту на конкретний момент часу, дозволяючи вам повертатися до попередніх версій та порівнювати їх.
2. Гілкована розробка: гілки в Git дозволяють вам розділяти роботу над різними функціями чи аспектами проекту. Це особливо корисно у магістерській роботі, де дослідження може вимагати різних напрямків вивчення.
3. Спільна робота: віддалені репозиторії в Git дозволяють кільком розробникам спільно працювати над проектом. Це важливо для колективних досліджень та розробки, які часто характерні для магістерських проектів.
4. Відстеження змін та Авторська Атрибуція: Git дозволяє вам відстежувати, хто та коли вніс зміни в проект. Це важливо для збереження відомостей про внесок кожного учасника у великому проекті.
5. Зручність у Вирішенні Конфліктів: завдяки важливій функції злиття (merge) та розгалуження (branching), Git полегшує вирішення конфліктів та інтеграцію змін між різними гілками розробки.
6. Захист від Втрати Даних: Git забезпечує ефективне збереження даних та можливість відновлення втрачених чи видалених файлів. Це особливо важливо при довгострокових дослідженнях.
7. Інтеграція з Іншими Інструментами: Git інтегрується з багатьма іншими інструментами для керування проектами, автоматизації тестування, неперервної інтеграції та іншими елементами розробки.
8. Зручна Робота з Кодом та Документацією: розробники можуть зручно вносити зміни, слідкувати за розвитком проекту та вести документацію, використовуючи Git для керування вмістом проекту.

Всі ці функції роблять Git важливим інструментом для ефективної та організованої роботи над магістерськими роботами, де командна робота та структурованість важливі для успішного завершення проекту.

Також для розробки WordPress плагіну використовувалась платформа Open Server Panel.

Open Server Panel є однією з популярних платформ для розгортання локального веб-сервера та роботи з веб-проектами. Це зручне інструментарій для розробки та тестування веб-додатків, включаючи плагіни для WordPress. Нижче розглянуті деякі з основних характеристик Open Server Panel, які зробили його популярним серед розробників.

1. Легке Розгортання Локального Сервера: Open Server Panel дозволяє швидко та легко розгортати локальний сервер на комп'ютері, що дозволяє вам працювати з веб-сайтами та додатками без доступу до інтернету.
2. Підтримка Різних Версій PHP та MySQL: платформа підтримує різні версії PHP та MySQL, що дозволяє розробникам легко переключатися між різними версіями та використовувати їх для тестування та сумісності.
3. Керування Серверними Налаштуваннями: Open Server Panel надає графічний інтерфейс для зручного керування налаштуваннями сервера, включаючи конфігурації PHP, Apache, та інші параметри.
4. Локальний Доступ до Баз Даних: зручний доступ до баз даних MySQL через phpMyAdmin дозволяє розробникам легко керувати та взаємодіяти з базами даних на локальному сервері.
5. Підтримка SSL та HTTPS: Open Server Panel надає можливість налаштувати SSL-сертифікати для роботи з HTTPS на локальному сервері, що важливо для тестування функцій, які вимагають безпечного з'єднання.

6. Резервне Копіювання та Відновлення Проектів: інтегрована можливість резервного копіювання та відновлення проектів спрощує управління робочими директоріями та забезпечує безпеку даних.
7. Підтримка Віртуальних Хостів: вбудована можливість налаштовування віртуальних хостів дозволяє одночасно працювати з декількома проектами та веб-сайтами, роблячи розробку більш гнучкою.
8. Керування Версіями PHP-Розширень: вбудований менеджер розширень PHP дозволяє додавати та видаляти розширення за допомогою простого інтерфейсу, забезпечуючи гнучкість у роботі з різними бібліотеками та функціями.
9. Зручний Доступ до Логів та Помилок: Open Server Panel надає зручний доступ до логів та помилок сервера, що полегшує виявлення та виправлення проблем в процесі розробки.
10. Безкоштовна та Відкрита Платформа: Open Server Panel є безкоштовним і відкритим проектом, що дає можливість використовувати його без обмежень та внесення власних змін.

Open Server Panel став необхідним інструментом для багатьох розробників, зокрема для тих, які працюють з плагінами для WordPress. Він дозволяє легко створювати та тестувати веб-додатки локально перед публікацією на сервері [29].

Open Server Panel є дуже зручним для користувачів операційної системи Windows, адже в ньому не потрібно багато різних речей налаштовувати вручну, у платформи дуже зручний інтерфейс, що дозволяє в один клік змінювати версії різних компонентів для роботи з базою даних чи JavaScript та PHP кодом всередині CMS WordPress.

3.2 Основні оператори мови програмування JavaScript

Оператори в JavaScript виконують різноманітні завдання, дозволяючи розробникам працювати з даними, порівнювати значення, виконувати

арифметичні операції та інше. Вони є фундаментальним елементом будь-якої програми і допомагають забезпечити логічність та функціональність коду. Давайте розглянемо основні категорії операторів:

Арифметичні оператори. Ці оператори використовуються для виконання математичних операцій:

- + (додавання): сумує два операнда.
- - (віднімання): віднімає один операнд від іншого.
- * (множення): перемножає два операнда.
- / (ділення): ділить перший операнд на другий.
- % (залишок від ділення): повертає залишок від ділення.

Оператори порівняння. Вони порівнюють два значення та повертають логічне значення:

- == (рівність): перевіряє, чи два значення рівні.
- === (строга рівність): перевіряє рівність значень та їх типів.
- != (нерівність): перевіряє, чи два значення не рівні.
- !== (строга нерівність): перевіряє нерівність значень та їх типів.
- > (більше): перевіряє, чи перше значення більше за друге.
- < (менше): перевіряє, чи перше значення менше за друге.
- >= (більше або рівне): перевіряє, чи перше значення більше або рівне другому.
- <= (менше або рівне): перевіряє, чи перше значення менше або рівне другому.

Логічні оператори. Вони використовуються для об'єднання та порівняння логічних виразів:

- && (логічне І): повертає true, якщо обидва операнди true.
- || (логічне АБО): повертає true, якщо хоча б один операнд true.
- ! (логічне НЕ): повертає true, якщо операнд false, і навпаки.

Оператори присвоювання. Ці оператори використовуються для присвоєння значень змінним:

- = (присвоювання): присвоює значення правого операнда лівому операнду.
- +=, -=, *=,** /=:** скорочені форми присвоєння для арифметичних операцій.

Тернарний оператор - це спеціальний оператор, який використовується для скороченого запису умовного виразу:

- condition ? expr1 : expr2: повертає expr1, якщо умова condition true, інакше повертає expr2.

Оператори в JavaScript дозволяють вам ефективно взаємодіяти з даними та контролювати хід програми, роблячи код більш зрозумілим та функціональним. З розумінням цих операторів розробники можуть створювати потужні та ефективні програми [30].

3.3 Особливості середовища Visual Studio Code для Front-end розробки

Visual Studio Code (VS Code) є одним з найпопулярніших та потужних текстових редакторів для розробки програмного забезпечення, включаючи front-end веб-розробку з використанням JavaScript. Нижче розглянуті основні особливості середовища VS Code, які роблять його привабливим для розробників front-end.

1. Легкий та Швидкий Редактор: VS Code має мінімалістичний інтерфейс та важко зосереджений на продуктивності редактор, що дозволяє розробникам швидко створювати та редагувати код.
2. Розширені Можливості Редакції: редактор підтримує автоматичне завершення коду, виправлення помилок на льоту, підказки, рефакторинг та інші корисні функції, що полегшують процес написання та редагування коду.
3. Вбудована Підтримка Git: VS Code інтегрується з системою контролю версій Git, що дозволяє розробникам ефективно взаємодіяти з кодовою базою, вносити зміни та переглядати історію комітів, всі це зручно із

вбудованою панеллю Git.

4. Розширюваність та Розширені Плагіни: VS Code надає масив плагінів та розширень для розробки на різних мовах програмування, включаючи JavaScript. За допомогою цих плагінів розробники можуть налаштовувати редактор згідно своїм потребам.
5. Вбудований Термінал: редактор має вбудований термінал, що дозволяє виконувати команди та запускати скрипти прямо в середовищі редактора, спрощуючи розробку та тестування.
6. Live Server та Вбудований Розгортання: для front-end розробки VS Code надає плагіни, такі як Live Server, які дозволяють швидко запустити локальний сервер та відстежувати зміни в реальному часі. Також можливості розгортання спрощують публікацію веб-сайтів.
7. Висока Підтримка JavaScript та Інших Мов: VS Code має високий рівень підтримки для JavaScript та інших мов, що використовуються для front-end розробки. Синтаксичне виділення, підказки, рефакторинг і багато іншого допомагають полегшити процес розробки.
8. Вбудовані Інструменти Дебагінгу: VS Code надає вбудовані засоби для дебагінгу JavaScript-коду, включаючи можливість встановлення точок зупинки, крокування по коду та аналізу значень змінних.
9. Активна Спільнота та Підтримка: спільнота VS Code є дуже активною, і через це розробники можуть легко знаходити підтримку, документацію та різноманітні рішення для своїх завдань.
10. Безкоштовний та Відкритий Код: Visual Studio Code є безкоштовним та відкритим проектом, що дозволяє розробникам користуватися його можливостями без витрат.

Visual Studio Code надає потужне та зручне середовище для front-end розробки на JavaScript, і відмінно підходить для розробки веб-ресурсів та додатків [31].

3.4 Розробка функціональності Front-end частини плагіна управління рекламою на веб-ресурсі

Розробка функціональності Front-end частини плагіну управління рекламою включає в себе створення інтерфейсу користувача, який дозволяє легко та ефективно керувати рекламними матеріалами та параметрами кампаній. Основні етапи цього процесу можуть бути наступними:

1. Дизайн та Макетування: розробка функціональності розпочинається з дизайну інтерфейсу, визначення структури та розташування елементів на сторінці. Макети можуть бути створені за допомогою інструментів дизайну, а також макетних мов, таких як HTML та CSS.
2. Розробка Адаптивності: оскільки користувачі використовують різні пристрої (комп'ютери, планшети, смартфони), важливо забезпечити адаптивність інтерфейсу. Використання адаптивного дизайну та медіа-запитів дозволяє забезпечити оптимальний вигляд на різних екранах.
3. Інтерактивність та Динамічні Елементи: JavaScript використовується для створення інтерактивних елементів та динамічних змін на сторінці. Наприклад, можуть бути реалізовані функції додавання та видалення рекламних блоків, налаштування параметрів кампаній, а також перегляд аналітики.
4. Взаємодія з Back-end: фронтенд взаємодіє з бекендом для отримання та відправлення даних. Запити до сервера можуть включати оновлення рекламних матеріалів, налаштування кампаній та отримання статистики.
5. Відображення Реклами: розробка функціональності для відображення рекламних блоків на сторінці. Це може включати в себе вбудування коду рекламних банерів та інших матеріалів з урахуванням встановлених параметрів.
6. Оптимізація Швидкості та Ваги Сторінки: застосування оптимізацій для зменшення часу завантаження сторінки, таких як мінімізація та компресія файлів, використання lazy loading для зображень та використання CDN.

7. Тестування: проведення тестувань для перевірки правильності роботи функціональності на різних браузерах та пристроях. Тестування включає в себе перевірку взаємодії з бекендом та коректність відображення рекламних матеріалів.
8. Удосконалення та Розширення: постійне удосконалення функціональності на основі відгуків користувачів та можливостей розширення плагіна у майбутньому [32].

Цей процес дозволяє створити ефективний та зручний інтерфейс для управління рекламними матеріалами на платформі WordPress, забезпечуючи користувачам зручність та можливість ефективного використання рекламних ресурсів.

Плагін управління рекламою виконує певні функції, частину з яких в адміністративній панелі сайту, а частину на сторінках чи постах сайту. В адміністративній панелі дається можливість створити рекламний банер чи групу рекламних банерів, обрати посилання на рекламу, картинку чи текст для відображення реклами. Також є варіант налаштування певного періоду часу для демонстрації банерів та увімкнення збирання статистичних даних про взаємодію користувача з рекламою. В свою чергу на сторінках чи постах сайту відображаються рекламні банери чи групи банерів у відповідно встановлений в адміністративній панелі час та дати та при натисканнях на рекламу передає дані про клік в розділ статистики в адміністративній панелі.

Почнемо розглядати функціональність Front-end частини плагіну управління рекламою з функції adsplugin_header() на рисунку 3.2.

```php
function adsplugin_header() {
  if(!function_exists('get_plugins')) require_once ABSPATH . 'wp-admin/includes/plugin.php';
  $plugins = get_plugins();
  $plugin_version = $plugins['adsplugin/adsplugin.php']['Version'];

  $output = "\n<!-- This site is using AdsPlugin v".$plugin_version." -->\n";
  echo $output;

  adsplugin_custom_css();
}
```

Рисунок 3.2 – Функція adsplugin_header()

Ця функція призначена для виведення інформації в шапці веб-сторінки, конкретно щодо використання плагіна AdsPlugin. Давайте розглянемо кожний рядок функції:

1. if(!function_exists('get_plugins')) require_once ABSPATH . 'wp-admin/includes/plugin.php';: цей рядок перевіряє, чи існує функція get_plugins у системі WordPress. Якщо вона не існує, то виконується require_once, щоб підключити файл plugin.php, де знаходиться ця функція.
2. $plugins = get_plugins();: функція get_plugins() повертає масив інформації про всі встановлені плагіни. Цей рядок отримує інформацію про всі плагіни і зберігає її у змінній $plugins.
3. $plugin_version = $plugins['adsplugin/adsplugin.php']['Version'];: цей рядок отримує версію конкретного плагіна AdsPlugin, звертаючись до елементу масиву з інформацією про плагін.
4. $output = "\n<!-- This site is using AdsPlugin v".$plugin_version." -->\n";: створюється рядок $output, який містить HTML-коментар з інформацією про використання плагіна AdsPlugin та його версію.
5. echo $output;: виводить рядок $output на веб-сторінці, тобто вставляє цю інформацію в HTML-код шапки сторінки.
6. adsplugin_custom_css();: викликає іншу функцію adsplugin_custom_css(), яка ймовірно відповідає за додавання власного CSS для стилізації елементів, пов'язаних з AdsPlugin.

Загальна мета цієї функції - вивести інформацію про версію плагіна AdsPlugin в шапці веб-сторінки.

Також є функції, які відповідають за відображення рекламних банерів на сайті, адже це основна мета плагіну. Розлянемо функцію adsplugin_shortcode, яка визначає короткий код (shortcode) для виведення рекламних банерів або груп з плагіна AdsPlugin на сторінках WordPress на рисунку 3.3. Ця функція дозволяє вставити шорткод в будь-яке місце на сайті й створити там певний рекламний банер чи групу банерів.

```php
function adsplugin_shortcode($atts, $content = null) {
  global $adsplugin_config;
  $banner_id = (!empty($atts['banner'])) ? trim($atts['banner'], "\r\t ") : 0;
  $group_ids = (!empty($atts['group'])) ? trim($atts['group'], "\r\t ") : 0;
  $output = '';
  if($adsplugin_config['w3caching'] == "Y") {
    $output .= '<!-- mfunc '.W3TC_DYNAMIC_SECURITY.' -->';
    if($banner_id > 0 AND ($group_ids == 0 OR $group_ids > 0)) { // Show one Ad
      $output .= 'echo adsplugin_ad('.$banner_id.');';
    }
    if($banner_id == 0 AND $group_ids > 0) { // Show group
      $output .= 'echo adsplugin_group('.$group_ids.');';
    }
    $output .= '<!-- /mfunc '.W3TC_DYNAMIC_SECURITY.' -->';
  } else if($adsplugin_config['borlabscache'] == "Y" AND function_exists('BorlabsCacheHelper')) {
    if(BorlabsCacheHelper()->willFragmentCachingPerform()) {
      $borlabsphrase = BorlabsCacheHelper()->getFragmentCachingPhrase();
      $output .= '<!--[borlabs cache start: '.$borlabsphrase.']--> ';
      if($banner_id > 0 AND ($group_ids == 0 OR $group_ids > 0)) { // Show one Ad
        $output .= 'echo adsplugin_ad('.$banner_id.');';
      }
      if($banner_id == 0 AND $group_ids > 0) { // Show group
        $output .= 'echo adsplugin_group('.$group_ids.');';
      }
      $output .= ' <!--[borlabs cache end: '.$borlabsphrase.']-->';
      unset($borlabsphrase);
    }
  } else {
    if($banner_id > 0 AND ($group_ids == 0 OR $group_ids > 0)) { // Show one Ad
      $output .= adsplugin_ad($banner_id);
    }
    if($banner_id == 0 AND $group_ids > 0) { // Show group
      $output .= adsplugin_group($group_ids);
    }
  }
  return $output;
}
```

Рисунок 3.3 – Функція відображення рекламних банерів

Шорткод (shortcode) - це короткий код в системі управління контентом (CMS), який використовується для вставки певного функціоналу або вмісту в веб-сторінку чи запис. Шорткоди дозволяють легко вбудовувати спеціальний функціонал, який виконується при відображенні сторінки або запису, без необхідності писати весь код вручну [33].

У багатьох системах управління контентом, таких як WordPress, шорткоди виглядають приблизно так: [shortcode_name]. Вміст, який розташований між

відкриваючим і закриваючим тегами шорткоду, буде оброблений відповідним функціоналом.

Розглянемо кожний рядок функції:

1. global $adsplugin_config;: Робить глобальною змінну $adsplugin_config, яка містить конфігураційні параметри для плагіна.
2. Отримання атрибутів короткого коду:
    - $banner_id = (!empty($atts['banner'])) ? trim($atts['banner'], "\r\t ") : 0;: Визначає $banner_id на основі атрибута banner короткого коду. Якщо атрибут не вказаний, встановлюється значення за замовчуванням - 0.
    - $group_ids = (!empty($atts['group'])) ? trim($atts['group'], "\r\t ") : 0;: Визначає $group_ids на основі атрибута group короткого коду. Якщо атрибут не вказаний, встановлюється значення за замовчуванням - 0.
    - Решта змінних, таких як $fallback, $weight, $site, $wrapper, мають значення за замовчуванням 0, і вони не підтримуються в безкоштовній версії плагіна.
3. Створення рядка $output: Цей рядок ініціалізує змінну $output як порожній рядок, який буде містити вихідні дані для виведення.
4. Обробка кешування:
    - Якщо включено кешування W3 Total Cache ($adsplugin_config['w3caching'] == "Y"), то вставляється коментар для W3TC, а потім викликається функція для виведення рекламного банера чи групи в цьому кешованому контексті.
    - Якщо включено кешування Borlabs Cache ($adsplugin_config['borlabscache'] == "Y" AND function_exists('BorlabsCacheHelper')), то вставляється токенизований фрагмент Borlabs Cache, а потім викликається функція для виведення рекламного банера чи групи в цьому контексті.

- У протилежному випадку (коли немає кешування), викликається функція для виведення рекламного банера чи групи прямо в змінну $output.

5. Повернення $output: Результат виведення рекламного банера чи групи, який знаходиться у змінній $output, повертається для використання в шаблоні WordPress.

Ця функція дозволяє використовувати шорткоди для вставки рекламних банерів або груп на сторінки сайту з врахуванням налаштувань конфігурації плагіна та ефективної роботи з кешуванням, як це показано на рисунку 3.4.

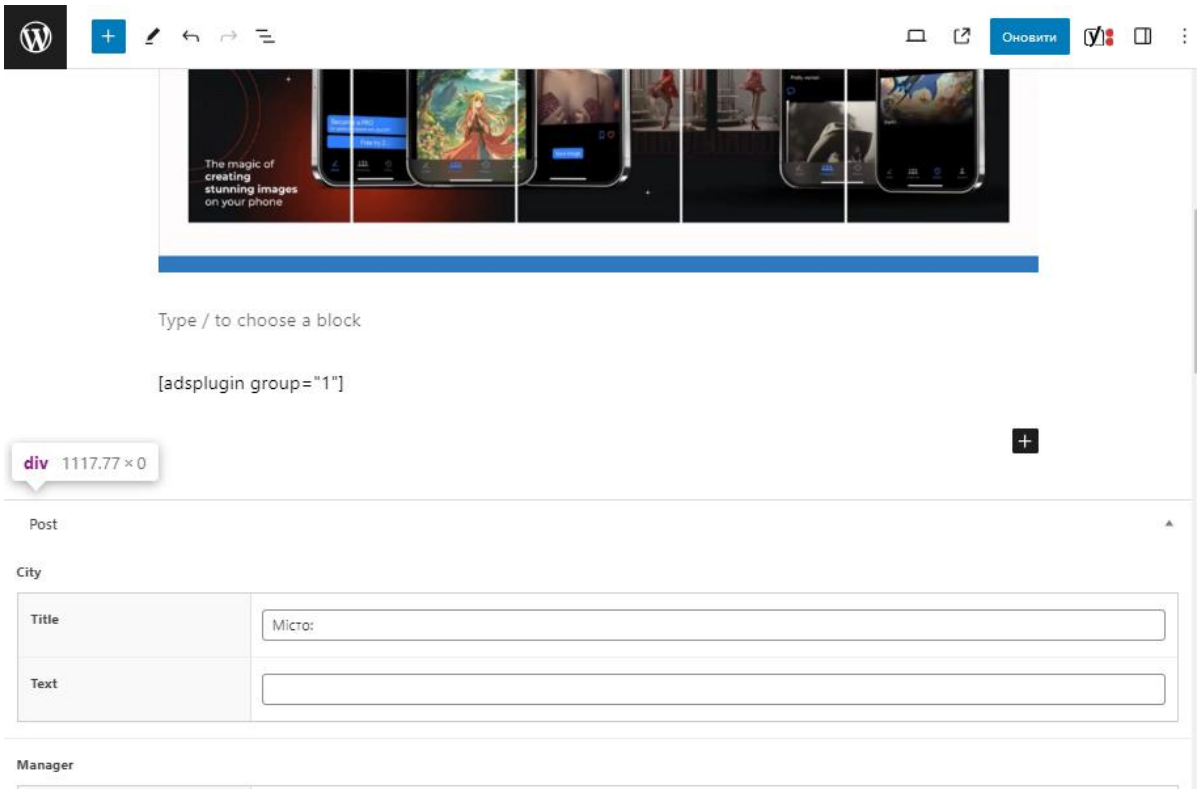

Рисунок 3.4 – Приклад застосування шорткоду для поста в адміністративній панелі сайту

Крім функції adsplugin_shortcode для виводу рекламних банерів на сайт використовується функція adsplugin_inject_posts, яка дозволяє вставляти рекламу в певні пости на сайті. Ця функція призначена для вставки рекламних блоків в контент сторінок та записів на WordPress, з урахуванням вказаних у налаштуваннях груп та категорій. Розглянемо цю функцію на рисунку 3.5.

```php
function adsplugin_inject_posts($post_content) {
  global $wpdb, $post;

  $categories_top = $categories_bottom = $categories_inside = array();
  if(is_page()) {
    $groups = $wpdb->get_results("SELECT `id`, `page`, `page_loc`, `page_par` FROM
    `{$wpdb->prefix}adsplugin_groups` WHERE `page_loc` > 0 AND  `page_loc` < 5;");
    foreach($groups as $group) {
      $pages_more = explode(",", $group->page);
      if(count($pages_more) > 0) {
        if(in_array($post->ID, $pages_more)) {
          if($group->page_loc == 1 OR $group->page_loc == 3) {
            $categories_top[$group->id] = $group->page_par;
          }
          if($group->page_loc == 2 OR $group->page_loc == 3) {
            $categories_bottom[$group->id] = $group->page_par;
          }
          if($group->page_loc == 4) {
            $categories_inside[$group->id] = $group->page_par;
          }
          unset($pages_more, $group);
        }
      }
    }
  }
  if(is_single()) {
    $groups = $wpdb->get_results("SELECT `id`, `cat`, `cat_loc`, `cat_par` FROM
    `{$wpdb->prefix}adsplugin_groups` WHERE `cat_loc` > 0 AND `cat_loc` < 5;");
    $wp_categories = wp_get_post_categories($post->ID, array('taxonomy' =>
    'category', 'fields' => 'ids'));
    foreach($groups as $group) {
      $categories_more = array_intersect($wp_categories, explode(",", $group->cat));
      if(count($categories_more) > 0) {
        if(has_category($categories_more, $post->ID)) {
          if(($group->cat_loc == 1 OR $group->cat_loc == 3)) {
            $categories_top[$group->id] = $group->cat_par;
          }
          if($group->cat_loc == 2 OR $group->cat_loc == 3) {
            $categories_bottom[$group->id] = $group->cat_par;
          }
          if($group->cat_loc == 4) {
            $categories_inside[$group->id] = $group->cat_par;
          }
          unset($categories_more, $group);
        }
      }
    }
  }
  if(count($categories_top) > 0) {
    $post_content = adsplugin_inject_posts_cache_wrapper(array_rand($categories_top)).$post_content;
  }
    foreach($categories_inside as $group_id => $group_paragraph) {
      if($group_paragraph == 99) {
        $group_paragraph = $post_content_count / 2; // Middle of content
      }
      $group_paragraph = intval($group_paragraph);
      if(!array_key_exists($group_paragraph, $inserted)) {
        $inserted[$group_paragraph] = $group_id;
      }
      unset($group_id, $group_paragraph);
    }
    foreach($post_content_exploded as $index => $paragraph) {
      $insert_here = $index + 1; // Deal with real offset
      if(array_key_exists($insert_here, $inserted)) {
        if($inserted[$insert_here] > 0) {
          $post_content_exploded[$index] .= adsplugin_inject_posts_cache_wrapper($inserted[$insert_here]);
          $inserted[$insert_here] = 0;
        }
      }
      unset($index, $paragraph, $insert_here);
    }
      $post_content = implode('', $post_content_exploded);
    unset($post_content_exploded, $post_content_count, $inserted);
  }
  unset($groups, $categories_top, $categories_bottom, $categories_inside);
  return $post_content;
}
```

Рисунок 3.5 – Функція відображення рекламних банерів в постах

Розглянемо її детально:

1. Перевірка, чи є сторінка (if(is_page())):
    - Якщо так, то витягуємо групи рекламних блоків для сторінок з бази даних ({$wpdb->prefix}adsplugin_groups).
    - Далі перевіряємо, чи поточна сторінка має відповідати одній зі сторінок, вказаних у налаштуваннях групи.
    - Якщо так, то визначаємо місце розміщення реклами (зверху, знизу, всередині) і зберігаємо інформацію у відповідний масив ($categories_top, $categories_bottom, $categories_inside).
2. Перевірка, чи є одиночна сторінка (if(is_single())):
    - Якщо так, то витягуємо групи рекламних блоків для категорій з бази даних.
    - Перевіряємо, чи поточна категорія запису відповідає категоріям, вказаним у налаштуваннях групи.
    - Якщо так, то визначаємо місце розміщення реклами (зверху, знизу, всередині) і зберігаємо інформацію у відповідний масив.
3. Вставка рекламних блоків в контент:
    - Якщо є групи для розміщення зверху, то вставляємо один блок перед контентом.
    - Якщо є групи для розміщення знизу, то вставляємо один блок після контенту.
    - Якщо є групи для розміщення всередині, то вставляємо блоки після параграфів контенту, на основі заданих у налаштуваннях параграфів (або середини контенту, якщо 99).
4. Очищення та повернення обробленого контенту:
    - Після вставки всіх рекламних блоків забираємо всі тимчасові дані, щоб не забруднювати оточення.
    - Повертаємо оброблений контент.

Ця функція є частиною розширення для рекламного плагіна на WordPress і дозволяє динамічно вставляти рекламу на сторінках і записах відповідно до вказаних у налаштуваннях груп та категорій.

На рисунку 3.6 показано як відбувається вбудування груп рекламних банерів у пости певних категорій чи інші сторінки сайту у адміністративній панелі сайту.

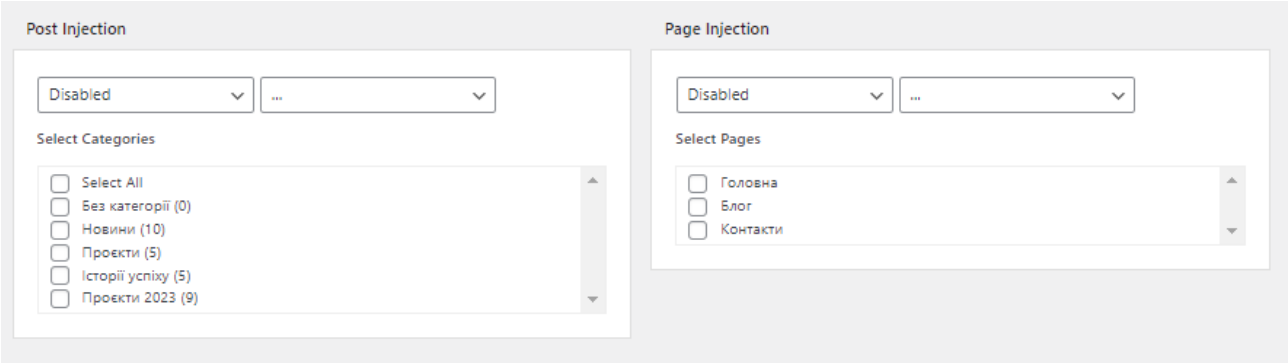

Рисунок 3.6 – Вбудування рекламних банерів в постах в адміністративній панелі

Також важливо відстежувати взаємодію користувача з рекламою на веб-ресурсі, записувати дані про це в адмністративній панелі та вираховувати статистичні дані. Розглянемо функцію adsplugin_click_callback(), що відповідає за обробку запитів на відстеження кліків на банерах і оновлення статистики кліків та бюджету в базі даних на рисунку 3.7.

Давайте розглянемо кожний рядок функції:

1. Визначення констант для запобігання кешуванню:
    - if(!defined('DONOTCACHEPAGE')) define('DONOTCACHEPAGE', true);: встановлює константу DONOTCACHEPAGE, щоб вказати системі кешування, що сторінка не повинна кешуватись.
    - if(!defined('DONOTCACHEDB')) define('DONOTCACHEDB', true);: встановлює константу DONOTCACHEDB, щоб вказати системі кешування, що не повинна використовуватись база даних для кешування.

- if(!defined('DONOTCACHEOBJECT')) define('DONOTCACHEOBJECT', true);: встановлює константу DONOTCACHEOBJECT, щоб вказати системі кешування, що не повинні кешуватись об'єкти.

2. Глобальні змінні:
   - global $wpdb, $adsplugin_config;: забезпечує доступ до глобальних об'єктів бази даних та конфігурації плагіна.
3. Отримання та розшифрування даних відстеження:
   - $meta = $_POST['track'];: отримується рядок, що представляє дані відстеження, передані методом POST.
   - $meta = base64_decode($meta);: розшифровується рядок, закодований у форматі base64.
4. Забезпечення безпеки даних:
   - $meta = esc_attr($meta);: використовується функція esc_attr, щоб екранувати дані та забезпечити безпеку.
5. Розбір розшифрованих даних:
   - list($ad, $group, $blog_id, $impression_timer) = explode(",", $meta, 4);: розбивається рядок на окремі змінні, використовуючи кому як роздільник. Ці дані вказують на конкретний банер, групу, ідентифікатор блогу та імпресійний таймер (не використовується у цьому контексті).
6. Перевірка числових значень та оновлення статистики кліків:
   - if(is_numeric($ad) AND is_numeric($group) AND is_numeric($blog_id)) { ... }: перевіряється, чи всі значення є числовими. Якщо так, то виконується логіка оновлення статистики кліків у базі даних.
7. Перехід до іншого блогу у мережі (якщо застосовано):
   - Якщо ідентифікатор блогу більше 0 і функція adsplugin_is_networked() повертає true, то виконується перехід до іншого блогу за допомогою switch_to_blog($blog_id);.

8. Отримання IP-адреси користувача та перевірка, чи це людина:
    - $remote_ip = adsplugin_get_remote_ip();: отримання IP-адреси користувача за допомогою функції adsplugin_get_remote_ip().
    - if(adsplugin_is_human() AND $remote_ip != "unknown" AND !empty($remote_ip)) { ... }: перевірка, чи користувач є людиною (не ботом) та чи отримано коректний IP-адрес.
9. Перевірка обмеження часу між кліками та оновлення статистики:
    - Отримання поточного часу, визначення часу початку сьогоднішнього дня.
    - $saved_timer = $wpdb->get_var($wpdb->prepare("SELECT timerFROM{$wpdb->prefix}adsplugin_trackerWHEREipaddress= '%s' ANDstat= 'c' ANDbannerid= %d ORDER BYtimer` DESC LIMIT 1;", $remote_ip, $ad.

```
function adsplugin_click_callback() {
  if(!defined('DONOTCACHEPAGE')) define('DONOTCACHEPAGE', true);
  if(!defined('DONOTCACHEDB')) define('DONOTCACHEDB', true);
  if(!defined('DONOTCACHEOBJECT')) define('DONOTCACHEOBJECT', true);
  global $wpdb, $adsplugin_config;
  $meta = $_POST['track'];
  $meta = base64_decode($meta);
  $meta = esc_attr($meta);
  list($ad, $group, $blog_id, $impression_timer) = explode(",", $meta, 4);
  if(is_numeric($ad) AND is_numeric($group) AND is_numeric($blog_id)) {
    if($blog_id > 0 AND adsplugin_is_networked()) {
      $current_blog = $wpdb->blogid;
      switch_to_blog($blog_id);
    }
    $remote_ip = adsplugin_get_remote_ip();
    if(adsplugin_is_human() AND $remote_ip != "unknown" AND !empty($remote_ip)) {
      $now = current_time('timestamp');
      $today = adsplugin_date_start('day');
      $click_timer = $now - $adsplugin_config['click_timer'];
      $saved_timer = $wpdb->get_var($wpdb->prepare("SELECT `timer` FROM `{$wpdb->prefix}adsplugin_tracker`
      WHERE `ipaddress` = '%s' AND `stat` = 'c' AND `bannerid` = %d ORDER BY `timer` DESC LIMIT 1;", $remote
      if($saved_timer < $click_timer) {
        $stats = $wpdb->get_var($wpdb->prepare("SELECT `id` FROM `{$wpdb->prefix}adsplugin_stats`
        WHERE `ad` = %d AND `group` = %d AND `thetime` = {$today};", $ad, $group));
        if($stats > 0) {
          $wpdb->query("UPDATE `{$wpdb->prefix}adsplugin_stats` SET `clicks` = `clicks` + 1
          WHERE `id` = {$stats};");
        } else {
          $wpdb->insert($wpdb->prefix.'adsplugin_stats', array('ad' => $ad, 'group' => $group,
          'thetime' => $today, 'clicks' => 1, 'impressions' => 1));
        }
        $wpdb->insert($wpdb->prefix.'adsplugin_tracker', array('ipaddress' => $remote_ip, 'timer' =>
        $now, 'bannerid' => $ad, 'stat' => 'c'));
      }
      $wpdb->query("UPDATE `{$wpdb->prefix}adsplugin` SET `budget` = `budget` - `crate` WHERE `id` = {$ad}
      AND `crate` > 0;");
    }
    if($blog_id > 0 AND adsplugin_is_networked()) {
      switch_to_blog($current_blog);
    }
    unset($remote_ip, $track, $meta, $ad, $group, $remote, $banner);
  }
  wp_die();
}
?>
```

Рисунок 3.7 – Функція обробки кліків

Авжеж лише за допомогою PHP коду неможливо відслідувати кліки на Front-end частині веб-ресурсу, для цього потрібно використовувати JavaScript та за допомогою Ajax запитів передавати зібрану та оброблену інформацію на Back-end.

Розглянемо JavaScript-код, призначений для відстеження кліків на посиланнях і відправлення відповідного асинхронного запиту на сервер для обробки цього кліку на рисунку 3.8. Цей механізм використовується для підрахунку кліків на рекламних банерах.

```
(function($) {
  $(document).ready(function() {
    $(document).on('click', 'a.gofollow', function(){
      $.post(click_object.ajax_url, {'action':'adsplugin_click','track':$(this).attr("data-track")});
    });
  });
}(jQuery));
```

Рисунок 3.8 – Функція відстеження кліків і відправлення асинхронного запиту на Back-end

Враховуючи що плагін має певну кількість JavaScript файлів, потрібно якимось чином їх приєднати на сторінки, де відображаються рекламні банери. За це відповідає функція adsplugin_scripts(), розглянемо її на рисунку 3.9.

```
function adsplugin_scripts() {
  global $adsplugin_config;
  $in_footer = false;
  if($adsplugin_config['jsfooter'] == "Y") {
    $in_footer = true;
  }
  if($adsplugin_config['jquery'] == 'Y') wp_enqueue_script('jquery', false, false, false, $in_footer);
  if(get_option('adsplugin_dynamic_required') > 0) wp_enqueue_script('jshowoff-adsplugin',
  plugins_url('/library/jquery.adsplugin.dyngroup.js', __FILE__), false, null, $in_footer);
  if($adsplugin_config['stats'] == 1) {
     wp_enqueue_script('clicktrack-adsplugin', plugins_url('/library/jquery.adsplugin.clicktracker.js',
     __FILE__), false, null, $in_footer);
    wp_localize_script('jshowoff-adsplugin', 'impression_object', array('ajax_url' =>
    admin_url( 'admin-ajax.php')));
    wp_localize_script('clicktrack-adsplugin', 'click_object', array('ajax_url' =>
    admin_url('admin-ajax.php')));
  }
  if(!$in_footer) {
    add_action('wp_head', 'adsplugin_custom_javascript');
  } else {
    add_action('wp_footer', 'adsplugin_custom_javascript', 100);
  }
}
```

Рисунок 3.9 – Функція приєднання скриптів

Розглянемо кожний рядок функції:

1. global $adsplugin_config;: цей рядок робить глобальною змінну $adsplugin_config, яка містить конфігураційні параметри для плагіна.
2. $in_footer = false;: створюється змінна $in_footer із значенням за замовчуванням false. Ця змінна вказує, чи скрипти мають бути включені внизу сторінки (true) чи у верхній частині (false).
3. Умова if($adsplugin_config['jsfooter'] == "Y") { $in_footer = true; }: перевіряє конфігураційний параметр jsfooter. Якщо він встановлений на "Y", то значення $in_footer встановлюється в true, і скрипти будуть включені внизу сторінки.
4. if($adsplugin_config['jquery'] == 'Y') wp_enqueue_script('jquery', false, false, false, $in_footer);: перевіряє конфігураційний параметр jquery. Якщо він встановлений на "Y", то використовується функція wp_enqueue_script для додавання скрипту jQuery.
5. if(get_option('adsplugin_dynamic_required') > 0) wp_enqueue_script('jshowoff-adsplugin', plugins_url('/library/jquery.adsplugin.dyngroup.js', __FILE__), false, null, $in_footer);: перевіряє значення опції adsplugin_dynamic_required. Якщо воно більше 0, то використовується функція wp_enqueue_script для додавання скрипту jshowoff-adsplugin з файлу jquery.adsplugin.dyngroup.js.
6. if($adsplugin_config['stats'] == 1) { ... }: перевіряє, чи включена статистика. Якщо так, то використовується wp_enqueue_script для додавання скриптів clicktrack-adsplugin та jshowoff-adsplugin.clicktracker.js. Також використовується wp_localize_script, щоб передати Ajax URL для обробки click та impression.
7. if(!$in_footer) { add_action('wp_head', 'adsplugin_custom_javascript'); } else { add_action('wp_footer', 'adsplugin_custom_javascript', 100); }: додає дії в WordPress, які викличуть функцію adsplugin_custom_javascript. Якщо $in_footer дорівнює false, то функція додає дію в wp_head. Якщо $in_footer дорівнює true, то дія додається в wp_footer з пріоритетом 100.

Загальна мета цієї функції - налаштувати та додати необхідні JavaScript-скрипти на сторінку WordPress відповідно до конфігурації плагіна.

3.5 Розробка методики тестування

Розробка методики тестування Front-end частини WordPress плагіну є критичним етапом у процесі забезпечення якості та надійності роботи веб-ресурсу. Ефективне тестування гарантує, що користувачі отримують зручний та безперебійний досвід взаємодії з рекламними функціями.

Метою тестування Front-end частини плагіну є перевірка його коректності, відповідності функціональним вимогам, адаптивності на різних пристроях та в різних браузерах, а також оптимальності відображення рекламних матеріалів.

Етапи тестування:

1. Функціональне тестування: перевірка основних функцій плагіну, таких як додавання та видалення рекламних блоків, управління параметрами кампаній, та взаємодія з іншими елементами інтерфейсу.
2. Адаптивне тестування: перевірка адаптивності та відзивчивості інтерфейсу на різних пристроях, таких як комп'ютери, планшети та смартфони. Використання інструментів розробника браузера для емуляції різних розмірів екрану.
3. Браузерне тестування: перевірка роботи плагіну в різних веб-переглядачах, таких як Chrome, Firefox, Safari, Edge та Internet Explorer. Виявлення та усунення можливих несумісностей.
4. Тестування швидкості та завантаження: вимірювання часу завантаження рекламних блоків та оптимізованих зображень. Використання інструментів для аналізу швидкості сторінки, таких як Google PageSpeed Insights.
5. Тестування безпеки: перевірка заходів безпеки, таких як валідація введених даних, захист від XSS та CSRF атак, аутентифікація та авторизація користувачів.

6. Тестування відображення реклами: відображення різних форматів рекламних матеріалів, включаючи банери, відео та інші. Впевнення, що реклама правильно адаптується до розмірів та типів контенту.
7. Тестування взаємодії із Back-end: взаємодія фронтенду з бекендом для передачі та отримання даних. Тестування коректності виконання запитів та обробки відповідей.
8. Тестування в специфічних сценаріях: відтворення специфічних сценаріїв використання, таких як великі обсяги даних, періоди інтенсивного трафіку, та інші сценарії, які можуть впливати на роботу плагіну [34].

Важливо перевірити правильне відображення реклами на веб-ресурсі, оскільки це гарантує коректний вигляд та передачу інформації. Вірне відображення брендованих зображень підтримує ідентичність бренду та створює професійний вигляд.

Також, перевірка статистики щодо кількості кліків на рекламу має важливе значення для оцінки ефективності рекламних кампаній та оптимізації їхнього результату. Аналіз цих даних допомагає рекламодавцям та власникам веб-ресурсу орієнтуватися на найефективніші стратегії та забезпечує максимальну прибутковість.

Використання інструментів тестування, таких як Cypress, є важливим етапом у забезпеченні якості фронт-енд частини рекламного плагіну. Cypress дозволяє створювати динамічні тести, які симулюють дії реального користувача, та перевіряти різні аспекти відображення реклами та її функціоналу. Його можливості тестування не обмежуються лише інтерфейсом, вони також охоплюють взаємодію фронт-енду та бек-енду, забезпечуючи повноту тестування та високу ефективність.

Cypress - це інструмент для автоматизованого тестування, спеціально розроблений для тестування веб-додатків. Основні особливості Cypress, які роблять його популярним для тестування фронт-енду, включають:

- Простота використання: Cypress має простий синтаксис та інтуїтивний

інтерфейс, що полегшує створення й виконання тестів. Також, відсутність складних налаштувань робить його доступним навіть для початківців.

- Реальний час: Cypress працює у реальному часі, надаючи інтерактивний досвід при написанні та відлагодженні тестів. Ви можете спостерігати за виконанням кожного кроку тесту в браузері, що спрощує виявлення помилок та відладку.
- Автоматичне очікування: Cypress автоматично очікує на елементи сторінки, що дозволяє уникнути проблем з завантаженням елементів та забезпечити стабільні тести.
- Спеціалізований на веб-додатках: Cypress призначений саме для веб-додатків і відомий своєю спрямованістю на фронт-енд. Його можливості орієнтовані на роботу з DOM, AJAX, інтерфейсом користувача, що робить його ідеальним для тестування фронт-енду.
- Можливості відлагодження: Cypress має вбудований інструмент відлагодження, що дозволяє вам легко знаходити та виправляти помилки в тестах [35].

Порівняно з аналогами, такими як Selenium чи WebDriver, Cypress визначається своєю простотою, швидкістю та спрямованістю на веб-додатки. Cypress може бути використаний як для одиночних тестів, так і для створення повноцінних тестових наборів. Порівняння Cypress з іншими інструментами тестування показано у таблиці 3.2.

Таблиця 3.2 – Порівняння інструментів тестування

Особливість	Cypress	Selenium WebDriver	Puppeteer
Мова програмування	JavaScript	Підтримка різних мов (Java, C#, Python, і ін.)	JavaScript

Продовження таблиці 3.2

Синтаксис	Легко зрозуміти та вивчати	Залежить від мови програмування	Легкий та зрозумілий
Взаємодія з DOM	Орієнтований на роботу з DOM	Взаємодія на рівні команд до браузера	Взаємодія на рівні команд до браузера
Взаємодія з AJAX	Підтримка AJAX вбудована, не потрібно чекати	Спеціальні команди для чекання AJAX запитів	Чекання вбудовано в команди Puppeteer
Продуктивність	Швидкий та легкий у використанні	Залежить від мови програмування та реалізації	Швидкий та ефективний
Відлагодження	Вбудований інструмент відлагодження	Використання інтегрованих середовищ відлагодження	Вбудований інструмент відлагодження
Спрямованість	Фронт-енд, спеціалізований на веб-додатках	Всебічний, використовується для різних типів тестів	Фронт-енд та back-енд на основі Node.js
Зручність використання	Легкий та дружелюбний інтерфейс	Потребує додаткових компонентів для повноцінної роботи	Легкий та простий у використанні

Основна перевага Cypress полягає в тому, що він дозволяє тестувати ваш фронт-енд на ранніх етапах розробки, прискорюючи процес тестування та полегшуючи виявлення помилок. Якщо ваш проект зосереджений на фронт-енді, Cypress може стати відмінним вибором для вашого інструментарію тестування.

Для тестування фронтенд частини WordPress плагіну управління рекламою можна використовувати різні види тестів, залежно від потреб і вимог. Одиничні тести дозволяють перевірити окремі функції чи методи, такі як обробка реклами або валідація введених даних. Інтеграційні тести допомагають визначити правильність взаємодії між фронтендом та бекендом плагіну. Компонентні тести призначені для перевірки компонентів та їх взаємодії, таких як розділи управління рекламою. Тести користувацького інтерфейсу важливі для перевірки коректності та працездатності інтерфейсу користувача, що відображає рекламу [36].

Щодо Unit тестів, вони є особливо корисними для фронтенд частини WordPress плагіну. Ці тести дозволяють перевірити окремі функції чи методи, такі як обробка реклами, валідація введених даних та інші елементарні операції. Вони особливо ефективні для виявлення та виправлення помилок на ранніх етапах розробки, забезпечуючи швидке виявлення та виправлення проблем. Unit тести також полегшують рефакторинг та покращення кодової бази, оскільки вони забезпечують впевненість у стабільності окремих компонентів без необхідності тестування всього додатку.

Загалом, Unit тести є ефективним інструментом для забезпечення якості фронтенд частини WordPress плагіну, допомагаючи забезпечити стабільність та функціональність окремих частин коду.

Прикладом Unit тесту для Front-end частини WordPress плагіну управління рекламою є автоматизація кліку по рекламному банеру на сторінці, перехід в адміністративну панеліь та перевірка збільшення кількості кліків у розділі статистики. Код такого Cypress тесту зображений на рисунку 3.10.

```
describe('Тест реклами', function () {
  it('Перевірка реклами та клікання', function () {
    cy.visit('http://startup.loc/imagenator/');
    cy.get('.a-single a').should('exist');
    cy.get('.a-single a').click();
    cy.url().should('eq', 'http://startup.loc/ads/');
  });
});
describe('Тест статистики', function () {
  it('Перевірка збільшення кількості кліків', function () {
    cy.visit('http://startup.loc/wp-admin/admin.php?page=adsplugin-statistics');
    cy.get('#user_login').type('admin');
    cy.get('#user_pass').type('**************');
    cy.get('#wp-submit').click();
    cy.get('tbody tr:first-of-type td:nth-of-type(6) .stats_large .number_large').then(($element) => {
      const currentClicks = parseInt($element.text());
      cy.go(-1);
      cy.get('.number_large').should(($elementAfterClick) => {
        const clicksAfterClick = parseInt($elementAfterClick.text());
        expect(clicksAfterClick).to.be.greaterThan(currentClicks);
      });
    });
  });
});
```

Рисунок 3.10 – Cypress тест

На рисунку 3.11 показано результат цього тесту у браузері chrome.

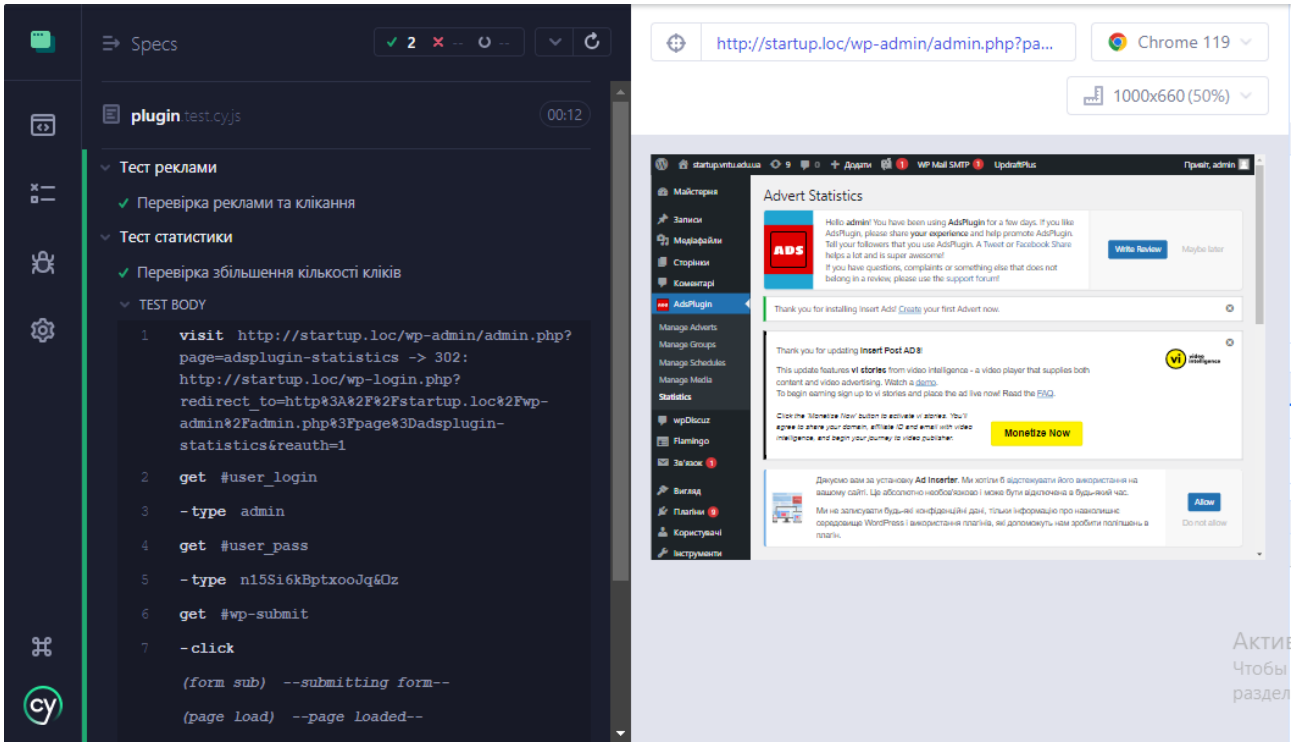

Рисунок 3.11 – Результат Cypress тесту

Розробка методики тестування Front-end частини WordPress плагіну управління рекламою є важливим етапом, спрямованим на забезпечення високої якості та ефективності веб-ресурсу. Тестування допомагає виявити та усунути можливі проблеми, забезпечуючи позитивний досвід користувачів та оптимальну роботу рекламної системи.

3.6 Тестування функціональності Front-end частини плагіна управління рекламою на веб-ресурсі

Основним функціоналом плагіну є виведення на сайт рекламного банеру чи групи банерів. Кожен банер складається з посилання та тексту чи картинки. Крім того важливо перевірити чи передаються дані про взаємодію користувачів з банером і відображаються в Back-end частині.

Почнемо з тестування відображення реклами тв створимо перший банер в адміністративній панелі як це показано на рисунку 3.12.

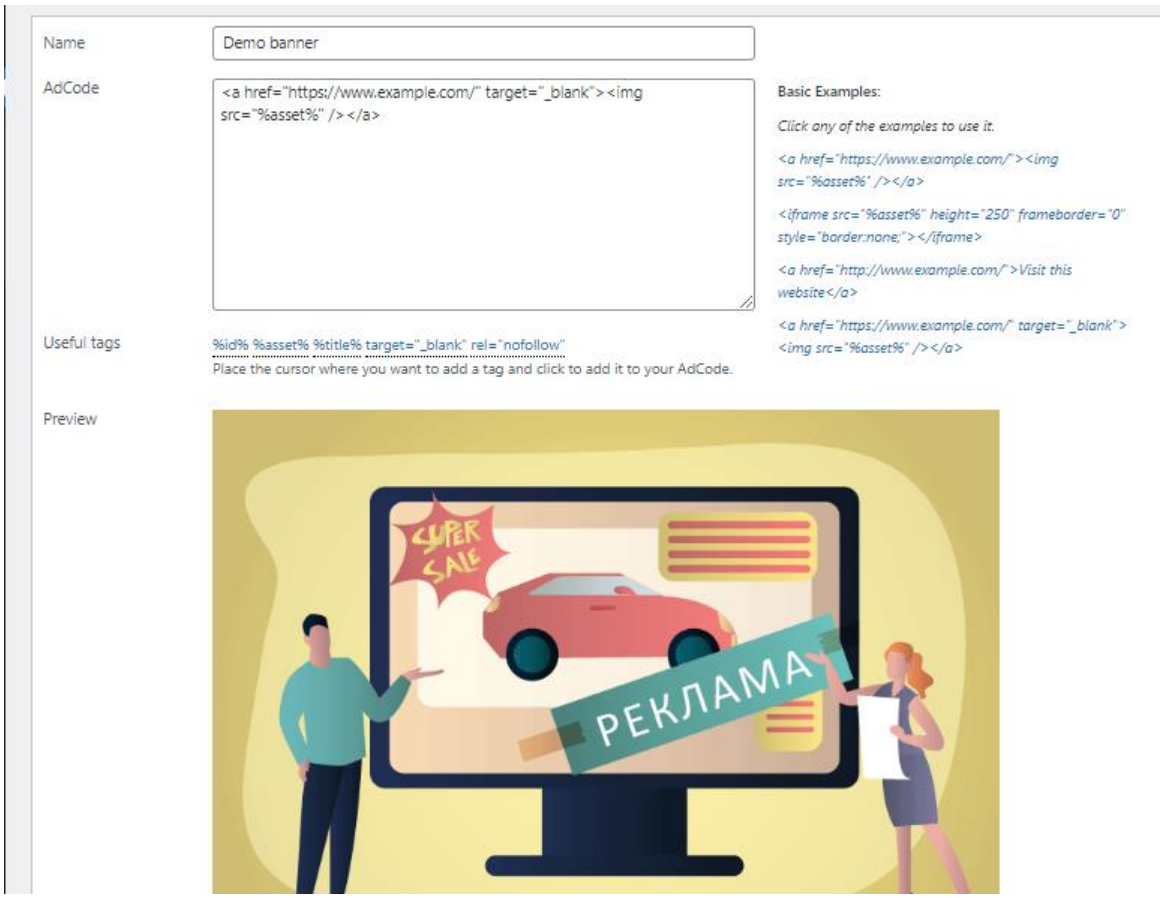

Рисунок 3.11 – Створення рекламного банеру

Після цього потрібно зберегти зміни та вставити шорткод на певну сторінку чи пост веб-ресурсу та отримуємо результат, представлений на рисунку 3.12. Рекламний банер відображається на сторінці.

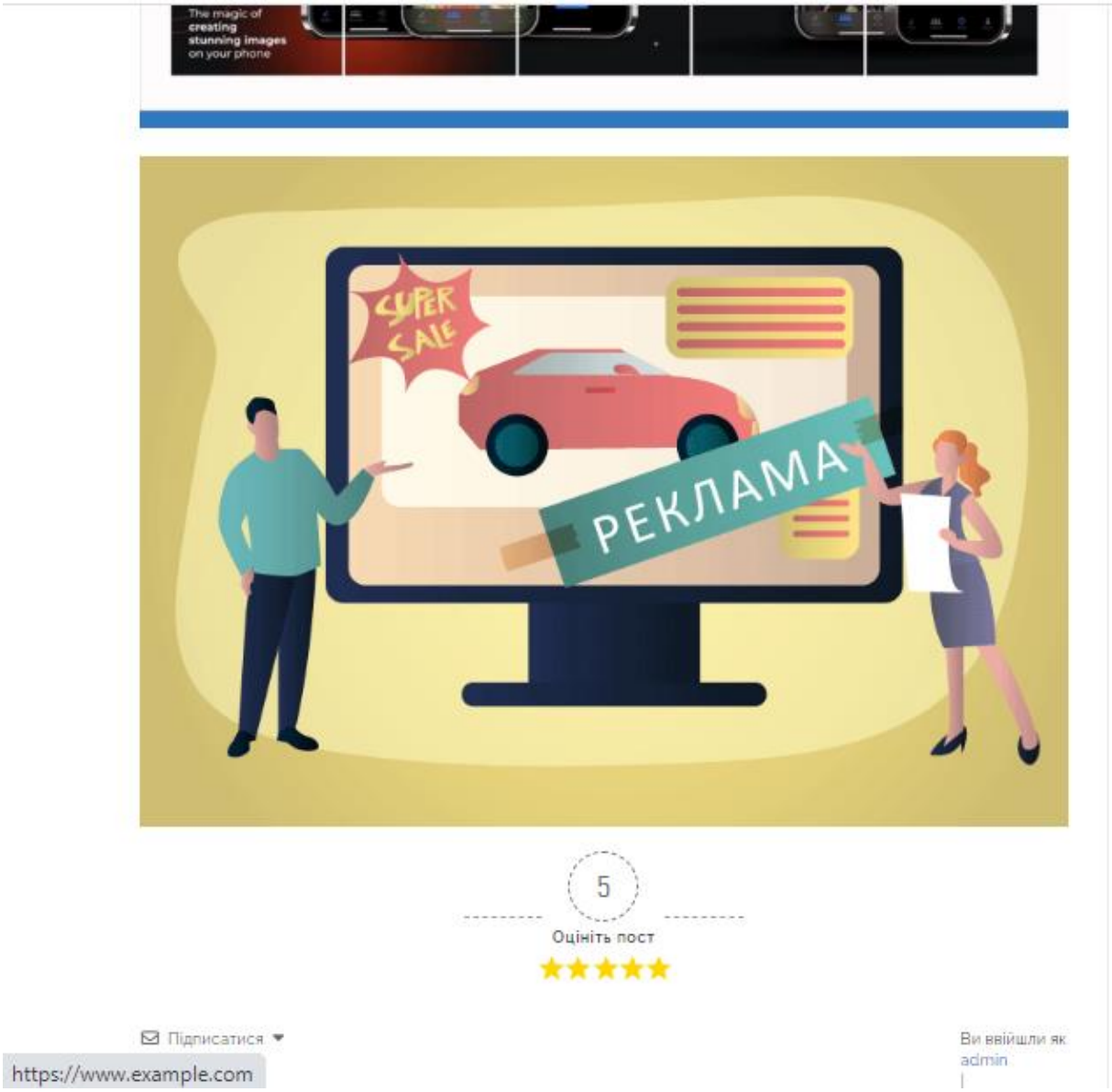

Рисунок 3.12 – Рекламний банер на сторінці веб-ресурсу

Також при створенні чи редагуванні рекламного банера можна вказати умови його зникання чи період відображення на сторінці (рис. 3.13). Вкажемо дату зникання банеру через 2 хвилини після творення та перевіримо чи банер перестане відображатись у вказаний час.

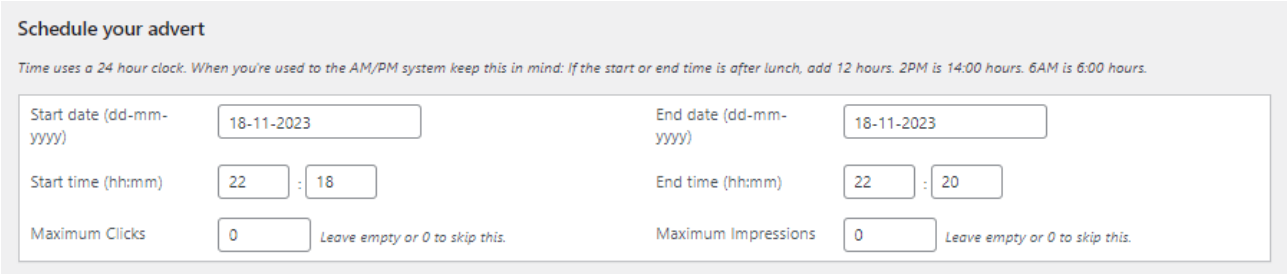

Рисунок 3.13 – Налаштування терміну відображення банера

Створюється банер у розділі Manage Adverts, також можна створити групу рекламний банерів у розділі Manage Groups у меню в dashboard адміністративної панелі (рис. 3.14).

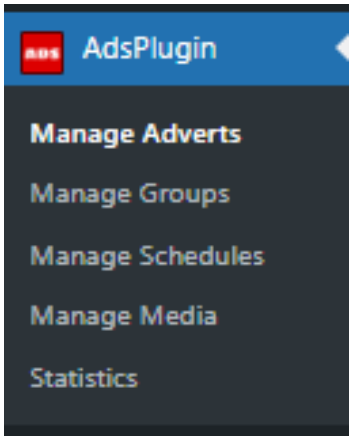

Рисунок 3.14 – Меню плагіна AdsPlugin

Створюємо групу банерів як це показано на рисунку 3.15, в розділі Mode обираємо dynamic mode, що означає, що обрані в групі рекламні банери будуть чергуватись з певною періодичністю. В розділі Automated refresh вказуємо кількість секунд для зміни банера.

Також потрібно обрати конкретні банери в налаштуваннях групи, які будуть в ній відображатись (рис. 3.16). Перед цим їх авжеж потрібно створити в розділі Manage Adverts.

На рисунках 3.17 та 3.18 можна побачити результат – відображення групи банерів зі зміною банера кожних 6 секунд.

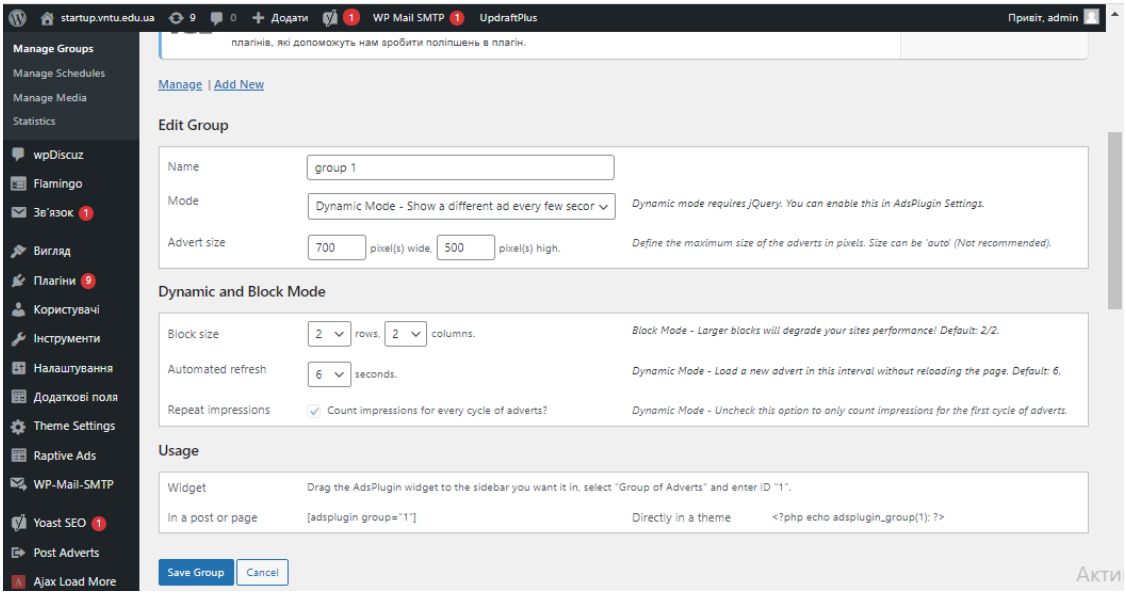

Рисунок 3.15 – Налаштування групи банерів в адмін панелі

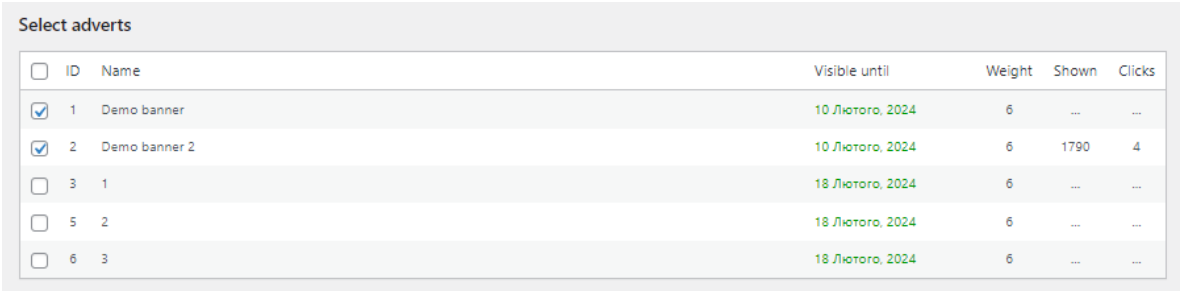

Рисунок 3.16 – Додавання рекламних банерів до групи

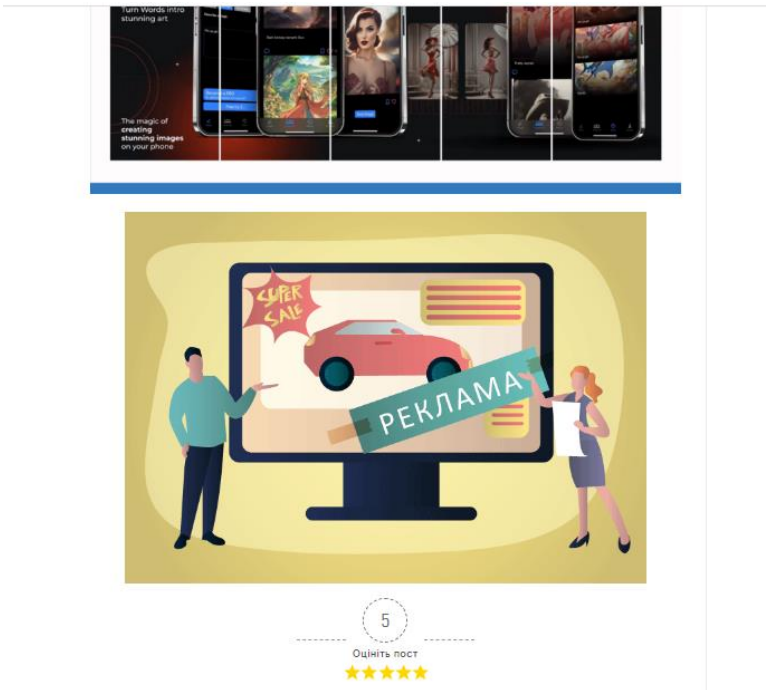

Рисунок 3.17 – Відображення першого рекламного банера з групи

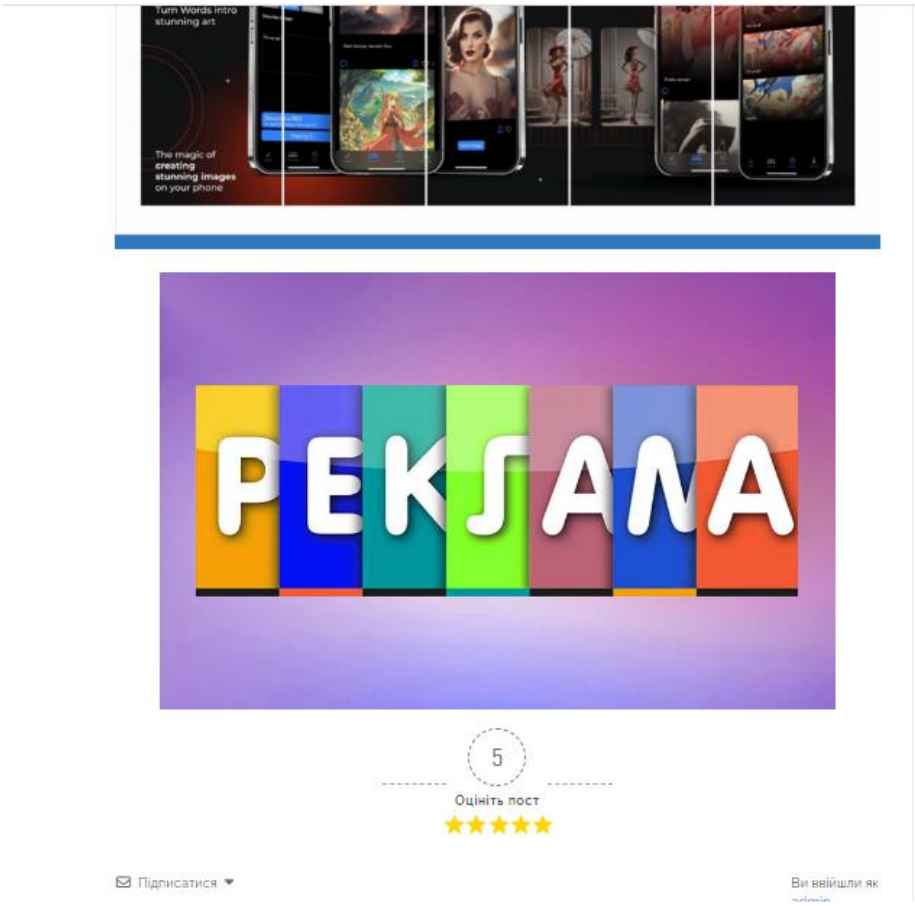

Рисунок 3.18 – Відображення другого рекламного банера з групи

3.7 Висновок

У процесі розробки Front-End частини плагіна для управління рекламою на платформі WordPress виявилося, що правильний вибір інструментів розробки та мов програмування має визначальне значення для досягнення успішного результату. Використання Visual Studio Code сприяло зручності та продуктивності розробки, а обрані мови програмування - JavaScript, HTML і CSS - забезпечили високий рівень функціональності та естетичного вигляду плагіна.

Застосування каскадних таблиць стилів (CSS) було ефективним для управління зовнішнім виглядом рекламних блоків та інших елементів, забезпечуючи легкість в розробці та підтримці. Використання HTML виступило як основний інструмент для створення структури та представлення вмісту веб-сторінок, сприяючи організації інформації та її коректному відображенню.

Процес розробки Front-End частини включав в себе важливі аспекти веб-розробки, такі як бокс-модель, позиціонування та використання флексбокс/грід для ефективного розміщення елементів. Робота з шрифтами та кольорами, а також адаптивний дизайн та зміна стилів при взаємодії, дозволили створити інтерфейс, який відповідає сучасним стандартам та вимогам користувачів.

Використання системи контролю версій Git сприяло ефективній співпраці розробників, відстеженню змін та керуванню версіями коду. Платформа Open Server Panel була важливим інструментом для локального розгортання веб-сервера та роботи з проектами на WordPress.

Завершеною частиною розробки стало впровадження функціональності Front-End частини плагіна, яка включає створення зручного та ефективного інтерфейсу для управління рекламою на веб-ресурсі. Процес охопив всі необхідні аспекти, включаючи дизайн, адаптивність, інтерактивність та оптимізацію швидкості завантаження сторінки.

Загалом, обрані інструменти розробки та підходи дозволили створити ефективний та функціональний Front-End для плагіна управління рекламою на платформі WordPress, відповідаючи вимогам завдання та забезпечуючи задоволення від користування кінцевими користувачами.

4 ЕКОНОМІЧНА ЧАСТИНА

Щоб науково-технічна розробка отримала прийняття та успішно впроваджувалася, важливо, щоб вона відповідала поточним вимогам науково-технічного прогресу та враховувала економічні аспекти. Таким чином, оцінка економічної ефективності результатів науково-дослідної роботи є невід'ємною частиною процесу.

Дослідження в магістерській роботі, присвяченій розробці та вивченню "Інформаційної технології проектування WEB-ресурсів. Частина 1. Клієнтська частина", відноситься до науково-технічних робіт, спрямованих на виведення на ринок (або рішення про виведення науково-технічної розробки на ринок може бути прийнято у процесі самої роботи), відзначаючи так звану комерціалізацію науково-технічної розробки. Цей напрямок розглядається як пріоритетний, оскільки отримані результати можуть бути корисними для різноманітних зацікавлених сторін, приносячи економічні вигоди. Проте для успішної реалізації цього процесу важливо знайти потенційного інвестора, зацікавленого у втіленні даного проекту, та переконати його в обґрунтованості вкладання інвестицій в цей конкретний проект.

Для цього визначені наступні етапи виконання робіт:

1. Проведено комерційний аудит науково-технічної розробки, що включає в себе визначення науково-технічного рівня та комерційного потенціалу.

2. Розраховані витрати на реалізацію науково-технічної розробки.

3. Проведено розрахунок економічної ефективності науково-технічної розробки в разі її впровадження та комерціалізації потенційним інвестором, а також обґрунтовано економічну доцільність комерціалізації для інвестора.

4.1 Проведення комерційного та технологічного аудиту науково-технічної розробки

Метою проведення комерційного і технологічного аудиту дослідження за темою "Інформаційної технології проектування WEB-ресурсів. Частина 1. Клієнтська частина" є оцінювання науково-технічного рівня та рівня комерційного потенціалу розробки front-end частини плагіна для WordPress з функціональними можливостями управління рекламою на веб-ресурсі з метою покращення ефективності та результативності рекламних зусиль.

Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням 5-ти бальної системи оцінювання за 12-ма критеріями, наведеними в табл. 4.1 [37]. Оцінювання показано в таблиці 4.1.

Таблиця 4.1 – Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

Бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено працездатність продукту в реальних умовах
Ринкові переваги (недоліки)					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку

Продовження таблиці 4.1

3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі влас-тивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає

Продовження таблиці 4.1

Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансуванн я є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуют ь ся у військово промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовую ться у виробництві

Продовження таблиці 4.1

11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання науково-технічного рівня та комерційного потенціалу науково-технічної розробки потрібно звести до таблиці. Для оцінки науково-технічного рівня і комерційного потенціалу розробки експертами було запрошено трьох незалежних експертів: Владислав Тютюнов, Front-end розробник, SkySoft; Максим Лешок, FullStack, COAX; Володимир Бурдейний, Back-end розробник, Luxoft. Результати оцінювання показано в таблиці 4.2.

Таблиця 4.2 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки експертами

Критерії	Експерт (ПІБ, посада)		
	Владислав Тютюнов	Максим Лешок	Володимир Бурдейний
	Бали:		
1. Технічна здійсненність концепції	3	3	3
2. Ринкові переваги (наявність аналогів)	2	2	2
3. Ринкові переваги (ціна продукту)	4	4	4
4. Ринкові переваги (технічні властивості)	1	2	2
5. Ринкові переваги (експлуатаційні витрати)	4	4	4
6. Ринкові перспективи (розмір ринку)	4	4	3
7. Ринкові перспективи (конкуренція)	1	2	1
8. Практична здійсненність (наявність фахівців)	3	4	4
9. Практична здійсненність (наявність фінансів)	3	4	4
10. Практична здійсненність (необхідність нових матеріалів)	4	4	4
11. Практична здійсненність (термін реалізації)	4	4	4
12. Практична здійсненність (розробка документів)	4	4	4
Сума балів	$СБ_1$=37	$СБ_2$=41	$СБ_3$=39
Середньоарифметична сума балів $СБ_c$	$\overline{СБ} = \frac{\sum_1^3 СБ_i}{3} = \frac{37+41+39}{3} = 39$		

За результатами розрахунків, наведених в таблиці 4.2, зробимо висновок щодо науково-технічного рівня і рівня комерційного потенціалу розробки. При цьому використаємо рекомендації, наведені в таблиці 4.3 [37].

Таблиця 4.3 – Науково-технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів СБ , розрахована на основі висновків експертів	Науково-технічний рівень та комерційний потенціал розробки
41…48	Високий
31…40	Вище середнього
21…30	Середній
11…20	Нижче середнього
0…10	Низький

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Інформаційної технології проектування WEB-ресурсів. Частина 1. Клієнтська частина.» становить 38,6 балів, що, відповідно до таблиці 4.3 рівень комерційного потенціалу розробки вище середнього, що свідчить про комерційну важливість проведення даних досліджень.

Плагін, який розробляється в магістеріській роботі буде реалізований у CMS WordPress, крім того він буде цікавий власникам веб-ресурсів, які хочуть оптимізувати роботу з рекламою на своїх сервісах.

В результаті реалізації результатів магістерської роботи покращеться управління рекламою на веб-ресурсах та економія часу користувачів.

4.2 Розрахунок витрат на проведення науково-дослідної роботи

Витрати, пов’язані з проведенням науково-дослідної роботи на тему «Інформаційної технології проектування WEB-ресурсів. Частина 1. Клієнтська частина», під час планування, обліку і калькулювання собівартості науково-дослідної роботи групуємо за відповідними статтями.

До статті «Витрати на оплату праці» належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці.

Витрати на основну заробітну плату дослідників ($З_о$) розраховуємо у відповідності до посадових окладів працівників, за формулою [37]:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p}, \qquad (4.1)$$

де k – кількість посад дослідників залучених до процесу досліджень;

M_{ni} – місячний посадовий оклад конкретного дослідника, грн;

t_i – число днів роботи конкретного дослідника, дн.;

T_p – середнє число робочих днів в місяці, T_p=22 дні.

$$З_о = 11500 \cdot 30 / 22 = 15682 \; грн.$$

Проведені розрахунки зведемо до таблиці 4.4.

Таблиця 4.4 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату, грн
Front-end розробник	11500	522,73	30	15682
Back-end розробник	11500	522,73	30	15682
Всього				31364

Додаткову заробітну плату розраховуємо як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%}, \quad (4.2)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати. Приймемо 11%.

$З_{дод}$ = (31364) · 11 / 100% = 3450 *грн.*

Нарахування на заробітну плату дослідників та робітників розраховуємо як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%} \quad (4.3)$$

де $Н_{зп}$ – норма нарахування на заробітну плату. Приймаємо 22%.

Зн = (31364+3450) · 22 / 100% = 7659 грн.

До статті «Сировина та матеріали» належать витрати на сировину, основні та допоміжні матеріали, інструменти, пристрої та інші засоби і предмети праці, які придбані у сторонніх підприємств, установ і організацій та витрачені на проведення досліджень за темою «Інформаційної технології проектування WEB-ресурсів. Частина 1. Клієнтська частина».

Витрати на матеріали (*М*), у вартісному вираженні розраховуються окремо по кожному виду матеріалів за формулою:

$$М = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j - \sum_{j=1}^{n} В_j \cdot Ц_{вj}, \quad (4.4)$$

де $Н_j$ – норма витрат матеріалу *j*-го найменування, кг;

n – кількість видів матеріалів;

$Ц_j$ – вартість матеріалу *j*-го найменування, грн/кг;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ = *1,1 … 1,15*);

$В_j$ – маса відходів *j*-го найменування, кг;

$Ц_{вj}$ – вартість відходів *j*-го найменування, грн/кг.

Проведені розрахунки зведемо до таблиці 4.5.

Таблиця 4.5 – Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за 1шт	Норма витрат, шт	Вартість витраченого матеріалу, грн
Упаковка паперу А4	180	1	180
Ручка	30	2	60
Диск оптичний OPTIMA CD	14	2	28
Flesh-пам'ять 64	400	1	400
Всього			668
З врахуванням коефіцієнта транспортування			734,8

Витрати на комплектуючі *($К_в$)*, які використовують при проведенні НДР на тему «Інформаційної технології проектування WEB-ресурсів. Частина 1. Клієнтська частина» відсутні.

До статті «Спецустаткування для наукових (експериментальних) робіт» належать витрати на виготовлення та придбання спецустаткування необхідного для проведення досліджень, також витрати на їх проектування, виготовлення, транспортування, монтаж та встановлення в роботі відсутні.

До статті «Програмне забезпечення для наукових (експериментальних) робіт» належать витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення, (програм, алгоритмів, баз даних) необхідних для проведення досліджень, також витрати на їх проектування, формування та встановлення в роботі відсутні.

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо, розраховуємо з використанням прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12}, \quad (4.5)$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

$A_{обл} = (35000 \cdot 1) / (2 \cdot 12) = 1458,33$ *грн.*

Проведені розрахунки зведемо до таблиці 4.6.

Таблиця 4.6 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Комп'ютер	35000	2	1	1458,33
Приміщення лабораторії	270000	20	1	1125,00
Всього				2583,33

Витрати на силову електроенергію ($B_е$) розраховуємо за формулою:

$$B_е = \sum_{i=1}^{n} \frac{W_{уі} \cdot t_i \cdot Ц_е \cdot K_{вті}}{\eta_i}, \quad (4.6)$$

де $W_{уі}$ – встановлена потужність обладнання на визначеному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_е$ – вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо $Ц_е$ = 7,5 грн;

$K_{впі}$ – коефіцієнт, що враховує використання потужності, $K_{впі} < 1$;

η_i – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

$B_е = 0{,}25 \cdot 300{,}0 \cdot 7{,}5 \cdot 0{,}5 / 0{,}8 = 351{,}56$ *грн.*

До статті «Службові відрядження» дослідної роботи на тему «Інформаційної технології проектування WEB-ресурсів. Частина 1. Клієнтська частина» належать витрати на відрядження штатних працівників, працівників організацій, які працюють за договорами цивільно-правового характеру, аспірантів, зайнятих розробленням досліджень, відрядження, пов'язані з проведенням випробувань машин та приладів, а також витрати на відрядження на наукові з'їзди, конференції, наради, пов'язані з виконанням конкретних досліджень. В нашій роботі відсутні.

Витрати за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації» відсутні.

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуємо як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$I_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \qquad (4.7)$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати», приймемо $Н_{ів} = 50\%$.

$І_в = (31364) \cdot 50 / 100\% = 15681{,}82$ грн.

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов'язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов'язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов'язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуємо як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (4.8)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв}$ = 100%.

$В_{нзв}$ = (31364) · 100 / 100% = 31363,64 грн.

Витрати на проведення науково-дослідної роботи на тему «Інформаційної технології проектування WEB-ресурсів. Частина 1. Клієнтська частина». розраховуємо як суму всіх попередніх статей витрат за формулою:

$$B_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + B_{спец} + B_{прг} + A_{обл} + B_е + B_{св} + B_{сп} + I_в + B_{нзв}. \quad (4.9)$$

$В_{заг}$= 31364+3450+7659+734,8+2583,33+351,56+15681,82+31363,64 = 93187,79 грн.

Загальні витрати *ЗВ* на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховується за формулою:

$$ЗВ = \frac{B_{заг}}{\eta}, \quad (4.10)$$

де η - коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи, приймемо η=0,7.

ЗВ = 93187,79 / 0,7 = 133125,41 грн.

4.3 Розрахунок економічної ефективності науково-технічної розробки при її можливій комерціалізації потенційним інвестором

В ринкових умовах узагальнюючим позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку.

Результати дослідження проведені за темою «Інформаційної технології проектування WEB-ресурсів. Частина 1. Клієнтська частина» передбачають комерціалізацію протягом 3-х років реалізації на ринку.

В цьому випадку основу майбутнього економічного ефекту будуть формувати:

ΔN – збільшення кількості споживачів яким надається відповідна інформаційна послуга у періоди часу, що аналізуються;

N – кількість споживачів яким надавалась відповідна інформаційна послуга у році до впровадження результатів нової науково-технічної розробки, приймемо 1 особа

$Ц_б$ – вартість послуги у році до впровадження інформаційної системи, приймемо 3000,00 грн;

$\pm\Delta Ц_o$ – зміна вартості послуги від впровадження результатів, приймемо зростання на 500,00 грн.

Можливе збільшення чистого прибутку у потенційного інвестора $\Delta П_i$ для кожного із 3-х років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково-технічної розробки, розраховуємо за формулою [37]:

$$\Delta П_i = (\pm\Delta Ц_o \cdot N + Ц_o \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \qquad (4.11)$$

де λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2021 році ставка податку на додану вартість складає 20%, а коефіцієнт $\lambda = 0,8333$;

ρ – коефіцієнт, який враховує рентабельність інноваційного продукту). Приймемо ρ =40%;

ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2023 році $\vartheta = 18\%$;

Збільшення чистого прибутку 1-го року:

$\Delta П_1 = (1 \cdot 1000 + 50000 \cdot 200) \cdot 0,83 \cdot 0,4 \cdot (1 - 0,18/100\%) = 299031,79$ грн.

Збільшення чистого прибутку 2-го року:

$\Delta П_2 = (1 \cdot 500 + 5000 \cdot (500 + 300)) \cdot 0,83 \cdot 0,4 \cdot (1 - 0,18/100\%) = 478814,2$ грн.

Збільшення чистого прибутку 3-го року:

$\Delta П_3 = (1 \cdot 500 + 5000 \cdot (500 + 300 + 200)) \cdot 0,83 \cdot 0,4 \cdot (1 - 0,18/100\%) = 598392,75$ грн.

Приведена вартість збільшення всіх чистих прибутків *ПП*, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \qquad (4.12)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн;

T – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ =18%;

t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

$ПП = 299031,79 /(1+0,18)^1+478814,2 /(1+0,18)^2+598392,75 /(1+0,18)^3=$

$= 929605,78$ грн.

Величина початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки:

$$PV = k_{інв} \cdot ЗВ, \tag{4.13}$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію, приймаємо $k_{інв}$=2;

$ЗВ$ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, приймаємо 133125,41грн.

$PV = k_{інв} \cdot ЗВ = 2 \cdot 133125,41 = 266250,82$ грн.

Абсолютний економічний ефект $E_{абс}$ для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$E_{абс} = ПП - PV, \tag{4.14}$$

де $ПП$ – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, 929605,78 грн;

PV – теперішня вартість початкових інвестицій, 266250,82 грн.

$E_{абс} = ПП - PV = 929605,78 - 266250,82 = 663354,96$ грн.

Внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1, \quad (4.18)$$

де $E_{абс}$ – абсолютний економічний ефект вкладених інвестицій, грн;

PV – теперішня вартість початкових інвестицій, грн;

$T_ж$ – життєвий цикл науково-технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, 3 роки.

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1 = (1+663354{,}96 /266250{,}82)^{1/3} - 1 = 0{,}82.$$

Мінімальна внутрішня економічна дохідність вкладених інвестицій $\tau_{мін}$:

$$\tau_{мін} = d + f, \quad (4.19)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2023 році в Україні d =0,1;

f – показник, що характеризує ризикованість вкладення інвестицій, приймемо 0,25.

$\tau_{мін}$ = 0,1+0,25 = 0,35 < 0,82 свідчить про те, що внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки вища мінімальної внутрішньої дохідності. Тобто інвестувати в науково-дослідну роботу за темою «Інформаційної технології проектування WEB-ресурсів. Частина 1. Клієнтська частина» доцільно.

Період окупності інвестицій $T_{ок}$ які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{E_в}, \quad (4.20)$$

де $E_в$ – внутрішня економічна дохідність вкладених інвестицій.

$T_{ок}$ = 1 / 0,82 = 1,2 р.

$T_{ок}$ < 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

4.4 Висновок

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Інформаційної технології проектування WEB-ресурсів. Частина 1. Клієнтська частина» становить 39 балів, що, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього).

Також термін окупності становить 1,2 р., що менше 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

Отже можна зробити висновок про доцільність проведення науково-дослідної роботи за темою «Інформаційної технології проектування WEB-ресурсів. Частина 1. Клієнтська частина»

ВИСНОВОКИ

Покращення технології проектування клієнтської частини WEB-ресурсів досягнуто за рахунок використання інтегрованого плагіну для розширення функціоналу по управлінню рекламою та автоматизації процесів статистичної обробки.

Всі задачі виконані в повному об’ємі. Було визначено мету, об’єкт, предмет та задачі подальшого дослідження. Під час виконання дослідження було здійснено аналіз різновидів веб-ресурсів. У результаті дослідження ключових аспектів розробки плагіна для WordPress з функціональними можливостями управління рекламою на веб-ресурсі було отримано глибоке розуміння інструментів та технологій, пов'язаних із створенням веб-ресурсів та управлінням рекламою на них. Були розглянуті види веб-ресурсів у сучасному світі та їхня роль у представленні інформації та бізнес-просуванні. Крім того, було надано детальний огляд інформаційних технологій проектування веб-ресурсів.

Було розроблено покращену технологію Front-end частини плагіна, розглянуто та розроблено UML діаграми, спроектовано складові частини плагіна, що відповідають за Front-end частину. Оптимізований front-end код є ключовим для забезпечення ефективної роботи веб-ресурсу. Використання засобів моделювання UML сприяє створенню докладного плану роботи та полегшує реалізацію та підтримку плагіна. Ці підходи сприяють покращенню продуктивності, надійності та доступності веб-ресурсу, також полегшують роботу розробників плагіна. Було розроблено UML-діаграми, а саме діаграму класів, діаграму послідовності, діаграму активностей та діаграму станів для клієнтської частини плагіна і описано його проектування.

Як результат розроблено Front-end частину плагіна у вибраних мовних середовищах. Описано розробку функціональності серверної частини плагіна. Розглянуто розроблені функції, а також проведено тестування, яке підтвердило правильність роботи розробленого функціоналу. Обрані інструменти виявилися

оптимальними для розробки плагіна для WordPress з управлінням рекламою. Використання PHP, JavaScript, HTML, CSS та Visual Studio Code дозволило зручно та ефективно реалізувати клієнтську частину, а Open Server Panel та GitHub сприяли легкій локальній розробці та спільній роботі над проектом. Час відгуку не перевищує 0,5 секунд.

Було також розроблено економічну чатсину Front-end частини плагіна. В економічній частині розраховано витрати на розробку нового програмного продукту, сума яких складає 133125,41 грн., а також основні економічні показники, за допомогою яких доводиться ефективність розробленого програмного продукту та доцільність використання розробки. Рівень комерційного потенціалу розробки становить 39 балів, що, свідчить про її комерційну важливість. Термін окупності складає 1,2 року, що свідчить про високу комерційну привабливість науково-технічної розробки. Це може стати стимулом для потенційного інвестора фінансувати впровадження цієї розробки та виведення її на ринок.

Розширення функціоналу полягає в відображенні рекламних банерів та груп банерів відповідно до вказаних користувачем налашувань місцезнаходження. Плагін забезпечує користувачам можливості для відслідковування, аналізу результатів рекламних кампаній та передачі запиту з даними на серверну частину. Розроблений плагін має функціонал відображення рекламних банерів у певний визначений час та дати.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Веб-ресурс стартап-школи ВНТУ Sikorsky challenge [Електронний ресурс] – Режим доступу до ресурсу: https://startup.vntu.edu.ua (дата звернення: 05.12.23).

2. Сілагін, О. В., Борисюк, О. О., Сторожук, А. С., Цвєткова, Ю. В. РОЗРОБКА САЙТУ СТАРТАПІВ "SIKORSKY CHALLENGE": ПРОЕКТУВАННЯ ТА ДИЗАЙН САЙТУ. НТКП ВНТУ. Факультет інтелектуальних інформаційних технологій та автоматизації, м. Вінниця, Україна. 2022. [Електронний ресурс] – Режим доступу до ресурсу: https://conferences.vntu.edu.ua/index.php/all-fksa/all-fksa-2022/paper/view/15233/12833 (дата звернення: 05.12.23).

3. Сторожук, А. С., Борисюк, О. О., Сілагін, О. В. РОЗРОБКА ПЛАГІНА У CMS WORDPRESS ДЛЯ УПРАВЛІННЯ РЕКЛАМОЮ НА WEB-РЕСУРСІ. НТКП ВНТУ. Факультет інтелектуальних інформаційних технологій та автоматизації, м. Вінниця, Україна. 2023. [Електронний ресурс] – Режим доступу до ресурсу: https://conferences.vntu.edu.ua/index.php/mn/mn2024/paper/view/19491 (дата звернення: 05.12.23).

4. WebTune [Електронний ресурс]. URL: https://webtune.com.ua/statti/web-rozrobka/vydy-sajtiv-ta-yih-funkczional/ (дата звернення: 10.10.23).

5. Технології розробки сайтів. Інструменти розробки сайтів. [Електронний ресурс]. URL: https://vseosvita.ua/lesson/tekhnolohii-rozrobky-saitiv-instrumenty-rozrobky-saitiv-317690.html (дата звернення: 10.10.23).

6. Node js vs Django and vs Ruby on Rails [Електронний ресурс]. URL: https://www.fortunesoftit.com/node-vs-django-vs-rails/ (дата звернення: 10.10.23).

7. Про SSL-сертифікати [Електронний ресурс]. URL: https://ssl.com.ua/blog/ukr/about-ssl-certificates/ (дата звернення: 10.10.23).

8. Веб-програмування. Що таке CMS? [Електронний ресурс]. URL: https://webstudio2u.net/ua/programming/96-cms.html (дата звернення: 10.10.23).

9. PWA додатки [Електронний ресурс]. URL: https://avada-media.ua/ua/services/pwa/ (дата звернення: 10.10.23).

10. Переваги та недоліки CMS WordPress [Електронний ресурс]. URL: https://esfirum.com/blog/pros-and-cons-cms-wordpress/ (дата звернення: 10.10.23).

11. Реклама в онлайн-джерелах [Електронний ресурс]. URL: https://bizmag.com.ua/dlia-choho-potribna-reklama-v-zmi/ (дата звернення: 10.10.23).

12. Реклама на сайті [Електронний ресурс]. URL: https://pbb.lviv.ua/statti-i-novyny/statti-shchodo-stvorennia-saitu/reklama-na-sayti/ (дата звернення: 10.10.23).

13. Оптимізація швидкості завантаження сайту [Електронний ресурс]. URL: https://seo-evolution.com.ua/blog/razrabotka/optymizatsiya-shvydkosti-zavantazhennya-saytu (дата звернення: 10.10.23).

14. Основи UML [Електронний ресурс]. URL: https://docs.kde.org/trunk5/uk/umbrello/umbrello/uml-basics.html (дата звернення: 10.10.23).

15. Розробка плагінів WordPress [Електронний ресурс]. URL: https://webbookstudio.com/ua/wordpress-development/custom-wordpress-plugin-development/ (дата звернення: 10.10.23).

16. Етапи створення веб сайтів [Електронний ресурс]. URL: https://webtune.com.ua/statti/web-rozrobka/etapy-stvorennya-veb-sajtiv/ (дата звернення: 10.10.23).

17. Знайомство з Visual Studio Code [Електронний ресурс]. URL: https://romul.name/blog/znayomstvo-z-visual-studio-code/ (дата звернення: 10.11.23).

18. JavaScript [Електронний ресурс]. URL: https://astwellsoft.com/uk/blog /tehnology/javascript.html (дата звернення: 10.11.23).

19. Чому JavaScript — перспективна мова програмування? [Електронний ресурс]. URL: https://dou.ua/forums/topic/35184/ (дата звернення: 10.11.23).

20. JavaScript - краща мова для програмування. Правда чи брехня? [Електронний ресурс]. URL: https://apeps.kpi.ua/javascript-krashcha-mova-dlia-programuvania (дата звернення: 10.11.23).

21. TypeScript: від хайпу до стандарту розробки [Електронний ресурс]. URL: https://www.gen.tech/post/typescript-vid-hajpu-do-standartu-rozrobki (дата звернення: 10.11.23).

22. CoffeeScript [Електронний ресурс]. URL: https://uk.wikipedia.org /wiki/CoffeeScript (дата звернення: 10.11.23).

23. Які мови програмування треба знати для створення сайтів [Електронний ресурс]. URL: https://foxminded.ua/movy-prohramuvannia-dlia-stvorennia-saitiv/ (дата звернення: 10.11.23).

24. Що таке HTML та як за допомогою нього увійти в IT [Електронний ресурс]. URL: https://mc.today/uk/shho-take-html-ta-yak-za-dopomogoyu-nogo-uvijti-do-it/ (дата звернення: 10.11.23).

25. Що таке CSS [Електронний ресурс]. URL: https://css.in.ua/article/shcho-take-html_10 (дата звернення: 10.11.23).

26. Повне керівництво по Flexbox [Електронний ресурс]. URL: https://devzone.org.ua/post/povne-kerivnitstvo-po-flexbox (дата звернення: 10.11.23).

27. Навіщо потрібна адаптивна верстка [Електронний ресурс]. URL: https://wezom.academy/ua/zachem-nuzhna-adaptivnaja-verstka/ (дата звернення: 10.11.23).

28. ЩО ТАКЕ GIT? [Електронний ресурс]. URL: https://freehost.com.ua /ukr/faq/wiki/chto-takoe-git/ (дата звернення: 10.11.23).

29. Налаштування Open Server [Електронний ресурс]. URL: https://armedsoft.com/ua/blog/nalashtuvannya-open-server (дата звернення: 10.11.23).

30. Оператори мови JavaScript. Класифікація операторів [Електронний ресурс]. URL: https://www.bestprog.net/uk/2022/05/22/javascript-javascript-language-operators-classification-of-operators-ua/ (дата звернення: 10.11.23).

31. Редагуйте код за допомогою веб-програми Visual Studio Code [Електронний ресурс]. URL: https://learn.microsoft.com/uk-ua/power-pages/configure/visual-studio-code-editor (дата звернення: 10.11.23).

32. Етапи створення сайту [Електронний ресурс]. URL: https://webcase.com.ua/uk/blog/iz-chego-sostoit-razrabotka-sajta/ (дата звернення: 27.11.23).

33. Як використовувати шорткод WordPress [Електронний ресурс]. URL: https://hostenko.com/uk/wpcafe/навчальні-посібники/shortkodi-wordpress/ (дата звернення: 27.11.23).

34. Види тестування та відмінності між ними [Електронний ресурс]. URL: https://qagroup.com.ua/publications/vydy-testuvannya-ta-vidminnosti-mizh-nymy/ (дата звернення: 27.11.23).

35. Cypress [Електронний ресурс]. URL: https://www.cypress.io/ (дата звернення: 27.11.23).

36. Top Javascript Testing Frameworks [Електронний ресурс]. URL: https://qagroup.com.ua/publications/top-javascript-testing-frameworks-2020/ (дата звернення: 27.11.23).

37. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. – Вінниця : ВНТУ, 2021. – 42 с.

Додаток А (обов'язковий)

Протокол перевірки кваліфікаційної роботи на наявність текстових запозичень

Назва роботи: Інформаційна технологія з проектування WEB-ресурсів. Частина 1. Клієнтська частина.

Тип роботи: магістерська кваліфікаційна робота
(БДР, МКР)

Підрозділ кафедра комп'ютерних наук, ФІТА
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 97,6% Схожість 2,4%

Аналіз звіту подібності (відмітити потрібне):

- ✓ Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.
- □ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.
- □ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ Сторожук А.С.

Керівник роботи _____ Сілагін О.В.

Опис прийнятого рішення

Магістерську кваліфікаційну роботу допущено до захисту

Особа, відповідальна за перевірку _____ Озеранський В.С.

Додаток Б (обов'язковий)

Лістинг програми

```
function adsplugin_header() {
	if(!function_exists('get_plugins')) require_once ABSPATH . 'wp-
admin/includes/plugin.php';
	$plugins = get_plugins();
	$plugin_version = $plugins['adsplugin/adsplugin.php']['Version'];

	$output = "\n<!-- This site is using AdsPlugin v".$plugin_version." -->\n";
	echo $output;

	adsplugin_custom_css();
}
function adsplugin_custom_css() {
	global $adsplugin_config;

	$generated_css = get_option('adsplugin_group_css');

	$output = "<!-- AdsPlugin CSS -->\n";
	$output .= "<style type=\"text/css\" media=\"screen\">\n";
	$output .= "\t.g { margin:20px 0px; padding:0px; overflow:hidden; line-height:1; zoom:1;
}\n";
	$output .= "\t.g img { height:auto; }\n";
	$output .= "\t.g-col { position:relative; float:left; }\n";
	$output .= "\t.g-col:first-child { margin-left: 0; }\n";
	$output .= "\t.g-col:last-child { margin-right: 0; }\n";
	if($generated_css) {
		foreach($generated_css as $group_id => $css) {
			if(strlen($css) > 0) {
				$output .= $css;
			}
		}
		unset($generated_css);
	}
	$output .= "\t@media only screen and (max-width: 480px) {\n";
	$output .= "\t\t.g-col, .g-dyn, .g-single { width:100%; margin-left:0; margin-right:0; }\n";
	$output .= "\t}\n";
	if($adsplugin_config['widgetpadding'] == "Y") {
		$output .= ".adsplugin_widgets, .ajdg_bnnrwidgets, .ajdg_grpwidgets {
overflow:hidden; padding:0; }\n";
	}
	$output .= "</style>\n";
	$output .= "<!-- /AdsPlugin CSS -->\n\n";

	echo $output;
}
function adsplugin_scripts() {
	global $adsplugin_config;

	$in_footer = false;
	if($adsplugin_config['jsfooter'] == "Y") {
```

```
		$in_footer = true;
	}

	if($adsplugin_config['jquery'] == 'Y') wp_enqueue_script('jquery', false, false, false,
$in_footer);
	if(get_option('adsplugin_dynamic_required') > 0) wp_enqueue_script('jshowoff-adsplugin',
plugins_url('/library/jquery.adsplugin.dyngroup.js', __FILE__), false, null, $in_footer);

	// Make clicktracking and impression tracking a possibility
	if($adsplugin_config['stats'] == 1) {
		 wp_enqueue_script('clicktrack-adsplugin',
plugins_url('/library/jquery.adsplugin.clicktracker.js', __FILE__), false, null, $in_footer);
		wp_localize_script('jshowoff-adsplugin', 'impression_object', array('ajax_url' =>
admin_url( 'admin-ajax.php')));
		wp_localize_script('clicktrack-adsplugin', 'click_object', array('ajax_url' =>
admin_url('admin-ajax.php')));
	}

	if(!$in_footer) {
		add_action('wp_head', 'adsplugin_custom_javascript');
	} else {
		add_action('wp_footer', 'adsplugin_custom_javascript', 100);
	}
}

function adsplugin_custom_javascript() {
	global $wpdb, $adsplugin_config;

	$groups = $wpdb->get_results("SELECT `id`, `adspeed` FROM `{$wpdb-
>prefix}adsplugin_groups` WHERE `name` != '' AND `modus` = 1 ORDER BY `id` ASC;");
	if($groups) {
		$output = "<!-- AdsPlugin JS -->\n";
		$output .= "<script type=\"text/javascript\">\n";
		$output .= "jQuery(document).ready(function(){\n";
		$output .= "if(jQuery.fn.gslider) {\n";
		foreach($groups as $group) {
			$output .= "\tjQuery('.g-".$group->id."').gslider({ groupid: ".$group->id.",
speed: ".$group->adspeed." });\n";
		}
		$output .= "}\n";
		$output .= "});\n";
		$output .= "</script>\n";
		$output .= "<!-- /AdsPlugin JS -->\n\n";
		unset($groups);
		echo $output;
	}
}
function adsplugin_shortcode($atts, $content = null) {
	global $adsplugin_config;

	$banner_id = (!empty($atts['banner'])) ? trim($atts['banner'], "\r\t ") : 0;
	$group_ids = (!empty($atts['group'])) ? trim($atts['group'], "\r\t ") : 0;
```

```
	$output = '';
	if($adsplugin_config['w3caching'] == "Y") {
		$output .= '<!-- mfunc '.W3TC_DYNAMIC_SECURITY.' -->';
		if($banner_id > 0 AND ($group_ids == 0 OR $group_ids > 0)) { // Show one Ad
			$output .= 'echo adsplugin_ad('.$banner_id.');';
		}
		if($banner_id == 0 AND $group_ids > 0) { // Show group
			$output .= 'echo adsplugin_group('.$group_ids.');';
		}
		$output .= '<!-- /mfunc '.W3TC_DYNAMIC_SECURITY.' -->';
	} else if($adsplugin_config['borlabscache'] == "Y" AND
function_exists('BorlabsCacheHelper')) {
		if(BorlabsCacheHelper()->willFragmentCachingPerform()) {
			$borlabsphrase = BorlabsCacheHelper()->getFragmentCachingPhrase();

			$output .= '<!--[borlabs cache start: '.$borlabsphrase.']--> ';
			if($banner_id > 0 AND ($group_ids == 0 OR $group_ids > 0)) { // Show one
Ad
				$output .= 'echo adsplugin_ad('.$banner_id.');';
			}
			if($banner_id == 0 AND $group_ids > 0) { // Show group
				$output .= 'echo adsplugin_group('.$group_ids.');';
			}
			$output .= ' <!--[borlabs cache end: '.$borlabsphrase.']-->';

			unset($borlabsphrase);
		}
	} else {
		if($banner_id > 0 AND ($group_ids == 0 OR $group_ids > 0)) { // Show one Ad
			$output .= adsplugin_ad($banner_id);
		}

		if($banner_id == 0 AND $group_ids > 0) { // Show group
			$output .= adsplugin_group($group_ids);
		}
	}

	return $output;
}
function adsplugin_inject_posts_cache_wrapper($group_id) {
	global $adsplugin_config;

	if($adsplugin_config['w3caching'] == 'Y') {
		$advert_output = '<!-- mfunc '.W3TC_DYNAMIC_SECURITY.' -->';
		$advert_output .= 'echo adsplugin_group('.$group_id.');';
		$advert_output .= '<!-- /mfunc '.W3TC_DYNAMIC_SECURITY.' -->';
	} else if($adsplugin_config['borlabscache'] == "Y" AND
function_exists('BorlabsCacheHelper')) {
		if(BorlabsCacheHelper()->willFragmentCachingPerform()) {
			$borlabsphrase = BorlabsCacheHelper()->getFragmentCachingPhrase();
```

```
                $advert_output = '<!--[borlabs cache start: '.$borlabsphrase.']-->';
                $advert_output .= 'echo adsplugin_group('.$group_id.');';
                $advert_output .= '<!--[borlabs cache end: '.$borlabsphrase.']-->';

                unset($borlabsphrase);
            }
        } else {
            $advert_output = adsplugin_group($group_id);
        }

        return $advert_output;
}
function adsplugin_inject_posts($post_content) {
        global $wpdb, $post;

        $categories_top = $categories_bottom = $categories_inside = array();
        if(is_page()) {
            // Inject ads into pages
            $groups = $wpdb->get_results("SELECT `id`, `page`, `page_loc`, `page_par` FROM
`{$wpdb->prefix}adsplugin_groups` WHERE `page_loc` > 0 AND  `page_loc` < 5;");

            foreach($groups as $group) {
                $pages_more = explode(",", $group->page);

                if(count($pages_more) > 0) {
                    if(in_array($post->ID, $pages_more)) {
                        if($group->page_loc == 1 OR $group->page_loc == 3) {
                            $categories_top[$group->id] = $group->page_par;
                        }
                        if($group->page_loc == 2 OR $group->page_loc == 3) {
                            $categories_bottom[$group->id] = $group->page_par;
                        }
                        if($group->page_loc == 4) {
                            $categories_inside[$group->id] = $group->page_par;
                        }
                        unset($pages_more, $group);
                    }
                }
            }
        }

        if(is_single()) {
            // Inject ads into posts in specified category
            $groups = $wpdb->get_results("SELECT `id`, `cat`, `cat_loc`, `cat_par` FROM
`{$wpdb->prefix}adsplugin_groups` WHERE `cat_loc` > 0 AND `cat_loc` < 5;");
            $wp_categories = wp_get_post_categories($post->ID, array('taxonomy' =>
'category', 'fields' => 'ids'));

            foreach($groups as $group) {
                $categories_more = array_intersect($wp_categories, explode(",", $group-
>cat));
```

```
                        if(count($categories_more) > 0) {
                                if(has_category($categories_more, $post->ID)) {
                                        if(($group->cat_loc == 1 OR $group->cat_loc == 3)) {
                                                $categories_top[$group->id] = $group->cat_par;
                                        }
                                        if($group->cat_loc == 2 OR $group->cat_loc == 3) {
                                                $categories_bottom[$group->id] = $group->cat_par;
                                        }
                                        if($group->cat_loc == 4) {
                                                $categories_inside[$group->id] = $group->cat_par;
                                        }
                                        unset($categories_more, $group);
                                }
                        }
                }
        }

        // Advert in front of content
        if(count($categories_top) > 0) {
                $post_content =
adsplugin_inject_posts_cache_wrapper(array_rand($categories_top)).$post_content;
        }

        // Advert behind the content
        if(count($categories_bottom) > 0) {
                $post_content =
$post_content.adsplugin_inject_posts_cache_wrapper(array_rand($categories_bottom));
        }

        // Adverts inside the content
        if(count($categories_inside) > 0) {
                // Setup
                $categories_inside = adsplugin_shuffle($categories_inside);
           $post_content_exploded = explode('</p>', $post_content);
                $post_content_count = ceil(count($post_content_exploded));
                $inserted = array();

                // Determine after which paragraphs ads should show
                foreach($categories_inside as $group_id => $group_paragraph) {
                        if($group_paragraph == 99) {
                                $group_paragraph = $post_content_count / 2; // Middle of content
                        }

                        $group_paragraph = intval($group_paragraph);

                        // Create $inserted with paragraphs numbers and link the group to it. This list
is leading from this point on.
                        if(!array_key_exists($group_paragraph, $inserted)) {
                                $inserted[$group_paragraph] = $group_id;
                        }
                        unset($group_id, $group_paragraph);
                }
```

```
                // Inject ads behind paragraphs based on $inserted created above, IF a group_id is set
higher than 0
                foreach($post_content_exploded as $index => $paragraph) {
                        $insert_here = $index + 1; // Deal with array offset
                        if(array_key_exists($insert_here, $inserted)) {
                                if($inserted[$insert_here] > 0) {
                                        $post_content_exploded[$index] .=
adsplugin_inject_posts_cache_wrapper($inserted[$insert_here]);
                                        $inserted[$insert_here] = 0;
                                }
                        }
                        unset($index, $paragraph, $insert_here);
                }

                // Re-assemble post_content and clean up
           $post_content = implode('', $post_content_exploded);
                unset($post_content_exploded, $post_content_count, $inserted);
        }

        unset($groups, $categories_top, $categories_bottom, $categories_inside);

        return $post_content;
}
function adsplugin_preview($banner_id) {
        global $wpdb;

        if($banner_id) {
                $now = current_time('timestamp');

                $banner = $wpdb->get_row($wpdb->prepare("SELECT * FROM `{$wpdb-
>prefix}adsplugin` WHERE `id` = %d;", $banner_id));

                if($banner) {
                        $image = str_replace('%folder%', '/banners/', $banner->image);
                        $output = adsplugin_ad_output($banner->id, 0, $banner->title, $banner-
>bannercode, $banner->tracker, $image);
                } else {
                        $output = adsplugin_error('ad_expired');
                }
        } else {
                $output = adsplugin_error('ad_no_id');
        }

        return $output;
}
function adsplugin_ad_output($id, $group, $name, $bannercode, $tracker, $image) {
        global $blog_id, $adsplugin_config;

        $banner_output = $bannercode;
        $banner_output = stripslashes(htmlspecialchars_decode($banner_output, ENT_QUOTES));
```

```
		if($adsplugin_config['stats'] > 0 AND $tracker == "Y") {
			if(empty($blog_id) or $blog_id == '') {
				$blog_id = 0;
			}

			if($adsplugin_config['stats'] == 1) { // Internal tracker
				preg_match_all('/<a[^>](?:.*?)>/i', $banner_output, $matches,
PREG_SET_ORDER);
				if(isset($matches[0])) {
					$banner_output = str_ireplace('<a ', '<a data-
track="'.adsplugin_hash($id, $group, $blog_id).'" ', $banner_output);
					foreach($matches[0] as $value) {
						if(preg_match('/<a[^>]+class=\"(.+?)\"[^>]*>/i', $value,
$regs)) {
						   $result = $regs[1]." gofollow";
							$banner_output = str_ireplace('class="'.$regs[1].'"',
'class="'.$result.'"', $banner_output);
						} else {
							$banner_output = str_ireplace('<a ', '<a
class="gofollow" ', $banner_output);
						}
						unset($value, $regs, $result);
					}
				}
			}
		}

		$image = apply_filters('adsplugin_apply_photon', $image);

		$banner_output = str_replace('%title%', $name, $banner_output);
		$banner_output = str_replace('%random%', rand(100000,999999), $banner_output);
		$banner_output = str_replace('%asset%', $image, $banner_output); // Replaces %image%
		$banner_output = str_replace('%image%', $image, $banner_output); // Depreciated, remove
in AdsPlugin 5.0
		$banner_output = str_replace('%id%', $id, $banner_output);
		$banner_output = do_shortcode($banner_output);

		return $banner_output;
}
function adsplugin_impression_callback() {
		if(!defined('DONOTCACHEPAGE')) define('DONOTCACHEPAGE', true);
		if(!defined('DONOTCACHEDB')) define('DONOTCACHEDB', true);
		if(!defined('DONOTCACHEOBJECT')) define('DONOTCACHEOBJECT', true);

		$meta = $_POST['track'];
		$meta = base64_decode($meta);

		$meta = esc_attr($meta);
		// Don't use $impression_timer - It's for impressions used in javascript
		list($ad, $group, $blog_id, $impression_timer) = explode(",", $meta, 4);
		if(is_numeric($ad) AND is_numeric($group) AND is_numeric($blog_id)) {
			adsplugin_count_impression($ad, $group, $blog_id);
```

```
	}

	wp_die();
}
var el = wp.element.createElement,
	__ = wp.i18n.__,
	registerBlockType = wp.blocks.registerBlockType,
	RichText = wp.blocks.RichText,
	BlockBoxStyle = { 'box-sizing': 'border-box', position: 'relative', padding: '1em', 'min-
height': '20px', width: '100%', margin: '0', color: '#1e1e1e', 'border-radius': '2px', 'background-color':
'#f7f7f7', 'box-shadow': 'inset 0 0 0 1px #1e1e1e', outline: '1px solid transparent', 'background-
image': 'linear-gradient(to bottom right, #f7f7f7, #1fa4d1)' };

registerBlockType('adsplugin/advert', {
	title: __('AdsPlugin Advert', 'adsplugin'),
	icon: 'editor-code',
	category: 'custom-adsplugin',
	description: __('Show a single advert by entering an advert ID.', 'adsplugin'),
	keywords: ['ad', 'advert', 'adsplugin', 'banner', 'ads'],

	attributes: {
		advert_id: {
			type: 'string',
			selector: 'input',
		},
	},
	supports: {
		html: false,
	},

	edit: function( props ) {
	  function onChangeText( e ) {
			props.setAttributes( { advert_id: e.target.value } );
	  }

		if(isNaN(props.attributes.advert_id)) props.attributes.advert_id = 0;

		return el('div', {
			className: props.className + 'components-placeholder widefat',
			style: BlockBoxStyle,
		},
			el('div', {
				className: 'components-placeholder__label',
				style: { 'font-size': '18pt', 'font-weight': '400' },
			}, __('AdsPlugin Advert', 'adsplugin')),
			el('label', {
				className: 'components-placeholder__instructions group-' +
props.attributes.advert_id,
			}, __('Enter an Advert ID (numbers only):', 'adsplugin')),
			el('input', {
	    	className: 'components-text-control__input group-' + props.attributes.advert_id,
	    	onChange: onChangeText,
```

```
            value: Number(props.attributes.advert_id),
            isSelected: props.isSelected,
            style: { 'background-color': '#fefefe' }
        }),
                el('div', {
                    className: 'components-placeholder__instructions group-' +
props.attributes.group_id,
                    style: { 'font-size': '70%', 'font-style': 'italic' },
                }, __('You can find the advert ID in Manage Adverts. Any special markup,
code or layout styles can be applied in the advert itself or by placing the advert in a group.',
'adsplugin')),
            );
        },

    save: function( props ) {
        return null;
    },
} );

registerBlockType('adsplugin/group', {
    title: __('AdsPlugin Group', 'adsplugin'),
    icon: 'editor-code',
    category: 'custom-adsplugin',
    description: __('Show a group of adverts by entering a group ID.', 'adsplugin'),
    keywords: ['ad', 'advert', 'adsplugin', 'banner', 'group'],

        attributes: {
            group_id: {
                type: 'string',
                selector: 'input',
            },
        },
        supports: {
            html: false,
        },

    edit: function( props ) {
        function onChangeText( e ) {
            props.setAttributes( { group_id: e.target.value } );
        }

            if(isNaN(props.attributes.group_id)) props.attributes.group_id = 0;

            return el('div', {
                className: props.className + 'components-placeholder widefat',
                style: BlockBoxStyle,
            },
                el('div', {
                    className: 'components-placeholder__label',
                    style: { 'font-size': '18pt', 'font-weight': '400' },
                }, __('AdsPlugin Group', 'adsplugin')),
                el('label', {
```

```
                                        className: 'components-placeholder__instructions group-' +
props.attributes.group_id,
                                }, __('Enter a group ID (numbers only):', 'adsplugin')),
                                el('input', {
                    className: 'components-text-control__input group-' + props.attributes.group_id,
                    onChange: onChangeText,
                    value: Number(props.attributes.group_id),
                    isSelected: props.isSelected,
                    style: { 'background-color': '#fefefe' }
                }),
                                el('div', {
                                        className: 'components-placeholder__instructions group-' +
props.attributes.group_id,
                                        style: { 'font-size': '70%', 'font-style': 'italic' },
                                }, __('You can find the group ID in Manage Groups. Any special markup,
code or layout styles can be applied in the group wrapper when editing the group.', 'adsplugin')),
                        );
        },

        save: function( props ) {
            return null;
        },
} );
(function($) {
        $(document).ready(function() {
                $(document).on('click', 'a.gofollow', function(){
                        $.post(click_object.ajax_url,
{'action':'adsplugin_click','track':$(this).attr("data-track")});
                });
        });
}(jQuery));
(function($) {
        $(document).ready(function() {
                $('#startdate_picker').datepicker({dateFormat: 'dd-mm-yy'});
                $('#enddate_picker').datepicker({dateFormat: 'dd-mm-yy'});
        });
}(jQuery));

(function($) {
        $.fn.gslider = function(settings) {
                var config = {groupid:0,speed:3000};
                if(settings) $.extend(true, config, settings)

                this.each(function(i) {
                        var $cont = $(this);
                        var gallery = $(this).children();
                        var length = gallery.length;
                        var timer = 0;
                        var counter = 1;

                        if(length == 1) {
                                // Impression tracker (Single ad)
```

```
                var tracker = $cont.find(".c-1 a").attr("data-track");
                        if(typeof tracker !== 'undefined') {
                                impressiontracker(tracker);
                        }
                }

                if(length > 1) {
                        $cont.find(".c-1").show();
                        for(n = 2; n <= length; n++) {
                                $cont.find(".c-" + n).hide();
                        }

                        timer = setInterval(function(){ play(); }, config.speed);
                }

                function transitionTo(gallery, index) {
                        if((counter >= length) || (index >= length)) {
                                counter = 1;
                        } else {
                                counter++;
                        }
                        $cont.find(".c-" + counter).show();

                        // Impression tracker (Multiple ads)
                var tracker = $cont.find(".c-" + counter + ' a').attr("data-track");
                        if(typeof tracker !== 'undefined') {
                                impressiontracker(tracker);
                        }
                        $cont.find(".c-" + index).hide();
                }

                function play() {
                        transitionTo(gallery, counter);
                }

                function impressiontracker(tracker) {
                admeta = atob(tracker).split(',');
                        var name = escape('adsplugin-'+admeta[0]);
                        var now = Math.round(Date.now()/1000);
                        var expired = now - admeta[3];
                        var session = sessionStorage.getItem(name); // Get session data

                        if(session == null) { // New session, no previous data
                                session = 0;
                        }

                        if(session <= expired) { // Count new impression?
                                $.post(impression_object.ajax_url, {'action':
'adsplugin_impression','track': tracker});
                           sessionStorage.setItem(name, now);
                                delete tracker;
                        }
```

```
                        }
                });
                return this;
        };
}(jQuery));

jQuery(function() {
        jQuery("table.manage-ads-main").tablesorter({
                headers: {
                        4: { sorter: false },
                        6: { sorter: false },
                        7: { sorter: false },
                }
        });
        jQuery("table.manage-ads-disabled").tablesorter({
                headers: {
                        1: { sorter: false },
                        3: { sorter: false },
                        4: { sorter: false },
                        5: { sorter: false },
                }
        });
        jQuery("table.manage-groups-main").tablesorter({
                headers: {
                        2: { sorter: false },
                        3: { sorter: false },
                        4: { sorter: false },
                        5: { sorter: false },
                        6: { sorter: false },
                }
        });
});

function textatcursor(areaId,text) {
        var txtarea = document.getElementById(areaId);
        var scrollPos = txtarea.scrollTop;
        var strPos = 0;
        var br = ((txtarea.selectionStart || txtarea.selectionStart == '0') ? "ff" : (document.selection ?
"ie" : false ) );

        if(br == "ie") {
                txtarea.focus();
                var range = document.selection.createRange();
                range.moveStart ('character', -txtarea.value.length);
                strPos = range.text.length;
        } else if(br == "ff") {
                strPos = txtarea.selectionStart;
        }

        var front = (txtarea.value).substring(0,strPos);
        var back = (txtarea.value).substring(strPos,txtarea.value.length);
        txtarea.value=front+text+back;
```

```
		strPos = strPos + text.length;
		if(br == "ie") {
			txtarea.focus();
			var range = document.selection.createRange();
			range.moveStart ('character', -txtarea.value.length);
			range.moveStart ('character', strPos);
			range.moveEnd ('character', 0);
			range.select();
		} else if(br == "ff") {
			txtarea.selectionStart = strPos;
			txtarea.selectionEnd = strPos;
			txtarea.focus();
		}
		txtarea.scrollTop = scrollPos;
}
jQuery(document).ready(function(){
		var custom_uploader;
		jQuery('#adsplugin_image_button').click(function(e) {
			e.preventDefault();
			if(custom_uploader) {
				custom_uploader.open();
				return;
			}
			custom_uploader = wp.media.frames.file_frame = wp.media({title: 'Choose
Banner',button: {text: 'Choose Banner'},multiple: false});
			custom_uploader.on('select', function() {
				attachment = custom_uploader.state().get('selection').first().toJSON();
				jQuery('#adsplugin_image').val(attachment.url);
			});
			custom_uploader.open();
		});
});
<?php

function adsplugin_register_blocks() {
   wp_register_script('adsplugin-block', plugins_url('/library/block.js', __FILE__), array('wp-
blocks', 'wp-element', 'wp-i18n'));

   register_block_type('adsplugin/advert', array('editor_script' => 'adsplugin-block',
'render_callback' => 'adsplugin_advert_block'));
   register_block_type('adsplugin/group', array('editor_script' => 'adsplugin-block', 'render_callback'
=> 'adsplugin_group_block'));
}
add_action('init', 'adsplugin_register_blocks');

function adsplugin_advert_block($attr) {
		global $adsplugin_config;

   if(!isset($attr['advert_id']) OR !is_numeric($attr['advert_id'])) return;

		$output = '';
		if($adsplugin_config['w3caching'] == "Y") {
```

```
                $output .= '<!-- mfunc '.W3TC_DYNAMIC_SECURITY.' -->';
                $output .= 'echo adsplugin_ad('.$attr['advert_id'].');';
                $output .= '<!-- /mfunc '.W3TC_DYNAMIC_SECURITY.' -->';
        } else if($adsplugin_config['borlabscache'] == "Y" AND
function_exists('BorlabsCacheHelper')) {
                if(BorlabsCacheHelper()->willFragmentCachingPerform()) {
                        $borlabsphrase = BorlabsCacheHelper()->getFragmentCachingPhrase();

                        $output .= '<!--[borlabs cache start: '.$borlabsphrase.']--> ';
                        $output .= 'echo adsplugin_ad('.$attr['advert_id'].');';
                        $output .= ' <!--[borlabs cache end: '.$borlabsphrase.']-->';

                        unset($borlabsphrase);
                }
        } else {
                $output .= adsplugin_ad($attr['advert_id']);
        }

        return $output;
}

function adsplugin_group_block($attr) {
        global $adsplugin_config;

   if(!isset($attr['group_id']) OR !is_numeric($attr['group_id'])) return;

        $output = '';
        if($adsplugin_config['w3caching'] == "Y") {
                $output .= '<!-- mfunc '.W3TC_DYNAMIC_SECURITY.' -->';
                $output .= 'echo adsplugin_group('.$attr['group_id'].');';
                $output .= '<!-- /mfunc '.W3TC_DYNAMIC_SECURITY.' -->';
        } else if($adsplugin_config['borlabscache'] == "Y" AND
function_exists('BorlabsCacheHelper')) {
                if(BorlabsCacheHelper()->willFragmentCachingPerform()) {
                        $borlabsphrase = BorlabsCacheHelper()->getFragmentCachingPhrase();

                        $output .= '<!--[borlabs cache start: '.$borlabsphrase.']--> ';
                        $output .= 'echo adsplugin_group('.$attr['group_id'].');';
                        $output .= ' <!--[borlabs cache end: '.$borlabsphrase.']-->';

                        unset($borlabsphrase);
                }
        } else {
                $output .= adsplugin_group($attr['group_id']);
        }

        return $output;
}

function adsplugin_add_block_category($categories, $editor_context) {
        array_unshift($categories, array(
                'slug'    => 'custom-adsplugin',
```

```
                'title' => __('AdsPlugin - Advertisements', 'adsplugin-pro'),
                'icon'  => null,
        ));

        return $categories;
}
add_filter('block_categories_all', 'adsplugin_add_block_category', 10, 2);
function adsplugin_click_callback() {
        if(!defined('DONOTCACHEPAGE')) define('DONOTCACHEPAGE', true);
        if(!defined('DONOTCACHEDB')) define('DONOTCACHEDB', true);
        if(!defined('DONOTCACHEOBJECT')) define('DONOTCACHEOBJECT', true);

        global $wpdb, $adsplugin_config;

        $meta = $_POST['track'];
        $meta = base64_decode($meta);

        $meta = esc_attr($meta);
        // Don't use $impression_timer - It's for impressions used in javascript
        list($ad, $group, $blog_id, $impression_timer) = explode(",", $meta, 4);

        if(is_numeric($ad) AND is_numeric($group) AND is_numeric($blog_id)) {
                if($blog_id > 0 AND adsplugin_is_networked()) {
                        $current_blog = $wpdb->blogid;
                        switch_to_blog($blog_id);
                }

                $remote_ip = adsplugin_get_remote_ip();

                if(adsplugin_is_human() AND $remote_ip != "unknown" AND !empty($remote_ip))
{
                        $now = current_time('timestamp');
                        $today = adsplugin_date_start('day');
                        $click_timer = $now - $adsplugin_config['click_timer'];

                        $saved_timer = $wpdb->get_var($wpdb->prepare("SELECT `timer` FROM
`{$wpdb->prefix}adsplugin_tracker` WHERE `ipaddress` = '%s' AND `stat` = 'c' AND `bannerid`
= %d ORDER BY `timer` DESC LIMIT 1;", $remote_ip, $ad));
                        if($saved_timer < $click_timer) {
                                $stats = $wpdb->get_var($wpdb->prepare("SELECT `id` FROM
`{$wpdb->prefix}adsplugin_stats` WHERE `ad` = %d AND `group` = %d AND `thetime` =
{$today};", $ad, $group));
                                if($stats > 0) {
                                        $wpdb->query("UPDATE `{$wpdb->prefix}adsplugin_stats`
SET `clicks` = `clicks` + 1 WHERE `id` = {$stats};");
                                } else {
                                        $wpdb->insert($wpdb->prefix.'adsplugin_stats', array('ad' =>
$ad, 'group' => $group, 'thetime' => $today, 'clicks' => 1, 'impressions' => 1));
                                }

                                $wpdb->insert($wpdb->prefix.'adsplugin_tracker', array('ipaddress' =>
$remote_ip, 'timer' => $now, 'bannerid' => $ad, 'stat' => 'c'));
```

```
                        }

                        // Advertising budget
                        $wpdb->query("UPDATE `{$wpdb->prefix}adsplugin` SET `budget` =
`budget` - `crate` WHERE `id` = {$ad} AND `crate` > 0;");
                }

                if($blog_id > 0 AND adsplugin_is_networked()) {
                        switch_to_blog($current_blog);
                }

                unset($remote_ip, $track, $meta, $ad, $group, $remote, $banner);
        }

        wp_die();
}
```

Додаток В (обов'язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ З ПРОЕКТУВАННЯ WEB-РЕСУРСІВ. ЧАСТИНА 1. КЛІЄНТСЬКА ЧАСТИНА

Виконав: студент 2-го курсу,
групи 2КН-22м
спеціальності 122 «Комп'ютерні науки»
(шифр і назва напряму підготовки, спеціальності)
Сторожук А.С.
(прізвище та ініціали)

Керівник: к.т.н., доцент каф. КН
Сілагін О.В.
(прізвище та ініціали)
«07. 12»__2022 р.

Вінниця ВНТУ - 2023 рік

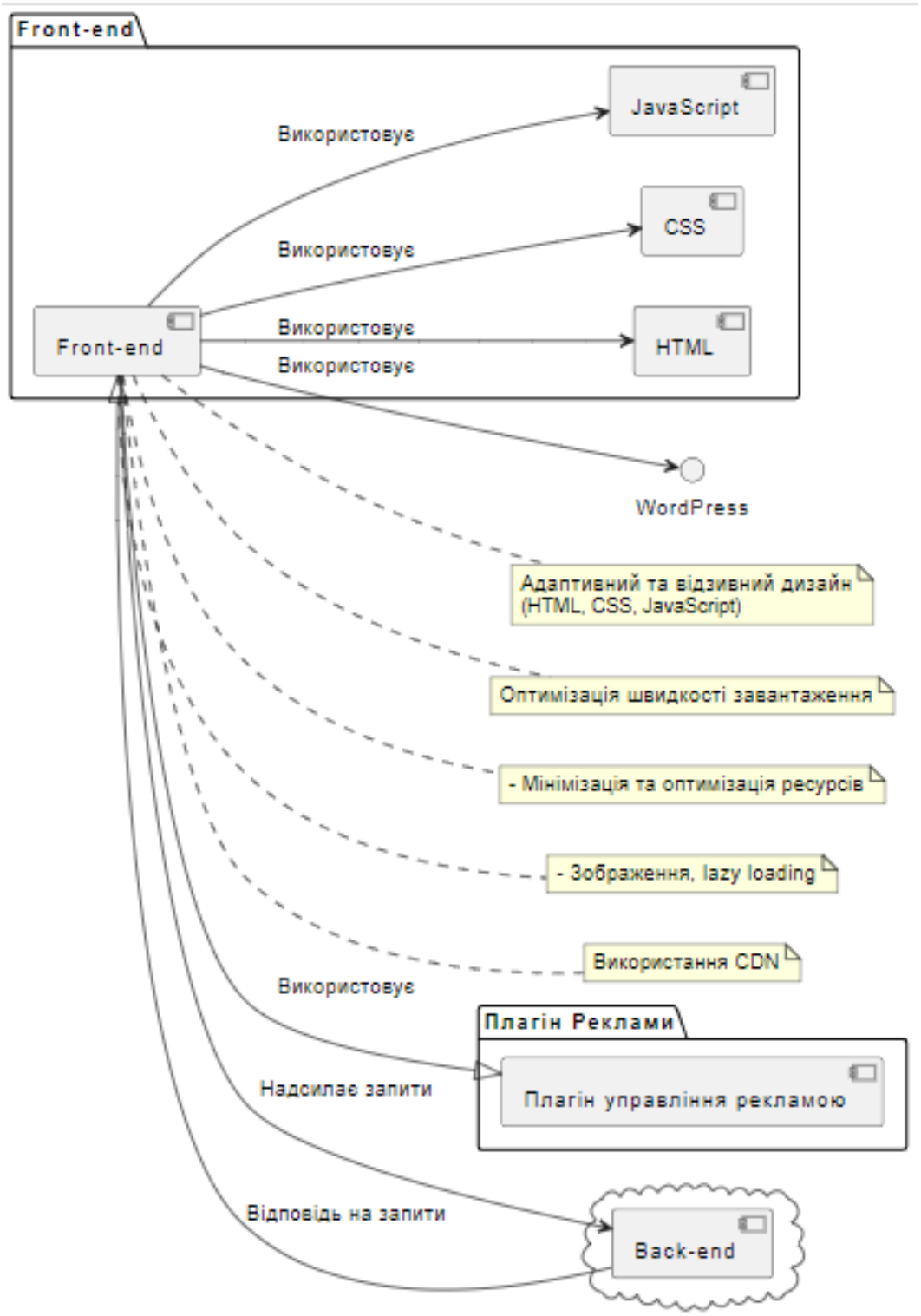

Рисунок В.1 – Схема покращеної технології проектування клієнтської частини WEB-ресурсів з використанням плагіна для управління рекламою

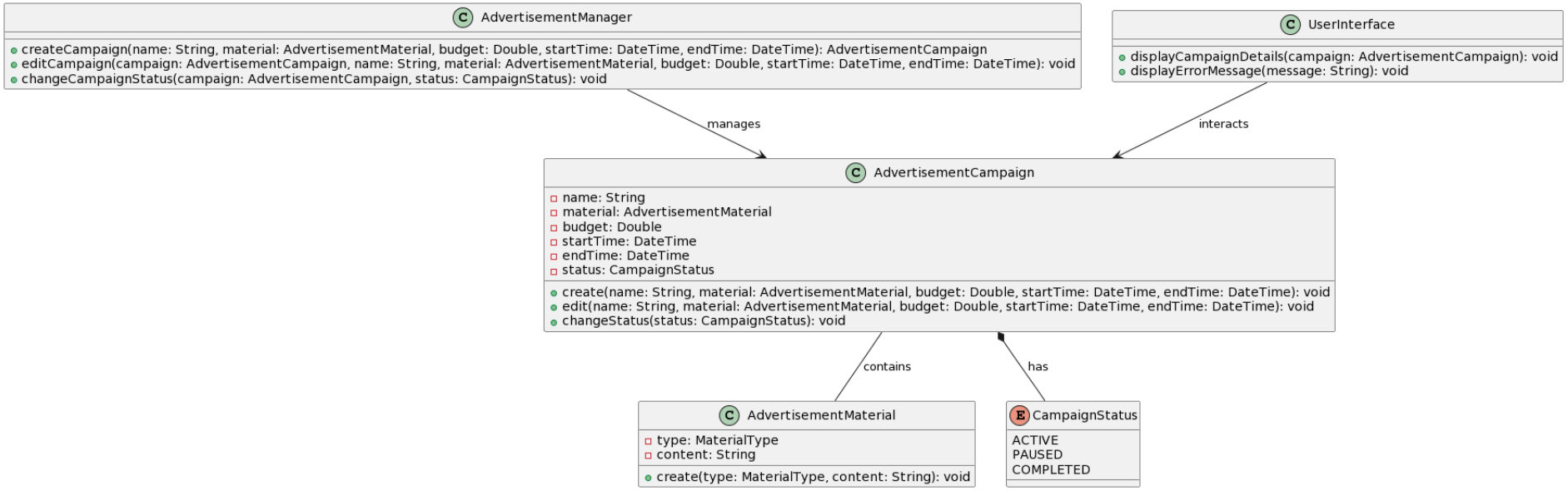

Рисунок В.2 – Діаграма класів

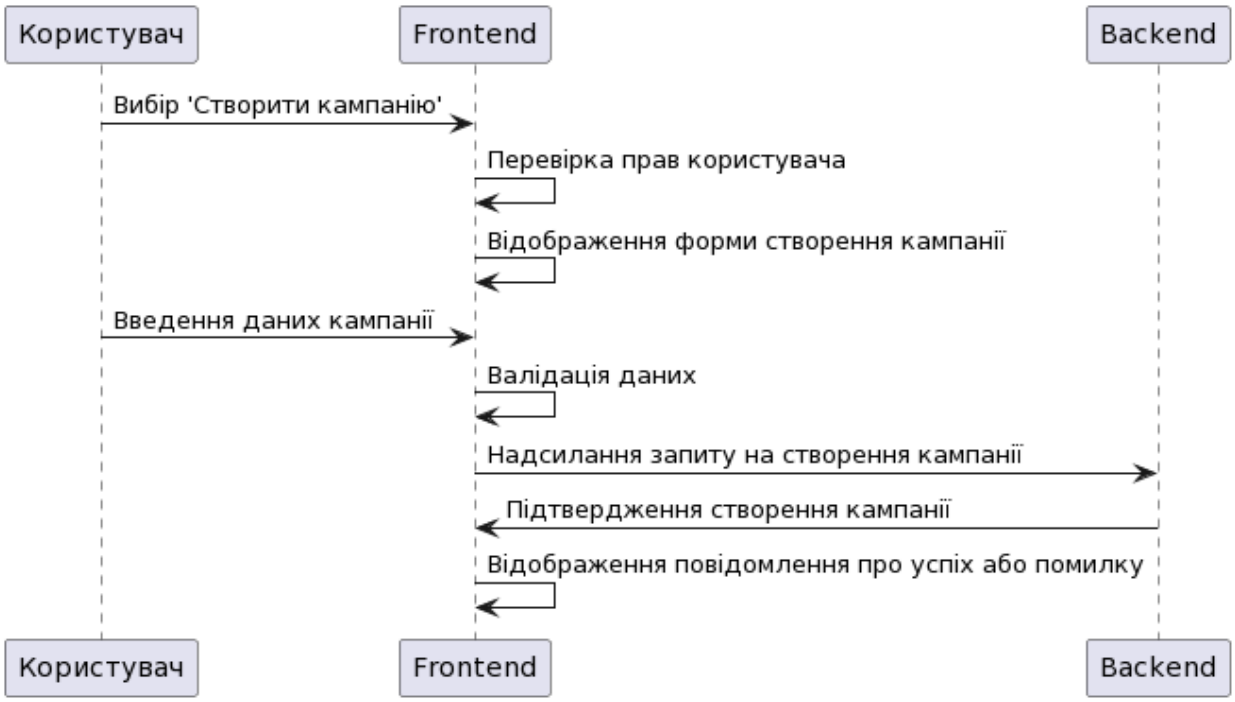

Рисунок В.3 – Діаграма послідовності

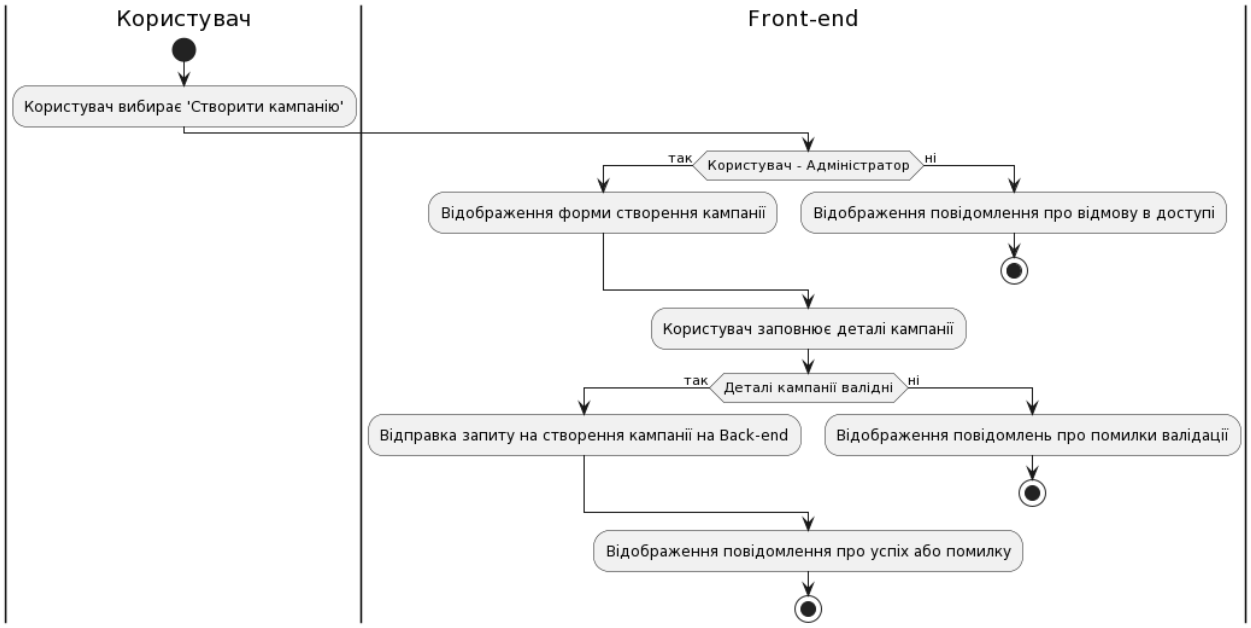

Рисунок В.4 – Діаграма активностей

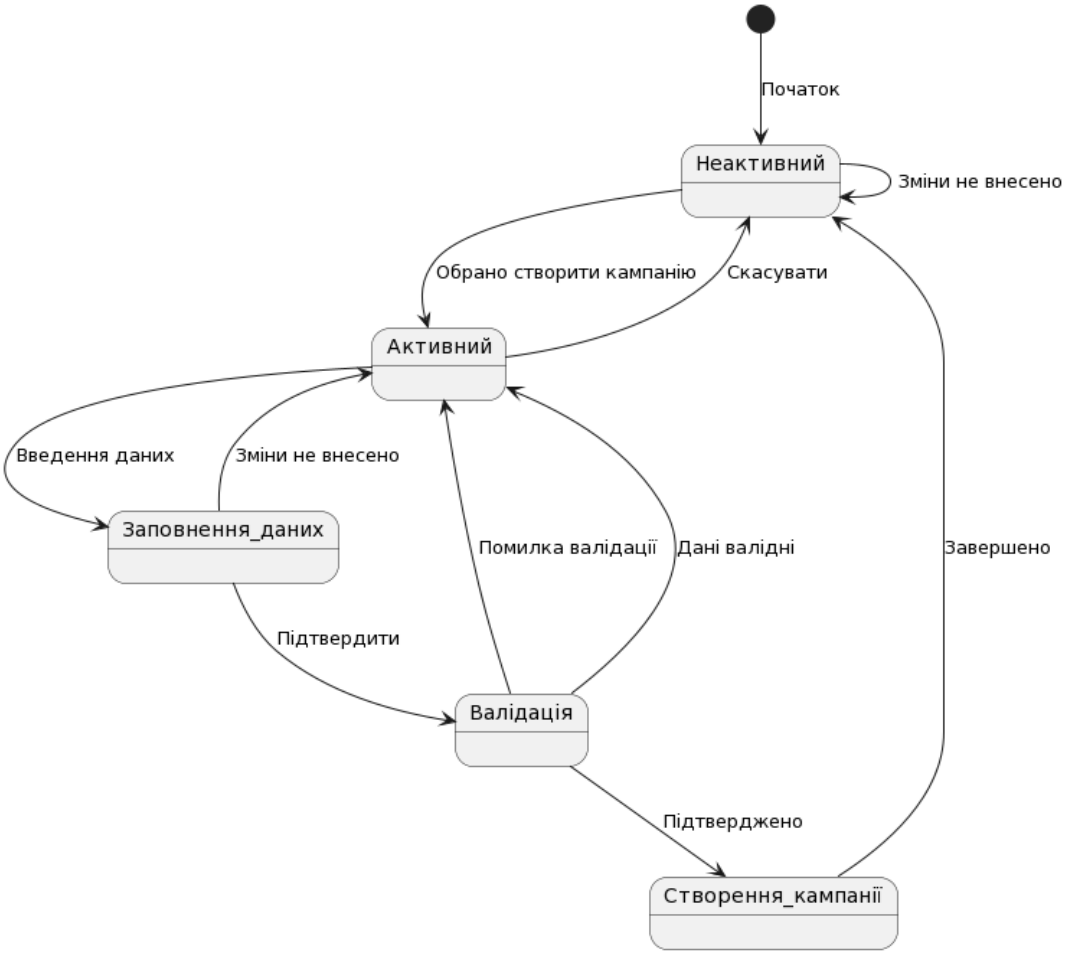

Рисунок В.5 – Діаграма станів

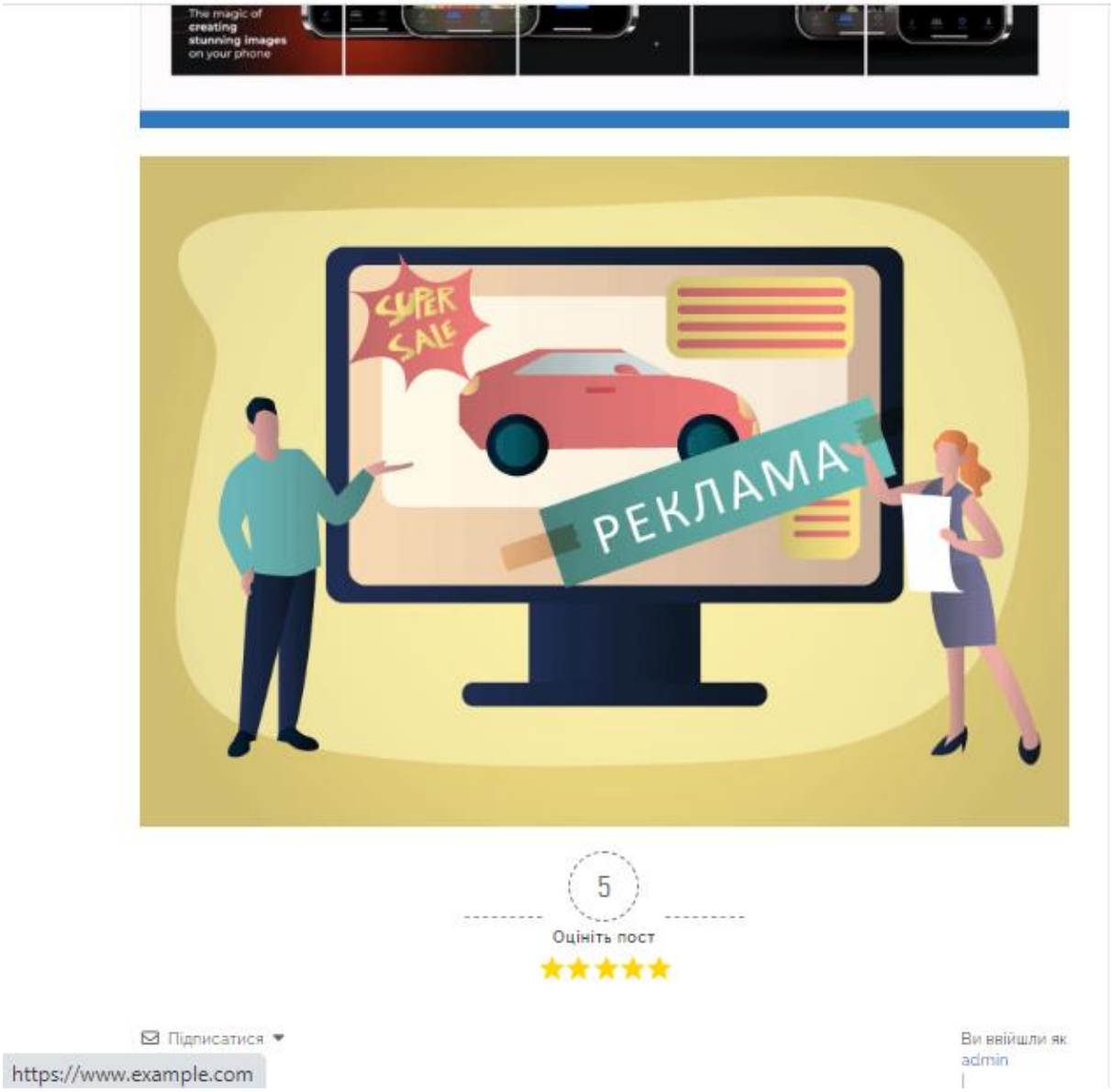

Рисунок В.6 – Рекласний банер на сторінці веб-ресурсу

Додаток Г (довідниковий)

Інструкція користувача

Розглянемо етапи створення нового рекламного блоку:

1. Спочатку потрібно обрати пункт «Manage Adverts» серед сторінок плагіна, це зображено на рисунку Г.1.

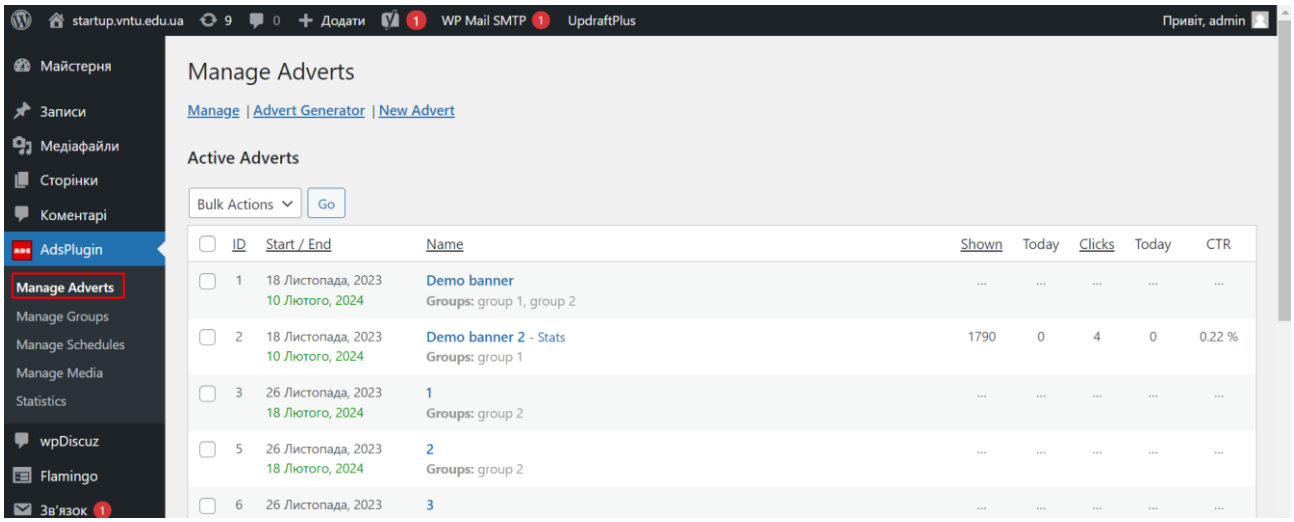

Рисунок Г.1 – Зображення сторінки «Manage Adverts»

2. Щоб створити новий рекламний банер потрібно перейти за одним з посилань «Advert Generator» і «New Advert», вони показані на рисунку Г.2.

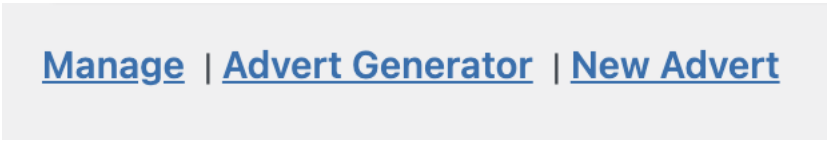

Рисунок Г.2 – Розділи сторінки «Manage Adverts»

3. Натискаємо на будь-який готовий рекламний банер та відкриваємо сторінку редагування чи створення банера. Зображення цієї сторінки показане на рисунку Г.3.

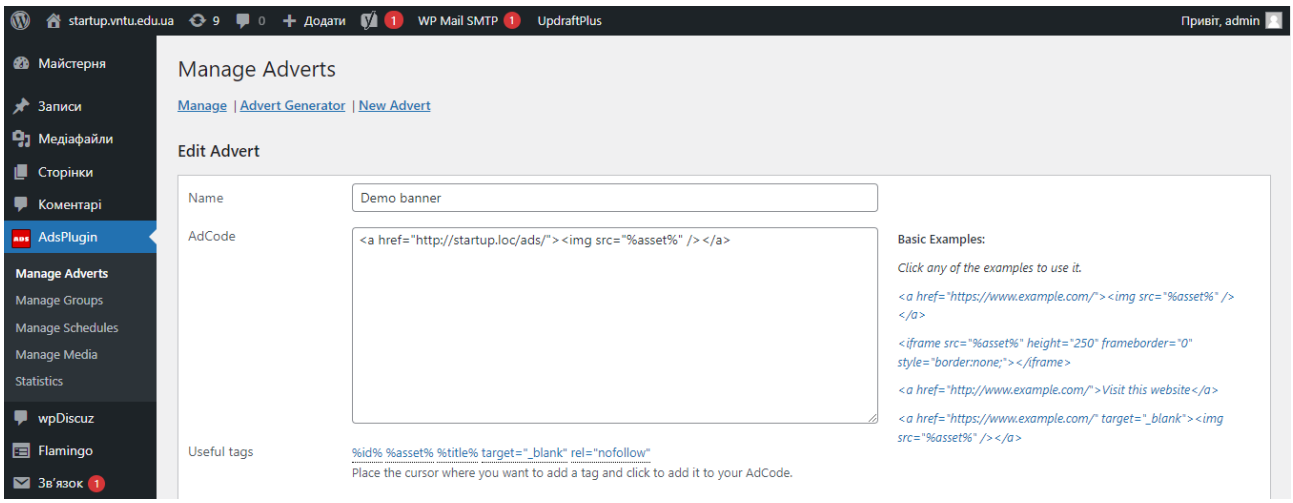

Рисунок Г.3 – Редагування рекламного банера

3.1. Оберемо зображення рекламного блоку використовуючи поле «Banner asset» та додамо посилання, використовуючи поле «AdCode». Після цього потрібно натиснути на кнопку «Save Advert». Поле для зображення зображено на рисунку Г.4.

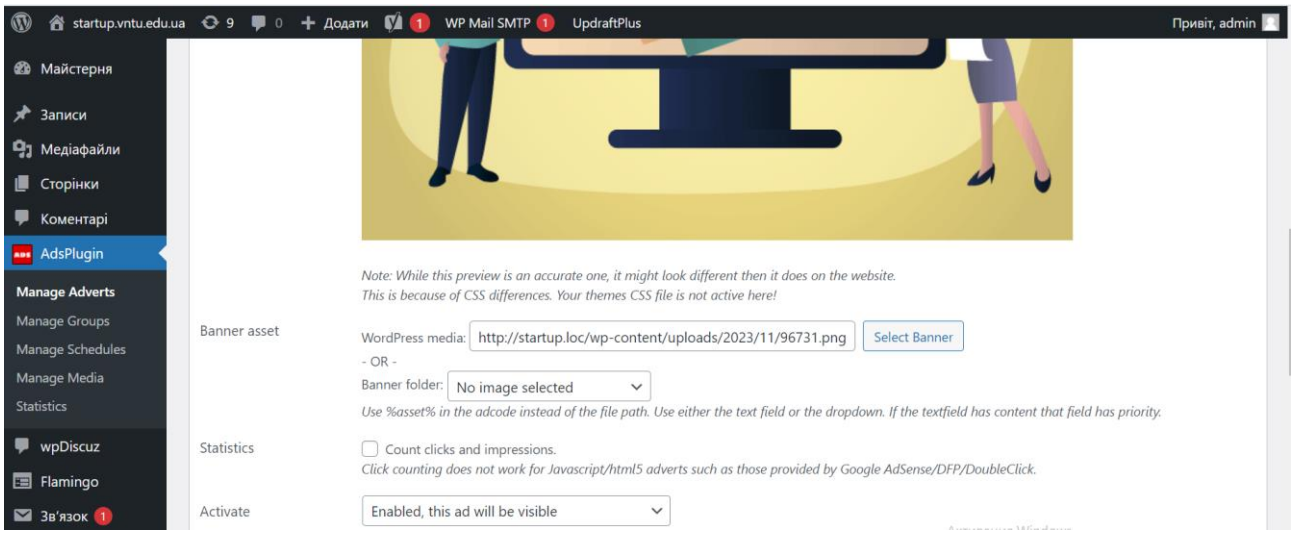

Рисунок Г.4 – Поле для зображення

3.2. Після цього відкриється будь-яка сторінка чи пост та вставляється шорткод, це зображено на рисунку Г.5.

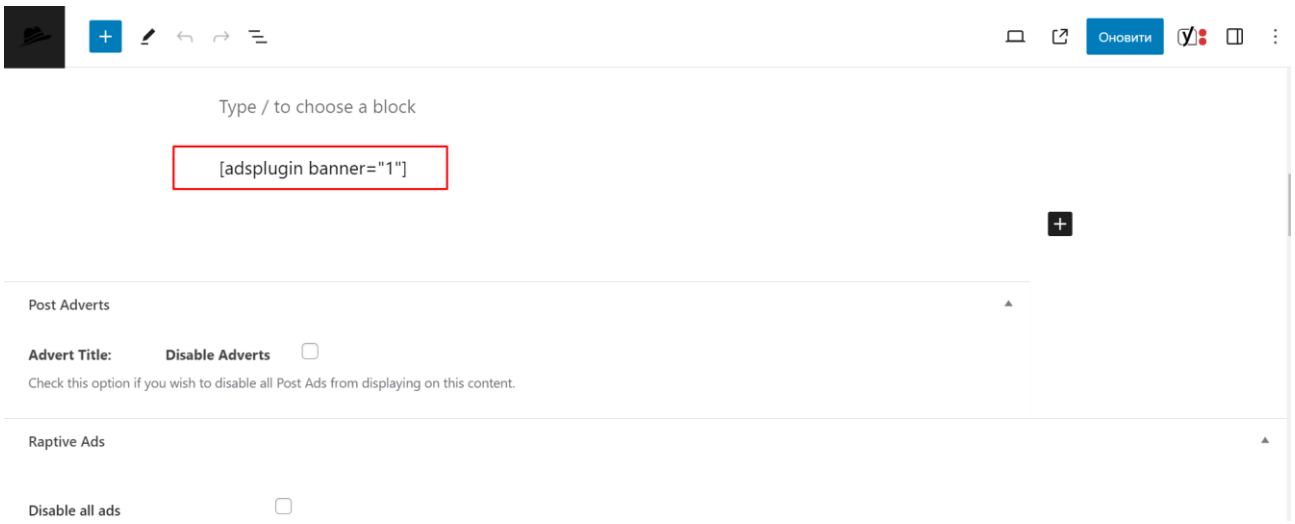

Рисунок Г.5 – Інтегрування шорткоду в пост блогу

4. Перейшовши на цей пост чи сторінку можна перевірити наявність рекламного банера, це продемонстровано на рисунку Г.6.

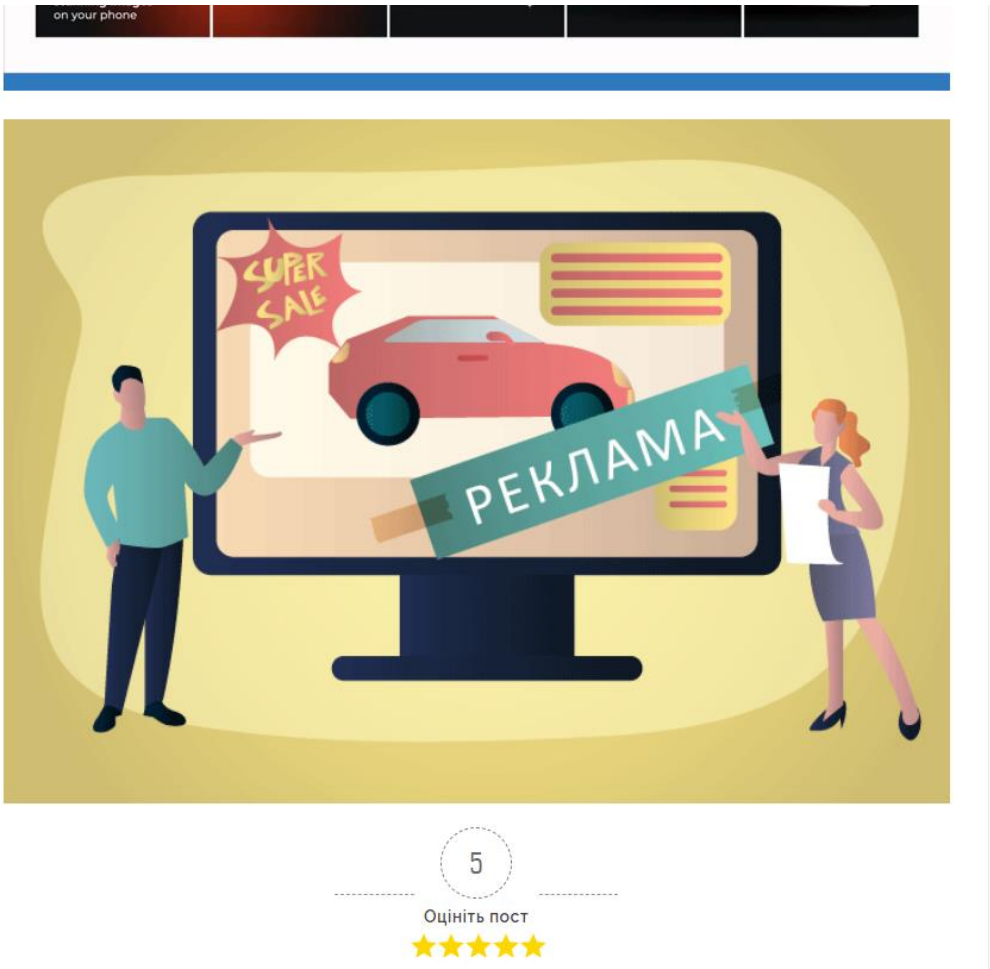

Рисунок Г.6 – Відображення рекламного банера в пості

5. Після цього можна натиснути на рекламний банер та відкрити рекламу і переглянути збільшення опрацьованих кліків в розділі статистики в адміністративній панелі, цей розділ зображено на рисунку Г.7.

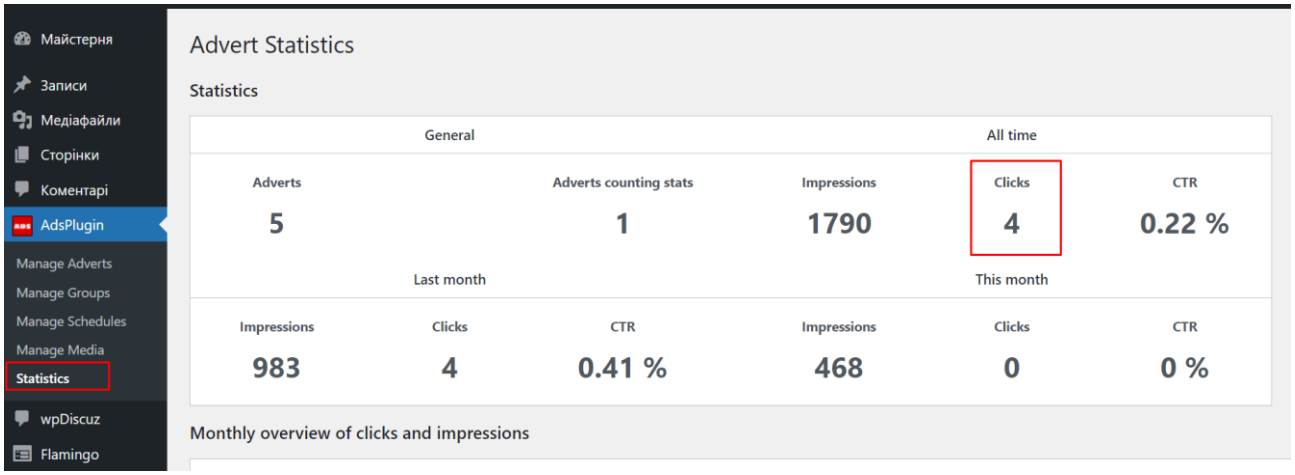

Рисунок Г.7 – Розділ статистики

Додаток Д (довідниковий)

Довідка про провадження

ДОВІДКА

Дана Сторожоку Антону Сергійовичу в тому що результати, одержані ним в процесі виконання магістерської кваліфікаційної роботи, а саме розроблений плагін для управління рекламою на веб-ресурсах, планується використати для стартап школи Sikorsky Challenge м. Вінниці.

Очільник стартап школи Барабан С.В.