

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування інституту, факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНА РОЗРОБКА ПЛАГІНУ CMS WORDPRESS ДЛЯ
РОБОТИ З ГАЛЕРЕЄЮ
SOFTWARE DEVELOPMENT OF THE CMS WORDPRESS PLUGIN TO
WORK WITH A GALLERY

Виконав: студент(ка) 4 курсу, групи КНТ-210сп
Спеціальності 122 Комп'ютерні науки

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерні науки

ТИХОНОВСЬКА Т.В.

(прізвище та ініціали)

Керівник СТЕПАНЕНКО О.О.

(прізвище та ініціали)

Рецензент ГОЛУБ Т.В.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут, факультет ФКНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 122 Комп'ютерні науки
(код і найменування)
Освітня програма (спеціалізація) Комп'ютерні науки
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
“ ” 2023 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ТИХОНОВСЬКОЇ Тетяни Володимирівни

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Програмна розробка плагіну CMS WordPress для роботи з галереєю. Software Development of the CMS WordPress Plugin to Work with a Gallery

керівник проєкту (роботи) к.т.н., доцент, СТЕПАНЕНКО Олександр Олексійович

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “4” квітня 2023 року № 87

2. Строк подання студентом проєкту (роботи) 1 червня 2023 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Розробка архітектури програми. 3. Розробка програми. 4. Експлуатація та тестування програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	СТЕПАНЕНКО О.О., доцент		
Нормоконтроль	БСЛОВА А.В., асистент		

7. Дата видачі завдання « 04 » квітня 2023 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	2 тиждень	Розділ 1
3	Розробка архітектури плагіну.	3 тиждень	Розділ 2
4	Розробка плагіну.	4-5 тижні	Розділ 3
5	Тестування та експериментальне дослідження плагіну.	6 тиждень	Розділ 4
6	Оформлення пояснювальної записки та документів до неї.	7 тиждень	Додатки
7	Захист роботи.	8 тиждень	

Студент(ка)

(підпис) Тетяна ТИХОНОВСЬКА
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис) Олександр СТЕПАНЕНКО
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 97 с., 40 рис., 1 табл., 3 дод., 11 джерел.

ГАЛЕРЕЯ, ПЛАГІН, ІМПОРТ, DROPBOX, GOOGLE DRIVE, WORDPRESS.

Об'єкт роботи – плагін CMS WordPress для роботи з галереєю.

Предмет роботи – плагін для роботи з галереєю у CMS WordPress, що також надає можливість імпорту зображень з Dropbox та Google Drive. Мета роботи – розробка плагіну для забезпечення роботи з галереєю у CMS WordPress, що включає створення самої галереї, її додавання на сторінку чи пост та імпорту зображень з Dropbox та Google Drive.

Матеріали, методи та технічні засоби: об'єктно-орієнтоване програмування, мови PHP, HTML, CSS, jQuery, система керування базами даних MySQL, система керування вмістом сайту WordPress.

Результат. Розробку плагіну роботи з галереєю у CMS WordPress виконано з визначення вимог до програми, розробленням структури і алгоритму плагіну на основі цих вимог, реалізацією відповідних одиниць. Плагін дозволяє створювати галерею зображень, редагувати її, видаляти, додавати на сторінку чи пост, імпортувати зображення з Dropbox та Google Drive, для їх використання під час створення галереї чи її редагування.

Висновки. Розроблений плагін дозволяє підвищити зручність зі створення галереї та роботою з нею.

Галузь використання. Плагін може використовуватися у CMS WordPress.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 97 pages, 40 figures, 1 tables, 3 appendixes, 11 sources.

GALLERY, PLUGIN, IMPORT, DROPBOX, GOOGLE DRIVE, WORDPRESS.

The object of work is a CMS WordPress plugin for working with a gallery.

The subject of work is a plugin for working with a gallery in CMS WordPress, which also provides the ability to import images from Dropbox and Google Drive. Purpose - to develop a plugin for working with a gallery in CMS WordPress, including creating the gallery itself, adding it to a page or post, and importing images from Dropbox and Google Drive.

Materials, methods and techniques: object-oriented programming, PHP, HTML, CSS, jQuery, MySQL database management system, WordPress content management system.

Result. The development of the gallery plugin in the WordPress CMS was carried out by defining the requirements for the program, developing the structure and algorithm of the plugin based on these requirements, and implementing the corresponding units. The plugin allows you to create a gallery of images, edit it, delete it, add it to a page or post, import images from Dropbox and Google Drive to use them when creating or editing a gallery.

Conclusions. The developed plugin allows you to increase the convenience of creating a gallery and working with it.

Field of application. The plugin can be used in CMS WordPress.

ЗМІСТ

	С.
Вступ.....	8
1 Аналіз предметної області.....	9
1.1 Використання галерей на вебсторінках	9
1.2 Огляд програмних аналогів.....	10
1.3 Висновки за розділом 1	14
2 Розробка архітектури програми.....	15
2.1 Вибір засобів проектування і розробки	15
2.2 Визначення функціональних вимог до плагіну	16
2.3 Взаємодія з базою даних	19
2.4 Визначення структури програми.....	20
2.5 Висновки за розділом 2	21
3 Розробка програми	22
3.1 Алгоритм функціонування програми.....	22
3.2 Опис реалізованих одиниць вебзастосунку.....	25
3.3 Висновки за розділом 3	28
4 Експлуатація та тестування плагіну.....	29
4.1 Опис плагіну	29
4.2 Експлуатація програми.....	29
4.3 Тестування програми	38
4.4 Висновки за розділом 4	39
Висновки	40
Перелік джерел посилання	41
Додаток А Технічне завдання	42
Додаток Б Текст програми	45
Додаток В Слайди презентації.....	89

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

API – Application Programming Interface;
AJAX – Asynchronous Javascript and XML;
CMS – Content Management System;
CSS – Cascading Style Sheets,
HTML – Hyper Text Markup Language;
REST – Representational State Transfer;
PHP – Hypertext Preprocessor.

ВСТУП

На сьогоднішній день, кожен має доступ до мережі у будь-який час і у будь-якому місці, що надає вільний доступ до цифрової інформації – це змушує мас-медіа переходити в електронний формат. Вони намагаються йти у ногу з часом, адже відсоток інформації у цифровому вираженні, яка публікується, щороку перевищує кількість друкованих примірників та означає втрату значної частину ринку. Тому у цьому їм допомагають вільні програмні засоби, які можливо використовувати для створення сайту, блогу чи вебдодатку. Одним з таких програмних засобів є WordPress, що надає широкий набір функціоналу для публікації інформації та її оформлення.

Жодне видання не може обійтися без публікацій фотографій, WordPress дозволяє легко додавати зображень на сторінки сайту чи блогу. Для організації великої кількості фотографії на сторінці, WordPress має вбудовану функцію галереї для її швидкого створення. Але також можливо використовувати плагіни у WordPress, що є більш надійними і надають ширший функціонал при їх створенні, адже стандартного набору інструментів може бути не достатньо аби досягти бажаного результату на сторінці.

Зважаючи на відсоток сайтів, що працюють на WordPress і який сягає 43% від усіх сайтів світу [1], не дивно, що існує велика кількість плагінів для роботи з фотографіями. Плагін - це інструменти, за допомогою яких можна додавати функції WordPress [2]. Кожен ж них знаходить свого користувача в залежності від потреб, адже кожен пропонує щось своє і імпорт зображень з хмарного сховища, наприклад Dropbox або Google Drive, не стандартною можливістю у WordPress, та й не всі плагіни її мають. Саме на надання такого функціоналу спрямована дана робота. Метою роботи було визначено розробку плагіну галереї з можливістю імпорту зображень з Dropbox або Google Drive до галереї WordPress.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Використання галерей на вебсторінках

Зазвичай зображення розміщуються на сторінці один за одним, або поміщаються у нерівномірно розподілені ряди. Це не тільки погіршує естетику сайту, а й змушує користувача виконувати більшу кількість дій, наприклад – прокручування сторінки, для їх перегляду (рис. 1.1).

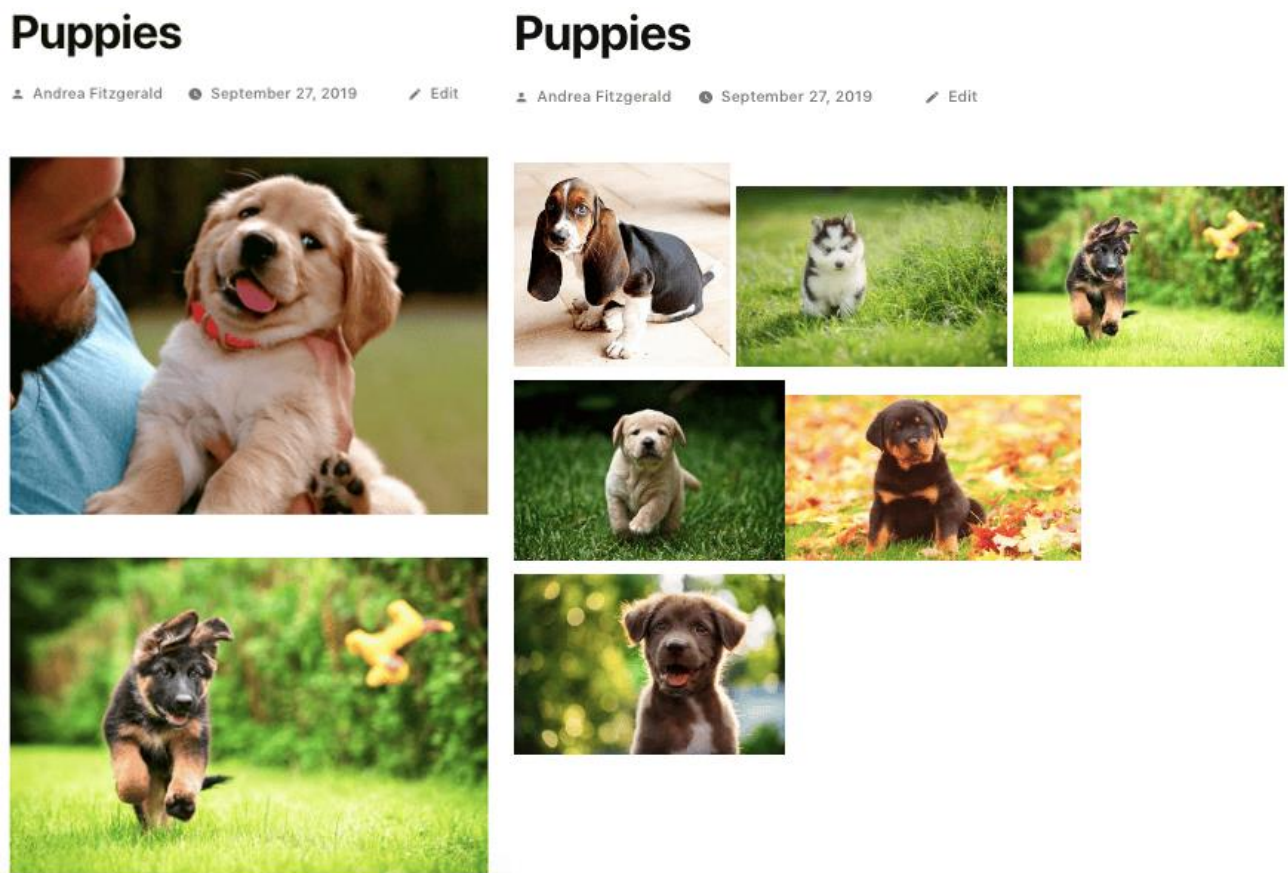


Рисунок 1.1 – Стандартний вигляд зображень на сторінці [3]

Використання галерея зображень дозволяє додати фотографії на сторінку та відображати їх більшу кількість на меншому просторі, а також спрощує перегляд для користувача. Вони вирішують проблеми розміщення, зовнішнього вигляду та загалом є більш гнучкими у використанні (рис. 1.2).

Puppies

Andrea Fitzgerald

September 27, 2019

Edit

Ut a molestie tortor. Morbi sed ligula et tortor imperdiet venenatis ut sit amet dui. Fusce elementum purus quis egestas tempor.



Too cute.



THESE EARS.



Too cute for words.



Donec mattis ex id nisi efficitur, quis venenatis eros maximus. Maecenas sit amet justo nisi. Nunc eget pellentesque nisi. Nunc et justo id massa tempor

Рисунок 1.2 – Вигляд зображень з використанням галереї [3]

1.2 Огляд програмних аналогів

Оскільки проблеми розміщення і відображення фотографій на сторінці не могли бути непоміченими, то звичайно існує велика кількість засобів, що дозволяє створити їх публікацію більш привабливою та організованою. У даному підрозділі такі рішення будуть розглянуті як аналоги і проаналізовані з урахуванням можливостей які вони надають користувачу.

Першим програмним аналог є плагін Envira Gallery (рис. 1.3).

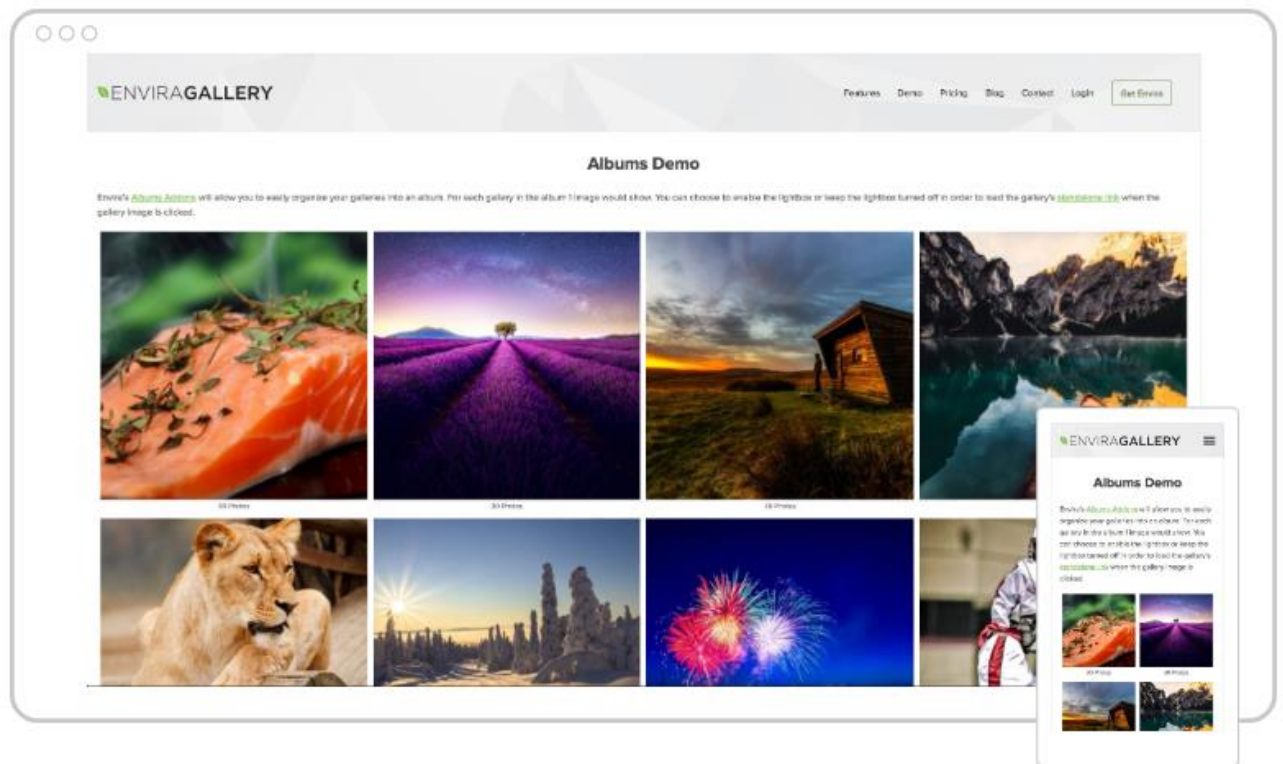


Рисунок 1.3 – Приклад галереї з використанням Envira Gallery [4]

Плагін Envira Gallery надає такі можливості користувачу:

- необмежена кількість галерей;
- збільшені зображення у лайтбоксах;
- доповнення імпортера Dropbox - імпорт фотографій з облікового запису Dropbox у галерею;
- слайд-шоу доповнення;
- готовий дизайн - готові шаблони, щоб настроїти зовнішній вигляд галереї;
- автономні галереї - незалежні галереї, не прив'язані до постів чи сторінок;
- чутливий дизайн - адаптивні галереї, які працюють на мобільних, планшетних та настільних пристроях.

Це лише частина того, що може отримати користувач, адже плагін не є безкоштовним та не має вільної версії хоча б з мінімальним набором для створення простої галереї, що є значним мінусом для його використання. Також додатковим мінусом є відсутність імпорту зображень з Google Drive.

Наступним аналогом є плагін Go Portfolio (рис. 1.4). Він також не безкоштовний, але коштує значно менше ніж попередній, а також є більш простим у використанні.

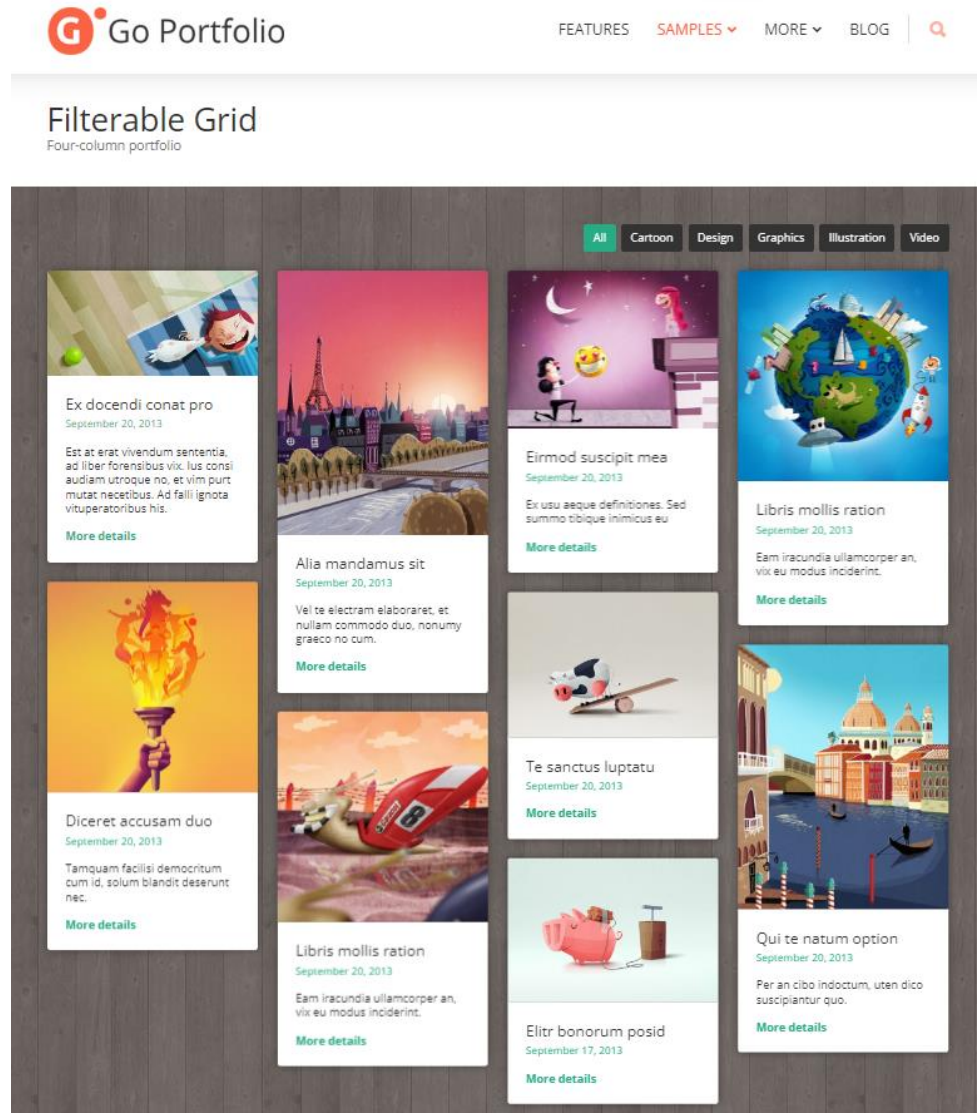


Рисунок 1.4 – Приклад галереї з використанням Go Portfolio [5]

Серед його функцій можна виділити наступні:

- сумісність із будь-якою темою WordPress;
- чуйна сітка або макет із горизонтальним слайдером;
- підтримка багатьох типів;
- працює з будь-яким типом постів (включаючи типи користувачів)

і ви можете створити необмежену кількість їх типів.

Але слід зазначити, що він не додає можливості імпорту зображень Dropbox або Google Drive, що вимагає їх окремого завантаження і безперечно є мінусом плагіну.

Останнім аналогом є Robo Gallery (рис. 1.5), що має безкоштовну версію. Його ключовими особливостями є:

- немає обмежень на кількість зображень/фотографій у кожному альбомі;
- повністю адаптивні та мобільні функції – плагін реалізований з розширеними налаштуваннями для різних розмірів екрана пристроїв;
- 15 ефектів при наведенні – всі ефекти при наведенні працюють у поєднанні з параметрами конфігурації інтерфейсу;
- класичний макет – макет може мати класичний стиль або макет сітки, і кожен елемент на сторінці може мати свої власні стилі та налаштування;
- індивідуальний порядок зображень – порядок для всіх зображень можна легко змінити вручну в медіа-менеджері, або змінити порядок за параметрами загальних налаштувань;
- можливо створити слайдер зображень із зручним для мобільних пристроїв інтерфейсом та підтримкою пристроїв із сенсорним екраном.



Рисунок 1.5 – Приклад галереї з використанням Robo Gallery [6]

Про те, як і в попереднього прикладу, він не має можливості імпорту зображень Dropbox або Google Drive.

1.3 Висновки за розділом 1

Перший розділ направлений на аналіз предметної області, який продемонстрував, що частина плагінів галерей спрямована на роботу з зображеннями завантаженими до медіа сайту, а інша частина – дозволяє імпортувати зображення з Dropbox, проте усі з наведених варіантів не мають можливості імпорту з Google Drive. Це змушує користувача спочатку завантажити необхідні фото на свій цифровий пристрій, а вже потім до медіа, для їх подальшого використання. Відповідно час роботи над галереєю збільшується, особливо тоді, коли необхідно додати більшу кількість зображень, ніж дві-три. Тому в роботі важливо об'єднати можливості різного імпорту.

Завданням роботи для сукупного досягнення мети є:

- аналіз програмних аналогів;
- визначення засобів проектування і розробки програми;
- проектування програми;
- реалізація і налагодження програми.

2 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ

2.1 Вибір засобів проектування і розробки

Оскільки плагін буде використовуватися у CMS WordPress, що написаний на PHP – ця мова програмування і буде основною для проекту. Також слід зазначити, що WordPress має хуки, що є інструментом WordPress, який дозволяє «підключитися» до ядра WordPress за допомогою коду користувача та щось зробити або змінити [7]. Вони активно використовуються для створення плагінів або тем, тому хуки використовуватимуться у розробці проекту.

Для роботи інтерфейсом користувача будуть використовуватися HTML – це мова розмітки гіпертекстових документів, CSS – спеціальна мова за допомогою якої оформлюються усі веб сторінки та jQuery – бібліотека JavaScript. jQuery було обрано через значні переваги у використанні над JavaScript (табл. 2.1).

Таблиця 2.1 – Відмінності jQuery та JavaScript [8]

JavaScript	jQuery
1	2
Для досягнення кросбраузерної сумісності необхідно написати додатковий код (або змінювати наявний).	jQuery вже має вбудовану кросбраузерну функціональність.
Для створення анімації з використанням JavaScript потрібно багато рядків коду. В основному це досягається зміною стилів HTML сторінки.	jQuery дозволяє додавати ефекти анімації набагато швидше, використовуючи меншу кількість коду.

Кінець табл. 2.1

1	2
Розробникам необхідно написати власний скрипт, який потребує багато часу.	У бібліотеці є вже готові сценарії, якими розробник може користуватися.
Часто, для досягнення певного результату, потрібно написати велику кількість коду.	Дозволяє писати менше коду, досягаючи аналогічного результату.

Тож jQuery є бібліотекою JavaScript, яка дозволяє значно спростувати використання JS у процесі створення веб-застосунків. Використовуючи jQuery, для виконання одного і того ж завдання розробник може використовувати лише один рядок коду замість десятків рядків у порівнянні з чистим JavaScript.

Таким чином, ключова відмінність між JavaScript і jQuery полягає в тому, що JavaScript - це самостійна мова програмування, а jQuery - бібліотека JavaScript, що значно полегшує його використання, але не є окремою мовою [8].

Також виникло питання як зв'язати імпорт зображень з Dropbox та Google Drive з WordPress, адже вони пропонують готові компоненти для розробників. У роботі їх використання повертає посилання на файл, а далі його потрібно обробити з використанням хуків WordPress і, аби це зробити, стало необхідними використання AJAX – технологія звернення до сервера без перезавантаження сторінки.

2.2 Визначення функціональних вимог до плагіну

Плагін галереї повинен надавати можливість роботи з цілим набором функцій, які мають використовуватися для зручної роботи:

- створення галереї;

- видалення галереї;
- редагування галереї;
- обрання для галереї зображення;
- видалення зображення у вже створеній галереї;
- додавання нових зображень до вже створеної галереї;
- вибір шаблону відображення галереї;
- зміна шаблону вже створеної галереї;
- додавати створенні галереї на сторінки або до поста;
- імпорт зображень з Dropbox;
- імпорт зображень з Google Drive;
- перегляд створеної галереї на сторінці сайту.

Відповідно до вимог, зазначених вище, є функції, які може виконувати користувач, зважаючи на те, авторизований він чи ні у WordPress та яку роль має.

Не авторизований користувач матиме тільки можливість перегляду галереї на сторінці сайту, не залежно від його ролі.

Якщо ж користувач авторизований і має таку роль як учасник, він має наступний сценарій роботи (рис. 2.1):

- створення галереї;
- видалення галереї;
- редагування галереї;
- обрання для галереї зображення;
- видалення зображення у вже створеній галереї;
- додавання нових зображень до вже створеної галереї;
- вибір шаблону відображення галереї;
- зміна шаблону вже створеної галереї;
- додавати створенні галереї на сторінки або до поста;
- перегляд створеної галереї на сторінці сайту.

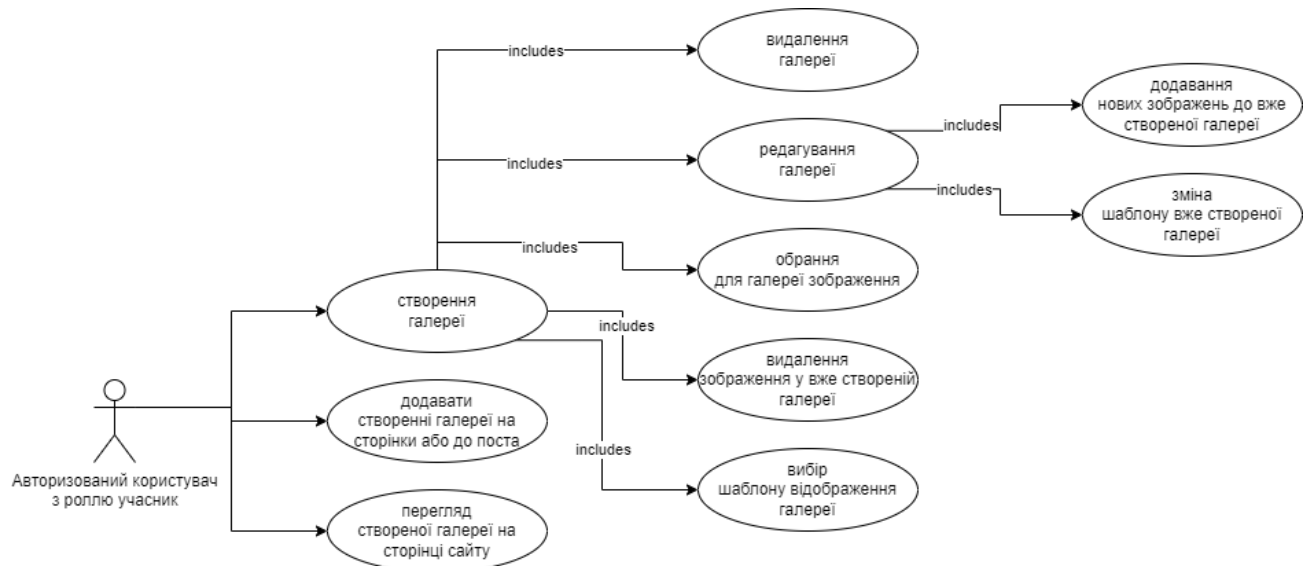


Рисунок 2.1 – Діаграма прецедентів авторизованого користувача з роллю учасник

Якщо ж користувач авторизований і має таку роль як адміністратор, редактор чи автор, він має наступний сценарій роботи (рис. 2.2):

- створення галереї;
- видалення галереї;
- редагування галереї;
- обрання для галереї зображення;
- видалення зображення у вже створеній галереї;
- додавання нових зображень до вже створеної галереї;
- вибір шаблону відображення галереї;
- зміна шаблону вже створеної галереї;
- додавати створенні галереї на сторінки або до поста;
- імпорт зображень з Dropbox;
- імпорт зображень з Google Drive;
- перегляд створеної галереї на сторінці сайту.

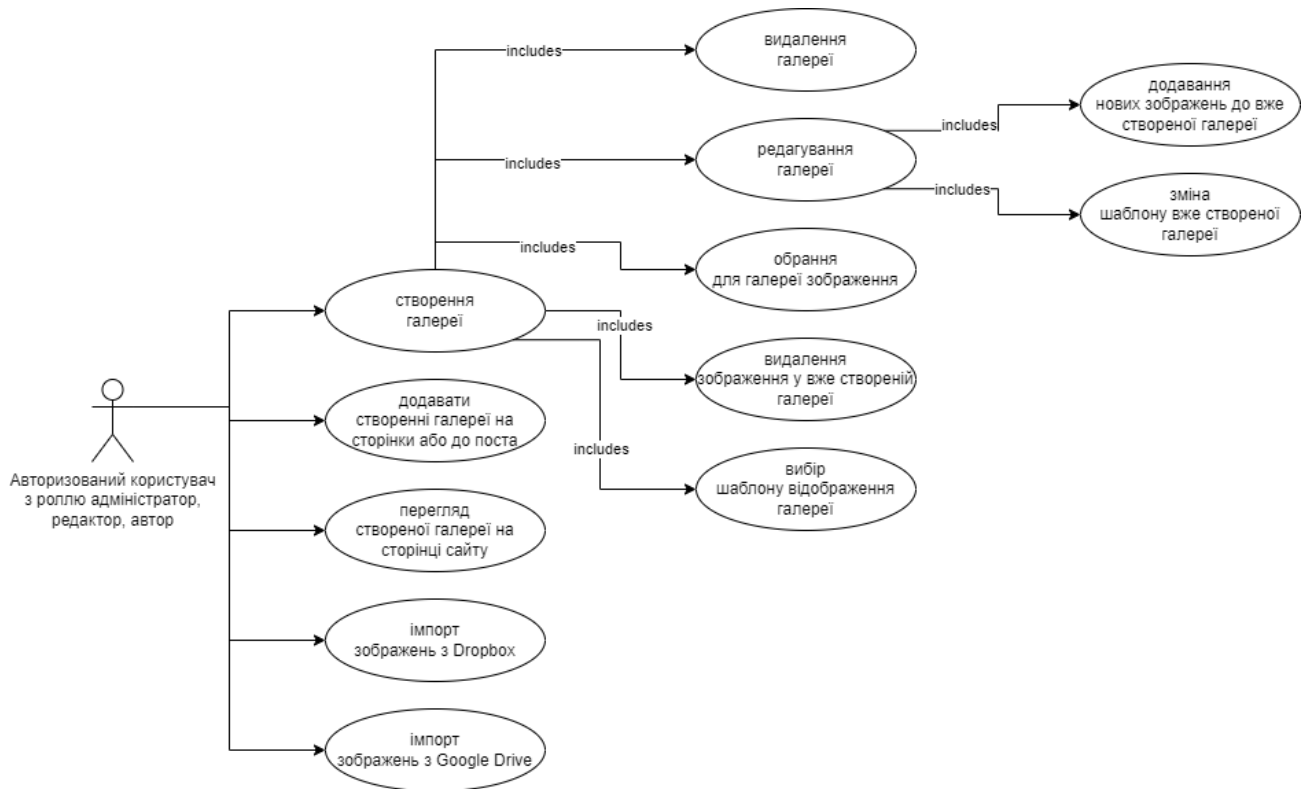


Рисунок 2.2 – Діаграма прецедентів авторизованого користувача з роллю адміністратор, редактор, автор

2.3 Взаємодія з базою даних

WordPress зберігає велику кількість інформації в базі даних, включаючи публікації, сторінки, коментарі, категорії, теги, кастомні поля, користувачів та інші налаштування WordPress.

Під час встановлення WordPress використовує надану вами інформацію про базу даних для створення таблиць і зберігання даних встановлення за замовчуванням у цих таблицях. Початкова збірка WordPress складається з 12 таблиць за замовчуванням у базі даних. Вони містять дані для різних розділів, функцій і можливостей WordPress [9].

Наразі стандартна інсталяція WordPress створює наступні таблиці [9]:

- wp_commentmeta – містить метайнформацію про коментарі;
- wp_comments – містить коментарі та інформацію про автора коментаря

- `wp_links` – керує списками блогів, створених у попередніх версіях WordPress;
- `wp_options` – містить більшість налаштувань WordPress для всього сайту;
- `wp_postmeta` – містить метайнформацію про пости, сторінки та кастомні типи постів;
- `wp_posts` – містить ваші пости, сторінки, кастомні типи постів та їхні версії;
- `wp_termmeta` – містить метадані для термінів у кастомних таксономіях;
- `wp_terms` – містить терміни у таксономіях, таких як кожна категорія та тег;
- `wp_term_relationships` – керує зв'язками між типами дописів і термінами;
- `wp_term_taxonomy` – визначає таксономії для розрізнення категорій і тегів;
- `wp_usermeta` – містить метайнформацію про зареєстрованих користувачів;
- `wp_users` – містить інформацію про користувачів, включаючи імена та паролі.

2.4 Визначення структури програми

Структура плагіну побудована на основі REST API, що передбачена WordPress (рис. 2.3).

REST API – забезпечує доступ до даних вмісту вашого сайту і реалізує ті ж самі обмеження автентифікації – загальнодоступний контент на вашому сайті, як правило, загальнодоступний через REST API, в той час як приватний контент, захищений паролем, внутрішні користувачі, кастомні типи постів і метадані доступні тільки з автентифікацією або якщо ви

спеціально налаштуєте його таким чином. Найважливіше, що потрібно розуміти про API – це те, що він дозволяє використовувати редактор блоків і сучасні інтерфейси плагінів без шкоди для безпеки або конфіденційності вашого сайту [10].

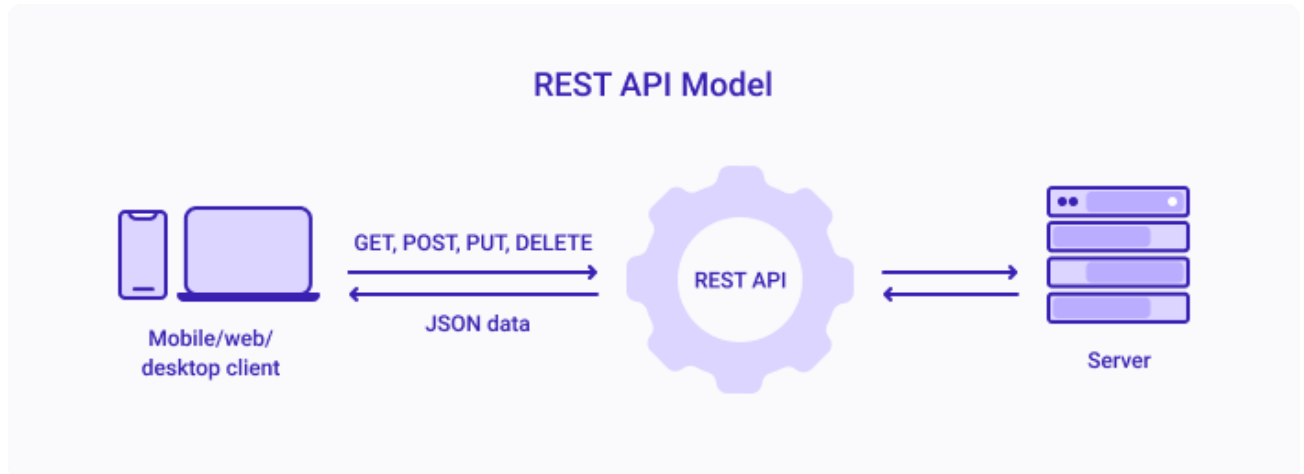


Рисунок 2.3 – Методологія REST API [11]

2.5 Висновки за розділом 2

Виконано вибір засобів проектування і розробки плагіну. Виділено основні сценарії роботи авторизованого користувача з роллю учасник та авторизованого користувача з роллю адміністратор, редактор чи автор. Визначено, що за база даних використовуватиметься та розібрано методологію REST API.

3 РОЗРОБКА ПРОГРАМИ

3.1 Алгоритм функціонування програми

Функціональність, яка надається плагіном для авторизованого користувача з ролю учасник та авторизованого користувача з ролями адміністратор, редактор чи автор, зумовлена набором дозволів, який надає WordPress у стандартному варіанті різним ролям авторизованих користувачів.

Роботу плагіну розділено на три частини – робота з галереєю, роботу з імпортом зображень та роботу з додаванням галереї до поста чи сторінки.

Під час роботи з галереєю (рис. 3.1) авторизований користувач повинен обрати, що він хоче працювати саме з нею, після чого визначитися яку дію з галереєю необхідно виконати. Якщо обрано «створити нову галерею», користувач переходить на сторінку створення де він може обрати необхідні зображення, видалити зайве, обрати шаблон галереї. Коли всі бажані дії виконано, відбувається збереження галереї. Якщо обрано редагування вже наявної галереї, користувач може обрати «змінити шаблон галереї» або «змінити раніше обрані зображення». Коли всі бажані дії виконано відбувається збереження галереї. Якщо обрано «видалити галерею», користувач має обрати, яку саме галерею необхідно прибрати, після чого її буде видалено, а список галерей оновлено.

Під час імпорту зображень (рис. 3.2) авторизований користувач може обрати з якого хмарного сховища відбуватиметься імпорт, Dropbox чи Google Drive. Коли вибір зроблено, завантажується діалогове вікно в якому можливо обрати зображення, якщо було обрано Google Drive, то перед цим діалоговим вікном буде ще одне для авторизації. Результатом обрання зображень буде відправлення запиту на сервер для завантаження обраного до WordPress, після чого користувач побачить повідомлення чи успішно пройшло завантаження.

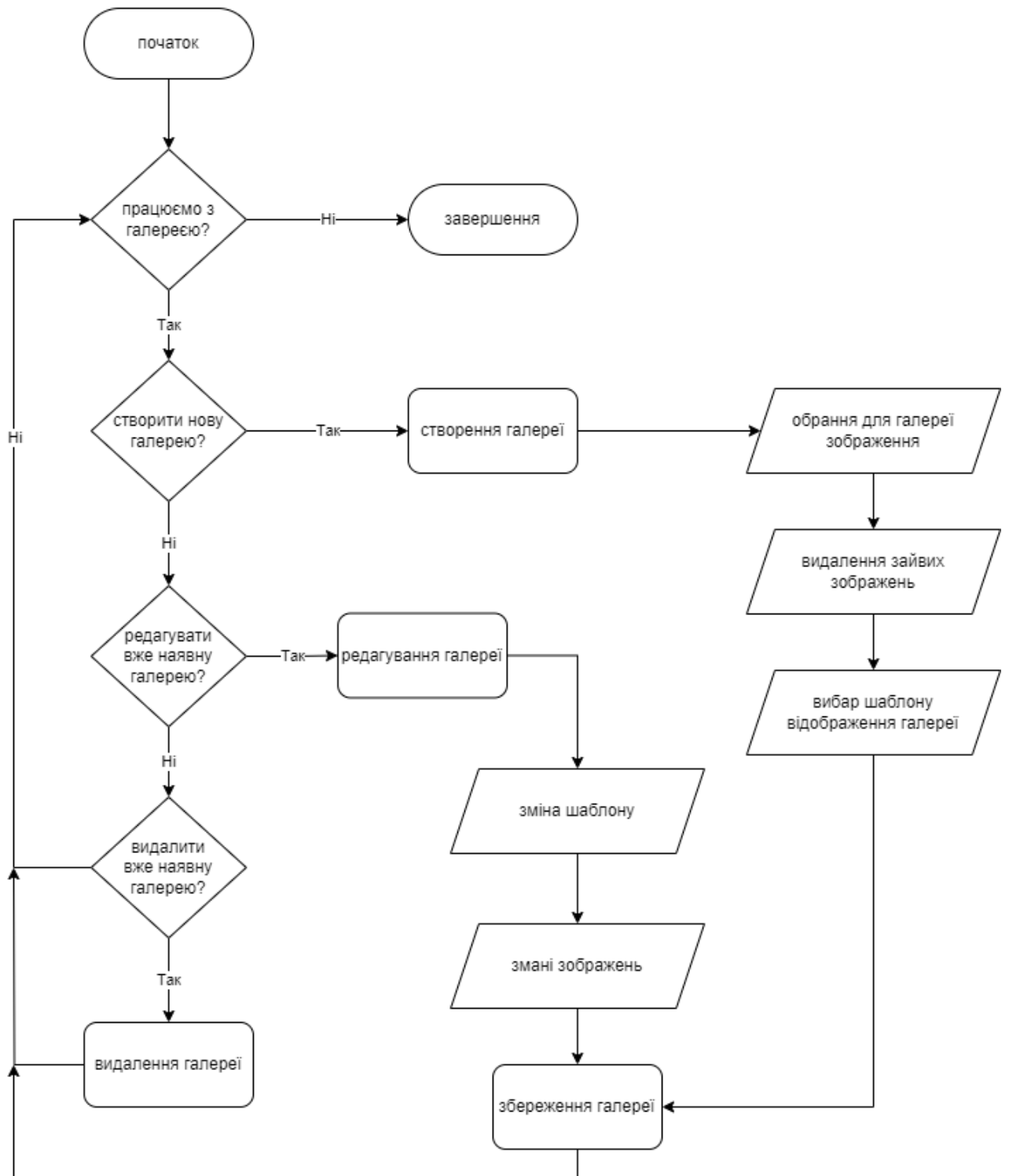


Рисунок 3.1 – Схема функціонування плагіну для роботи з галереєю

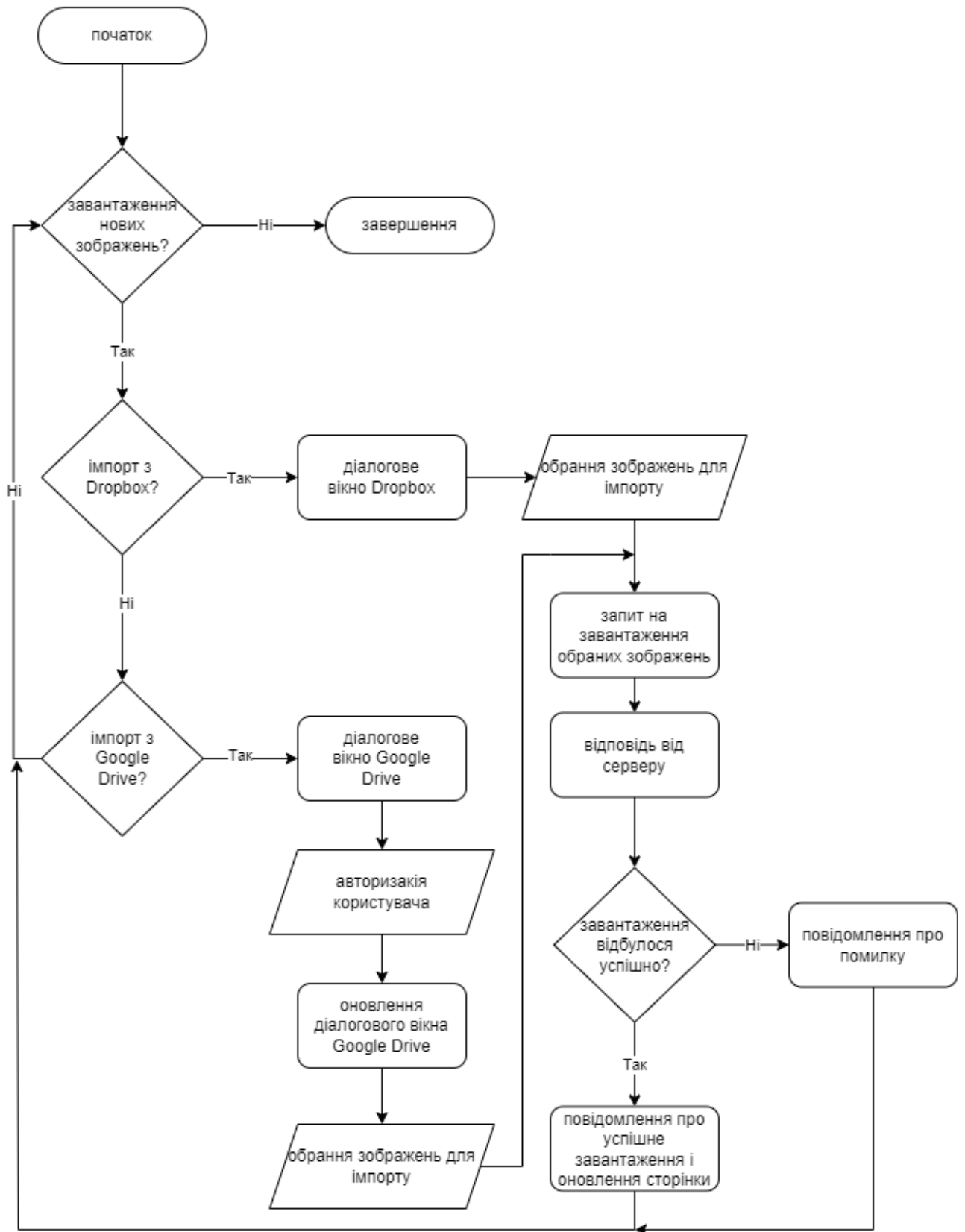


Рисунок 3.2 – Схема функціонування плагіну для роботи з імпортом зображень

Під час додавання галереї до посту чи сторінки (рис. 3.3) авторизований користувач має обрати галерею зі списку створених галерей у діалоговому вікні, після чого спрацює скрипт, який додасть shortcode до контенту поста чи сторінки.

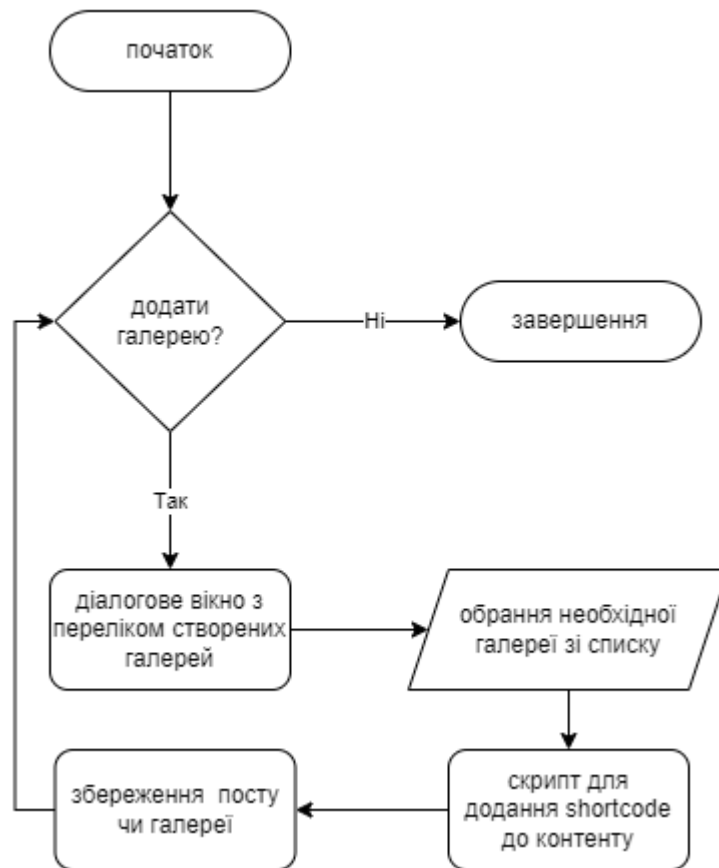


Рисунок 3.3 – Схема функціонування плагіну для роботи з додаванням галереї до посту чи сторінки

3.2 Опис реалізованих одиниць вебзастосунку

Хук `admin_menu` використовується для додавання додаткових підменю та пунктів меню до структури меню панелі адміністратора. Він викликає функцію, з такою ж назвою - `admin_menu()`, що додає пункт підменю під назвою `Connect API`.

Хук `admin_head` спрацьовує в розділі `head` для всіх сторінок адміністратора. За його допомогою спрацьовують функції

`dropbox_custom_js()`, `add_gallery_custom_js()` та `google_custom_js_header()`, аби підключити необхідні скрипти.

Хук `post-plupload-upload-ui` спрацьовує після завантаження інтерфейсу завантаження, тобто після завантаження інтерфейсу на сторінці Media. Функції `mediaButton()` та `GoogleMediaButton()` працюють з його допомогою, аби додати кнопки для імпорту зображень до інтерфейсу сторінки Media.

Хук `p_ajax_{ $action }` запускає дії Ajax для користувачів, а частина імені хука `$action` відноситься до імені зворотного виклику дії Ajax, що запускається. Функції `dropbox_ajax_action()` та `google_ajax_action()` виконуються з його допомогою та необхідні для обробки

Хук `media_buttons_context` виконується після відображення кнопок мультимедіа за замовченням. Він необхідний для додавання Add Gallery на сторінках створення та редагування постів або сторінок.

Хук `init` спрацьовує після завершення завантаження WordPress і далі починаються відпрацьовувати хуки `init`. Цей хук часто використовується у різних плагінах, тому у проекті він необхідний для реєстрації нового типу поста для галереї.

Хук `add_meta_boxes` необхідний для додавання нових метабоксів після завантаження вбудованих. За його допомогою створюється поле Gallery images на сторінках створення і редагування галереї, у якому відображаються обрані зображення.

Хук `save_post` спрацьовує при збереженні поста і необхідний для збереження інформації про обрані зображення для галереї та обраного шаблону.

Хук `edit_form_after_editor` додає контент в кінець на сторінці редагування поста чи сторінки. Він використовується для додавання діалогового вікна з переліком створених галерей, які можливо додати до поста чи сторінки.

Функція `add_shortcode` – додає новий короткий код, що записується у квадратних дужках. Короткий код необхідний аби додати інформацію про

галерею у пост чи сторінку в короткому вигляді і аби перетворити цю інформацію на сторінці вебсайту на зображення розміщені за обраним шаблоном.

Для імпорту зображень з Dropbox використовується селектор – готовий компонент для розробників, приклад якого наведено в документації. Так само, для імпорту зображень з Google Drive, використовується Google Picker приклад якого наведено в документації. Кожен з цих прикладів повертає перелік посилань на зображення обрані для імпорту. Аби обробити отримані посилання використано AJAX запит. Кожен з цих запитів активується після отримання посилань на зображення від Dropbox селектору та Google Drive Picker, обидва використовують наступний набір параметрів:

- type() – тип запиту. Можливі значення GET або POST;
- dataType() – опис типу даних, який очікується у відповідь від сервера. Можливі значення xml, json, script або html;
- url() – URL адрес для запиту;
- data() – дані користувача, які передаються на сторінку запиту;
- success() – ця функція викликається після успішного завершення запиту. Функція отримує від 1 до 3 параметрів. Але перший параметр завжди містить дані, що повертаються з сервера. Нижче наведено фрагмент коду, що реалізує AJAX запит:

```
var data = {  
    'action': 'dropbox_ajax_action',  
    'files_link': files  
};  
jQuery.ajax({  
    type: "post",  
    dataType: "json",  
    url: dropbox_img_url.ajax_url,  
    data: data,
```

```
success: function(result){  
    if (result != 0) {  
        alert ('Files uploaded successfully. Page will be reload.');        location.reload();  
    } else {  
        alert ('Something went wrong ...');    }  
}  
});
```

3.3 Висновки за розділом 3

Описано основні послідовності, що надається для реалізації роботи функціями, який надає плагін. Описано основні хуки, що використовуються плагіном та принцип взаємодії користувача з сервером.

4 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПЛАГІНУ

4.1 Опис плагіну

Робота зі створенням контенту у веб просторі супроводжується не лише текстом, а й зображеннями. Їх розміщення на сторінці є доволі важливим, адже це має бути зручним для перегляду користувачами, особливо якщо їх велика кількість. Тому існують плагіни галереї, що допомагають організувати зображення, проте не всі вони мають засоби для імпорту фото з хмарних сховищ. Саме на реалізацію подібного функціоналу направлено роботу даного плагіну.

Для підтримки роботи плагіну необхідно, щоб хостинг включав дисковий простір від 500 МБ, оперативну пам'ять від 512 МБ, вебсервер з підтримкою PHP і MySQL, версія PHP від 7 і вище, версія MySQL від 5.6 і вище, HTTPS протокол.

4.2 Експлуатація програми

Для експлуатації плагіну потрібно спочатку авторизуватися в панелі адміністратора CMS WordPress, встановити плагін та активувати його (рис. 4.1). Після чого в правому меню панелі адміністратора з'явиться новий пункт під назвою Image Galleries, кнопки для імпорту зображень на сторінці Media Library під час завантаження нових фото та кнопку додавання галереї для її подальшого відображення на вебсайті на сторінці створення чи редагування поста або сторінки.

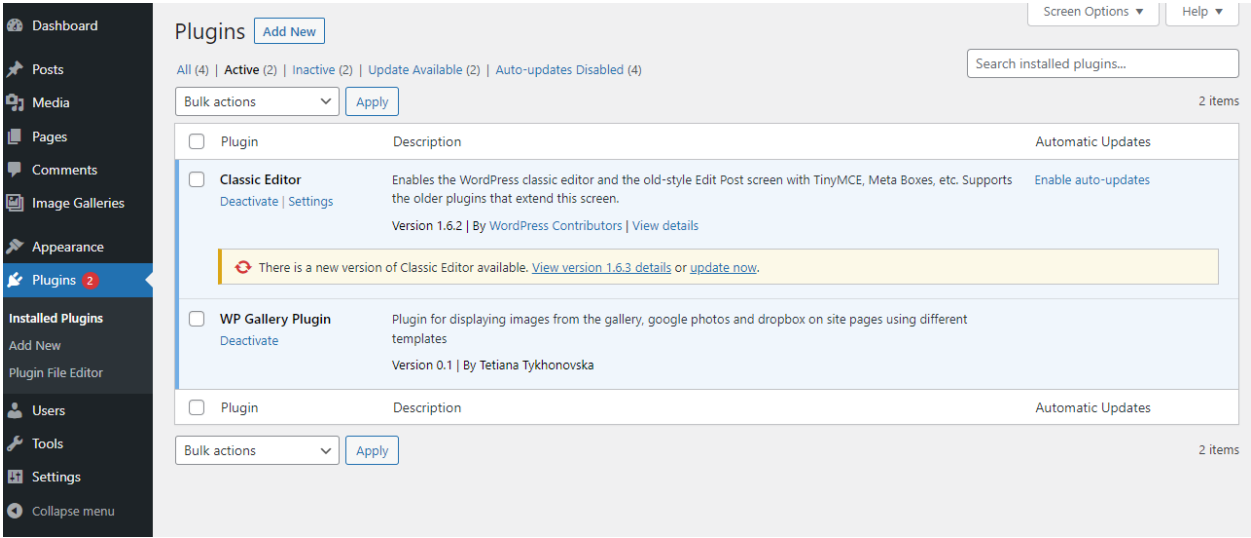


Рисунок 4.1 – Активація плагіну в панелі адміністратора

При наведенні миші на пункт меню Image Galleries буде відображено його підменю (рис. 4.2), яка складається з наступних сторінок – Image Galleries, Add New Gallery, Connect API.

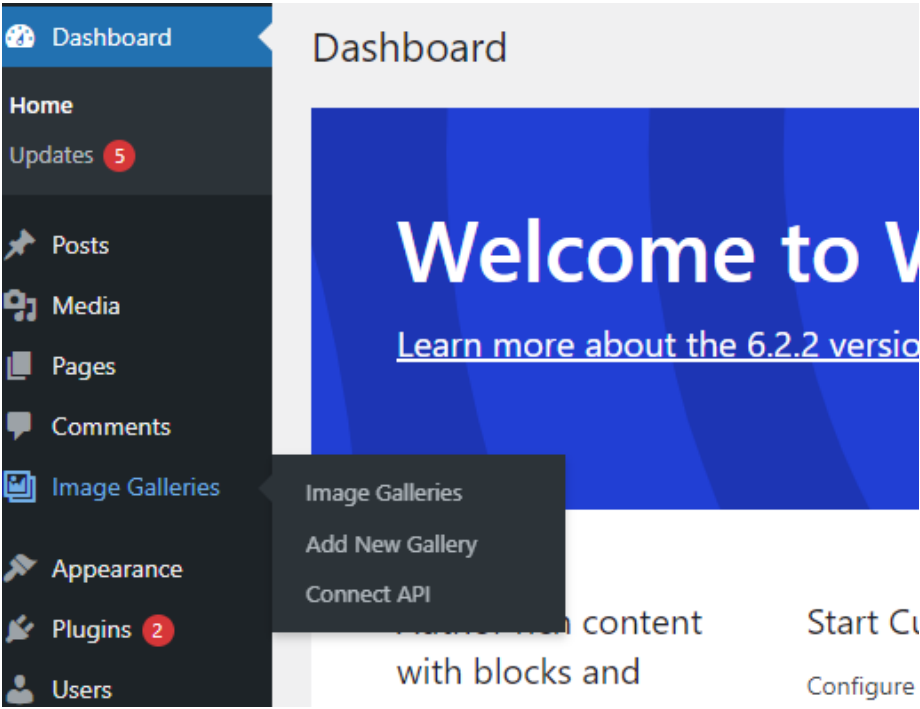


Рисунок 4.2 – Підменю Image Galleries

Під час обрання пункту меню Image Galleries (рис. 4.3) користувач переходить на сторінку на якій буде відображено перелік всіх створених

галерей у вигляді таблиці з короткою інформацією про них. Також на цій сторінці можливо:

- перейти до перегляду створених галерей, що створив користувач, який є авторизованим;
- видалити обрану галерею;
- перейти на сторінку її редагування;
- обрати швидке редагування на цій же сторінці;
- перейти до перегляду галереї на сторінці вебсайту.

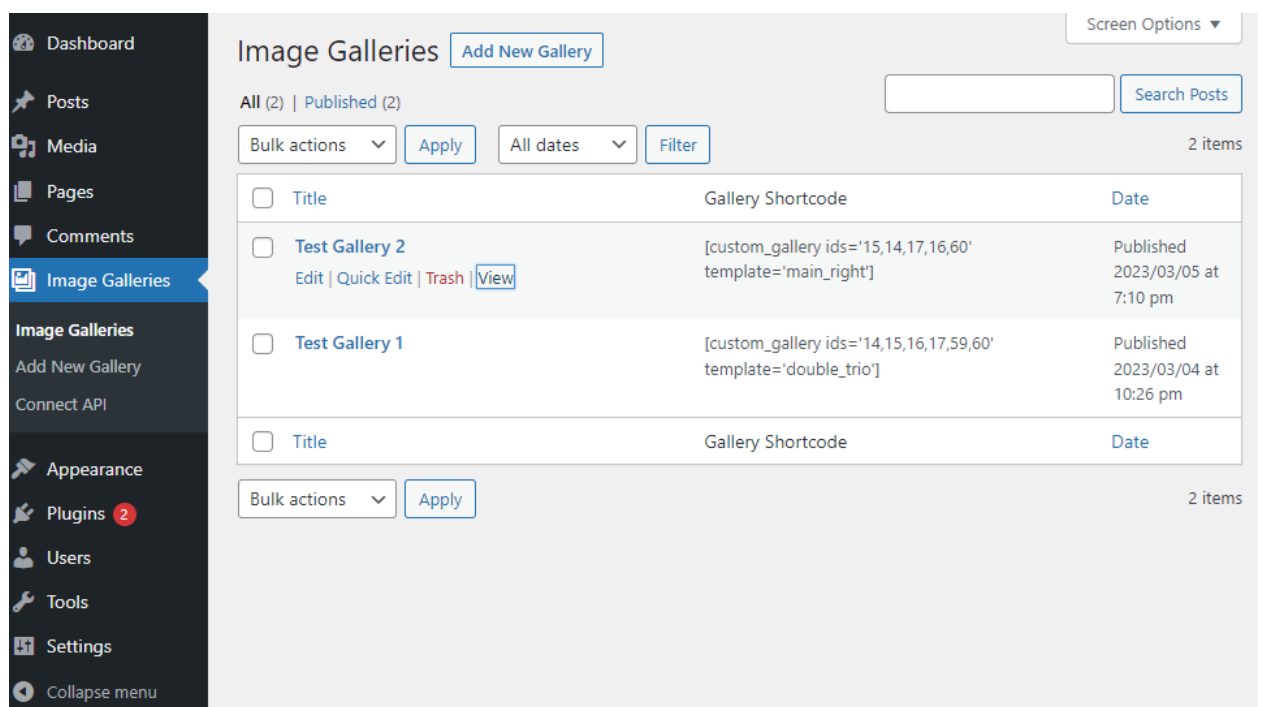


Рисунок 4.3 – Сторінка Image Galleries

Для переходу на сторінку створення нової галереї користувач може скористуватися кнопкою **Add New Gallery**, що знаходиться біля заголовку сторінки **Image Galleries** або пунктом з бокового меню, що має туж саму назву. На сторінці для створення нової галереї можливо (рис. 4.4):

- ввести назву нової галереї;
- обрати необхідні зображення, що з'являться у діалоговому вікні, після натискання кнопки **Select Images**;
- обрати шаблон відображення галереї.

Після чого галерея може бути опублікованою, якщо натиснути кнопку Publish.

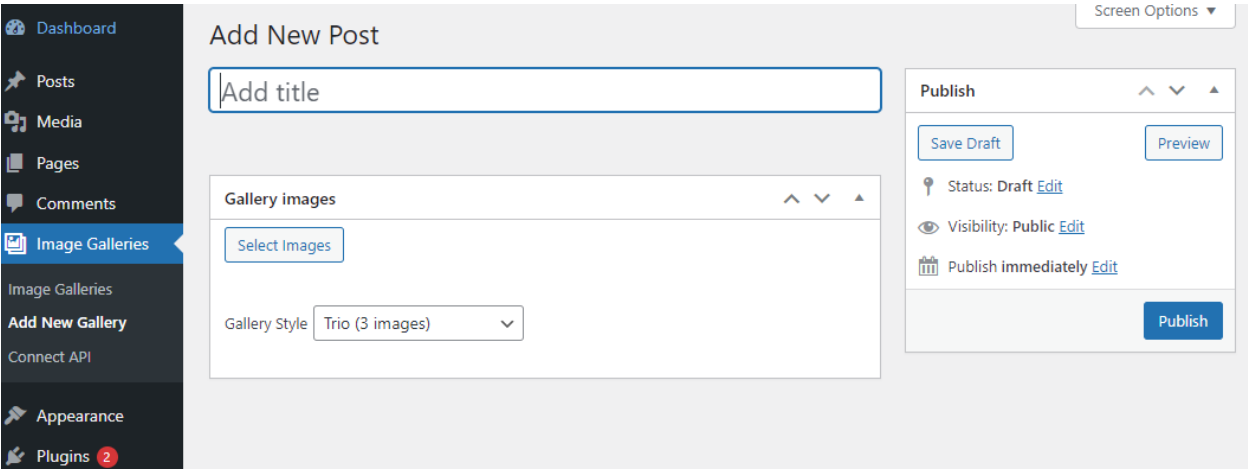


Рисунок 4.4 – Сторінка Add New Gallery

Якщо обрати редагування існуючої галереї (рис. 4.5), буде відкрито сторінку схожу на Add New Gallery, де можливо змінити зображення, додати нові, обрати інший шаблон або змінити назву галереї після чого оновити цю галерею натиснувши кнопку Update.

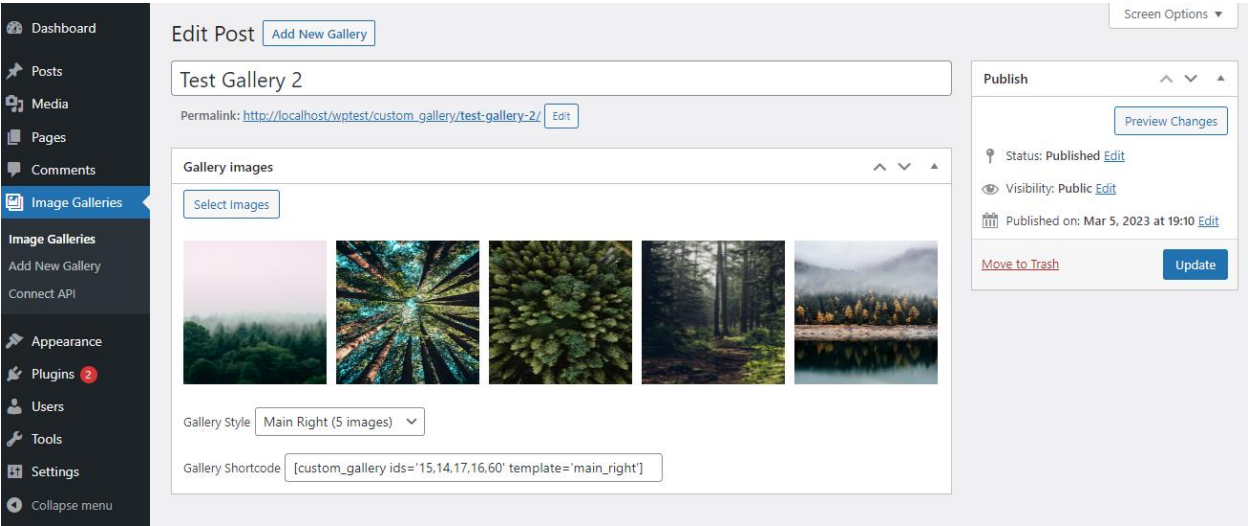


Рисунок 4.5 – Сторінка редагування галереї

У випадку обрання пункту Connect API з підменю Image Galleries, буде відображено відповідну сторінку на якій необхідно ввести ключ від

Dropbox App аби мати змогу імпортувати зображення зі свого Dropbox та зберегти зміни (рис. 4.6). Також на цій сторінці розміщено посилання на сторінку для створення Dropbox App, якщо користувач ще його не має.

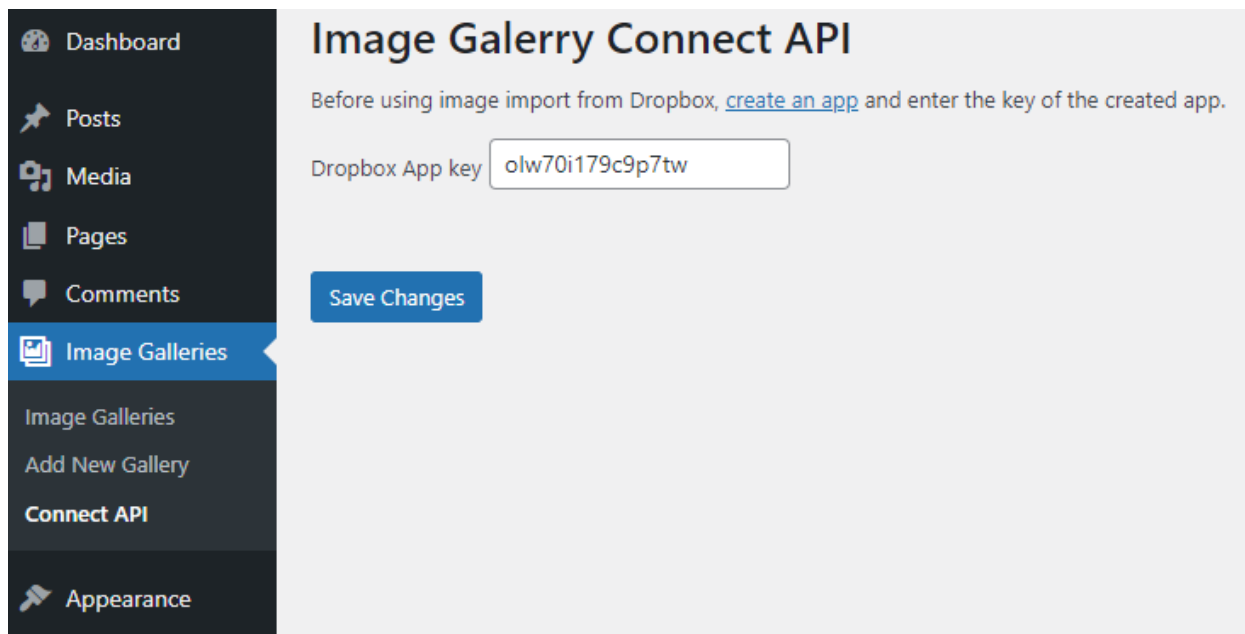


Рисунок 4.6 - Сторінки Connect API

Для імпорту зображень з хмарних сховищ, необхідно перейти на сторінку Media, посилання на яку знаходиться у боковому меню панелі адміністратора, де натиснути кнопку Add New, після чого з'явиться стандартна кнопка WordPress для завантаження зображень з пристрою, а під нею кнопки для імпорту з Dropbox та Google Drive, що мають відповідні підписи (рис. 4.7).

Якщо обрати імпорт з Dropbox та натиснути кнопку Choose from Dropbox, з'явиться діалогове вікно від Dropbox (рис. 4.8) в якому можливо обрати одне чи декілька фото для завантаження, після чого фони будуть додані до медіа файлів WordPress і будуть відображені серед списку мініатюр на сторінці Media, а також вони стануть доступними до вибору при створенні чи редагуванні галереї.

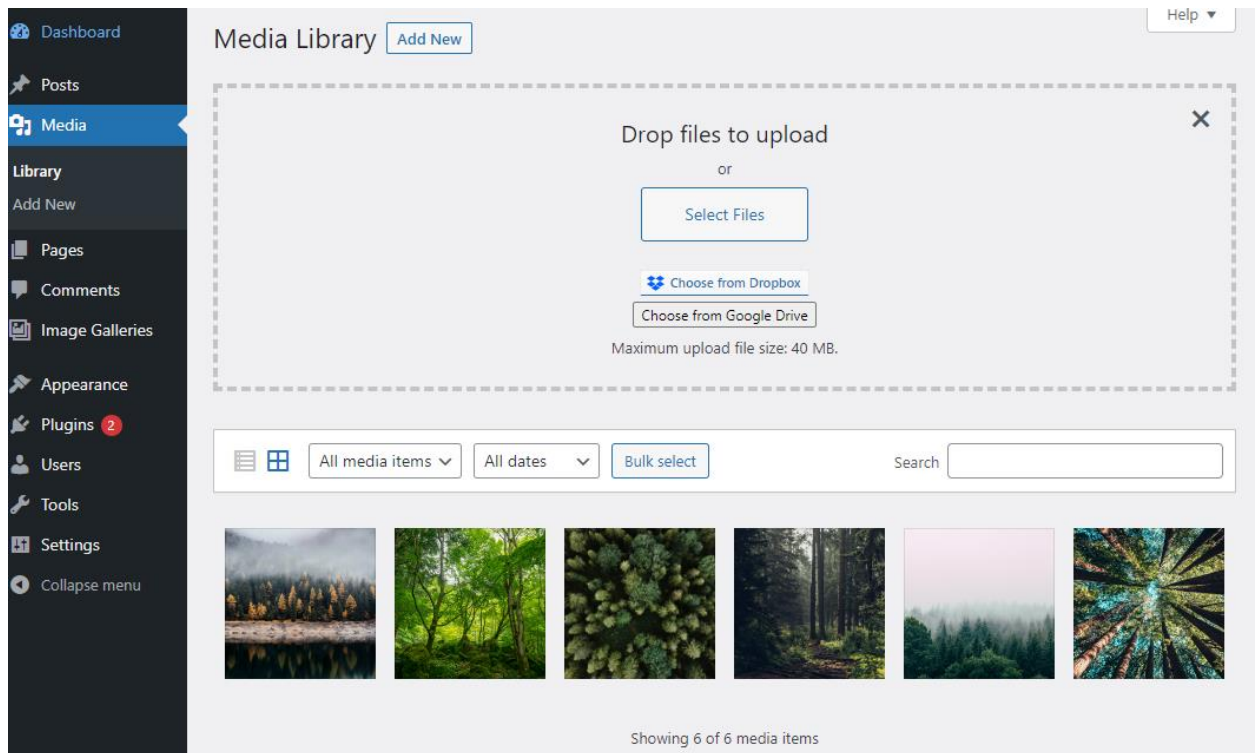


Рисунок 4.7 – Розташу кнопок імпорту з Dropbox та Google Drive

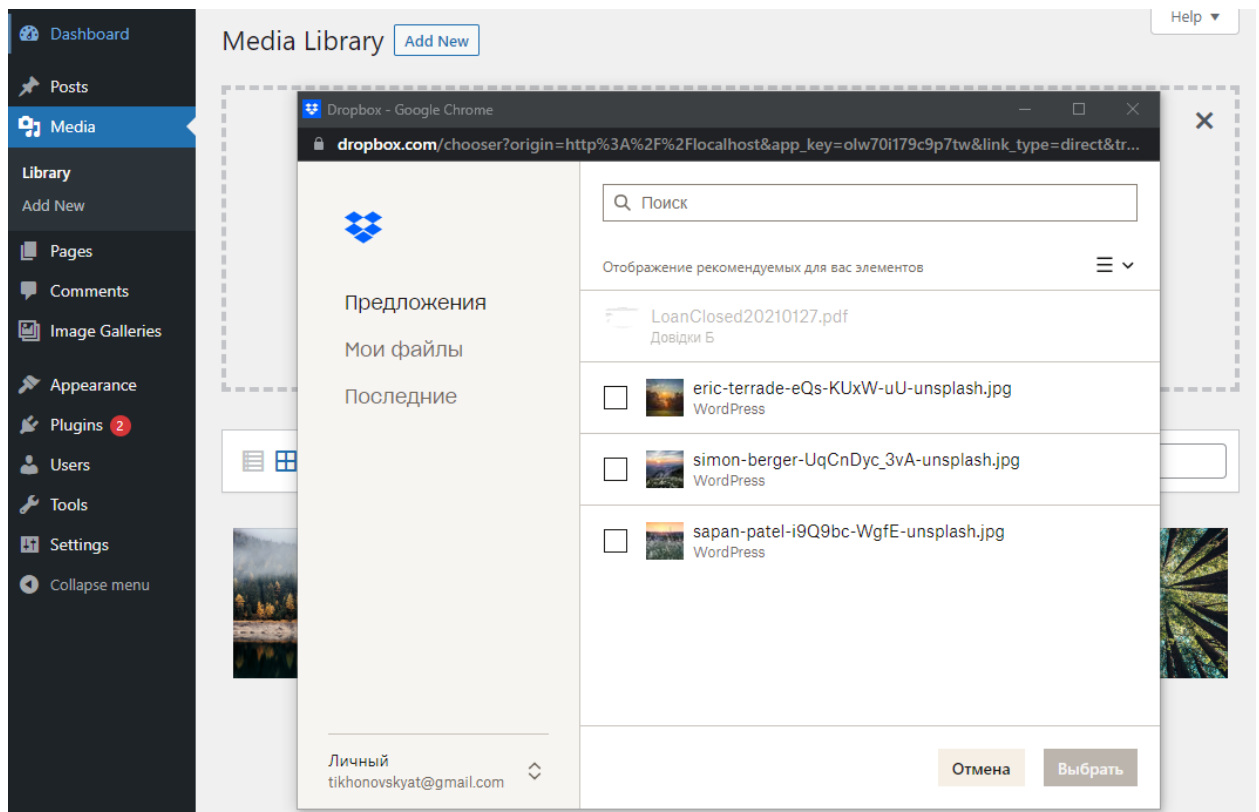


Рисунок 4.8 – Діалогове вікно Dropbox

Якщо обрати імпорт з Google Drive та натиснути кнопку Choose from Google Drive, з'явиться діалогове вікно від Google Drive (рис. 4.9) в якому необхідно обрати необхідний обліковий запис, після чого підтвердити доступ до файлів Google Drive. Коли доступ дозволено, діалогове вікно оновиться і в ньому будуть відображені фото (рис. 4.10), які можливо обрати для завантаження. Їх так само можливо обрати одне чи декілька та вони будуть додані до медіа файлів WordPress, після чого відображатимуться серед списку мініатюр на сторінці Media і стануть доступними до вибору при створенні чи редагуванні галереї.

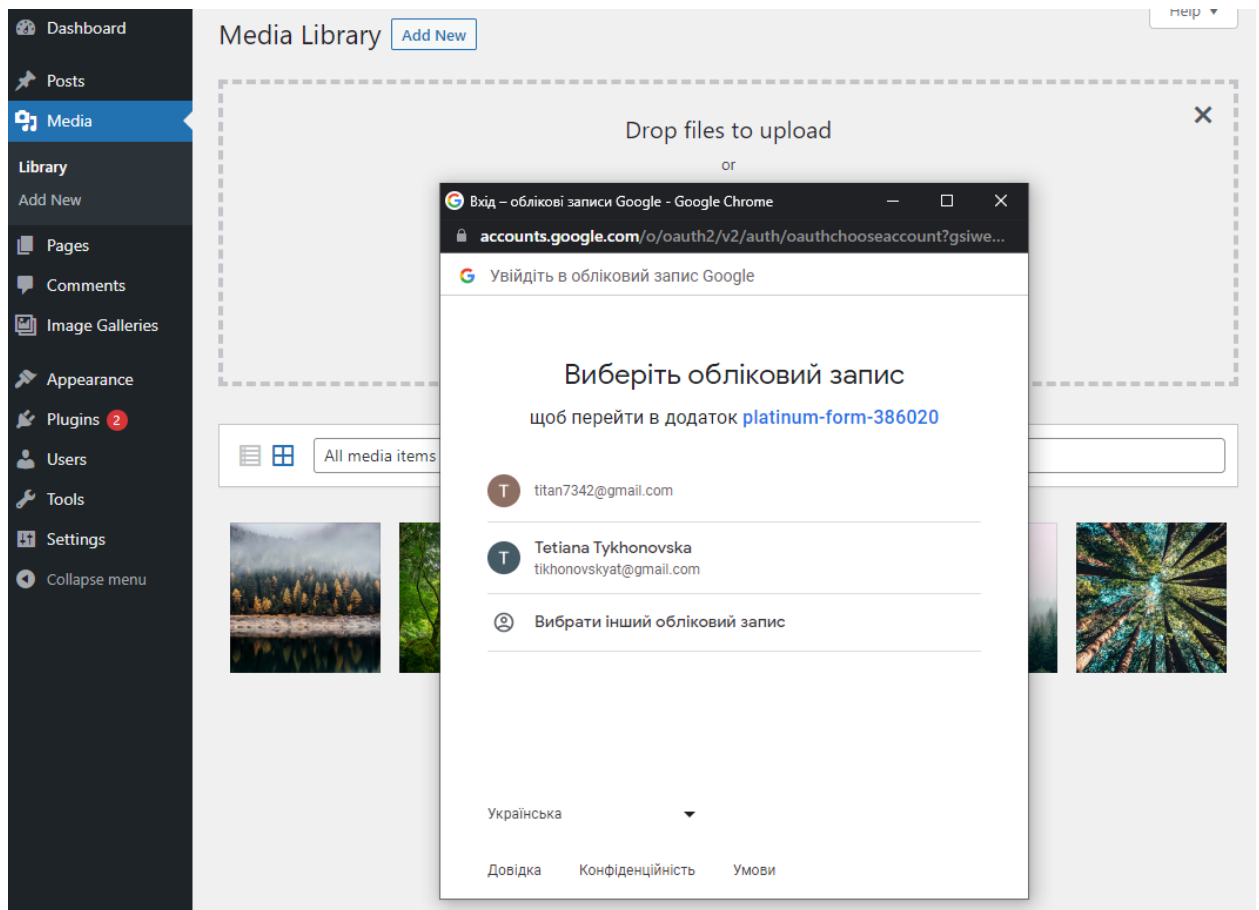


Рисунок 4.9 – Діалогове вікно Google Drive

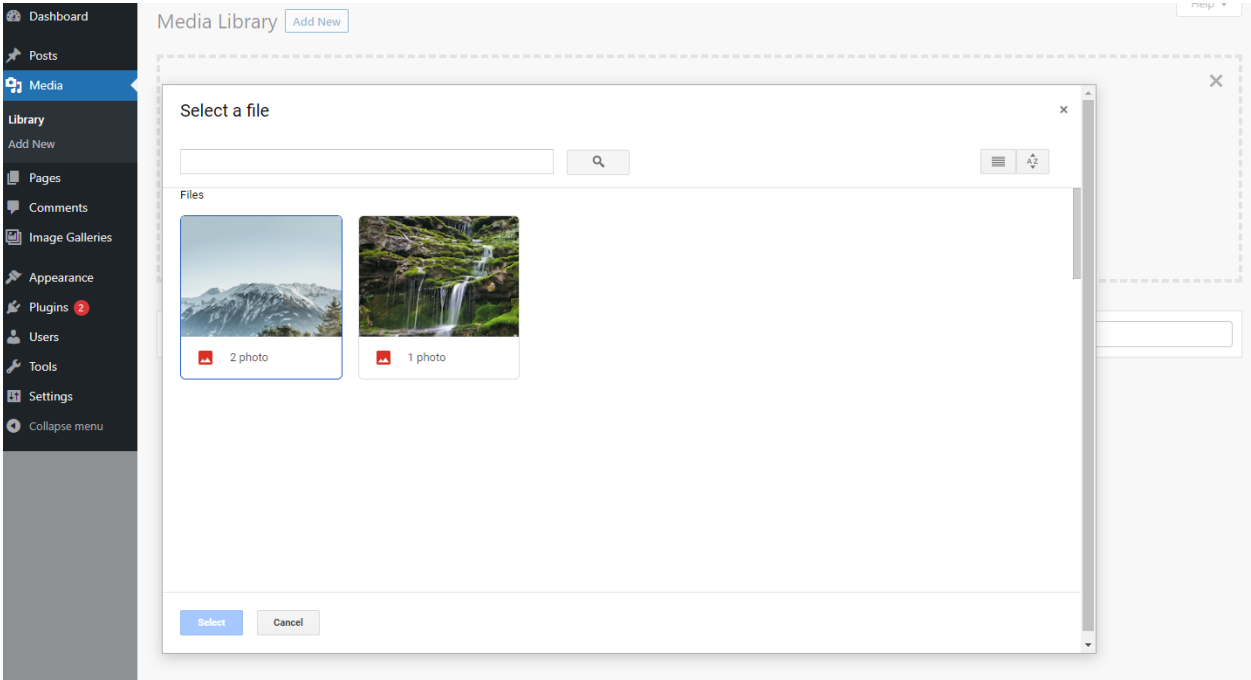


Рисунок 4.10 – Оновлене діалогове вікно Google Drive для обрання зображень

Для додавання готової галереї на сторінку чи пост, необхідно перейти до його редагування де буде відображено кнопку Add Gallery (рис. 4.11). Натиснувши цю кнопку з’явиться діалогове вікно за списком опублікованих галерей, з якого можливо обрати одну і вона буде додана до тексту сторінки.

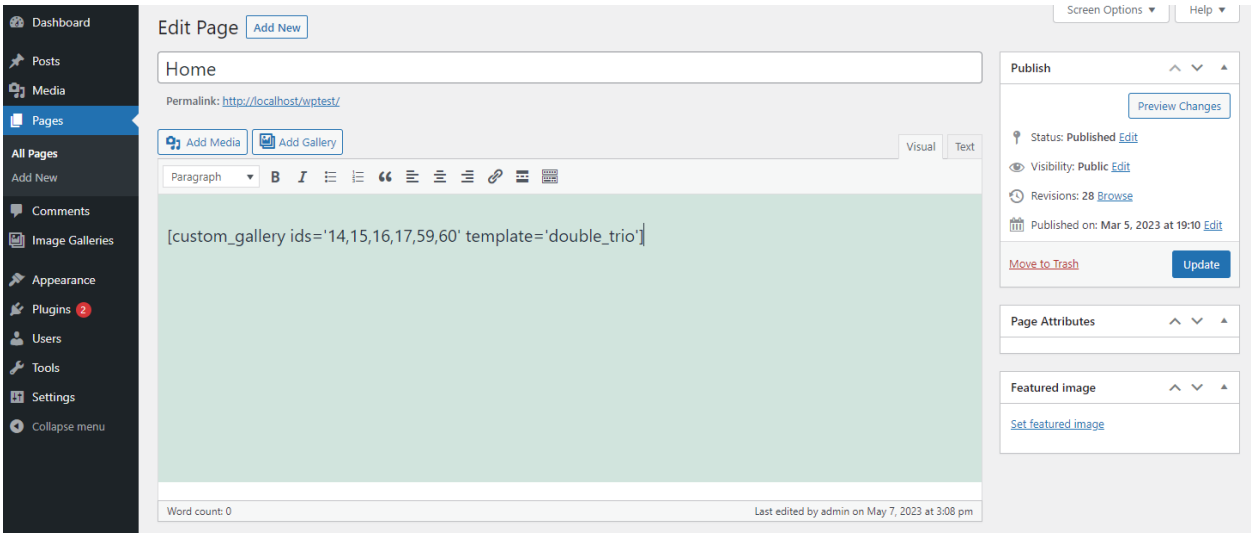


Рисунок 4.11 – Додавання галереї на сторінку

Аби переглянути який вигляд має галерея на сторінці вебсайту, необхідно перейти за посиланням необхідної сторінки (рис. 4.12), після чого буде відображено галерею за обраним шаблоном. А для перегляду зображення у більшому форматі (рис. 4.13) достатньо натиснути на нього і з'явиться лаутбокс, відповідний оригінальному розміру фотографії, що б його закрити, достатньо натиснути у будь яке місце на екрані за межами збільшеного зображення.

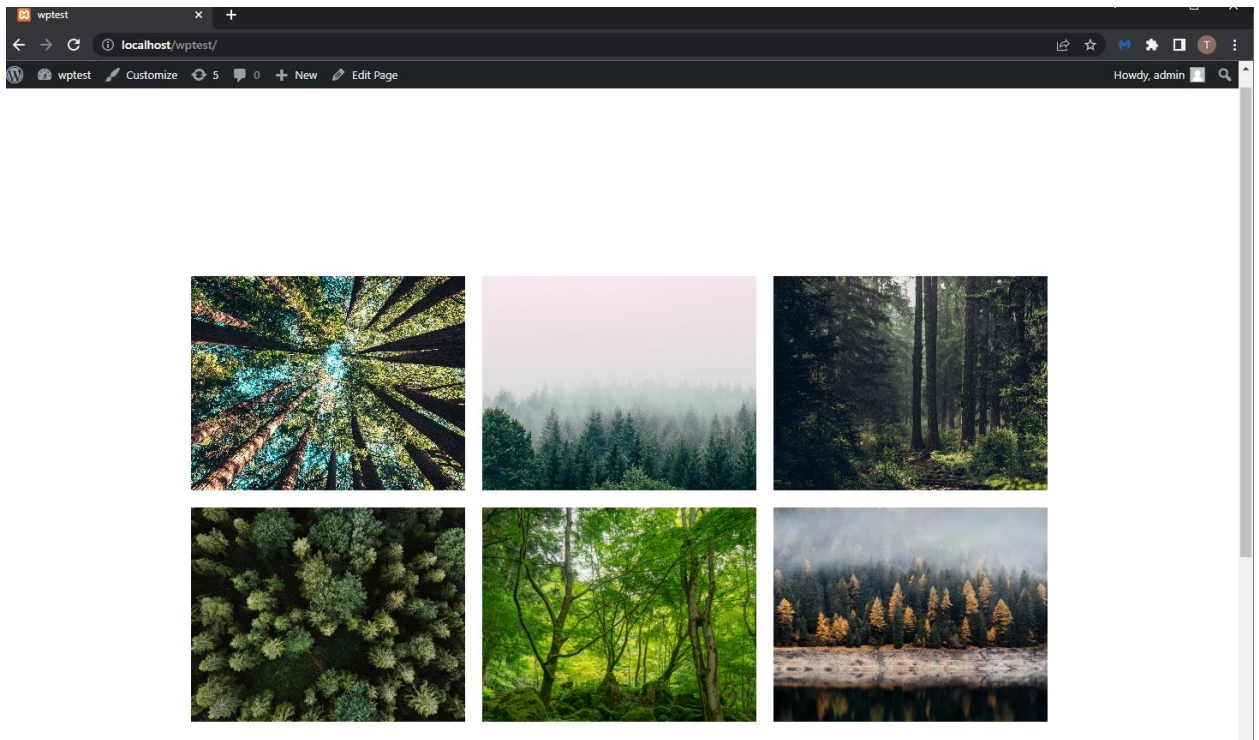


Рисунок 4.12 – Приклад відображення галереї на сторінці вебсайту

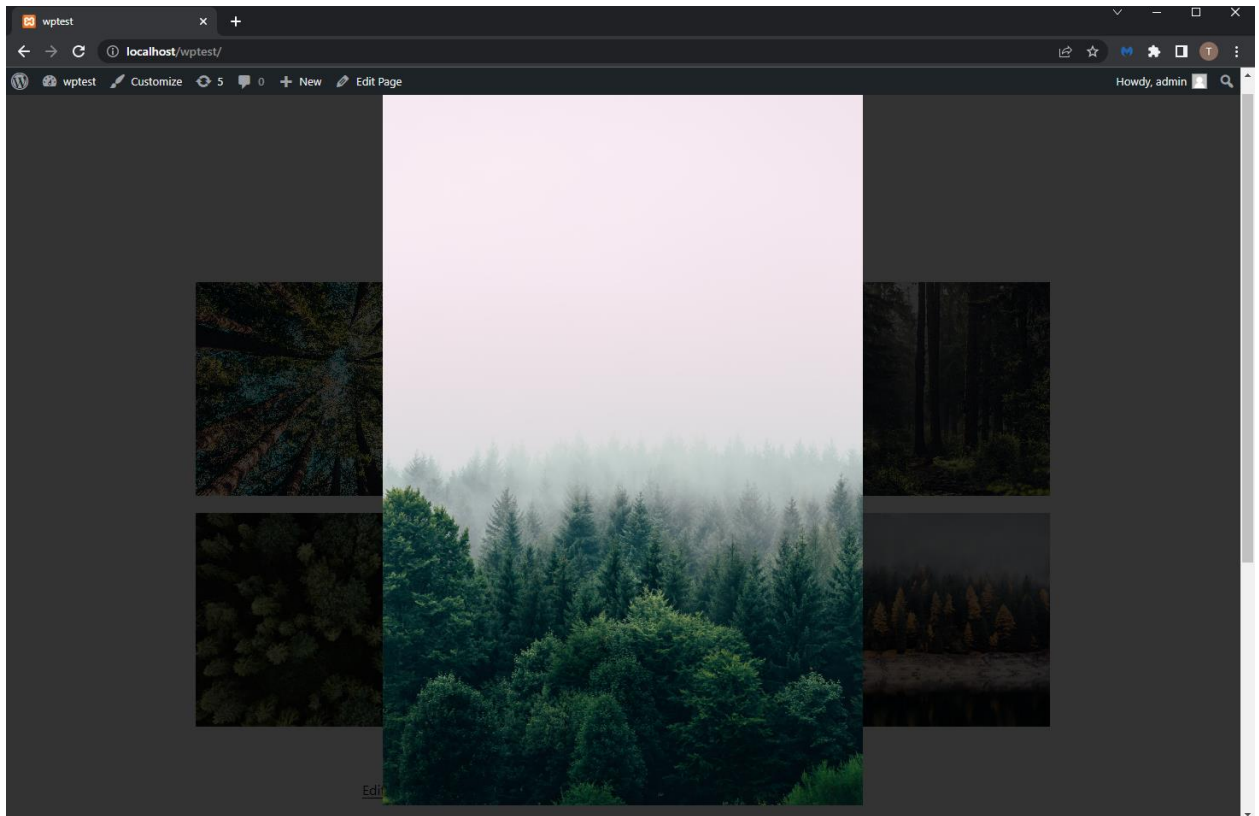


Рисунок 4.13 – Лайтбокс зі збільшеним зображенням

4.3 Тестування програми

Тестування програми було проведено у трьох напрямках.

Перший напрямок включав перевірку функціональних вимог плагіну. Для цього розроблено набір тестів, що включав коректну роботу з галереєю, її вбудовування на сторінку чи пост та відображення при перегляді. Також важлива частина тестування коректного відпрацювання ролей.

Другий напрямок був спрямований на перевірку імпорту зображень з хмарних сховищ до WordPress.

Було окремо протестовано весь список функціональних вимог, що підтверджує дієздатність плагіну.

Третій напрямок включав роботу шаблонів галереї через різні браузери. Цей напрямок включав браузери Google Chrome (версія 113.0.5672.127), Mozilla Firefox (версія 113.0.1), Microsoft Edge (версія

113.0.1774.50). Результати продемонстрували, що шаблони плагіну коректно відображаються у всіх зазначених браузерах.

4.4 Висновки за розділом 4

Плагін працює в режимах неавторизованого користувача, авторизованого користувача з роллю учасник, адміністратор, редактор, автор, кожен з яких було описано. Виконано успішне тестування плагіну.

ВИСНОВКИ

Виконано аналіз предметної області, в межах якого було проведено огляд програмних аналогів, до яких входять плагіни Envira Gallery, Go Portfolio, Robo Gallery. Враховуючи в них відсутність імпорту зображень одразу з двох хмарних сховищ, таких як Dropbox та Google Drive, саме дану можливість визначено як особливість розробленого плагіну.

Розробку плагіну було виконано на основі CMS WordPress з визначенням до плагіну, розробленням структури плагіну і алгоритму роботи на основі цих вимог, реалізацією відповідних одиниць вебзастосунку. Для роботи використано мову програмування PHP, мову розмітки HTML, мову стилю сторінок CSS, бібліотеку jQuery.

Плагін дозволяє створювати галерею, її редагувати та видаляти, додавати галерею на сторінку чи у пост, імпорт зображень з Dropbox та Google Drive. Також до створення галереї та її редагування відноситься можливість вибору зображень, їх видалення, додавання нових та вибір типу шаблону, що буде використовуватися, або його зміна.

Розроблений плагін дозволяє підвищити зручність роботи з організації зображень для перегляду на сторінках вебсайту, та за потреби імпортувати зображення з таких хмарних сховищ як Dropbox та Google Drive для їх подальшого використання.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Meet WordPress [Electronic resource]. - Access mode : <https://uk.wordpress.org/>.
2. Plugins [Electronic resource]. - Access mode : <https://wordpress.com/support/plugins/>.
3. How to Create an Image Gallery in WordPress [Electronic resource] / Anna Fitzgerald. - Access mode : <https://blog.hubspot.com/website/create-image-gallery-wordpress>.
4. Envira Gallery [Electronic resource]. - Access mode : <https://enviragallery.com/>.
5. Go Portfolio - WordPress Responsive Portfolio [Electronic resource] / Granth. - Access mode : <https://codecanyon.net/item/go-portfolio-wordpress-responsive-portfolio/5741904>.
6. Photo Gallery, Images, Slider in Rbs Image Gallery [Electronic resource] / RoboSoft. - Access mode : <https://wordpress.org/plugins/robo-gallery/>.
7. The WordPress Hooks Bootcamp: How to Use Actions, Filters, and Custom Hooks [Electronic resource] / Salman Ravoof. - Access mode : <https://kinsta.com/blog/wordpress-hooks/>.
8. Differences between JavaScript and jQuery [Electronic resource] / Natalie. - Access mode : <https://it-incubator.io/blog/javascript-jquery-differences>.
9. What is: MySQL [Electronic resource]. - Access mode : <https://www.wpbeginner.com/glossary/mysql/>.
10. REST API Handbook [Electronic resource]. - Access mode : <https://developer.wordpress.org/rest-api/>.
11. What is a REST API? [Electronic resource]. - Access mode : <https://voximplant.com/blog/what-is-a-rest-api>.

ДОДАТОК А
Технічне завдання

Вступ

Робота зі створенням контенту у веб просторі супроводжується не лише текстом, а й зображеннями. Їх розміщення на сторінці є доволі важливим, адже це має бути зручним для перегляду користувачами, особливо якщо їх велика кількість. Тому існують плагіни галереї, що допомагають організувати зображення, проте не всі вони мають засоби для імпорту фото з хмарних сховищ.

А.1 Підстави для розробки

Підставою для виконання розробки є завдання на дипломний проект від 4 квітня 2023 року на тему «Програмна розробка плагіну CMS WordPress для роботи з галереєю», затверджене наказом № 87 по НУ «Запорізька політехніка».

А.2 Призначення розробки

Плагін має використовуватися для створення і роботи з галереєю в CMS WordPress, а також для імпорту зображень з Dropbox та Google Drive.

А.3 Основні вимоги до програми

А.3.1 Вимоги до функціональних характеристик

Набір функціональних вимог, які має забезпечувати плагін, включає:

- створення галереї;
- видалення галереї;
- редагування галереї;
- обрання для галереї зображення;
- видалення зображення у вже створеній галереї;

- додавання нових зображень до вже створеної галереї;
- вибір шаблону відображення галереї;
- зміна шаблону вже створеної галереї;
- додавати створенні галереї на сторінки або до поста;
- імпорт зображень з Dropbox;
- імпорт зображень з Google Drive;
- перегляд створеної галереї на сторінці сайту.

A.3.2 Умови експлуатації

Умовами експлуатації є виділення адміністратора, редактора або автора та учасника для роботи з галереєю і імпорту зображень. Редактор та автор мають ті ж самі можливості роботи з плагіном, що й адміністратор, але краще виділити його окремо. Учасники можуть працювати з галереєю, але не можуть імпортувати зображення.

A.3.3 Вимоги до складу та параметрів технічних засобів

Для підтримки роботи плагіну необхідно, щоб хостинг включав дисковий простір від 500 МБ, оперативну пам'ять від 512 МБ, вебсервер з підтримкою PHP і MySQL, версія PHP від 7 і вище, версія MySQL від 5.6 і вище, HTTPS протокол.

A.4 Порядок контролю та приймання

Дипломна кваліфікаційна робота має виконуватися у відповідності з визначеним календарним планом згідно встановлених завдань, їх строків і результатів виконання.

ДОДАТОК Б
Текст програми

Б.1 Текст файла gallery_plugin.php

```

<?php
/*
Plugin Name:  WP Gallery Plugin
Description:  Plugin for displaying images from the gallery,
google photos and dropbox on site pages using different templates
Version:      0.1
Author:       Tetiana Tykhonovska
*/

function admin_menu() {
    add_submenu_page( 'edit.php?post_type=custom_gallery', __(
'Connect API'), __( 'Connect API'), 'edit_posts',
'custom_gallery_connect_api', 'connect_api');
}
add_action('admin_menu', 'admin_menu');

function connect_api() {
    include_once sprintf("%s\admin\connect_api.php",
dirname(__FILE__));
}

include_once sprintf("%s\php\dropbox.php", dirname(__FILE__));
include_once sprintf("%s\php\upload_image_from_url.php",
dirname(__FILE__));

include_once sprintf("%s\php\google_picker.php",
dirname(__FILE__));

include_once sprintf("%s\php\gallery_post_type.php",
dirname(__FILE__));

```

```
include_once sprintf("%s\php\show_galleries.php",
dirname(__FILE__));
```

```
include_once sprintf("%s\php\galleries_button_in_post_edit.php",
dirname(__FILE__));
```

Б.2 Текст файла /admin/connect_api.php

```
<?php
$dropbox_app_key = "dropbox_app_key";
$google_app_key = "google_app_key";

if (array_key_exists($dropbox_app_key, $_POST))
    update_option($dropbox_app_key, $_POST["dropbox_app_key"]);

$value_dropbox_app_key = get_option($dropbox_app_key);

?>

<h1><?php _e( 'Image Galerry Connect API'); ?></h1>
<div class="for_dropbox">
    <p>Before using image import from Dropbox, <a
target="_blank"
href="https://www.dropbox.com/developers/apps/create">create an
app</a> and enter the key of the created app.</p>
    <form method="post">
        <label for="dropbox_app_key">Dropbox App key</label>
        <input type="text" class="dropbox_app_key"
name="dropbox_app_key" value="<?php echo $value_dropbox_app_key; ?>">
        <br><br>
        <?php submit_button(); ?>
    </form>
</div>
<?
```

Б.3 Текст файла /css/admin-style.css

```
.selected-gallery-content ul {
    display: flex;
    flex-direction: row;
}

.selected-gallery-content ul li {
    position: relative;
    margin-right: 10px;
}

.selected-gallery-content img {
    cursor: pointer;
}

.selected-gallery-content .gallery-image-remove {
    display: none;
    top: 0;
    right: 0;
    position: absolute;
    cursor: pointer;
}

.selected-gallery-content .gallery-image-remove:hover {
    display: block;
}

.selected-gallery-content img:hover+.gallery-image-remove {
    display: block;
}

.gallery-image-remove .dashicons-dismiss {
    color: red;
}
```

```

}

.custom-gallery-shortcode {
    width: 50%;
}

#add-img-to-gallery {
    margin-bottom: 10px;
}

```

Б.4 Текст файла /css/lightbox.css

```

.lightbox {
    display: none;
    position: fixed;
    z-index: 999;
    top: 0;
    left: 0;
    right: 0;
    bottom: 0;
    padding: 1em;
    background: rgba(0, 0, 0, 0.8);
}

.lightbox:target {
    display: block;
}

.lightbox span {
    display: block;
    width: 100%;
    height: 100%;
    background-position: center;
    background-repeat: no-repeat;
}

```

```
        background-size: contain;
    }
```

Б.5 Текст файла /css/window_with_galleries.css

```
.window-with-galleries {
    display: none;
    position: absolute;
    top: 30%;
    left: 40%;
    z-index: 999;
    background-color: #ffffff;
    width: 300px;
    border: 1px solid #000000;
    cursor: pointer;
    padding: 10px;
    overflow: auto;
}

.window-with-galleries li:hover {
    background-color: #f3f3f3;
}

.window-with-galleries .gallery-image-remove {
    top: 0;
    right: 0;
    position: absolute;
    cursor: pointer;
}

.window-with-galleries .gallery-image-remove:hover {
    opacity: 0.7;
}
```

Б.6 Текст файла /css/templates/double_trio.css

```
.gal-img-imglist {
    display: flex;
    position: relative;
    flex-direction: row;
    flex-wrap: wrap;
    max-width: 100% !important;
    width: 1000px;
    height: 520px;
    justify-content: space-between;
    margin: 10px auto !important;
}

.gal-img-imglist img {
    height: 250px;
    width: calc(960px/3);
    object-fit: cover;
}

.gal-img-imglist a:nth-child(-n + 3) {
    margin-bottom: 20px;
}

@media only screen and (max-width: 1000px) {
    .gal-img-imglist {
        width: 600px;
        height: 900px;
    }

    .gal-img-imglist img {
        width: calc(580px/2);
        height: 300px;
        padding-bottom: 20px;
    }
}
```

```

    }

    .gal-img-imglist a:nth-child(-n + 3) {
        margin-bottom: 0px;
    }
}

@media only screen and (max-width: 600px) {
    .gal-img-imglist {
        width: 300px;
        height: 600px;
        flex-direction: column;
    }

    .gal-img-imglist img {
        width: calc(280px/2);
        height: 200px;
        padding-bottom: 20px;
    }

    .gal-img-imglist a:nth-child(-n + 3) {
        margin-bottom: 0px;
    }
}

```

Б.7 Текст файла /css/templates/main_center.css

```

.gal-img-imglist {
    display: flex;
    position: relative;
    flex-direction: row;
    flex-wrap: nowrap;
    max-width: 100% !important;
    width: 1000px;

```

```
    height: 500px;
    justify-content: space-between;
    margin: 10px auto !important;
}
```

```
.left_part {
    display: flex;
    flex-direction: column;
    justify-content: space-between;
    height: 500px;
    width: calc(1000px/3);
}
```

```
.left_part a {
    width: 100%;
    height: 240px;
    margin-left: auto;
    margin-right: auto;
}
```

```
.left_part a>img {
    width: 100%;
    height: 100%;
    object-fit: cover;
}
```

```
.center_part {
    height: 500px;
    width: calc(1000px/3);
    margin-left: 20px;
    margin-right: 20px;
}
```

```
.center_part img {
```

```

        width: 100%;
        height: 100%;
        object-fit: cover;
    }

```

```

.right_part {
    display: flex;
    flex-direction: column;
    justify-content: space-between;
    height: 500px;
    width: calc(1000px/3);
}

```

```

.right_part a {
    width: 100%;
    height: 240px;
    margin-left: auto;
    margin-right: auto;
}

```

```

.right_part a>img {
    width: 100%;
    height: 100%;
    object-fit: cover;
}

```

```

@media only screen and (max-width: 1000px) {
    .gal-img-imglist {
        width: 500px;
        height: 400px;
    }

    .left_part {
        height: 400px;
    }
}

```

```

        width: calc(500px/3);
    }

    .left_part a {
        height: 190px;
    }

    .center_part {
        height: 400px;
        width: calc(500px/3);
    }

    .right_part {
        height: 400px;
        width: calc(500px/3);
    }

    .right_part a {
        height: 190px;
    }
}

@media only screen and (max-width: 600px) {
    .gal-img-imglist {
        width: 300px;
        height: 640px;
        flex-direction: column;
    }

    .left_part {
        flex-direction: row;
        height: 200px;
        width: 300px;
    }
}

```

```

.left_part a {
    width: 140px;
    margin-left: 0;
    margin-right: 0;
    margin-bottom: auto;
}

.center_part {
    height: 200px;
    width: 300px;
    margin-left: 0px;
    margin-right: 0px;
    margin-bottom: 20px;
    margin-top: 20px;
}

.right_part {
    flex-direction: row;
    height: 200px;
    width: 300px;
}

.right_part a {
    width: 140px;
    margin-left: 0;
    margin-right: 0;
    margin-top: auto;
}
}

```

Б.8 Текст файла /css/templates/main_left.css

```

.gal-img-imglist {

```

```
    display: flex;
    position: relative;
    flex-direction: row;
    flex-wrap: nowrap;
    max-width: 100% !important;
    width: 1000px;
    height: 500px;
    justify-content: space-between;
    margin: 10px auto !important;
}
```

```
.left_part {
    height: 500px;
    width: calc(960px/3);
}
```

```
.left_part img {
    width: 100%;
    height: 100%;
    object-fit: cover;
}
```

```
.center_part {
    display: flex;
    flex-direction: column;
    justify-content: space-between;
    height: 500px;
    width: calc(960px/3);
}
```

```
.center_part a {
    width: 100%;
    height: 240px;
    margin-left: auto;
```

```

        margin-right: auto;
    }

    .center_part a>img {
        width: 100%;
        height: 100%;
        object-fit: cover;
    }

    .right_part {
        display: flex;
        flex-direction: column;
        justify-content: space-between;
        height: 500px;
        width: calc(960px/3);
    }

    .right_part a {
        width: 100%;
        height: 240px;
        margin-left: auto;
        margin-right: auto;
    }

    .right_part a>img {
        width: 100%;
        height: 100%;
        object-fit: cover;
    }

    @media only screen and (max-width: 1000px) {
        .gal-img-imglist {
            width: 500px;
            height: 400px;
        }
    }

```

```

}

.left_part {
    height: 400px;
    width: calc(500px/3);
}

.center_part {
    height: 400px;
    width: calc(500px/3);
}

.center_part a {
    height: 190px;
    padding-left: 20px;
}

.right_part {
    height: 400px;
    width: calc(500px/3);
}

.right_part a {
    height: 190px;
    padding-left: 20px;
}
}

@media only screen and (max-width: 600px) {
    .gal-img-imaglist {
        width: 300px;
        height: 600px;
        flex-direction: column;
    }
}

```

```
.left_part {  
    height: 200px;  
    width: 300px;  
    padding-bottom: 20px;  
}  
  
.center_part {  
    flex-direction: row;  
    height: 200px;  
    width: 300px;  
}  
  
.center_part a {  
    width: 140px;  
    padding-left: 0px;  
    margin-left: 0px;  
    margin-right: 0px;  
}  
  
.right_part {  
    flex-direction: row;  
    height: 200px;  
    width: 300px;  
}  
  
.right_part a {  
    width: 140px;  
    padding-left: 0px;  
    margin-left: 0px;  
    margin-right: 0px;  
    margin-top: auto;  
}  
}
```

Б.9 Текст файла /css/templates/main_right.css

```
.gal-img-imglist {
    display: flex;
    position: relative;
    flex-direction: row;
    flex-wrap: nowrap;
    max-width: 100% !important;
    width: 1000px;
    height: 500px;
    justify-content: space-between;
    margin: 10px auto !important;
}

.left_part {
    display: flex;
    flex-direction: column;
    justify-content: space-between;
    height: 500px;
    width: calc(960px/3);
}

.left_part a {
    width: 100%;
    height: 240px;
    margin-left: auto;
    margin-right: auto;
}

.left_part a>img {
    width: 100%;
    height: 100%;
    object-fit: cover;
}
```

```

.center_part {
    display: flex;
    flex-direction: column;
    justify-content: space-between;
    height: 500px;
    width: calc(960px/3);
}

.center_part a {
    width: 100%;
    height: 240px;
    margin-left: auto;
    margin-right: auto;
}

.center_part a>img {
    width: 100%;
    height: 100%;
    object-fit: cover;
}

.right_part {
    height: 500px;
    width: calc(960px/3);
}

.right_part img {
    width: 100%;
    height: 100%;
    object-fit: cover;
}

@media only screen and (max-width: 1000px) {
    .gal-img-imglist {

```

```

        width: 500px;
        height: 400px;
    }

    .right_part {
        height: 400px;
        width: calc(500px/3);
    }

    .center_part {
        height: 400px;
        width: calc(500px/3);
    }

    .center_part a {
        height: 190px;
        padding-right: 20px;
    }

    .left_part {
        height: 400px;
        width: calc(500px/3);
    }

    .left_part a {
        height: 190px;
        padding-right: 20px;
    }
}

@media only screen and (max-width: 600px) {
    .gal-img-imglist {
        width: 300px;
        height: 600px;
        flex-direction: column;
    }
}

```

```
}
```

```
.right_part {  
    height: 200px;  
    width: 300px;  
    padding-top: 20px;  
}
```

```
.center_part {  
    flex-direction: row;  
    height: 200px;  
    width: 300px;  
}
```

```
.center_part a {  
    width: 140px;  
    padding-right: 0px;  
    margin-left: 0px;  
    margin-right: 0px;  
    margin-top: auto;  
}
```

```
.left_part {  
    flex-direction: row;  
    height: 200px;  
    width: 300px;  
}
```

```
.left_part a {  
    width: 140px;  
    padding-right: 0px;  
    margin-left: 0px;  
    margin-right: 0px;  
    margin-bottom: auto;
```

```
    }
}
```

Б.10 Текст файла /css/templates/trio.css

```
.gal-img-imglist {
    display: flex;
    position: relative;
    flex-direction: row;
    flex-wrap: nowrap;
    max-width: 100% !important;
    width: 1000px;
    height: 500px;
    justify-content: space-between;
    margin: 10px auto !important;
}

.gal-img-imglist img {
    height: 500px;
    width: calc(960px/3);
    object-fit: cover;
}

@media only screen and (max-width: 1000px) {
    .gal-img-imglist {
        width: 600px;
        height: 300px;
    }

    .gal-img-imglist img {
        height: 300px;
        width: calc(560px/3);
    }
}
```

```

@media only screen and (max-width: 600px) {
    .gal-img-imglist {
        flex-direction: column;
        width: 300px;
        height: 600px;
    }

    .gal-img-imglist img {
        width: 300px;
        height: calc(560px/3);
    }
}

```

Б.11 Текст файла /js/admin-scripts.js

```

jQuery(document).ready(function($) {
    var frame;
    var _attachments = $('.selected-images-ids').val();
    var _template = $('.gal-style').find(":selected").val();

    $(document).on('click', '#add-img-to-gallery', function
(event) {
        event.preventDefault();

        if (frame) {
            frame.open();
            return;
        }
        frame = wp.media({
            title: 'Select Media',
            button: {
                text: 'Use this media'
            },
            multiple: 'add',

```

```

        autoSelect: true
    });

    frame.on('select', function() {
        var attachments = frame.state().get('selection').map(
            function( attachment ) {
                attachment.toJSON();
                return attachment;
            }
        );

        var i;
        for (i = 0; i < attachments.length; i++) {
            $('.selected-gallery-content ul').append('<li data-attachment_id="'+attachments[i].id+'"><div class="gallery-image-remove"><span class="dashicons dashicons-dismiss"></span></div></li>');

            if (_attachments.length === 0) {
                _attachments = attachments[i].id;
            } else {
                _attachments = _attachments + ',' + attachments[i].id;
            }
        };

        $(".selected-images-ids").val(_attachments);
        $(".custom-gallery-shortcode").val("[custom_gallery ids='"+_attachments+"' template='"+_template+"']");
    });

    frame.open();
});

```

```

$(document).on("click", ".gallery-image-remove",
function(event) {
    event.preventDefault();
    event.stopPropagation();
    _attachments = $(".selected-images-ids").val();
    $(this).parent().remove();
    _attachments = '';
    $('.selected-gallery-content ul li').each(function(){
        if ( _attachments === "" ) {
            _attachments = $(this).attr('data-attachment_id');
        } else {
            _attachments = _attachments + ',' + $(this).attr('data-
attachment_id');
        }
    });
    $(".selected-images-ids").val(_attachments);
    $(".custom-gallery-shortcode").val("[custom_gallery
ids='"+_attachments+"' template='"+_template+"']");
});

$(document).on("click", ".gal-style", function() {
    _template = $('.gal-style').find(":selected").val();
    $(".custom-gallery-shortcode").val("[custom_gallery
ids='"+_attachments+"' template='"+_template+"']");
});
});

```

Б.12 Текст файла /js/dropbox-chooser.js

```

jQuery(document).ready(function($) {
    options = {
        success: function(files){
            var data = {
                'action': 'dropbox_ajax_action',

```

```

        'files_link': files
    };
    jQuery.ajax({
        type: "post",
        dataType: "json",
        url: dropbox_img_url.ajax_url,
        data: data,
        success: function(result){
            if (result != 0) {
                alert ('Files uploaded successfully.
Page will be reload.');
```

location.reload();

```

            } else {
                alert ('Something went wrong ...');
```

}

```

        }
    });
},
cancel: function(){
},
linkType: "direct",
multiselect: true,
extensions:['.png', '.jpg', '.jpeg'],
};
var button = Dropbox.createChooseButton(options);

document.getElementById("dropboxContainer").appendChild(button);
});

```

Б.13 Текст файла /js/embed-gallery-in-post.js

```

jQuery(document).ready(function($) {
    $(document).on('click', '.add_gallery_in_content',
function() {

```

```

        if ($('.window-with-galleries').css('display') ==
'none') {
            $('.window-with-galleries').css({"display":
"block"});
        } else {
            $('.window-with-galleries').css({"display":
"none"});
        }

    });

    $(document).on('click', '.post-item-box-content-title',
function() {
        var gallery_slug = $(this).attr('gallery_shortcode');

parent.tinyMCE.activeEditor.setContent(parent.tinyMCE.activeEditor.get
Content() + gallery_slug);
        $('.window-with-galleries').css({"display": "none"});
    });

    $(document).on('click', '.window-with-galleries .gallery-
image-remove', function() {
        $('.window-with-galleries').css({"display": "none"});
    });
});

```

Б.14 Текст файла /js/google-picker.js

```

jQuery(document).ready(function($) {
    const SCOPES =
'https://www.googleapis.com/auth/drive.metadata.readonly';
    const CLIENT_ID = '326033712728-
vhf41fkha67n8rpo19dn3bgditi8k7u.apps.googleusercontent.com';
    const API_KEY = 'AIzaSyBuz7Q4g8dYHC32ufUQh_8N390B6zR2kCw';

```

```

const APP_ID = '326033712728';
let tokenClient;
let accessToken = null;
let gisInited = false;

$('.google_drive_button').click(function() {
    gapi.load('client:picker', initializePicker);
});

async function initializePicker() {
    await
gapi.client.load('https://www.googleapis.com/discovery/v1/apis/drive/v
3/rest');

    tokenClient = google.accounts.oauth2.initTokenClient({
        client_id: CLIENT_ID,
        scope: SCOPES,
        callback: '',
    });
    tokenClient.callback = async (response) => {
        if (response.error !== undefined) {
            throw (response);
        }
        accessToken = response.access_token;
        await createPicker();
    };
    if (accessToken === null) {
        tokenClient.requestAccessToken({prompt: 'consent'});
    } else {
        tokenClient.requestAccessToken({prompt: ''});
    }
}

function createPicker() {

```

```

        const view = new
google.picker.View(google.picker.ViewId.DOCS);
        view.setMimeTypes('image/png,image/jpeg,image/jpg');
        const picker = new google.picker.PickerBuilder()
            .enableFeature(google.picker.Feature.NAV_HIDDEN)

.enableFeature(google.picker.Feature.MULTISELECT_ENABLED)
            .setDeveloperKey(API_KEY)
            .setAppId(APP_ID)
            .setOAuthToken(accessToken)
            .addView(view)
            .addView(new google.picker.DocsUploadView())
            .setCallback(pickerCallback)
            .build();
        picker.setVisible(true);
    }

    async function pickerCallback(data) {
        if (data.action === google.picker.Action.PICKED) {
            const document =
data[google.picker.Response.DOCUMENTS][0];
            const fileId = document[google.picker.Document.ID];
            const parentId = document['parentId'];

            $.ajax({
                method: "PATCH",
                url:
"https://www.googleapis.com/drive/v3/files/" + fileId + "?addParents=" +
parentId,

                headers: {
                    "Authorization": "Bearer " + accessToken,
                }
            })
            .done(function (result) {

```

```

        console.log(result);
        var data = {
            'action': 'google_ajax_action',
            'files_link': result
        };
        jQuery.ajax({
            type: "post",
            dataType: "json",
            url: google_img_url.ajax_url,
            data: data,
            success: function(result){
                if (result != 0) {
                    alert ('Files uploaded successfully.
Page will be reload.');
```

location.reload();

```

                } else {
                    alert ('Something went wrong ...');
```

}

```

            }
        });
    })
    .fail(function (result) {
        console.log(result);
    });
}
}
});

```

Б.15 Текст файла /js/lightbox.js

```

(function($) {
    $('.gal-img-imglist a').click(function(e){
        e.preventDefault();
        var href = $(this).attr('href');
```

```

        $('.lightbox span').css({'background-image': 'url("' +
href + '")' });
        $('.lightbox').css({'display': 'block'});
    });

    $('.lightbox').click(function(){
        $('.lightbox').css({'display': 'none'});
    });
})(jQuery);

```

Б.16 Текст файла /php/dropbox.php

```

<?php
function dropbox_custom_js() {
    $value_dropbox_app_key = get_option("dropbox_app_key");
    if (!empty($value_dropbox_app_key)) {
        echo '<script type="text/javascript"
src="https://www.dropbox.com/static/api/2/dropins.js" id="dropboxjs"
data-app-key="'. $value_dropbox_app_key .' "></script>';
        wp_enqueue_script('dropbox_chooser',
plugins_url('../js/dropbox-chooser.js', __FILE__), array(), '0.0.4');
        wp_localize_script('dropbox_chooser', 'dropbox_img_url',
array('ajax_url' => admin_url('admin-ajax.php')));
    }
}

add_action('admin_head', 'dropbox_custom_js');
add_action('wp_ajax_dropbox_ajax_action',
'dropbox_ajax_action');

function dropbox_ajax_action() {
    $result = '';
    $files = $_POST['files_link'];

    foreach($files as $file){
        $result .= rudr_upload_file_by_url($file["link"]);
    }
}

```

```

        $result .= '';
    }

    if (!empty($result)){
        echo $result;
    } else {
        echo 0;
    }

    wp_die();
}

add_action('post-plupload-upload-ui', 'mediaButton');
function mediaButton() {
    if (current_user_can('upload_files')) {
        echo '<div id="dropboxContainer"></div>';
    }
}

```

Б.17 Текст файла /php/galleries_button_in_post_edit.php

```

<?php
function add_gallery_custom_js() {
    wp_enqueue_script('embed_gallery_in_post',
plugins_url('../js/embed-gallery-in-post.js', __FILE__), array(),
'0.0.4');
}
add_action('admin_head', 'add_gallery_custom_js');

function window_with_galleries_style_admin($hook) {
    global $current_screen, $post;

    if (isset($post) && !empty($post)) {

```

```

        if (isset($current_screen->post_type) &&
$current_screen->post_type != "custom_gallery") {
            wp_enqueue_style('custom_gallery_adminstyle',
plugins_url('../css/window_with_galleries.css', __FILE__), array(),
'0.0.3');

            wp_enqueue_media();
        }
    }

    add_action ('admin_enqueue_scripts',
'window_with_galleries_style_admin');

    add_filter('media_buttons_context', function($context) {
        global $pagenow;

        if (in_array($pagenow, array('post.php', 'page.php', 'post-
new.php', 'post-edit.php'))) {
            $context .= '<button type="button" class="button
add_gallery_in_content"><span class="dashicons dashicons-format-
gallery" style="margin-top: 3px"></span> Add Gallery</button>';
        }

        return $context;
    });

    function gallery_window_with_posts(){
        global $current_screen, $post;
        if (isset($post) && !empty($post)) {
            if (isset($current_screen->post_type) &&
$current_screen->post_type != "custom_gallery") {
                $custom_query = new WP_Query(
                    array(
                        'post_type'=>'custom_gallery',
                        'posts_per_page'=>'10',

```

```

        'orderby' => 'post_modified',
        'post_status' => "publish",
        'order' => 'DESC',));

$posts = $custom_query->posts;

?>
<div class="window-with-galleries">
    <div class="gallery-image-remove"><span
class="dashicons dashicons-dismiss"></span></div>
    <ul class="window-with-galleries-body">
        <?php
            foreach($posts as $post) {
                $post_slug = $post->post_name;
                $post_name = $post->post_title;
                $ids = get_post_meta($post->ID,
'selected-images-ids', true);
                $gal_template = get_post_meta($post-
>ID, 'gal-style', true);
                ?>
                <li id="<?php echo $post_slug;
?>" class="el-gallery">
                    <div class="post-item-box-
content-title" gallery_shortcode="[custom_gallery ids='<?php echo
$ids; ?>' template='<?php echo $gal_template; ?>']">
                        <?php echo $post_name;
?>
                    </div>
                </li>
            <?php
        }
    ?>
</ul>
</div>

```

```

        <?php
            }
        }
    }
    add_action('edit_form_after_editor',
'gallery_window_with_posts', 10, 3);

```

Б.18 Текст файла /php/gallery_post_type.php

```

<?php
function gallery_create_post_type() {
    register_post_type('custom_gallery',
        array(
            'labels' => array(
                'name' => __('Image Galleries'),
                'add_new' => __('Add New Gallery')
            ),
            'public' => true,
            'show_ui' => true,
            'capability_type' => 'post',
            'query_var' => true,
            'menu_icon' => 'dashicons-format-gallery',
            'supports' => array('title'),
        )
    );
}
add_action('init', 'gallery_create_post_type');
function gallery_new_images_add_meta_box() {
    add_meta_box(
        'gallery_new_images_edit',
        __('Gallery images'),
        'gallery_new_images_edit_meta_box_field',
        'custom_gallery',
        'normal',

```

```

        'core'
    );
}
add_action('add_meta_boxes', 'gallery_new_images_add_meta_box',
1);

function gallery_new_images_edit_meta_box_field($post) {
    $ids = '';
    $arr = array();
    $ids = get_post_meta(get_the_ID(), 'selected-images-ids',
true);

    if (!empty($ids)) {
        $arr = explode(',', $ids);
    }
    $gal_template = get_post_meta(get_the_ID(), 'gal-style',
true);

    ?>
    <input type="hidden" class="selected-images-ids"
name="selected-images-ids" value="<?php if (isset($ids) &&
!empty($ids)) echo $ids; ?>">
    <a id="add-img-to-gallery" href="#" class="rl-gallery-
select button"><?php _e('Select Images')?></a>
    <div class="selected-gallery-content">
        <ul>
            <?php if (!empty($arr) && is_array($arr)):
foreach ($arr as $attach_id): ?>
                <li data-attachment_id="<?php echo
$attach_id; ?>">

                    <div class="gallery-image-remove"><span
class="dashicons dashicons-dismiss"></span></div>

```

```

        </li>
        <?php endforeach; endif; ?>
    </ul>
</div>
<label for="gal-style">Gallery Style</label>
<select class="gal-style" name="gal-style">
    <option value="trio" <?php selected($gal_template,
'trio'); ?>>Trio (3 images)</option>
    <option value="double_trio" <?php
selected($gal_template, 'double_trio'); ?>>Double trio (6
images)</option>
    <option value="main_center" <?php
selected($gal_template, 'main_center'); ?>>Main Center (5
images)</option>
    <option value="main_left" <?php
selected($gal_template, 'main_left'); ?>>Main Left (5 images)</option>
    <option value="main_right" <?php
selected($gal_template, 'main_right'); ?>>Main Right (5
images)</option>
</select>
<br><br>
<?php
    if (isset($ids) && !empty($ids)) {
        ?>
        <label for="custom-gallery-shortcode">Gallery
Shortcode</label>
        <input type="text" class="custom-gallery-shortcode"
name="custom-gallery-shortcode" value="[custom_gallery ids='<?php echo
$ids;?>' template='<?php echo $gal_template;?>']">
        <?php
    }
}

function gallery_new_postdata_save($post_id) {

```

```

        if (array_key_exists('selected-images-ids', $_POST)) {
            update_post_meta($post_id, 'selected-images-ids',
sanitize_text_field($_POST['selected-images-ids']));
        }
        if (array_key_exists('gal-style', $_POST)) {
            update_post_meta($post_id, 'gal-style',
sanitize_text_field($_POST['gal-style']));
        }
    }
    add_action('save_post', 'gallery_new_postdata_save');

function gallery_new_interface_enqueue_style_admin($hook) {
    global $current_screen, $post;

    if (isset($post) && !empty($post)) {
        if (isset($current_screen->post_type) &&
$current_screen->post_type == "custom_gallery") {
            wp_enqueue_style('custom_gallery_adminstyle',
plugins_url('../css/admin-style.css', __FILE__), array(), '0.0.3');
            wp_enqueue_script('custom_gallery_adminscripts',
plugins_url('../js/admin-scripts.js', __FILE__));
            wp_enqueue_media();
        }
    }
}

add_action ('admin_enqueue_scripts',
'gallery_new_interface_enqueue_style_admin');

function set_custom_gallery_columns($columns) {
    $newArray = array_slice($columns, 0, 2, true) +
array('gallery_shortcode' => 'Gallery Shortcode') +
array_slice($columns, 2, NULL, true);
    return $newArray;
}

```

```

    add_filter('manage_custom_gallery_posts_columns',
'set_custom_gallery_columns');

function set_custom_gallery_val__columns($column, $post_id) {
    $ids = '';
    $ids = get_post_meta(get_the_ID(), 'selected-images-ids',
true);
    $gal_template = get_post_meta(get_the_ID(), 'gal-style',
true);
    if (isset($ids) && !empty($ids)) {
        echo "[custom_gallery ids='". $ids ."' template='".
$gal_template ."'"]";
    }
}
add_action('manage_custom_gallery_posts_custom_column' ,
'set_custom_gallery_val__columns', 10, 2);

```

Б.19 Текст файла /php/google_picker.php

```

<?php
function google_custom_js_header() {
    echo '<script async defer
src="https://apis.google.com/js/api.js"></script>';
    echo '<script async defer
src="https://accounts.google.com/gsi/client"></script>';
    wp_enqueue_script('google_picker',
plugins_url('../js/google-picker.js', __FILE__), array(), '0.0.6');
    wp_localize_script('google_picker', 'google_img_url',
array('ajax_url' => admin_url('admin-ajax.php')));
}
add_action('admin_head', 'google_custom_js_header');

add_action('post-plupload-upload-ui', 'GoogleMediaButton');
function GoogleMediaButton() {

```

```

        if (current_user_can('upload_files')) {
            echo '<button class="google_drive_button" style="margin-top: 5px;">Choose from Google Drive</button>';
        }
    }

    add_action('wp_ajax_google_ajax_action', 'google_ajax_action');
    function google_ajax_action() {
        $result = '';
        $files = $_POST['files_link'];

        foreach($files as $file){
            $result .= rudr_upload_file_by_url($file["link"]);
            $result .= ' ';
        }

        if (!empty($result)){
            echo $result;
        } else {
            echo 0;
        }

        wp_die();
    }

```

Б.20 Текст файла /php/show_galleries.php

```

<?php
add_shortcode('custom_gallery', 'custom_gallery_func');
function custom_gallery_func($atts) {
    $attributes = shortcode_atts([
        'ids' => 'empty',
        'template' => 'empty',
    ], $atts);

```

```

$return = '<script
src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.1/jquery.min.js"
></script><script
src="https://ajax.googleapis.com/ajax/libs/jqueryui/1.12.1/jquery-
ui.min.js"></script>';

$return .= '<style gal_style="gal_style">';
if (isset($attributes['template'])) {
    $gal_style =
file_get_contents(plugins_url('../css/templates/'.
$attributes['template'] .'.css', __FILE__));
    $return .= $gal_style;
}
$lightbox_style =
file_get_contents(plugins_url('../css/lightbox.css', __FILE__));
$return .= $lightbox_style;
$return .= '</style>';

$_ids = array();
if (isset($attributes['ids'])) {
    $_ids = explode(",", $attributes['ids']);
}
$return .= '<div class="gal-img-imglist">';
if ($attributes['template'] == "trio" ||
$attributes['template'] == "double_trio") {
    foreach ($_ids as $_id) {
        $return .= '<a href="'.
wp_get_attachment_url((int)$_id) .'"></a>';
    }
} else if ($attributes['template'] == "main_left") {

```

```

        $return .= '<div class="left_part"><a href="'.
wp_get_attachment_url((int)$_ids[0]) .'"></a></div>';

        $return .= '<div class="center_part"><a href="'.
wp_get_attachment_url((int)$_ids[1]) .'"></a>
                <a href="'.
wp_get_attachment_url((int)$_ids[2]) .'"></a></div>';

        $return .= '<div class="right_part"><a href="'.
wp_get_attachment_url((int)$_ids[3]) .'"></a>
                <a href="'.
wp_get_attachment_url((int)$_ids[4]) .'"></a></div>';

    } else if ($attributes['template'] == "main_right") {
        $return .= '<div class="left_part"><a href="'.
wp_get_attachment_url((int)$_ids[0]) .'"></a>
                <a href="'.
wp_get_attachment_url((int)$_ids[1]) .'"></a></div>';

        $return .= '<div class="center_part"><a href="'.
wp_get_attachment_url((int)$_ids[2]) .'"></a>
                <a href="'.
wp_get_attachment_url((int)$_ids[3]) .'"></a></div>';

        $return .= '<div class="right_part"><a href="'.
wp_get_attachment_url((int)$_ids[4]) .'"></a></div>';

    } else if ($attributes['template'] == "main_center") {

```

```

        $return .= '<div class="left_part"><a href="'.
wp_get_attachment_url((int)$_ids[0]) .'"></a>
        <a href="'.
wp_get_attachment_url((int)$_ids[1]) .'"></a></div>';
        $return .= '<div class="center_part"><a href="'.
wp_get_attachment_url((int)$_ids[2]) .'"></a></div>';
        $return .= '<div class="right_part"><a href="'.
wp_get_attachment_url((int)$_ids[3]) .'"></a>
        <a href="'.
wp_get_attachment_url((int)$_ids[4]) .'"></a></div>';
    }

    $return .= '<a class="lightbox"
id="img1"><span></span></a></div>';

    $return .= '<script>';
    $lightbox_script =
file_get_contents(plugins_url('../js/lightbox.js', __FILE__));
    $return .= $lightbox_script;

    $return .= '</script>';

    return $return;
}

```

Б.21 Текст файла /php/upload_image_from_url.php

```

<?php
function rudr_upload_file_by_url( $image_url ) {

```

```

require_once( ABSPATH . 'wp-admin/includes/file.php' );

$temp_file = download_url( $image_url );

if( is_wp_error( $temp_file ) ) {
    return false;
}

$file = array(
    'name'      => basename( $image_url ),
    'type'      => mime_content_type( $temp_file ),
    'tmp_name' => $temp_file,
    'size'      => filesize( $temp_file ),
);

$sideoad = wp_handle_sideoad(
    $file,
    array(
        'test_form' => false
    )
);

if( ! empty( $sideoad[ 'error' ] ) ) {
    return false;
}

$attachment_id = wp_insert_attachment(
    array(
        'guid'           => $sideoad[ 'url' ],
        'post_mime_type' => $sideoad[ 'type' ],
        'post_title'     => basename( $sideoad[ 'file' ]
    ),
    'post_content'      => '',
    'post_status'       => 'inherit',

```

```
        ),
        $sideload[ 'file' ]
    );

    if( is_wp_error( $attachment_id ) || ! $attachment_id ) {
        return false;
    }

    require_once( ABSPATH . 'wp-admin/includes/image.php' );

    wp_update_attachment_metadata(
        $attachment_id,
        wp_generate_attachment_metadata( $attachment_id,
    $sideload[ 'file' ] )
    );

    return $attachment_id;

}
```

ДОДАТОК В
Слайди презентації

Національний університет «Запорізька політехніка»

Дипломна кваліфікаційна робота бакалавра

Програмна розробка плагіну CMS WordPress для роботи з галереєю
Software development of the CMS WordPress plugin to work with a gallery

Виконав
студент групи КНТ-210сп

Т.В. ТИХОНОВСЬКА

Керівник к.н.т., доцент

О.О. СТЕПАНЕНКО

Рисунок В.1 – Слайд № 1

Об'єкт, предмет, мета, завдання роботи

Об'єкт роботи – плагін CMS WordPress для роботи з галереєю.

Предмет роботи – плагін для роботи з галереєю у CMS WordPress, що також надає можливість імпорту зображень з Dropbox та Google Drive.

Мета роботи – розробка плагіну для забезпечення роботи з галереєю у CMS WordPress, що включає створення самої галереї, її додавання на сторінку чи пост та імпорту зображень з Dropbox та Google Drive.

Завдання роботи:

- аналіз програмних аналогів;
- визначення функціональних вимог до плагіну роботи з галереєю;
- проектування плагіну;
- реалізація і налагодження плагіну.

Рисунок В.2 – Слайд № 2

Вибір засобів проектування і розробки

Оскільки плагін буде використовуватися у CMS WordPress, що написаний на PHP – ця мова програмування і буде основною для проекту.

Для роботи інтерфейсом користувача будуть використовуватися HTML – це мова розмітки гіпертекстових документів, CSS – спеціальна мова за допомогою якої оформлюються усі веб сторінки та jQuery – бібліотека JavaScript. Ключова відмінність між JavaScript і jQuery полягає в тому, що JavaScript - це самостійна мова програмування, а jQuery - бібліотека JavaScript, що значно полегшує його використання, але не є окремою мовою.

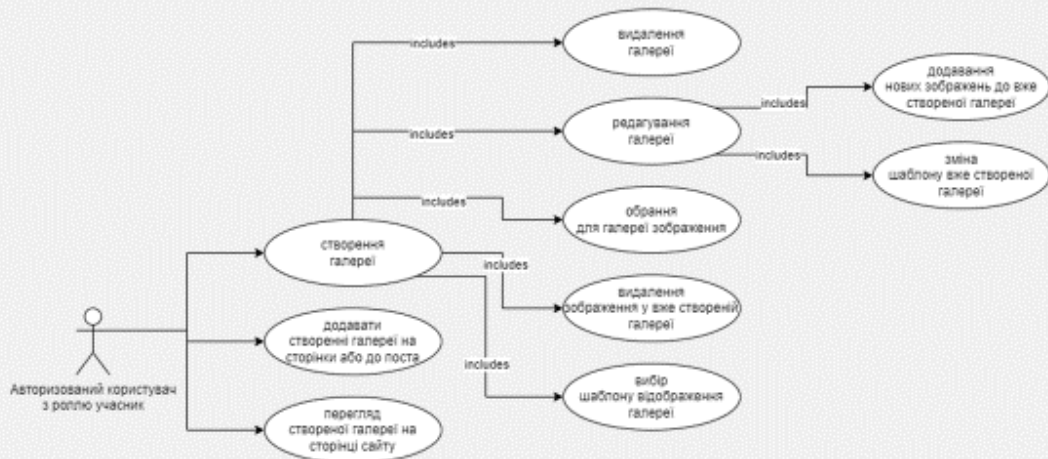
Також виникло питання як зв'язати імпорт зображень з Dropbox та Google Drive з WordPress, адже вони пропонують готові компоненти для розробників. У роботі їх використання повертає посилання на файл, а далі його потрібно обробити з використанням хуків WordPress і аби це зробити стало необхідним використання AJAX – технологія звернення до сервера без перезавантаження сторінки.

JavaScript	jQuery
Для досягнення кросбраузерної сумісності необхідно написати додатковий код (або змінювати наявний).	jQuery вже має вбудовану кросбраузерну функціональність.
Для створення анімації з використанням JavaScript потрібно багато рядків коду. В основному це досягається зміною стилів HTML сторінки.	jQuery дозволяє додавати ефекти анімації набагато швидше, використовуючи меншу кількість коду.
Розробникам необхідно написати власний скрипт, який потребує багато часу.	У бібліотечі є вже готові сценарії, якими розробник може користуватися.
Часто, для досягнення певного результату, потрібно написати велику кількість коду.	Дозволяє писати менше коду, досягаючи аналогічного результату.

3

Рисунок В.3 – Слайд № 3

Діаграма прецедентів



4

Рисунок В.4 – Слайд № 4



Рисунок В.5 – Слайд № 5

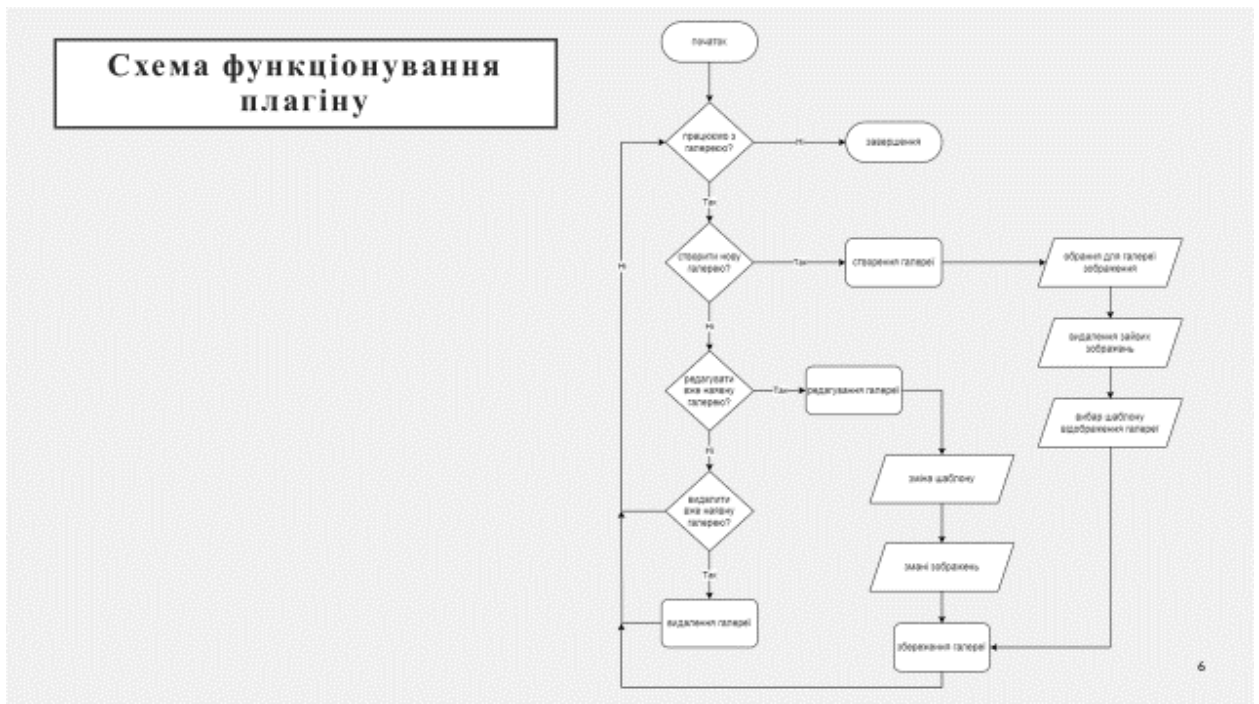


Рисунок В.6 – Слайд № 6

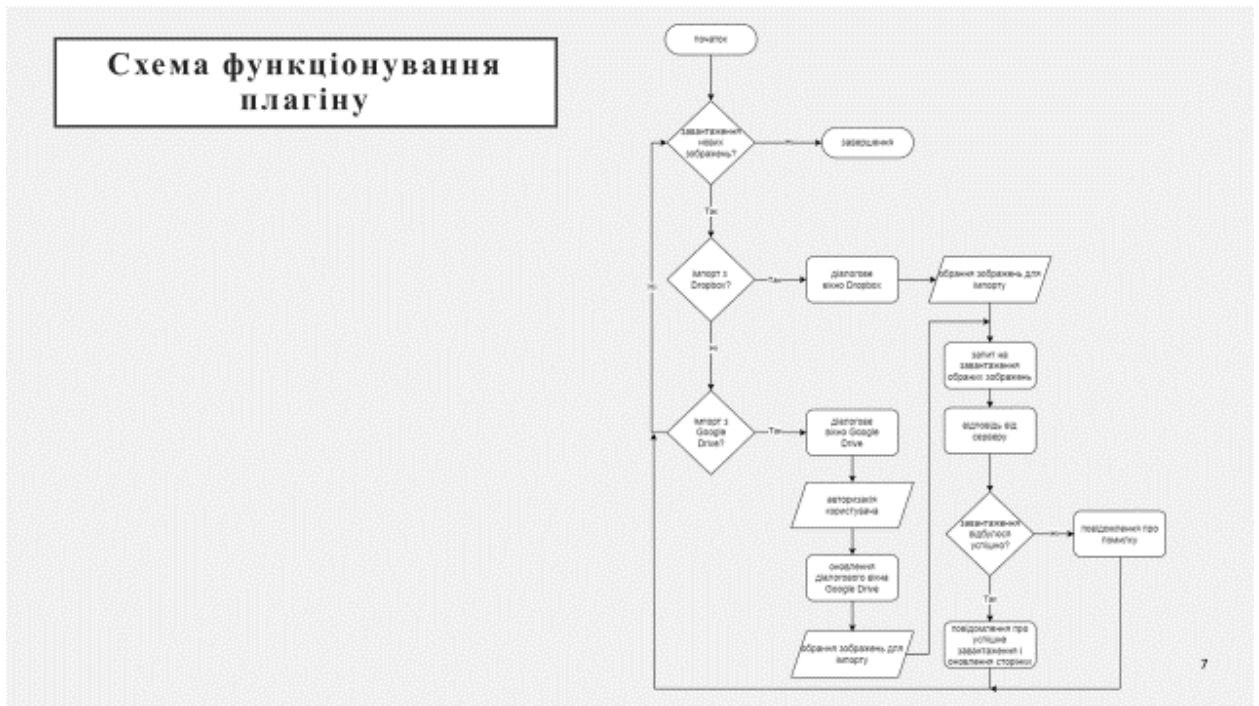


Рисунок В.7 – Слайд № 7

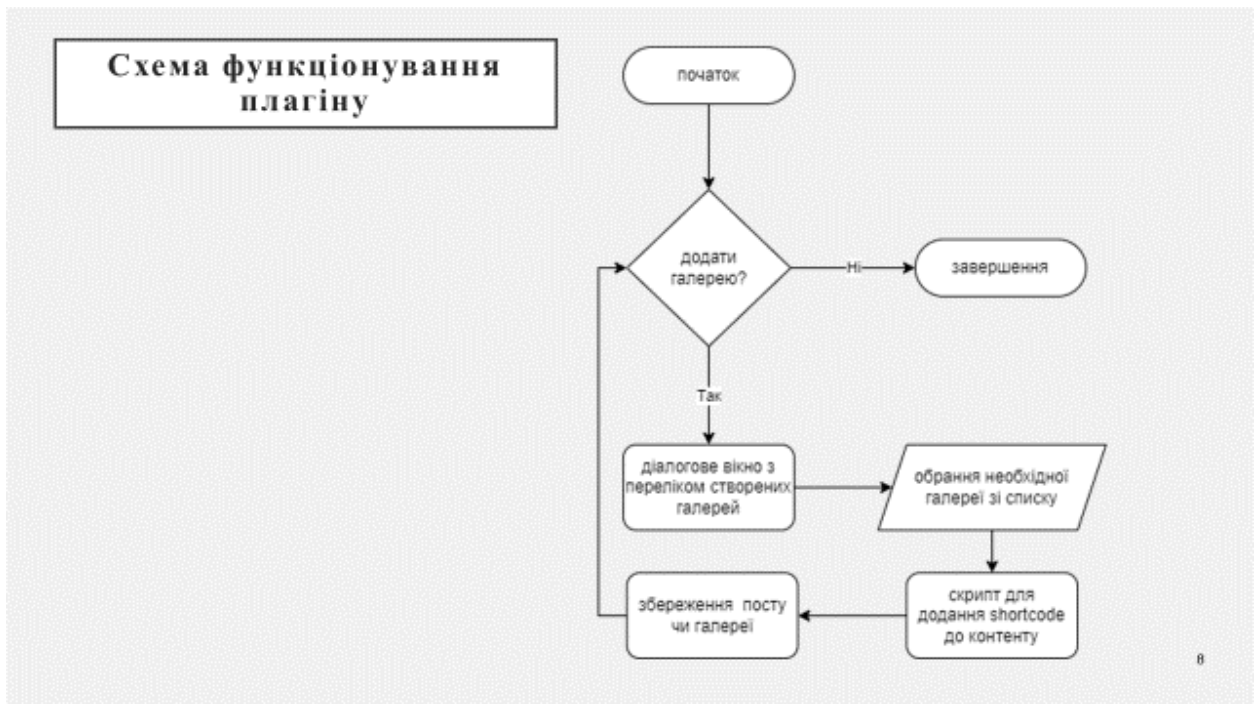


Рисунок В.8 – Слайд № 8

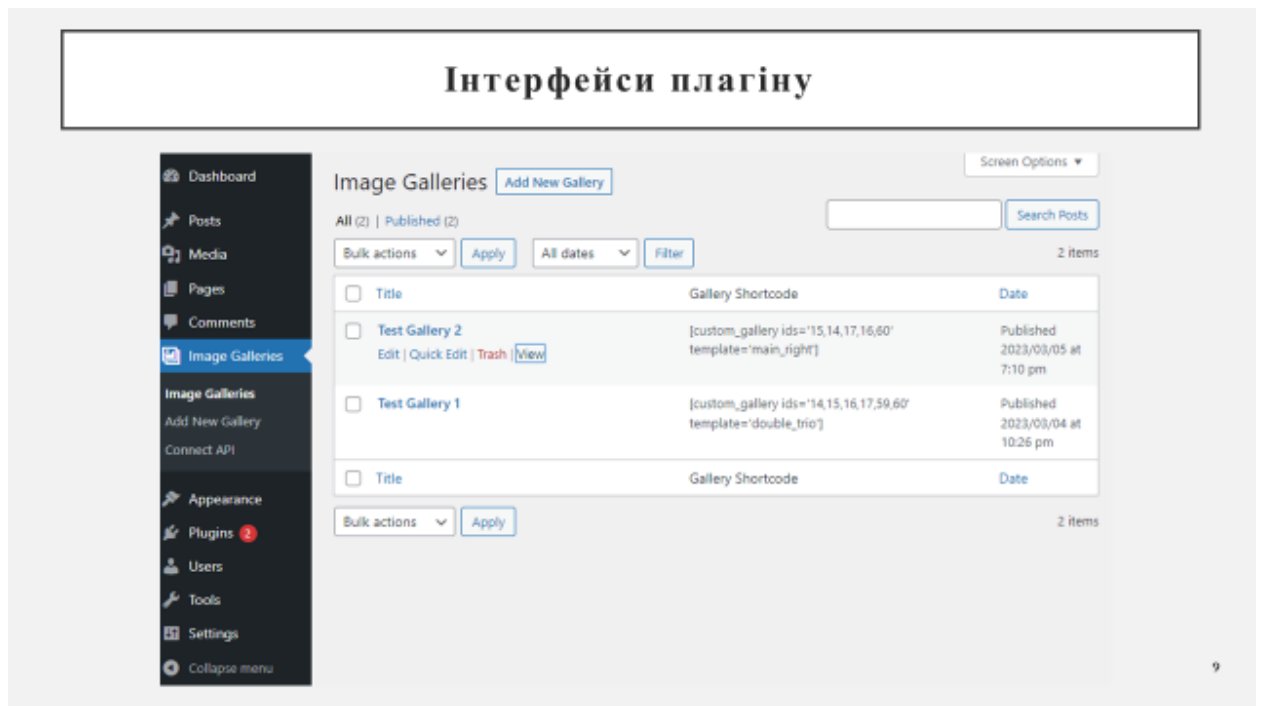


Рисунок В.9 – Слайд № 9

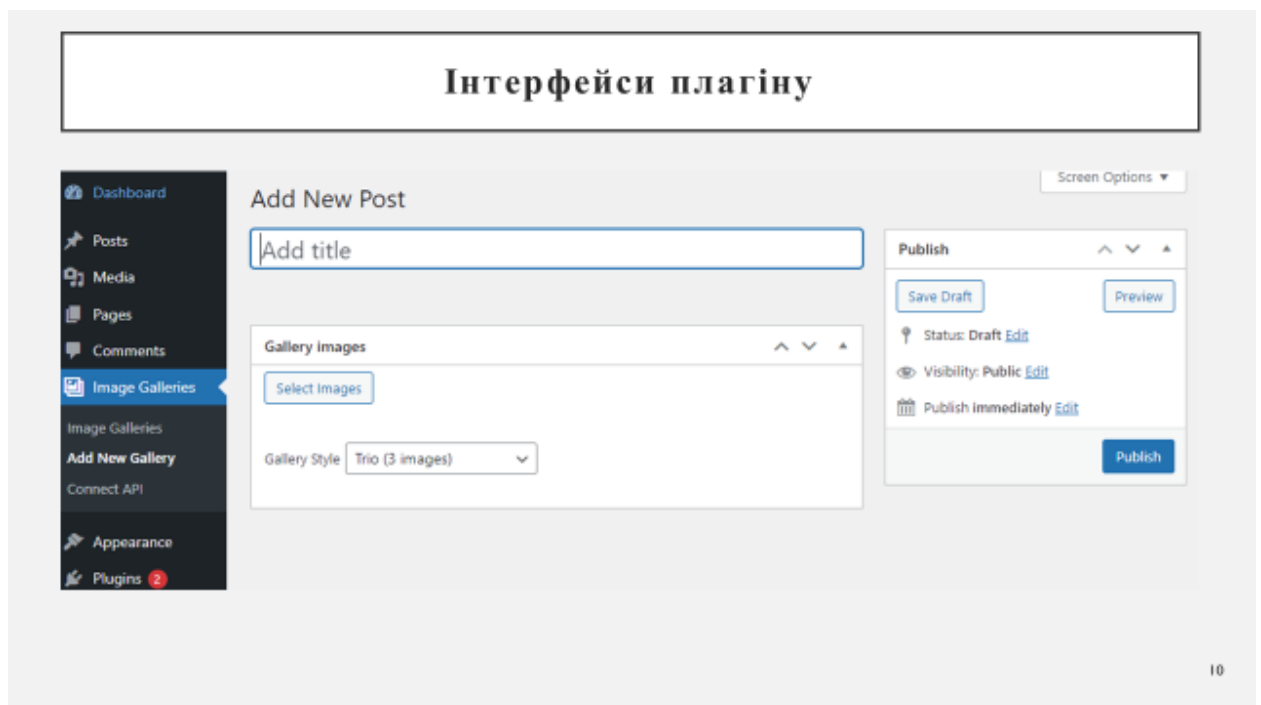


Рисунок В.10 – Слайд № 10

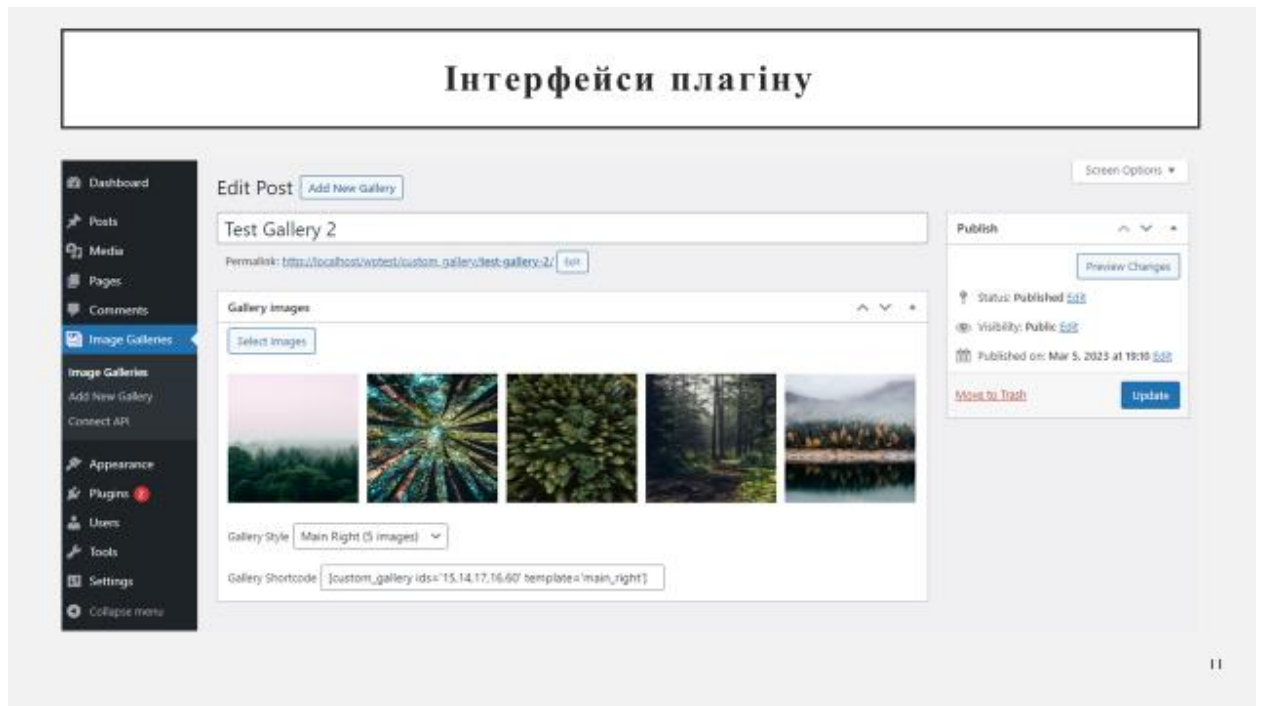


Рисунок В.11 – Слайд № 11

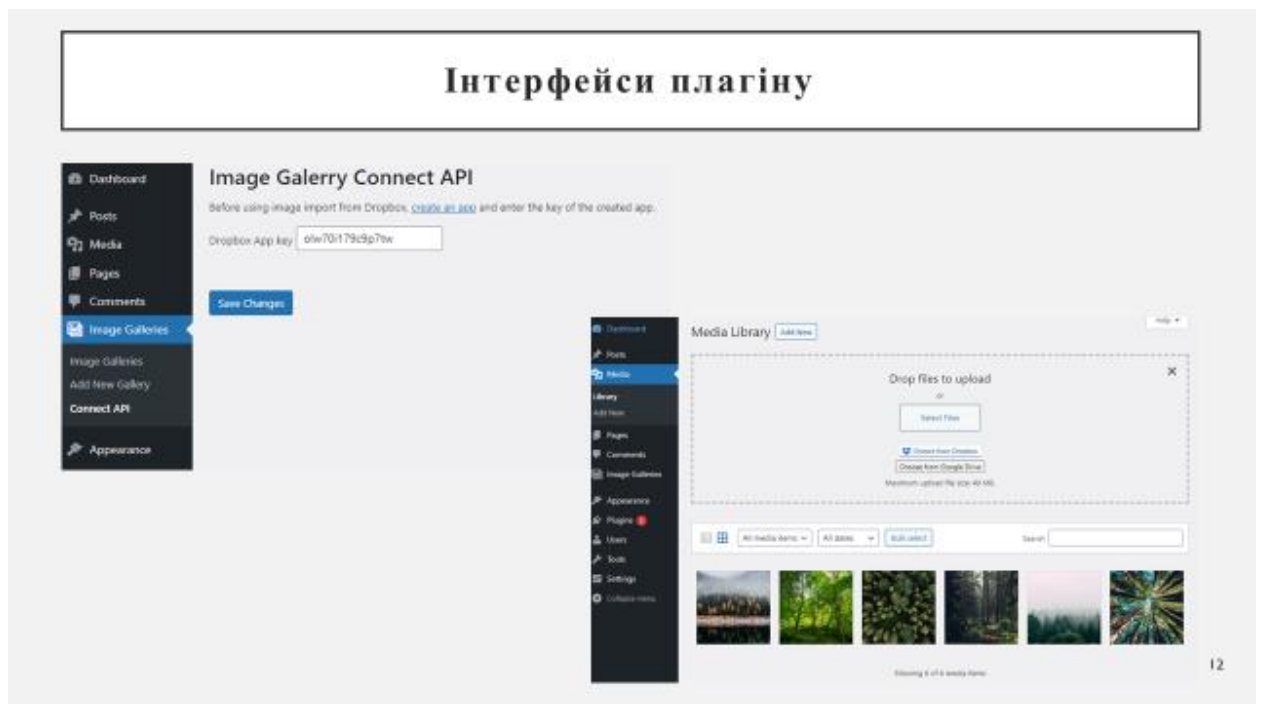


Рисунок В.12 – Слайд № 12

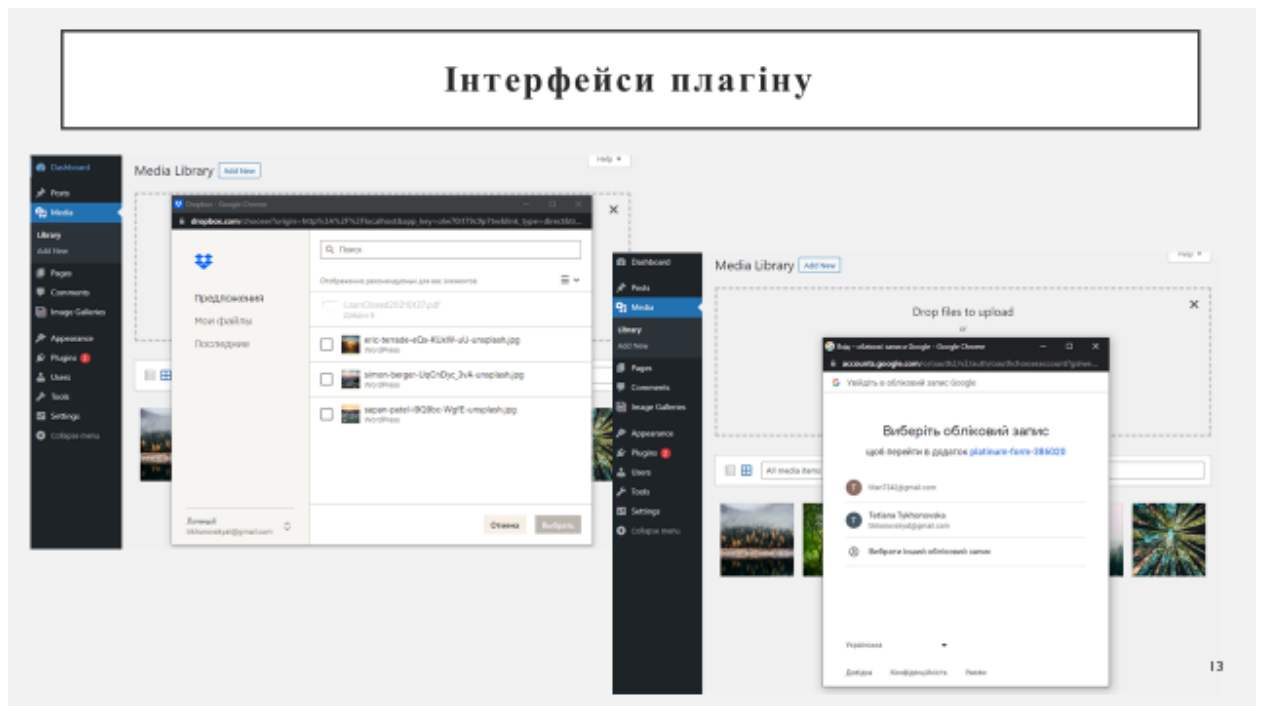


Рисунок В.13 – Слайд № 13

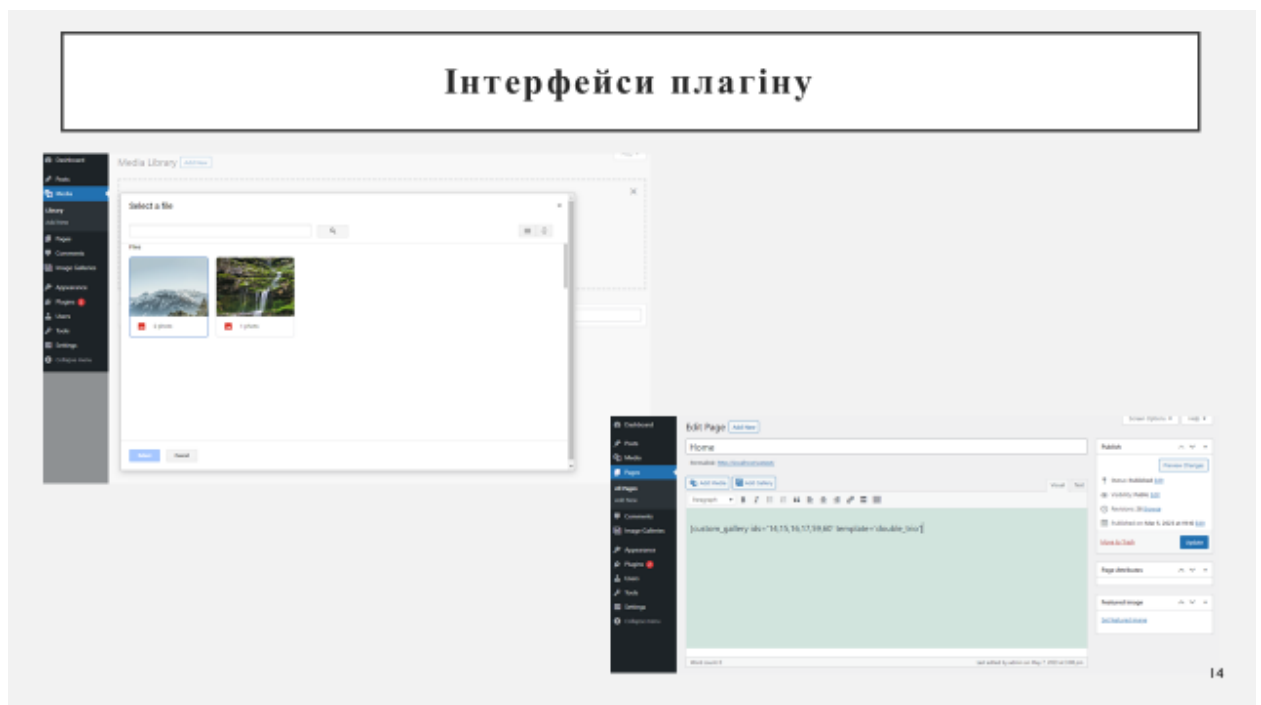


Рисунок В.14 – Слайд № 14

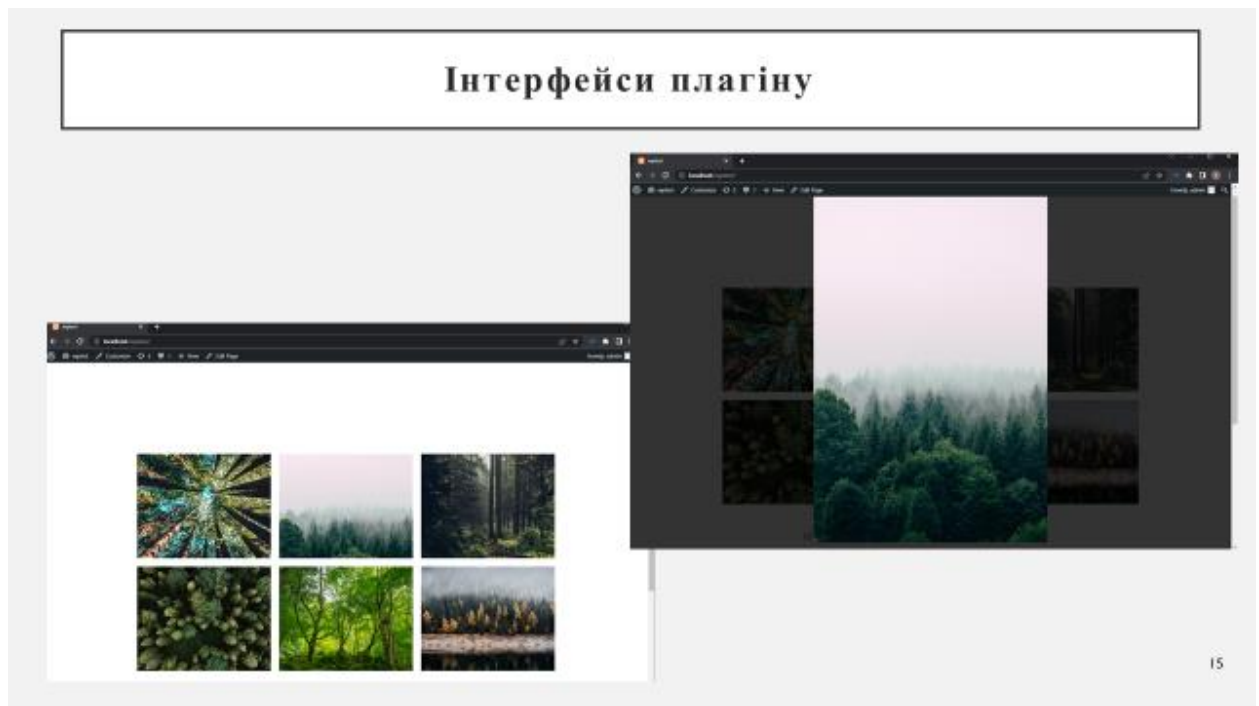


Рисунок В.15 – Слайд № 15

Висновки

Розробку плагіну було виконано на основі CMS WordPress з визначенням до плагіну, розробленням структури плагіну і алгоритму роботи на основі цих вимог, реалізацією відповідних одиниць вебзастосунку. Для роботи використано мову програмування PHP, мову розмітки HTML, мову стилю сторінок CSS, бібліотеку jQuery.

Плагін дозволяє створювати галерею, її редагувати та видаляти, додавати галерею на сторінку чи у пост, імпорт зображень з Dropbox та Google Drive. Також до створення галереї та її редагування відноситься можливість вибору зображень, їх видалення, додавання нових та вибір типу шаблону, що буде використовуватися, або його зміна.

Рисунок В.16 – Слайд № 16