МІНІСТЕРСТВО ФІНАНСІВ УКРАЇНИ
ДЕРЖАВНИЙ ПОДАТКОВИЙ УНІВЕРСИТЕТ

ФАКУЛЬТЕТ ФІНАНСІВ ТА ЦИФРОВИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І СИСТЕМ

КВАЛІФІКАЦІЙНА РОБОТА
на тему:

РОЗРОБКА СИСТЕМИ КЕРУВАННЯ ЗАХИСТОМ ВЕБ-САЙТУ

добувача вищої освіти першого (бакалаврського) рівня,
курсу, групи КБз-19-1, спеціальності 122 «Комп'ютерні науки»,
світня програма «Інформаційні управляючі системи і технології (за галузями)»

_____ Шульги Дмитра Миколайовича
(підпис) (прізвище, ім'я, по батькові)

Науковий керівник _____ Котух Євген Володимирович, к.т.н., доцент
(підпис) (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Керівник освітньо-професійної програми _____ Федотова - Півень Ірина Миколаївна, к.т.н, доцент
(підпис) (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Допущено до захисту за рішенням кафедри
протокол від «01» червня 2023 № 14

Завідувач кафедри _____ Філоненко Михайло Миколайович, к.ф.-м.н., доцент
(підпис) (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Ірпінь – 2023

Реєстрація № 44 07.06.2023 _____
(дата) (підпис відповідальної особи)

Результати перевірки до захисту 01.06.2023 _____
(до захисту, на доопрацювання) (дата) (підпис)

Результат захисту 5 (відмінно) 16.06.2023 _____
(оцінка) (дата) (підпис)

Голова комісії: _____ В.М. Франчук
(підпис)

Члени комісії: _____ М.М. Філоненко
(підпис)

_____ В.О. Ніжегородцев
(підпис)

_____ Є.В. Котух
(підпис)

БЛАНК ОЦІНЮВАННЯ ВИПУСКНОЇ РОБОТИ

Об'єкти контролю	Діапазон оцінки в балах	Фактична кількість балів	Примітки	Дата, П.І.Б.: керівника, голови комісії членів комісії,секретаря, (підписи)
ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ КОМІСІЄЮ ПО ЗАХИСТУ				
1. Зміст випускної роботи (якість теоретичного матеріалу, його зв'язок з практичним, вміння представляти матеріал в логічній послідовності, наявність висновків та пропозицій)	від 0 до 30	29		16.06.2023 Франчук В.М. Філоненко М.М Ніжегородцев В.О. Котух Є.В. Суховецька Л.П.
2. Дотримання вимог щодо оформлення	від 0 до 10	8		
3. Вчасне виконання роботи на всіх етапах у встановлені кафедрою терміни	від 0 до 5	3		
4. Рецензент	від 0 до 5	5		
РАЗОМ БАЛІВ	від 0 до 50	45		

Оцінювання кваліфікаційної роботи науковим керівником - 45 балів.

ПЕРЕВЕДЕННЯ ОЦІНКИ ЗА 100-БАЛЬНОЮ ШКАЛОЮ ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалоюЄКТС	Оцінка за національною шкалою	Дата	П.І.Б.: голови, членів комісії, секретаря, (підпис)
90	A	5 (відмінно)	16.06.2023	Франчук В.М. Філоненко М.М Ніжегородцев В.О. Котух Є.В. Суховецька Л.П.

ПРАВИЛА ПЕРЕВЕДЕННЯ ОЦІНОК ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалою ЄКТС	Оцінка за національною шкалою
90-100	A	5 (відмінно)
80-89	B	4 (добре)
70-79	C	
60-69	D	3 (задовільно)
50-59	E	
35-49	FX	2 (незадовільно)

культет фінансів та цифрових технологій
федра комп'ютерних та інформаційних технологій і систем
узь знань: 12 «Інформаційні технології»
еціальність: 122 «Комп'ютерні науки»
вітня програма: «Інформаційні управляючі системи і технології (за галузями)»

Затверджую
Завідувач кафедри
_____ Філоненко М.М.
«16» 11 2022

Завдання на кваліфікаційну роботу
здобувачу вищої освіти
Шульзі Дмитру Миколайовичу
(прізвище, ім'я, по батькові)

Тема кваліфікаційної роботи
озробка системи керування захистом веб-сайту"
тверджена наказом ректора від «15» листопада 2022 № 1205 (зі змінами наказ
д «04» квітня 2023 № 361).
Строк здачі закінченої роботи «01» червня 2023.
Мета роботи: дослідити захищеність веб-сайтів, створених за допомогою
истем керування вмістом.
б'єкт дослідження: системи керування вмістом
редмет дослідження: дослідження систем керування захисту веб-сайту.
аза дослідження: посібники та підручники.
Вихідні дані до роботи розробка системи керування захистом веб-сайту.
Перелік графічного матеріалу рисунки та таблиці.
Зміст випускної кваліфікаційної роботи (перелік питань за кожним розділом)

ВСТУП
1. ІНФОРМАЦІЙНИЙ ОГЛЯД
1.1 Аналіз можливих загроз для вебсайтів.
1.2 Загальна характеристика систем керування вмістом
1.3 Постановка задачі
2. СТВОРЕННЯ ВЕБСАЙТІВ ЗА ДОПОМОГОЮ CMS ТА АНАЛІЗ ПЛАГІНІВ БЕЗПЕКИ
2.1 Створення вебсайту за допомогою WordPress
2.2 Створення вебсайту за допомогою Joomla
2.3 Опис плагінів, використаних для розроблення вебсайтів
3. МЕТОДИКА ТА РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ
3.1 Методи та засоби для аналізу захищеності вебсайтів
3.2 Організація дослідження
3.3 Результати проведеного дослідження
ВИСНОВКИ
СПИСОК ЛІТЕРАТУРИ

7. Календарний план виконання роботи

№ п/п	Назва етапів кваліфікаційноїроботи	Строк виконання етапів роботи	
		За планом	Фактичн
1	2	3	4
1	Вибір теми, підбір літературних джерел, формування структури кваліфікаційної роботи	До 15.11.2022	15.11.2022
2	**І розділ** (розкрити сутність і стан дослідження наукової проблематики роботи, після чого навести її загальну характеристику; огляд літератури за темою і вибір конкретних напрямів досліджень)	06.02.2023	06.02.2023
3	**ІІ розділ** (аналіз експериментальної частини про методики досліджень)	31.03.2023	31.03.2023
4	**ІІІ розділ** (проведені теоретичні і (або) експериментальні дослідження та проектні рішення)	05.05.2023	05.05.2023
5	**Висновки**	16.05.2022	16.05.2022
6	Доопрацювання роботи з урахуванням зауважень наукового керівника	01.06.2023	01.06.2023
7	Отримання відгуку керівника	08.06.2023	08.06.2023
8	Отримання зовнішньої рецензії	11.06.2023	11.06.2023

8. Дата видачі завдання «16» листопада 2022

9. Керівник випускної кваліфікаційної роботи _____ (підпис) Котух Є.В. (прізвище, ініціали)

10. Керівник освітньо-професійної програми _____ (підпис) Федотова-Пі (прізвище, ініціали)

11. Завдання до виконання отримав здобувач вищої освіти _____ (підпис) Шульги Д.М. (прізвище, ініціали)

АНОТАЦІЯ

РОЗРОБКА СИСТЕМИ КЕРУВАННЯ ЗАХИСТОМ ВЕБСАЙТУ

Метою роботи є аналіз систем керування вмістом веб-сайтів з точки зору їх можливостей для створення безпечних продуктів, а також проведення дослідження створених веб-сайтів з використанням спеціалізованого інструментарію на їх захищеність від кібератак.

Завданнями роботи є: проаналізувати можливі загрози для веб-сайтів; дати загальну характеристику систем керування вмістом; поставити задачі щодо системи керування захистом веб-сайту; запропонувати підходи щодо створення веб-сайту за допомогою WordPress; виробити підходи щодо створення веб-сайту за допомогою Joomla; описати плагіни, що використовуються для розроблення веб-сайту; запропонувати методи та засоби для аналізу захищеності веб-сайтів.

Об'єктом дослідження є система керування захистом веб-сайту.

Предметом дослідження є підходи до розробки системи керування захистом веб-сайту.

Базою досліджень є аналітичні дані, а також наукові публікації та методичні матеріали.

За результатами дослідження сформульовані висновки та рекомендації, що можуть бути використані у діяльності щодо розробки системи керування захистом веб-сайту.

Одержані результати є важливими для користувачів, які займаються розробкою системи керування захистом веб-сайту, оскільки він має значну практичну цінність.

Кваліфікаційна робота містить 52 сторінки, 1 таблицю, 52 рисунки, список використаних джерел із 25 найменувань.

Ключові слова: веб-сайт, плагін, захист, система керування.

Рік виконання кваліфікаційної роботи – 2022 – 2023 рр.

Рік захисту роботи – 2023 р.

ABSTRACT

DEVELOPMENT OF WEBSITE PROTECTION MANAGEMENT SYSTEM

The purpose of the work is to analyze website content management systems from the point of view of their ability to create secure products, as well as to conduct a study of created websites using specialized tools for their security against cyber attacks.

The tasks of the work are: to analyze possible threats to websites; give a general description of content management systems; set tasks regarding the website's security management system; suggest approaches to creating a website using WordPress; develop approaches to creating a website using Joomla; describe the plugins used to develop the website; propose methods and tools for website security analysis.

The object of the study is the website security management system.

The subject of research is approaches to the development of a website security management system.

The basis of research is analytical data, as well as scientific publications and methodical materials.

Based on the results of the research, conclusions and recommendations are formulated that can be used in activities related to the development of a website protection management system.

The obtained results are important for users involved in the development of a website security management system, as it has significant practical value.

The qualification work contains 52 pages, 1 table, 52 figures, a list of used sources from 25 titles.

Keywords: website, plugin, protection, management system.

The year of the qualification work is 2022-2023.

The year of job protection is 2023.

ЗМІСТ

ВСТУП 1
1 ІНФОРМАЦІЙНИЙ ОГЛЯД 3
1.1 Аналіз можливих загроз для вебсайтів 3
1.2 Загальна характеристика систем керування вмістом 5
1.3 Постановка задачі 11
2 СТВОРЕННЯ ВЕБСАЙТІВ ЗА ДОПОМОГОЮ CMS ТА АНАЛІЗ ПЛАГІНІВ БЕЗПЕКИ 12
2.1 Створення вебсайту за допомогою WordPress 12
2.2 Створення вебсайту за допомогою Joomla 19
2.3 Опис плагінів, використаних для розроблення вебсайтів 21
3 МЕТОДИКА ТА РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ 24
3.1 Методи та засоби для аналізу захищеності вебсайтів 24
3.2 Організація дослідження 26
3.3 Результати проведеного дослідження 31
ВИСНОВКИ 49
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 50

ВСТУП

Із розвитком технологій засоби зберігання та передачі інформації у мережі стали головним фактором, що визначає загальний рівень розвитку суспільства та держави у цілому. Отже, можна сказати, що інформація - це один з головних ресурсів, за допомогою якого можна здійснити будь-які дії, а безпеку інформаційного простору є головною складовою безпеки будь-якої держави. Із кожним роком зростає роль інформаційного чинника у сучасному суспільстві, постійне вдосконалення інформаційних технологій актуалізують питання щодо інформаційної безпеки.

Все більша частина компаній почали створювати власні вебсайти, що спочатку слугували як візитівка що знаходиться у мережі, та з плином часу ці вебсайти почали використовуватися в якості основного інструменту у бізнесі, за допомогою якого можна буде отримати доступ до важливої інформації, контактувати з клієнтами, підтверджувати замовлення тощо.

Хоча у сучасному світі багато людей мають особисті вебсайти, проте далеко не кожен може створити вебсайт з нуля. Для такого користувача були розроблені Content Management System (CMS) системи, що є зручними й інтуїтивно зрозумілими інструментами задля створення вебсайтів, які не потребують спеціальних знань для їх ефективного використання. У мережах існує безліч CMS систем, де кожна з яких різниться від іншої та має свою специфіку.

З кожним днем різних ебсайтів стає дедалі більше, що підтверджує те, що у мережі Інтернет є перспективи для розвитку. Але, з іншого боку, це надає більше можливостей для кіберзлочинців. Тим самим, забезпечення безпеки користувачів у мережі стає сьогодні нагальною необхідністю.

Метою роботи є аналіз систем керування вмістом вебсайтів з точки зору їх можливостей для створення безпечних продуктів, а також проведення

дослідження створених вебсайтів з використанням спеціалізованого інструментарію на їх захищеність від кібератак.

Об’єктом дослідження є система керування захистом вебсайту.

Предметом дослідження є підходи до розробки системи керування захистом вебсайту.

- Основними завдання роботи є:
- Проаналізувати можливі загрози для вебсайтів;
- Дати загальну характеристику систем керування вмістом;
- Поставити задачі щодо системи керування захистом вебсайту
- Запропонувати підходи щодо створення вебсайту за допомогою WordPress
- Виробити підходи щодо створення вебсайту за допомогою Joomla;
- Описати плагіни, що використовуються для розроблення вебсайту
- Запропонувати методи та засоби для аналізу захищеності вебсайтів

1 ІНФОРМАЦІЙНИЙ ОГЛЯД

1.1 Аналіз можливих загроз для вебсайтів

В світі існує величезна кількість вебсайтів, на які можна здійснити кібератаки. На перший погляд, може так здатись, що шанс піддатись атаці є дуже низьким, і мотивувати це тим, що нікому не цікавий сам звичайний вебсайт, проте завжди є шанс відносно того, що веб-ресурс є тренувальною площиною для кіберзлочинця. Тож завжди потрібно звернути увагу на безпеку інтернет-ресурсів у першу чергу, задля того, щоб уберегти цілісність та конфіденційність інформації, що знаходиться у середині веб-ресурсів. Таким чином, кожний власник має бути ознайомленим із рекомендаціями щодо безпеки.

Як приклад того, що захист вебсайту є необхідним, можна навести DDoS-атаки на вебсайт Президента України, яку було нейтралізовано спеціалістами у сфері кібербезпеки, і це була не єдина атака, яку було направлено на державний або комерційний вебсайт. Із жовтня 2019 року у мережах ширився ботнет за назвою KashmirBlack, який атакував веб-ресурс, що використовували популярні системи управління вмістом, він використав вже відомі вразливості на сервері жертви, у середньому виконуючи мільйон атак на день, тим самим нагадуючи користувачам про необхідність дотримуватися певних правил безпеки [1].

У системах управління вмістом (Content Management System, CMS) більшість експлойту припадає на сторону браузерів, наприклад, довільні коди виконуються браузером локально, що призводить до непередбачуваних наслідків, таких, як витік інформації або викликає критичні помилки. Також додатковий ризик для користувачів CMS систем створюють спливаюче вікно, плагін та додаток, що працюють у веб-браузерах. Велика кількість таких атак можлива, скільки кібер-злочинці викрадають акаунти розробників і випускають оновлення, до яких додано небезпечний код. Не слід давати доступ до коду

доповнення кінцевим користувачам і необхідно забезпечити розроблення додатків за допомогою API WCMS.

Безпека CMS може бути порушено із допомогою стороннього коду на стороні клієнтів. Основна проблема полягає у тому, що більшість додатків включають у себе сценарії, що викликають інтерфейси API для оновлення серверних даних системи управління вмістом, за допомогою чого кіберзлочинець може надіслати шкідливий запит. Одним із прикладів є SQL команди, несподівані формати та сценарії. Таким чином було зламано API Drupal [2].

Однак більшість корустувачів вважають, що оскільки WordPress, Drupal і Joomla є популярними і через свою популярність вони повинні мати надійний захист. Однак через свою популярність вони є привабливими мішенями для кібер-злочинців. Як приклад можна навести систему управління вмістом під назвою WordPress, яка містить у своєму ядрі, темах та плагінах приблизно 14 000 тис. відомих вразливостей. Приблизно 55% усіх атак проводяться через встановлені плагіни, які мають слабкий захист або були покинуті своїми розробниками. Також однією з головних вразливостей є встановлення простого логіну та пароля, що надає змогу зламати вебсайт за допомогою простого підбору даних відомого як “Brute force” [10]. Одним з надійних способів захисту, який може надати перевагу, – це використання без головного підходу, який відповідає за універсальний контент (бекенд), за допомогою якого можна автоматизувати розповсюдження контенту на інші платформи, які використовують дане «тіло».

Системи управління контентом, що використовують такий підхід, надають можливість до більш жорсткого контролю за доступом до самої CMS.

Не зважаючи на все описане вище, необхідно оновлювати додатки, сценарії й інструменти. Через те, що старі версії можуть містити у собі ресурси, які закриваються з виходом нової версії, і зменшують шанси того, що сайт буде зламаний, але необхідно пам’ятати, що оновлення може бути випущене без

відома автора, і може містити небезпечний код. Ще одним способом надійних захистів є пароль, який відповідає усім сучасним стандартам, тобто пароль має містити велику літеру, спеціальний символ, маленьку літеру, цифру та містити вісім символів. Задля збільшення надійності, розташування символів може бути довільним. Є випадки, коли користувачі не може придумати надійні паролі, у таких випадках більшість сайтів дають користувачу випадково згенерований пароль, використання якого унеможливить швидкий злам акаунту користувача та підвищить рівень безпеки даних, які зберігаються на сайті [3].

1.2 Загальна характеристика систем управління вмістом

Усі CMS системи виконують одну Й ту саму функцію швидкого створення вебсайтів без спеціального знання. Розглянемо деякі найбільш поширені CMS системи.

1.2.1 WordPress

WordPress – це система управління вмістом із відкритим кодом, що використовує мови програмування PHP і СУБД MySQL, що підтримуються великою кількістю хостингів. Проте з-поміж безкоштовного використання можна придбати план, який дає більше різних можливостей. Зазвичай цю CMS систему використовують задля створення простих і маленьких вебсайтів, наприклад, блогів. За допомогою вбудованих інструментів вебсайт, який було створено в WordPress можна перетворити на будь-який інший тип ресурсів.

Основними перевагами є:

- Величезна кількість мов;
- Простота в освоєнні;
- Велика різноманітність інструментів персоналізації;
- Використання різних плагінів;
- Простота в оновленні;

- Відкриті програмні коди.

Основними недоліками є:

- Час завантаження сторінок;
- Проблеми із системою безпеки [4].

Для розробників сторінки WordPress мають вигляд, що наведено на рис. 1.1, 1.2 і 1.3.

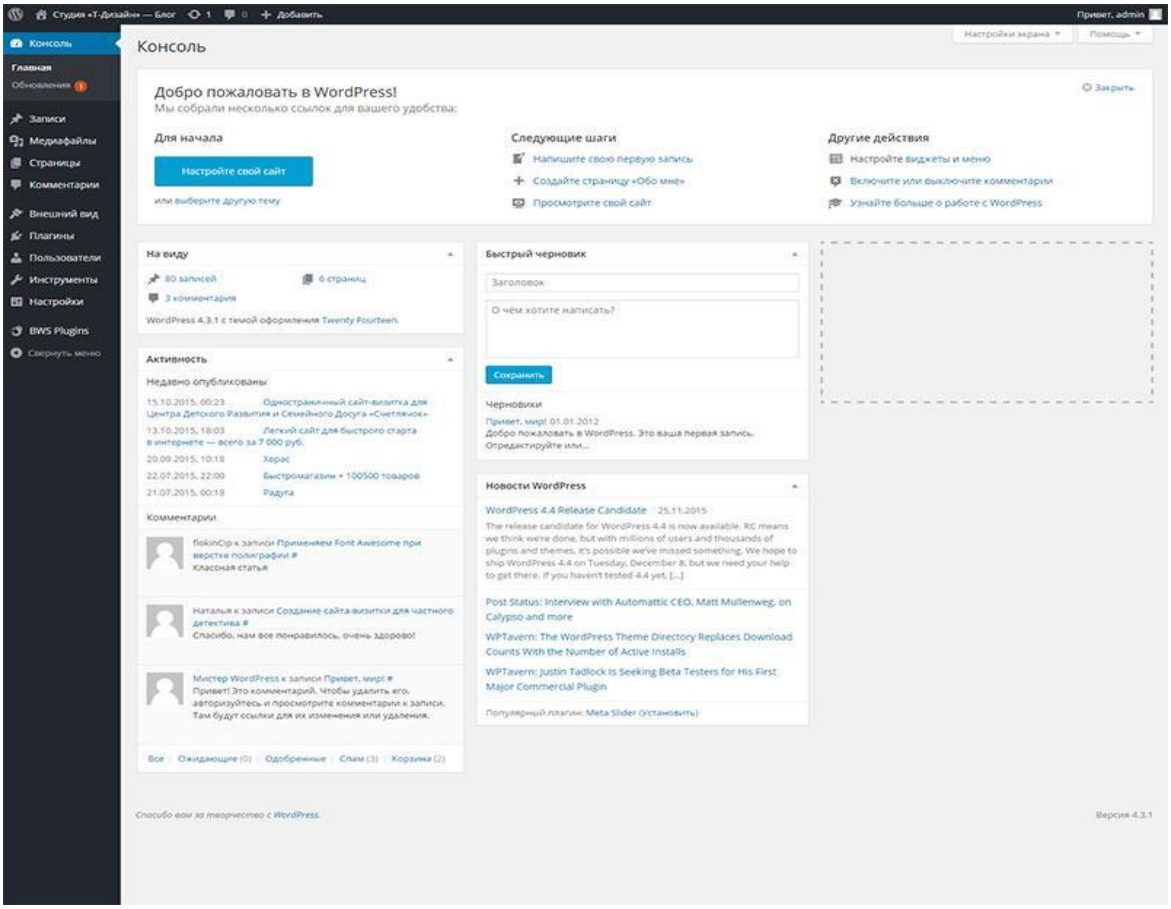


Рис. 1.1. Інтерфейс головної сторінки розробника

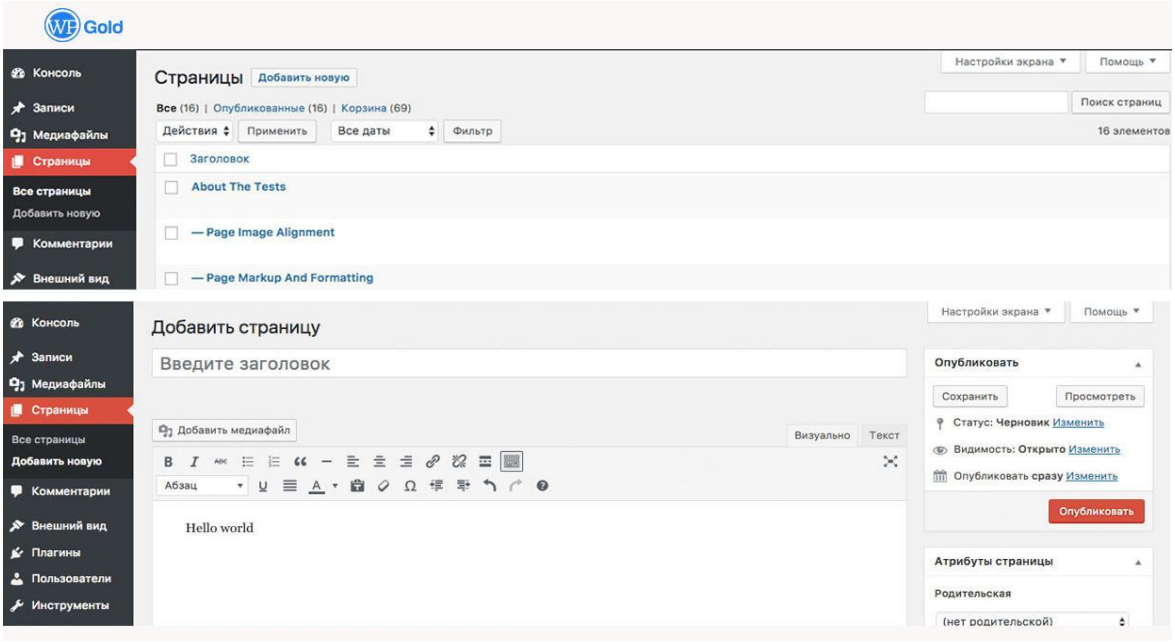


Рис. 1.2. Інтерфейс створення нової сторінки

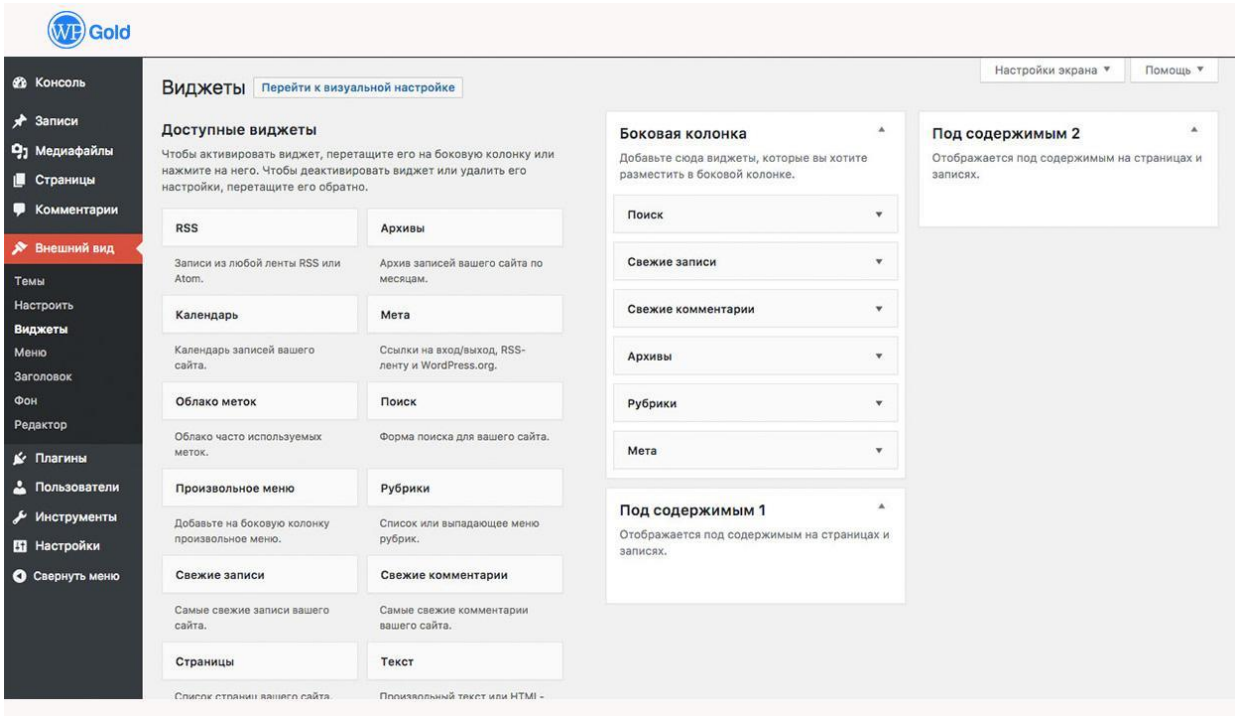


Рис.1.3. Інтерфейс налаштування зовнішнього виду

1.2.2 Drupal

Систему управління вмістом під назвою Drupal було засновано на стеці LAMP, що є набором програмного забезпечення із відкритим кодом [6]. Drupal відрізняється модульною структурою, що надає можливість змінювати

зовнішній вид вебсайтів, а основою є ядро, що містить у собі PHP скрипти, необхідні задля коректної роботи CMS.

Основними перевагами є:

- Можливості роботи на пристроях із різними конфігураціями;
- Простота налаштування, модифікації й оновлення веб-ресурсів;
- Велика доступність;
- Висока швидкість роботи;
- Підтримання великої кількості мов;
- SEO-оптимзація.

Основний недолік – система управління вмістом є загально спрямованою, і це не надає користувачу можливості вибрати тип вебсайтів [5].

Для розробників сторінки мають вигляд, що наведено на рис. 1.4 і 1.5.



Рис. 1.4. Інтерфейс головної сторінки Drupal

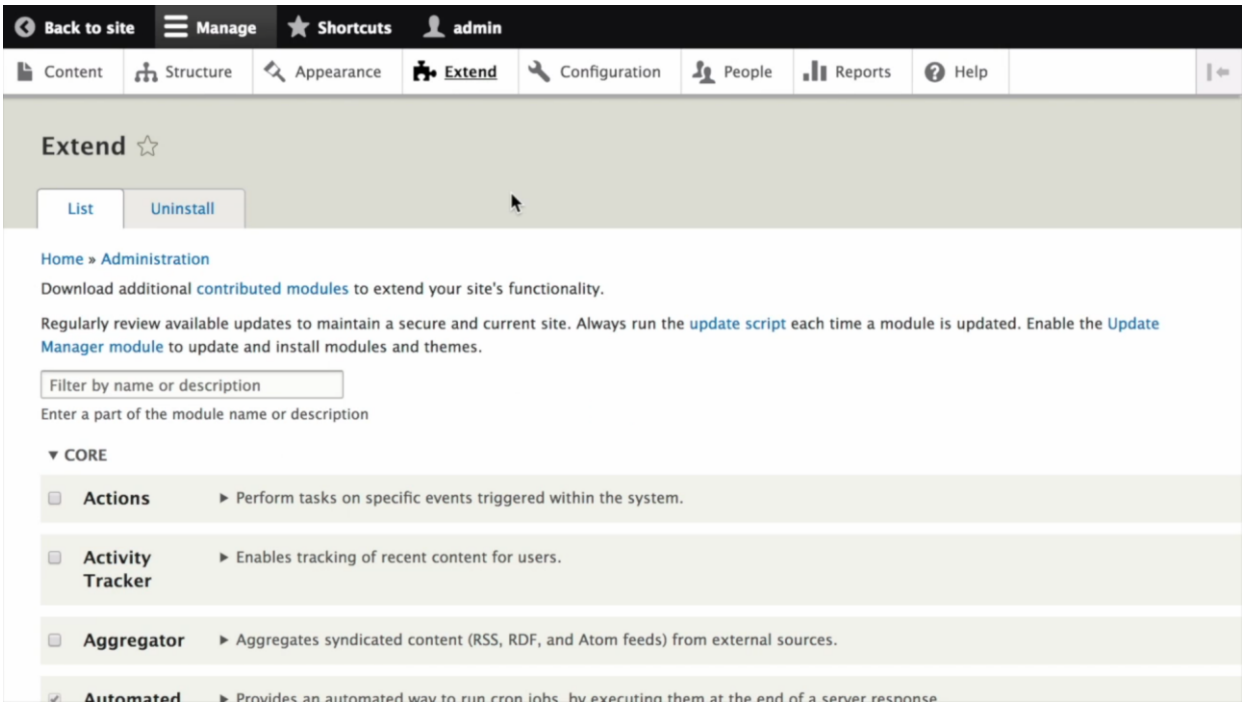


Рис. 1.5. Інтерфейс налаштування додатку

Але незважаючи на усі переваги, що надають ці системи управління контентом, вони, як і інші системи управлінням вмістом мають свої недоліки. Наприклад, вразливість, що була знайдено у 2013 році і яка отримала назву DRUPALGEDDON. Ця вразливість давала можливість відправити спеціально створені SQL-запити просто напряму до бази даних, в це могло призвести до несправностей у роботі PHP скрипту. Також у Drupal було виявлено проблеми із захистом від загальновідомої вразливості Cross-Site Scripting (XSS), що їх було знайдено у одній з останніх версій. Ця вразливість надавала змогу у дистанційно впроваджувати шкідливі скрипти і без дозволу власників або користувачів [11].

1.2.3 Joomla

Joomla, як і інші вищеописані системи управління вмістом, реалізується за ліцензією General Public License (GPL), що надає змогу вільно використовувати, модифікувати і розповсюджувати різне програмне забезпечення. Joomla було написано на мові програмування PHP, вона використовує СУБД, зокрема MySQL, PostgreSQL або MS SQL.

За встановлення системи Joomla користувач отримує комплект необхідних інструментів задля створення повноцінного вебсайту і з плином часу цей набір збільшуватиметься завдяки іншим користувачам, які створюватимуть свої інструменти.

Основні переваги полягають у:

- Зручному управлінні обліковими записами;
- Великій кількості готових шаблонів;
- Зручному інтерфейсі;
- Підтримці великої кількості мов;
- Стандартизованому інтерфейсі;
- Функціональності [7].

Однією з переваг для Joomla було виявлено кілька вразливостей, які призводили до витоку даних або некоректної роботи. Одна з таких – вразливість системи управлінням вмістом SQL-ін’єкцій, через що виявлені непередбачувані наслідки. Таку вразливість було знайдено у 2016 році. Також можна згадати про вразливість, що дала можливість створити акаунти з правами адміністратора, який, у свою чергу, надавав можливість встановити небезпечні додатки [12].

Для розробників сторінки Joomla мають вигляд, що наведено на рис. 1.6, 1.7 і 1.8.

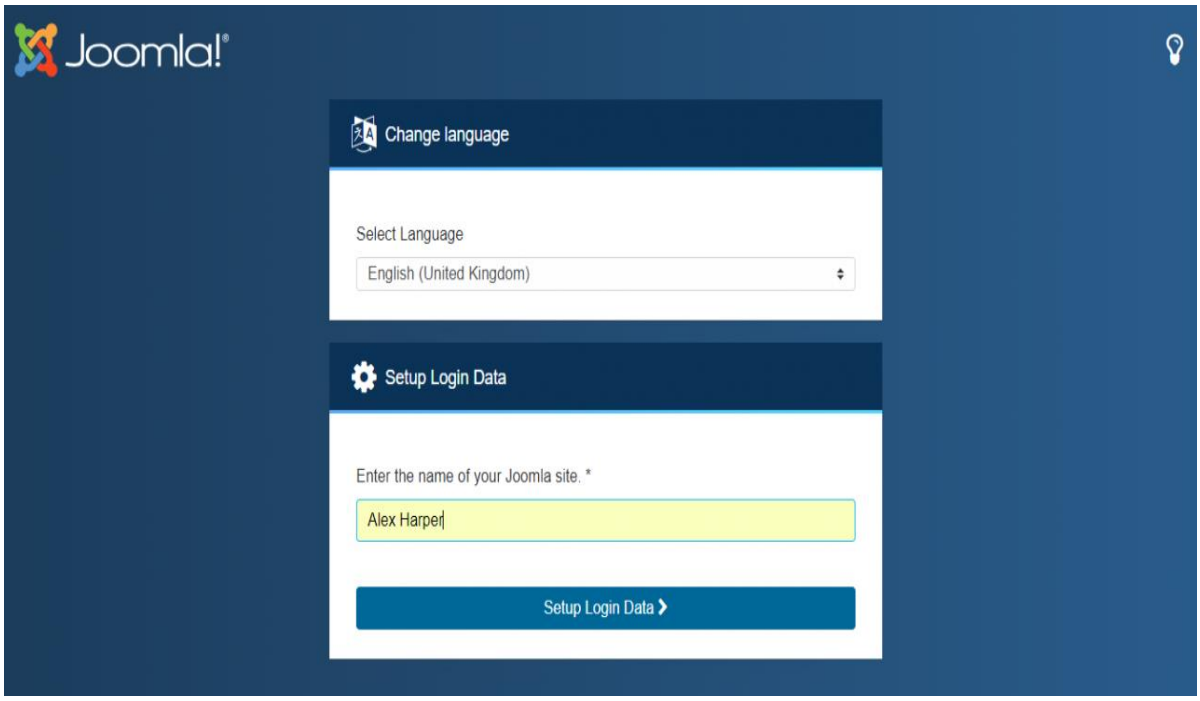


Рис. 1.6. Інтерфейс створення сайту

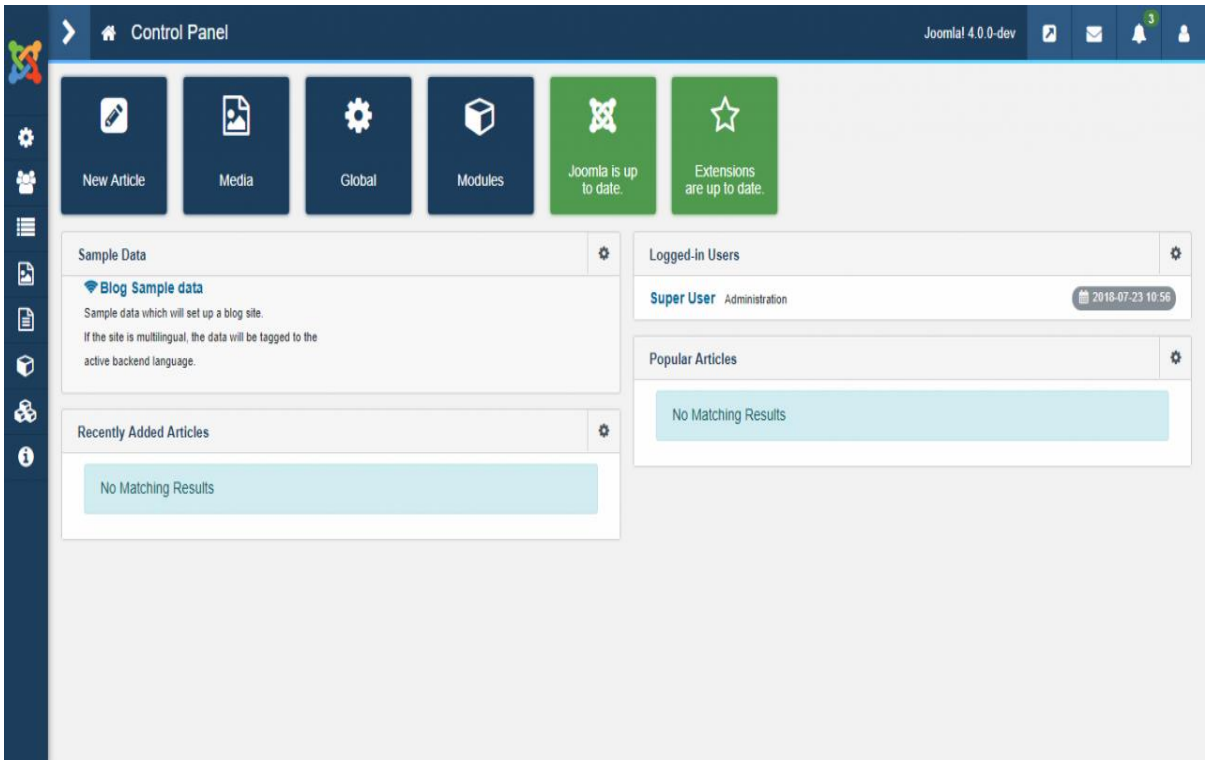


Рис. 1.7. Інтерфейс контрольної панелі

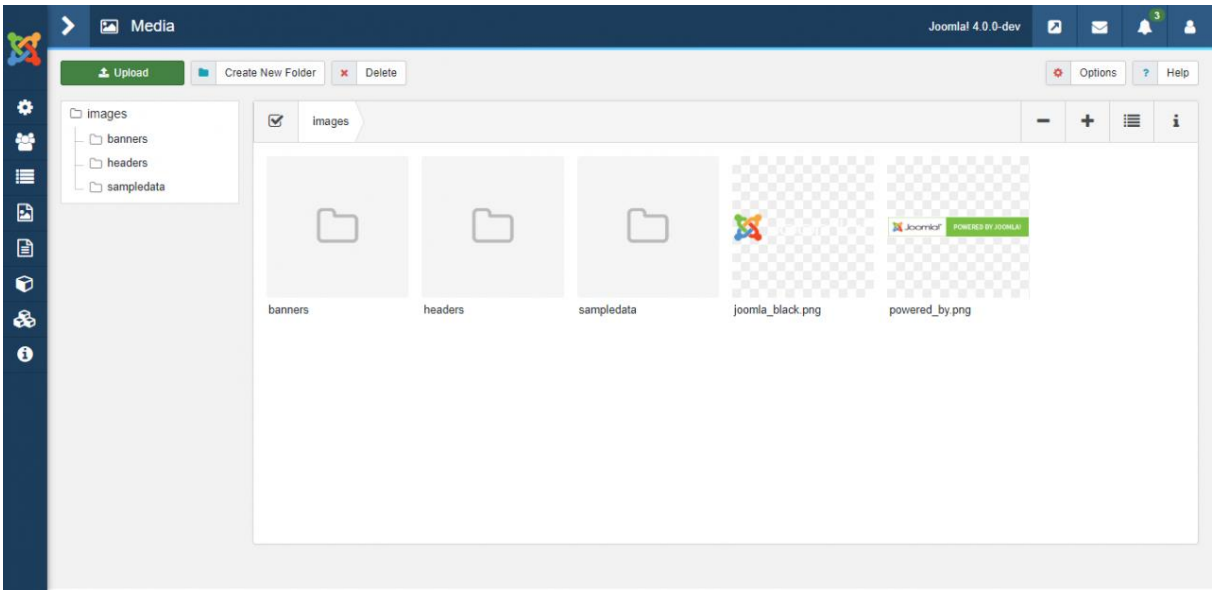


Рис. 1.8. Інтерфейс для вставки медіа файлів

1.3 Постановка задачі

На основі проведеного аналізу поставлено такі завдання:

1. Обрати дві системи управління контентом із високим рівнем безпеки і довіри користувачів. Рівень відповідної безпеки визначається за кількістю існуючих вразливостей, дані про які знаходяться в Інтернеті.

2. Дослідити можливості обраних систем управління вмістом і створити два вебсайта на цій основі.

3. Визначити етапи й інструменти для дослідження захищення вебсайтів.

4. Провести тестування вебсайтів задля аналзу їх захищеності і зробити відповідні висновки.

2 СТВОРЕННЯ ВЕБСАЙТІВ ЗА ДОПОМОГОЮ CMS ТА АНАЛІЗ ПЛАГІНІВ БЕЗПЕКИ

Задля подальшого коректного аналізу створюватимемо вебсайти на основі таких CMS-систем:

- WordPress;
- Joomla.

Створення вебсайту із допомогою CMS-систем розпочинається з завантаження цієї системи на робочі комп'ютери, на яких проходитиме створення веб-сайту, та вибору тематики вебсайту, що створюється.

2.1 Створення вебсайту за допомогою WordPress

Для початку роботи необхідно встановити та налаштувати CMS систему. Після завантаження CMS системи на локальні сервери користувачі потрапляють на сторінку з описом CMS системи, після ознайомлення з якою користувачі переходять на наступну сторінку, де проходить конфігурування доступу до баз даних (рис. 2.1).

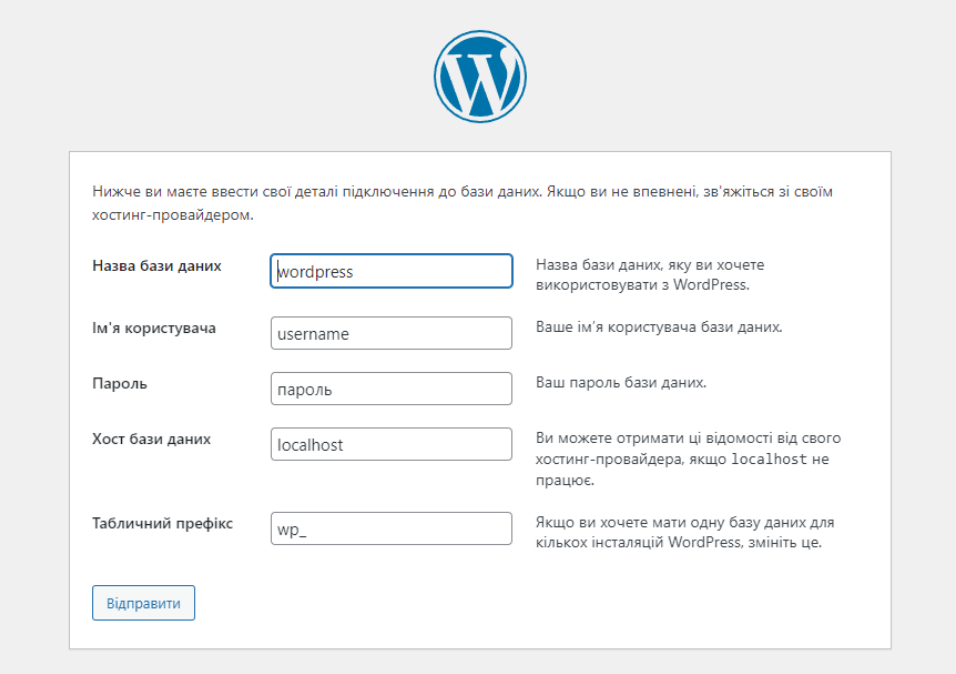


Рис. 2.1. Налаштування з'єднання з базою даних

Надалі необхідно вказати назву вебсайту, ім’я головного користувача, з профіля якого буде проводитися подальша робота та електрону адресу (рис. 2.2).

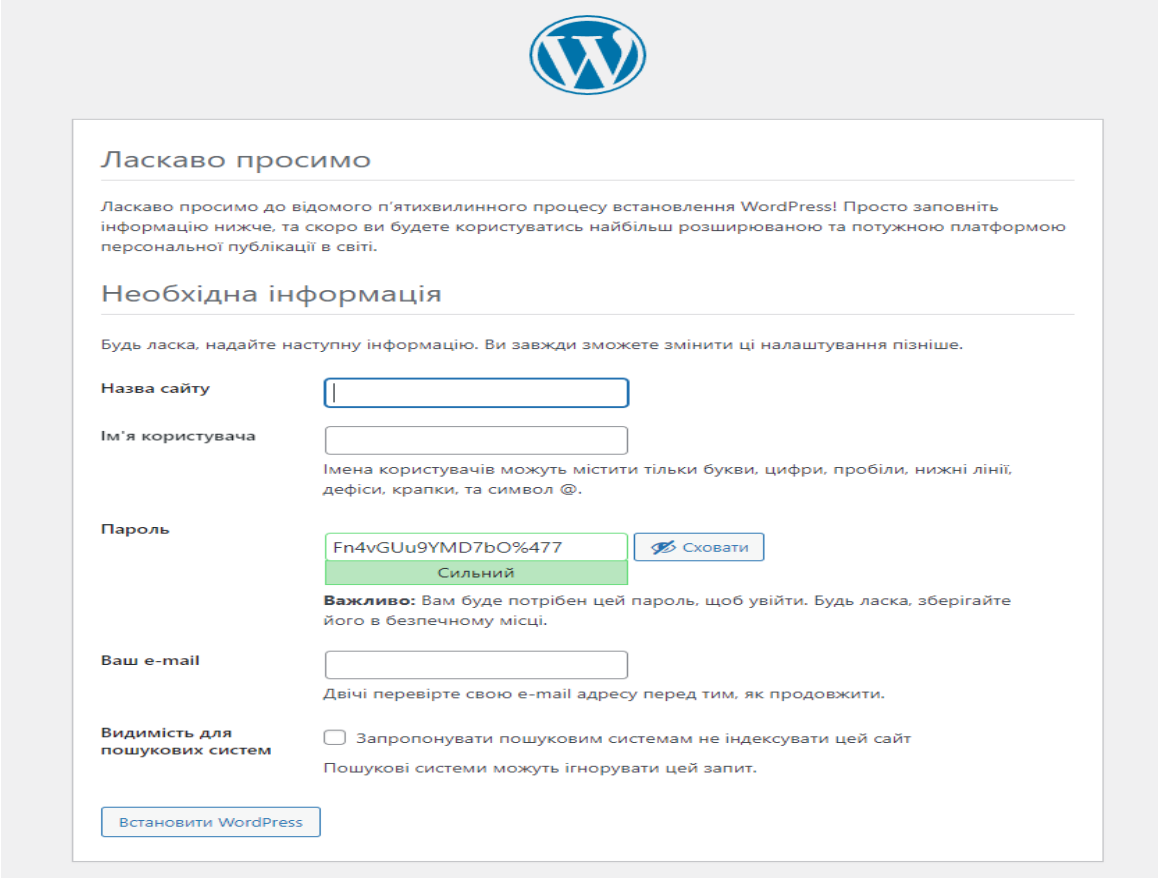


Рис. 2.2. Сторінка створення назви та профілю адміністратора

Одразу після створення користувачів з’являється сторінка, що повідомляє про те, що WordPress була встановлена коректно (рис. 2.3).

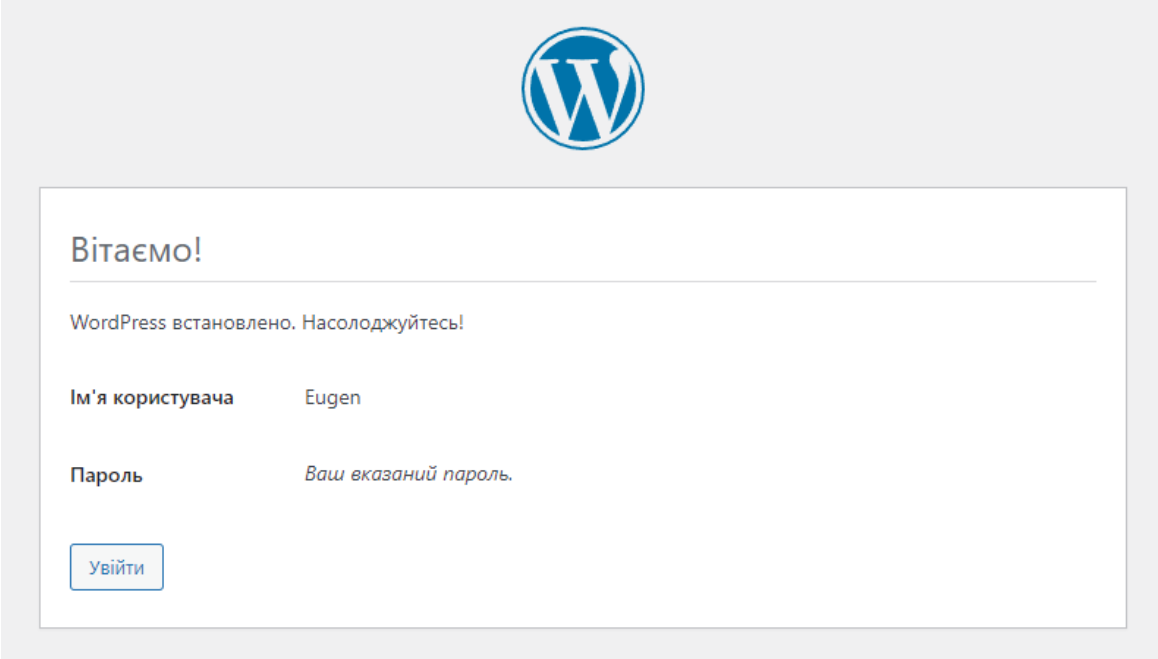


Рис. 2.3. Повідомлення про коректне встановлення WordPress

Наступний крок – вхід до облікового запису користувачів з правами адміністраторів.

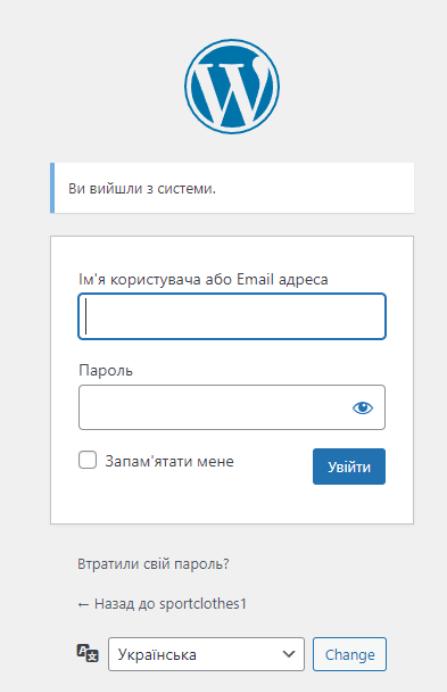


Рис. 2.4. Вікно входу у профіль

Після введення даних одразу для входу у профілі користувачі переходять на головну сторінку, де можна детальніше налаштувати CMS-систему у найкращий спосіб (рис. 2.5).

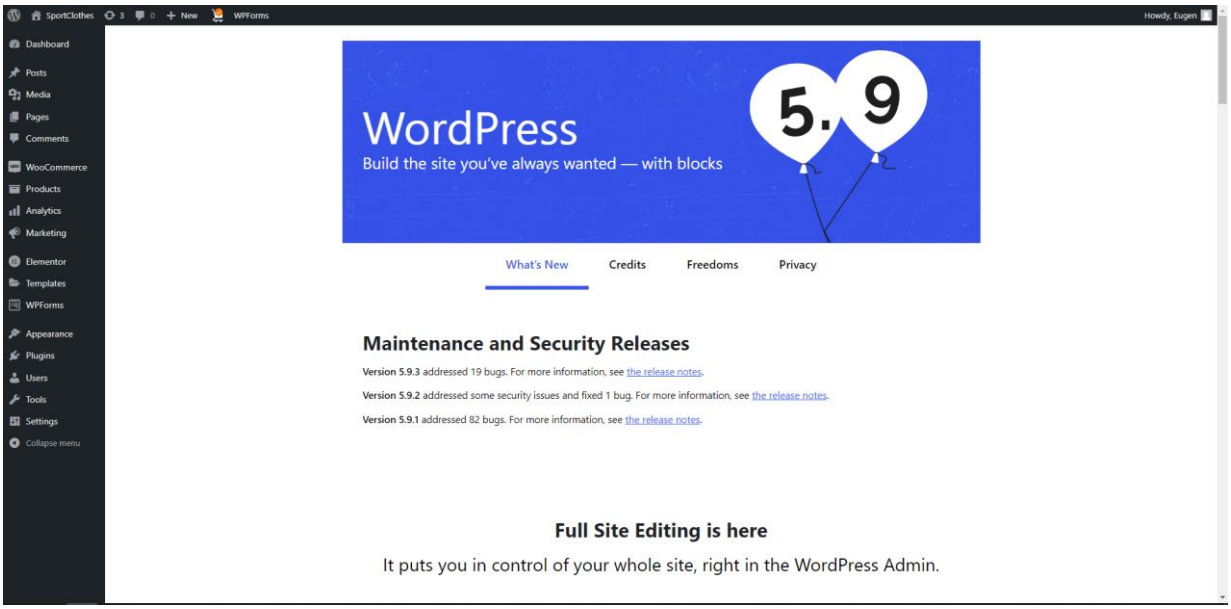


Рис. 2.5. Головна сторінка WordPress

Для створення сайтів із мінімальними функціональними можливостями необхідно встановити тему «Astra» і плагін Starter templates (рис. 2.6 і 2.7).

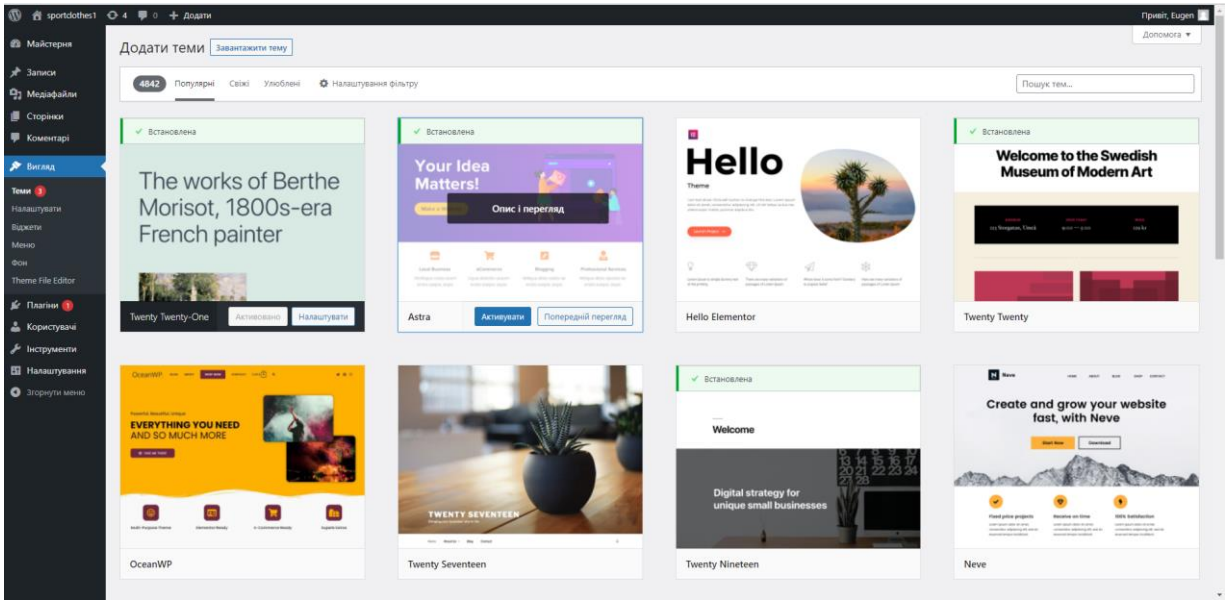


Рис. 2.6. Вибір теми

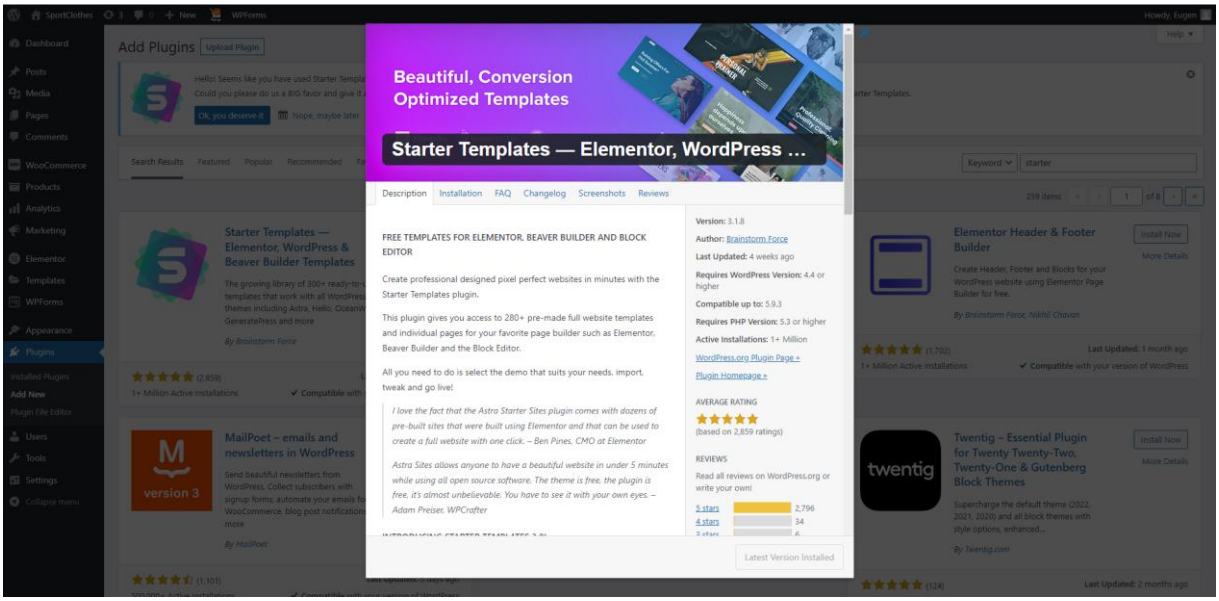


Рис. 2.7. Встановлення плагіну Starter templates

Після встановлення плагінів необхідно вибрати builder, за допомогою якого буде створюватись веб-сторінка (рис. 2.8).

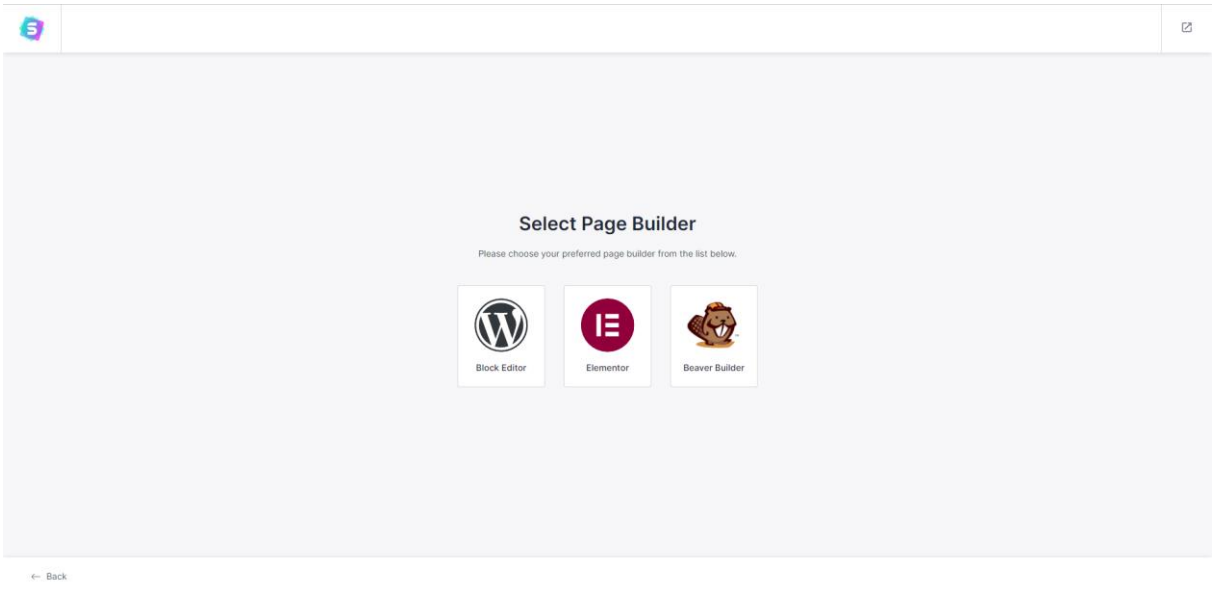


Рис. 2.8. Меню вибору Builder у плагіні Starter templates

Наступним кроком користувачі обирають макет (рис. 2.9).

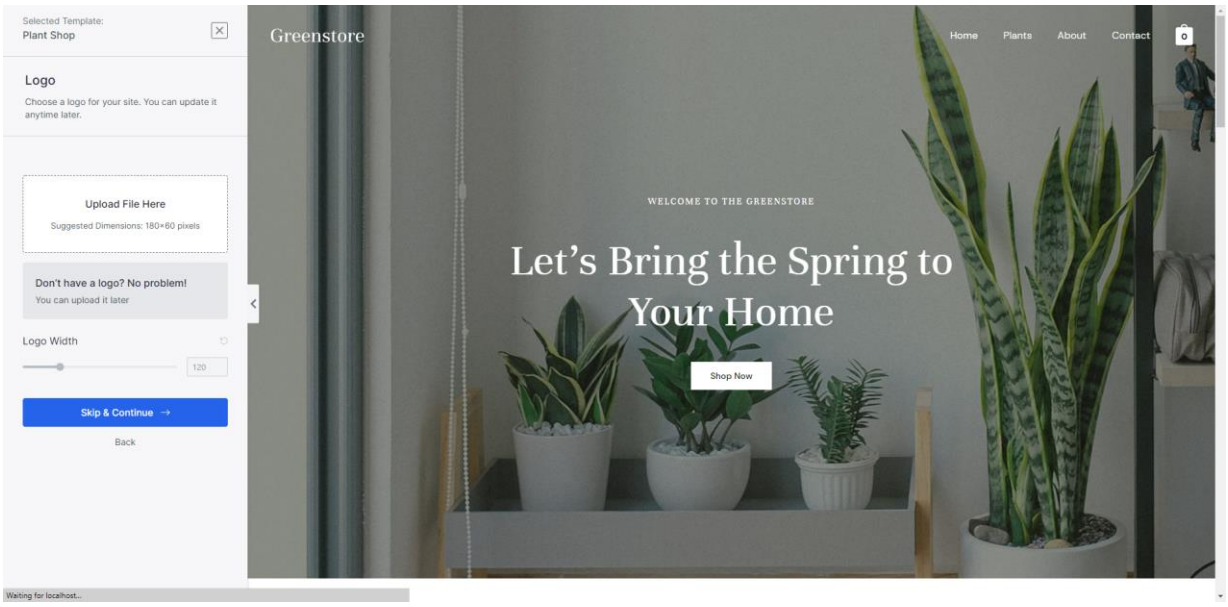


Рис. 2.9. Вибір шаблонів

Після обрання макету буде проходити процес будування веб-сторінки, який зображено на рис. 2.10.

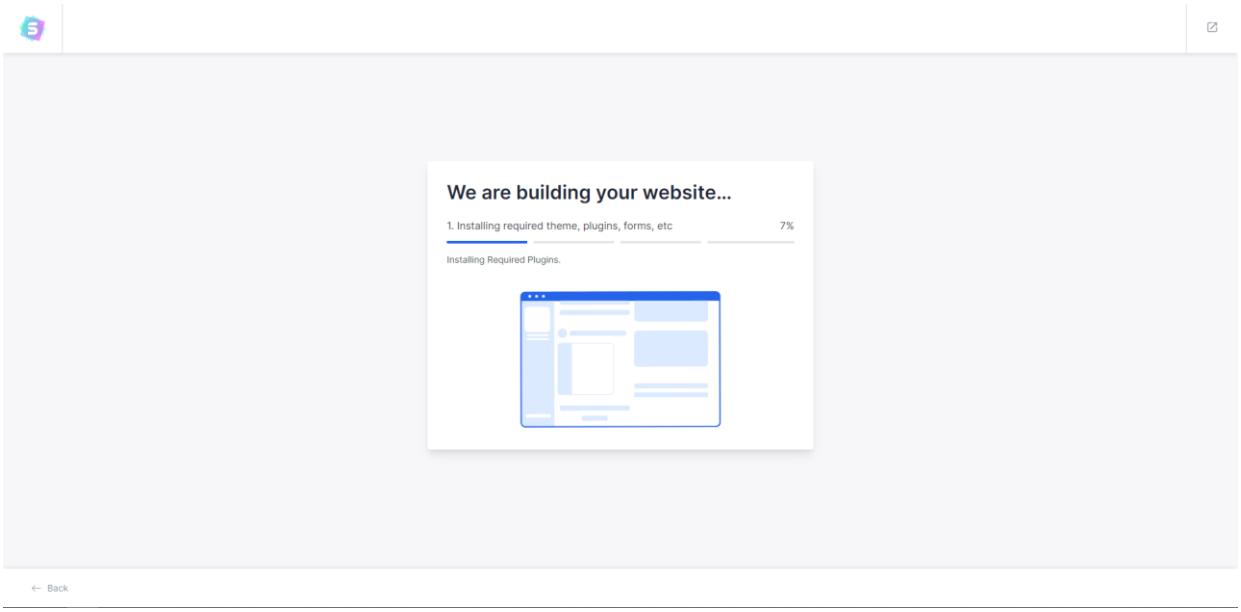


Рис. 2.10. Побудова веб-сторінки на базі вибраного шаблону

Для подальшої роботи необхідною є зміна дизайну і наповнення веб-сторінки щоб він відповідав відповідній тематиці. На рис. 2.11 зображено головну сторінку веб-сайту.

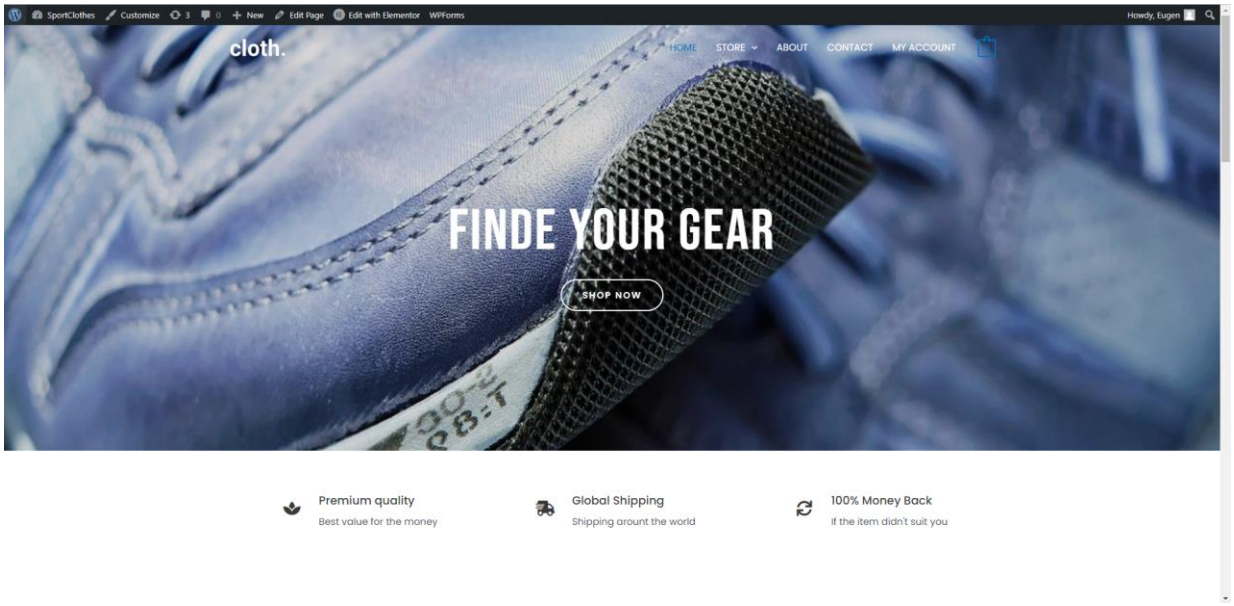


Рис. 2.11. Головна сторінка побудованого веб-сайту

Задля покращення захищеності вебсайту застосовується плагін Jetpack. Цей плагін використовує відповідний набір інструментів, резервне копіювання у реальному часі, захист від спаму і відновлення усіх даних (рис. 2.12).



Рис. 2.12. Сторінка плагіну Jetpack

2.2 Створення вебсайту за допомогою Joomla

Процес створення вебсайтів на базі CMS системи Joomla розпочинається з завантаження цієї системи.

Процес завантаження Joomla проходить у три етапи, де першим є вибір назви вебсайту й мови інтерфейсу (рис. 2.13).

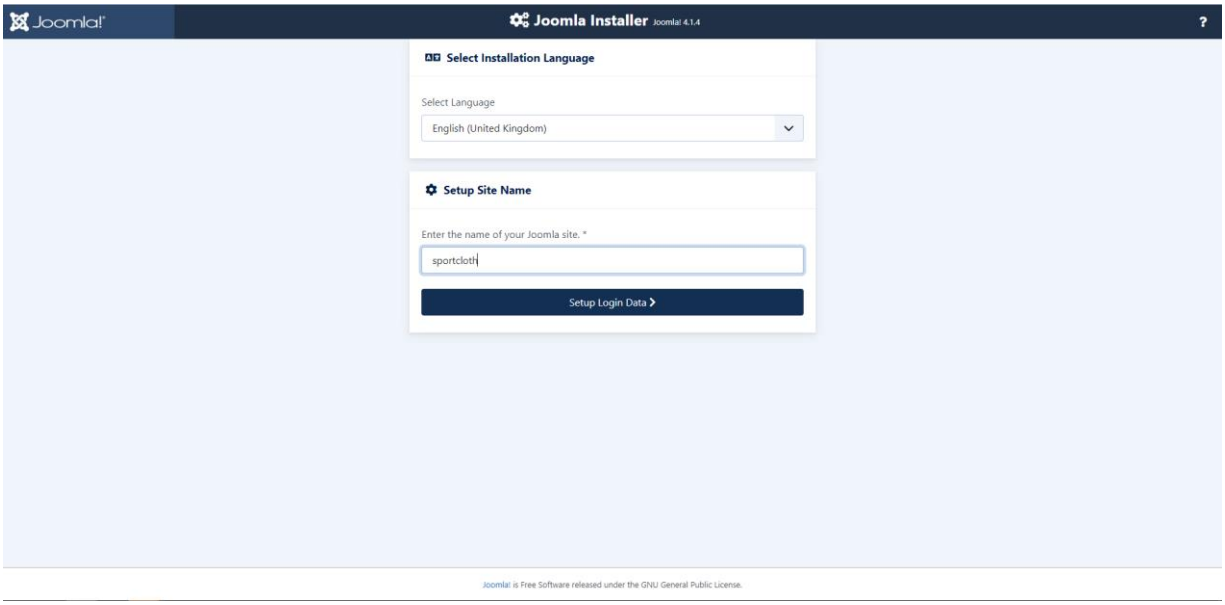


Рис. 2.13. Вибір назви вебсайту

Другий етап встановлення – створення облікового запису адміністратора, за допомогою якого виконуватиметься подальші роботи (рис. 2.14).

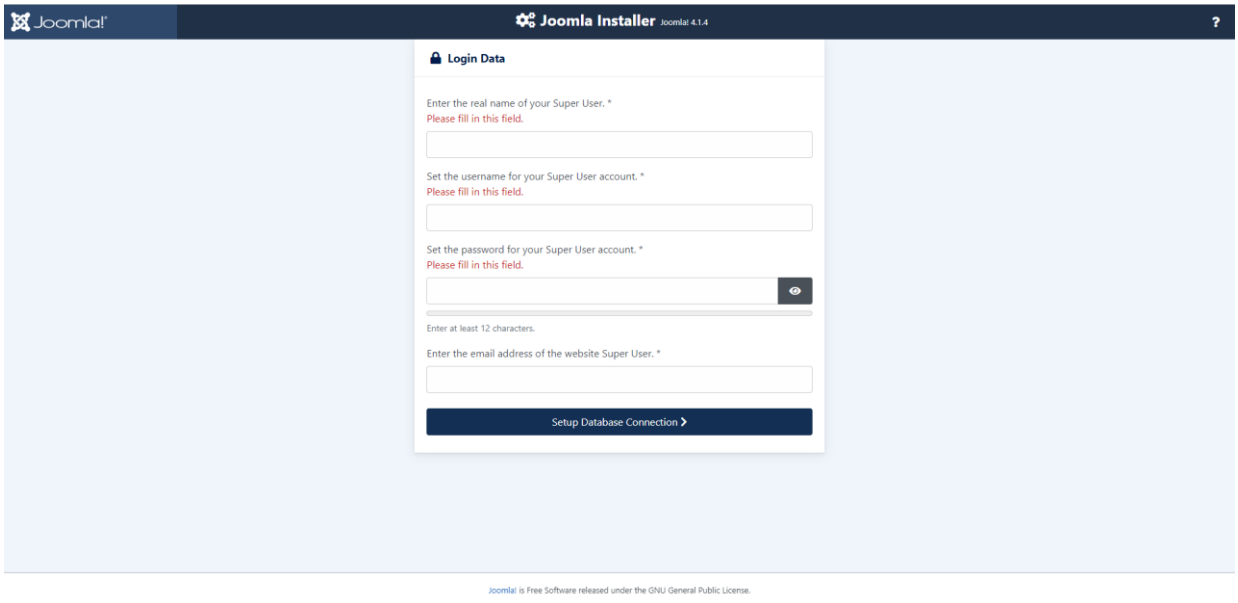


Рис. 2.14. Створення облікового запису адміністраторів

Третій етап встановлення – налаштування з’єднання з базою даних (рис. 2.15).

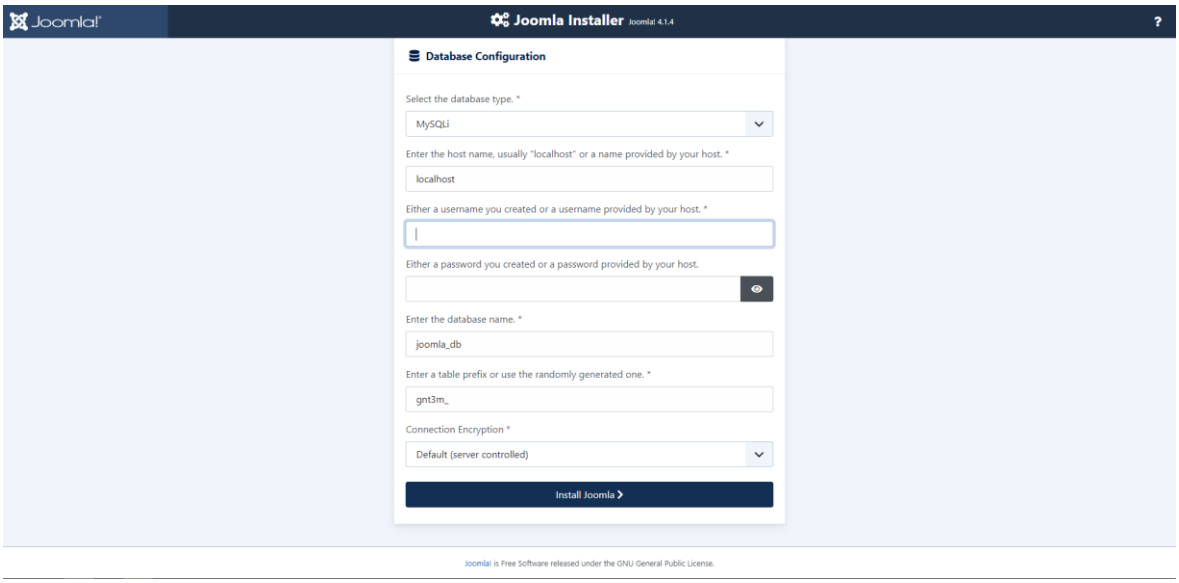


Рис. 2.15. Налаштування з’єднання з базою даних

Опісля встановлення CMS систем з’явиться вікно з повідомленням щодо готовності до роботи і можливістю перейти до меню адміністратора або безпосередньо вебсайту (рис. 2.16).

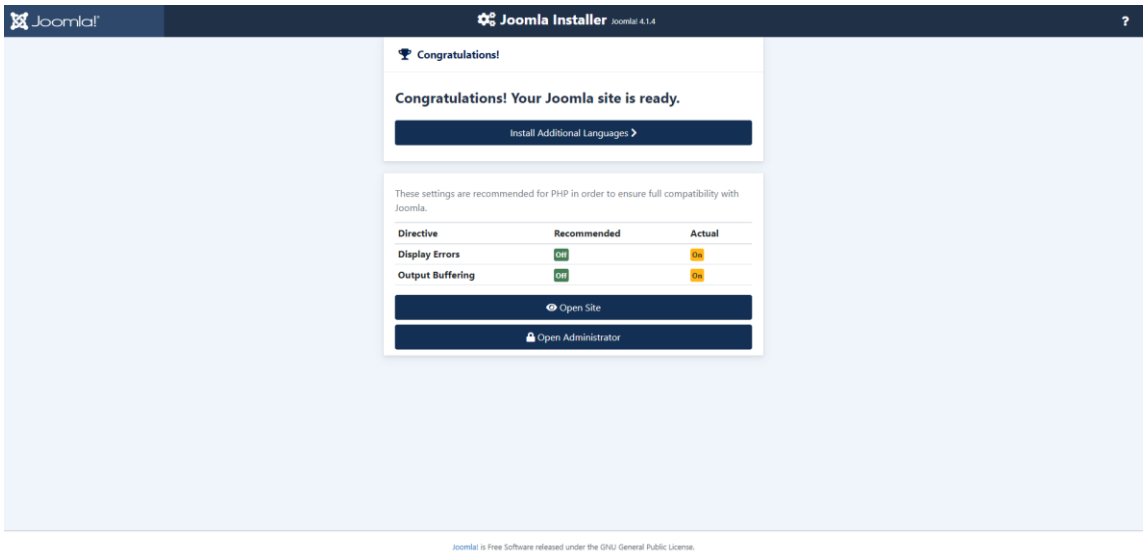


Рис. 2.16. Повідомлення щодо коректного встановлення CMS системи

Для початку безпосередньої розробки вебсайту необхідно провести аналіз доступних та цільових плагінів, оскільки не всі плагіни та теми є безкоштовними.

Для створення вебсайту інтернет-магазинів за допомогою CMS систем Joomla знайдено і встановлено плагін HikaShop, шаблон ET_SHOES, а задля поліпшення безпеки вебсайтів встановлено плагін Watchful.

2.3 Опис плагінів, використаних для розроблення вебсайтівStarter Templates

Плагіни Starter Templates надають доступ до більше яке 280 шаблонів й індивідуальних сторінок для різних інструментів побудови сторінок. Окрім шаблонів, ці плагіни надають можливість завантажувати демоверсію вебсайту. Плагіни використовують більше одного мільйона людей з різними навичками і завданнями через те, що він швидкодійний, легкий щодо налаштування, має піксельно-ідеальний дизайн і більш глибшу інтеграцію щодо інших інструментів.

Основні функції, що їх виконує плагін:

- побудова вебсайтів на базі вибраних шаблонів;
- можливість повних редагувань окремих блоків сторінки;
- велика кількість різних безкоштовних зображень [13].

WooCommerce

WooCommerce – це одна з найопулярніших платформ для інтернет-магазину з відкритим кодом. Плагіни мають відкритий початковий код, що робить їх гнучким і безкоштовними через велику кількість розробників.

Ці плагіни допомагають створювати онлайн-магазини і мають такі функції:

- можливості розробки вітрин на основі обраної тематики;

- можливості виставлення на продаж товарів, завантаження і сортування інформації щодо товарів;
- різні способи розрахунків;
- налаштування процесу купівлі [14].

Jetpack

Jetpack забезпечує захист вебсайтів із допомогою комплексного набору простого засобу захисту. Плагіни включають у себе можливість створення резервної копії у реальному часі і зручне відновлення, пошук шкідливих програмних забезпечень й захист від спаму. Крім цих функцій, також можна полегшити перехід вебсайтів на новий хостинг, створити дублікати сайтів для тестування, можливості відкату до старої версії вебсайту і створення резервних копій баз даних, що має можливість вести облік внесених змін і тих, хто здійснив такі зміни [15].

HikaShop

HikaShop є це плагіном, що допомагає створити інтернет-магазини на всіх версіях Joomla.

HikaShop також має широкий спектр маркетингових інструментів, а також потужні інструменти для складання статистики, яка відображається на інформаційній панелі, щоб полегшити користувачу користування своїм інтернет-магазином. Також цей додаток надає розширене управління податками, зонами, мовами та валютами для продажів, а також розширені можливості налаштування вашого інтернет-магазину [16].

Watchful

Watchful — це набір інструментів, який заощаджує багато часу на управління вебсайтами. Watchful дозволяє автоматизувати такі завдання, як створення резервних копій веб-сайтів, сканування на наявність ознак вторгнення і масове оновлення плагінів. Також дозволяє сканувати вебсайт на предмет використання найкращих методів безпеки, виконувати глибоке

сканування для того щоб знайти потенційні загрози безпеці, а також створювати звіти вебсайту для своїх клієнтів.

Основним функціоналом плагіну є:

- налаштування інформаційної панелі для того, щоб отримати огляд всього свого агентства;
- створення звіту щодо технічного обслуговування своїх клієнтів;
- можливість цілодобового автоматичного резервного копіювання, а також щотижня або щомісяця;
- необмежена наявна кількість резервних копій;
- автоматичне виявлення резервних застарілих копій, а також помилок конфігурації резервних копій;
- постійне відстежування вебсайту із допомогою сканування виявлення вторгнень щодня;
- оновлення комерційного плагіну, використовуючи базу даних ліцензії оновлення і централізованих ключів [17].

3 МЕТОДИКА ТА РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

3.1 Методи та засоби для аналзу захищеності вебсайтів

Аналіз захищеності вебсайтів може бути реалізований двома методами: ручним та автоматизованим. Ручне тестування представляє собою таке тестування, де безпосередню участь бере людина, тобто людина вибирає, які параметри тестувати та яким чином. Автоматизоване тестування представляє собою тестування з використанням спеціалізованих утиліт. Розглянемо деякі з них.

Nikto2

Nikto2 це сканер з відкритим кодом. Сканер виконує комплексне сканування веб-серверів, містить в собі список, який складається з більш ніж 6700 небезпечних файлів або програм, перевіряє 1200 серверів на застаріле програмне забезпечення та 270 серверів на специфічні вразливості, котрі притаманні лише цим серверам. Окрім цього, Nikto2 також перевіряє конфігураційні файли, файли з назвою «index», опції протоколу HTTP та може ідентифікувати встановлене програмне забезпечення та веб-сервер. Сканер дозволяє сканувати встановлені плагіни та перевіряти їх версію. Ця утиліта не є засобом для пасивного тестування, вона тестує вебсервер у швидкісному режимі [18].

OWASP ZAP

OWASP ZAP (Zed Attack Proxy) може проводити активне і пасивне сканування вразливостей. Пасивний тест це такий тест, який перевіряє вебсайт на наявність відомих вразливостей та ніяким чином не впливає на дані вебсайту, що є більш безпечним способом сканування. Активне сканування може використовувати різноманітні методи проникнення, воно може змінювати дані та вставляти небезпечні скрипти. Тому активним видом сканування краще користуватися з метою перевірки вебсайту, який запущений на локальному хості. OWASP ZAP може генерувати тестові рядки різними методами, що, в свою чергу, доволі корисно при первісному виявлені загрози. Ще однією

особливістю цього інструменту є те, що в ньому є можливість генерувати запити безпосередньо при виявлені загрози, відповідь на ці запити відображається у тому ж робочому вікні, що значно покращує швидкість прийняття рішення щодо певної загрози [19].

Sqlmap

Sqlmap використовується для тестування на проникнення та має відкритий початковий код, що за потребою дозволяє модифікувати його під нетривіальні задачі. Цей інструмент автоматизує процес виявлення та експлуатації вразливостей, також містить в собі багато різноманітних функцій від збору відбитків баз даних за отриманою від них інформації до можливості отримати доступ до файлової системи. Підтримує майже усі системи управлінням базами даних (СУБД), може напряму підключатися до баз даних, автоматично розпізнає формати зашифрованих даних, наявна можливість створення бази даних та інше [20].

Nmap

Nmap це безкоштовна утиліта з відкритим кодом, яку використовують для аудиту безпеки та пошуку мереж. Ця утиліта використовує немодифіковані IP пакети, що дозволяє визначити, які хости доступні у мережі, які сервіси ці хости пропонують, яку операційну систему використовують, який файервол або фільтр пакетів використовується на даний момент. Цей інструмент розроблений для сканування великих мереж та ефективно працює з одинокими хостами. В додаток до класичного функціоналу Nmap надає просунутий інтерфейс, гнучку передачу даних, перенаправлення та інструменти для дебагінгу, також надає можливість порівнювати результати та інструмент для генерації та аналізу пакетів [21].

VegScanner

Цей інструмент є безкоштовним сканером та тестовою платформою з відкритим кодом для тестування безпеки вебсайтів, має вбудований сканер вразливостей для швидкого тестування. VegScanner допоможе знайти та

підтвердити SQL ін’єкції, кроссайтинговий скріптинг, незахищену чутливу інформацію та інше. Окрім цього може зондувати налаштування безпеки протоколу TLS/SSL і запропонувати можливі рішення для покращення захисту сервера. Розроблено цей інструмент на мові програмування Java. Він має графічний інтерфейс та сумісний з усіма відомими операційними системами [22].

WPScan

WPScan це сканер вразливостей, який має закритий початковий код і розроблений для сканування вебсайтів, створених на базі Wordpress. Цей інструмент має свою базу даних вразливостей, які присутні в WordPress. Сканер працює за принципом «чорної» скриньки і під час сканування він імітує хакерські атаки [23].

WhatWeb

WhatWeb створений для ідентифікації вебсайтів. Метою цього інструмент є сканування вебсайтів та ідентифікація технологій, які були використані при створенні ресурсу, а також CMS систему, якщо така використовувалася. WhatWeb має більше 1700 функціонуючих плагінів, кожен з яких виконує свою задачу. Цей засіб передбачає можливість регулювання рівня агресивності, який використовується у випадках коли з першого разу не вдалося отримати усю необхідну інформацію. Більшість плагінів, що використовуються, є фундаментальними та у своїй роботі враховують велику кількість рзноманітних даних. Деякі вебсайти видаляють тег <meta>, за яким можна визначити використану CMS систему, тому WhatWeb при тестуванні використовує 15 різноманітних тестів [24].

3.2 Організація дослідження

Тестування безпеки це метод оцінки безпеки системи чи мережі, що перевіряє та підтверджує ефективність системи контролю безпеки ресурсів. Тестування вебсайтів зосереджене на оцінці безпеки вебсайтів. Процес тестування включає активний аналіз вебсайту на вразливості. Інформація про

будь-яку вразливість, що виявлена під час тестування, доводиться до відома власника вебсайтів, разом з поясненням того, як саме вразливість може впливати на безпеку та шляхи вирішення цієї проблеми.

Для повноцінного розуміння необхідності тестування потрібно уточнити, що таке вразливість, небезпека та тест. Вразливість це недолік дизайну системи, який може бути використаний для проникнення до програмного коду вебсайтів. Небезпеку для вебсайту представляє усе, що піддає систему небезпеці, чи то зовнішні атаки, чи то внутрішня недбалість розробників. Під тестом розуміють дію, яка спрямована на перевірку системи безпеки вебсайтів на вразливість та відповідність вимогам безпеки.

Для проведення дослідження захищеності створених вебсайтів як основу використаємо методологію OWASP Web Security Testing Guide (OWASP WSTG - v4.2) [25]. Методологія OWASP базується на підході «чорної скриньки». Суть підходу «чорна скринька» полягає у том, що тестувальник володіє певною мінімальною кількістю інформації про систему, що тестується.

Згідно методології OWASP Web Security Testing Guide тестування поділяють на два види: пасивне та активне.

Пасивне тестування полягає у тому, що тестувальник намагається зрозуміти логіку побудови вебсайтів та досліджує вебсайти, як користувач. Також тестувальник може використовувати різноманітні інструменти для збору інформації. Наприклад, HTTP проксі можна використати з метою огляду усіх запитів та відповідей на них. По завершенню цієї стадії тестувальник повинен знати усі точки доступу та увесь функціонал системи.

Активне тестування поділяється на 12 категорій, кожна з яких має свою мету та спосіб реалізації.

Під час активного тестування за методологією OWASP Web Security Testing Guide передбачаються такі етапи перевірки:

- збирання інформації;

- конфігурація і розгорнення тесту менеджменту;
- тестування управління ідентифікацією;
- тестування аутентифікації;
- тестування авторизації;
- тестування сесій менеджменту;
- тестування перевірки введення;
- обробка помилок;
- криптографія;
- тестування бзнес-логіки;
- тестування на стороні клієнта;
- тестування API.

Хоча за методологією OWASP WSTG передбачається перевірка у 12 етапів, кожен з яких містить ще й підетапи, для аналзу захищеності вебсайтів, створених за допомогою CMS систем, обрано саме такі етапи:

- збір інформації про вебсайтии;
- тестування методів HTTP;
- тестування на можливість введення сторонніх скриптів;
- тестування на SQL ін’єкції.

Розглянемо детальніше кожен з етапів.

1. Збір інформації про вебсайти.

У кожного вебсайту є HTML заголовки, cookies та структура директорій. Усі ці частини вебсайтів допомагають ідентифікувати, що саме було використано при розробленні вебсайтів. Більшість CMS систем мають спеціальні маркери, які допомагають відрізнити їх від інших. Це в основному

те, що автоматизовані інструменти здійснюють: шукають ці маркери, а потім порівнюють з базами даних.

Звичайно ці маркери розташовуються у таких місцях:

- заголовки HTTP;
- cookies;
- вихідний HTML код;
- специфічні файли та папки;
- розширення файлів;
- повідомлення про помилки.

Основним способом ідентифікації є перевірка поля X-Powered-By в HTTP заголовках. Іншим надійним способом є перевірка Cookies.

2. Тестування методів HTTP пропонує набір методів, які Звичайно використовуються при роботі вебсайтів. Самими відомими HTTP методами є методи POST та GET, за допомогою яких надається доступ до інформації вебсерверів. Але окрім них існують інші відомі методи, за допомогою яких, зловмисник може отримати доступ до інформації.

Перелік існуючих HTTP методів:

- GET;
- POST;
- HEAD;
- PUT;
- DELETE;
- CONNECT;
- OPTIONS;
- TRACE.

Незважаючи на таку велику кількість різних методів, у вебсайтів. Зазвичай використовують методи POST та GET, які отримують дані користувача з рядка URL, що вводяться користувачем. Стандартні посилання, як і форми, визначені без ініціації методу GET. JavaScrips та AJAX запити можуть надсилати інші методи, але Звичайно цього не роблять. Через те, що інші методи використовуються не дуже часто, більшість розробників не беруть до уваги те, як вебсайт взаємодіє з цими методами.

3. Тестування на можливість введення сторонніх скриптів.

Кроссайтовий скриптинг або XSS вразливість це такий вид вразливості в інтерактивних частинах вебсайтів, коли існує можливість введення сторонніх скриптів. XSS вразливості виникають у ситуаціях, коли у згенеровану сервером сторінку потрапляють користувацькі скрипти. Відображені XSS як один з рзновидів кроссайтового скриптингу з’являються коли зловмисник вводить виконуваний код в браузер в одному HTTP запиті. Проте код, який введено не зберігається всередині вебсайтів, і при перезавантаженні сторінки все повернеться до початкового стану. Атакуючий рядок входить до створеного URL рядка або HTTP параметра, які некоректно обробляються вебсайтами. Відображені XSS це один з найбільш використовуваних видів XSS атак. Такі атаки ще відомі, як непостійні, оскільки навантаження атаки постачається та виконується за допомогою одного запиту та відповіді, їх ще можна побачити під назвою атаки першого порядку.

Коли вебсайт вразливий до такого типу атак, він пропускає неперевірені вхідні дані. Звичайним сценарієм є надання користувачу зміненого URL посилання та переконання користувача у легітимності цього посилання. Після переходу за наданим посилання небезпечний скрипт починає діяти з використанням браузера користувача.

Однією з головних частин протидії XSS скриптам є правильне кодування символів. У деяких випадках вебсервер або вебсайт не можуть фільтрувати

деякі кодування, наприклад, вебсайт може відфільтрувати <script>, але не зможе відфільтрувати %3cscript%3.

4. Тестування на SQL ін’єкції.

Тестування на SQL ін’єкції перевіряє чи можливо впроваджувати дані у вебсайт так, щоб запит виконувався на стороні клієнта. Вдала експлуатація цієї вразливості дозволяє неавторизованому користувача маніпулювати даними.

Атака за допомогою SQL ін’єкцій виконується за допомогою ін’єкції команди в підготовленому запиті. В основному такі вразливості можливі через наявність помилок у написаній розробником SQL команди, яка приймає дані від користувача. Наприклад, select title, text from news where id=$id, де змінна $id, яка являє собою дані, введені користувачем, а інша частина запиту написана розробником та не змінюється, що робить запит динамічним.

Успішна SQL ін’єкція потребує створення такого синтаксично правильного запиту, що вебсайт не помітить втручання і виконає написаний запит. Якщо вебсайт повертає повідомлення про неправильний запит, це може допомогти зловмиснику переписати запит, так, щоб його не можна було виявити.

3.3 Результати проведеного дослідження

Для експериментальної частини було проаналзовано два вебсайти, що було створено на CMS системах Joomla та WordPress. План дослідження кожного з вебсайтів передбачав аналз ресурсу без використання захисного плагіну та з використанням відповідного плагіну. Для кожної CMS системи обрано найпопулярніші та безкоштовні плагіни безпеки.

Аналз захищеності розроблених вебсайтів складається з чотирьох частин. По-перше, необхідно зібрати інформацію про вебсайт, на якому сервері розташована, які технології використовувалися тощо. Для реалзації цієї задачі буде використовуватися інструмент WhatWeb. Цей інструмент має три рівні

агресивності, але для збору інформації щодо вебсайту, який тестується достатньо першого рівня.

Для того, щоб почати збір інформації необхідно ввести у консоль команду whatweb [опції] <URL>. Першим було проведено збір інформації про вебсайт на базі CMS системи WordPress з деактивованим плагіном (рис. 3.1 - 3.2).

Після отримання результатів аналізу з деактивованим плагіном було проведено збір інформації про вебсайт на базі CMS системи WordPress з активованим плагіном (рис. 3.3 – 3.4).

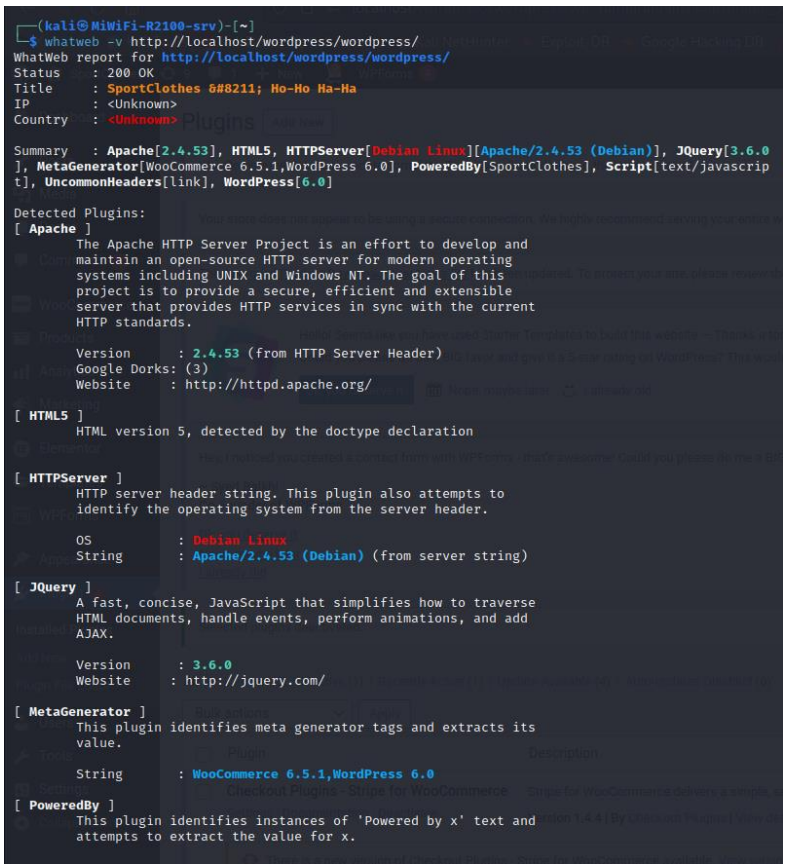


Рис. 3.1. Перша частина збору інформації про вебсайт без активованого захисного плагіну на базі CMS системи WordPress

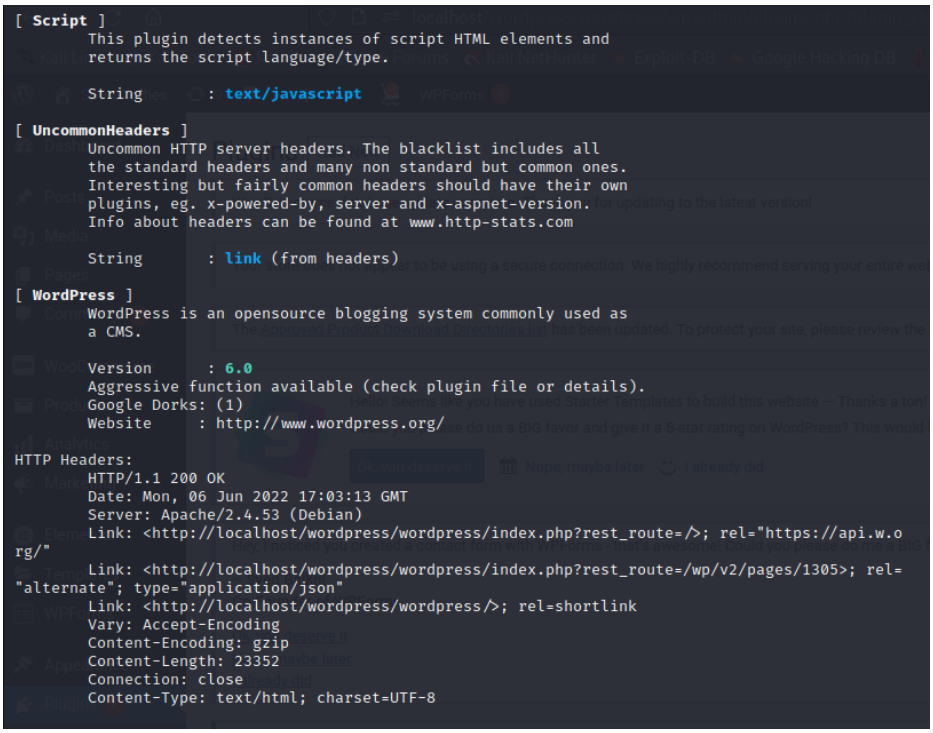


Рис. 3.2. Друга частина збору інформації про вебсайт без активованого захисного плагіну на базі CMS системи WordPress

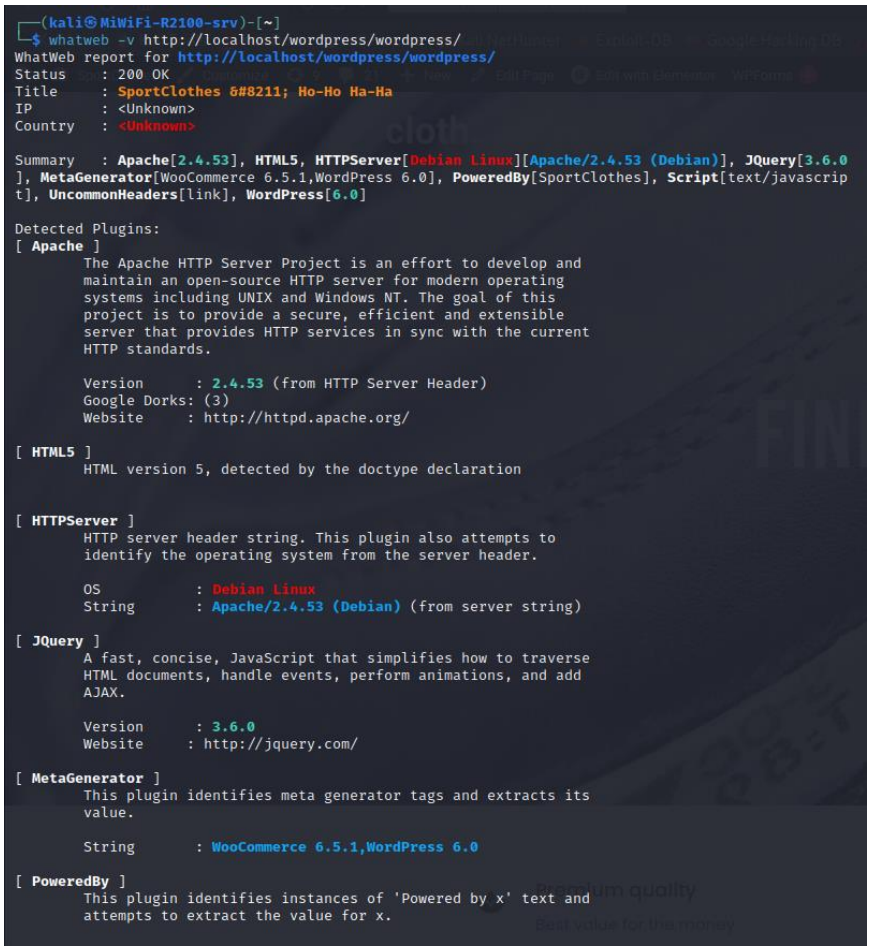


Рис. 3.3. Перша частина збору інформації про вебсайт з активованим захисним плагіном на базі CMS системи WordPress

За результатами збору інформації про вебсайт на базі CMS системи WordPress як з використанням захисного плагіну, так і без нього, було визначено, що захисний додаток не впливає на результати зібраної інформації про вебсайт.

Після збору інформації про вебсайт, який був створений на базі CMS системи WordPress, необхідно провести збір інформації про вебсайт, який був створений на базі CMS системи Joomla, за тією ж схемою як і попередній. Результати збору інформації без захисного плагіну наведені на рис. 3.5, 3.6 і 3.7.

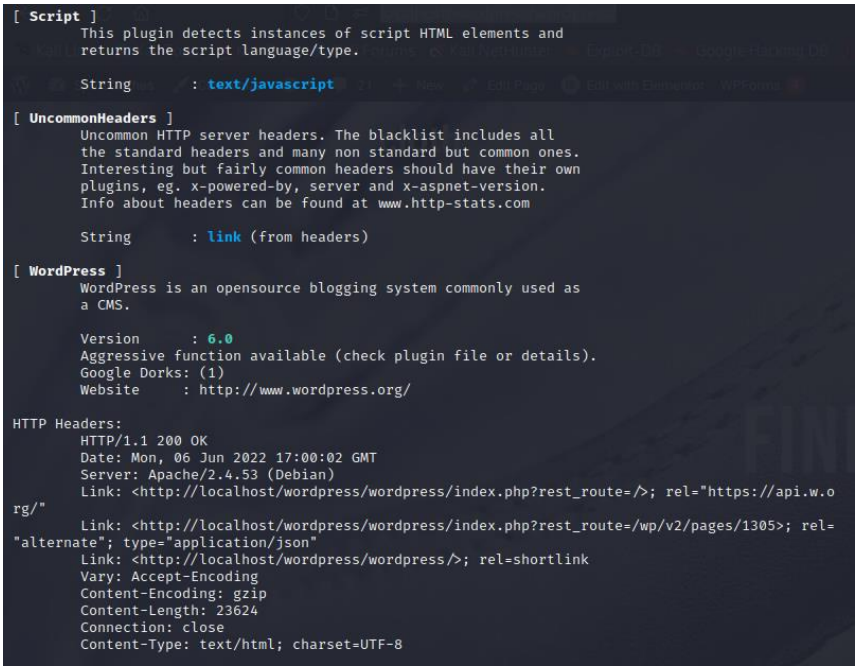


Рис. 3.4. Друга частина збору інформації про вебсайт з активованим захисним плагіном на базі CMS системи WordPress

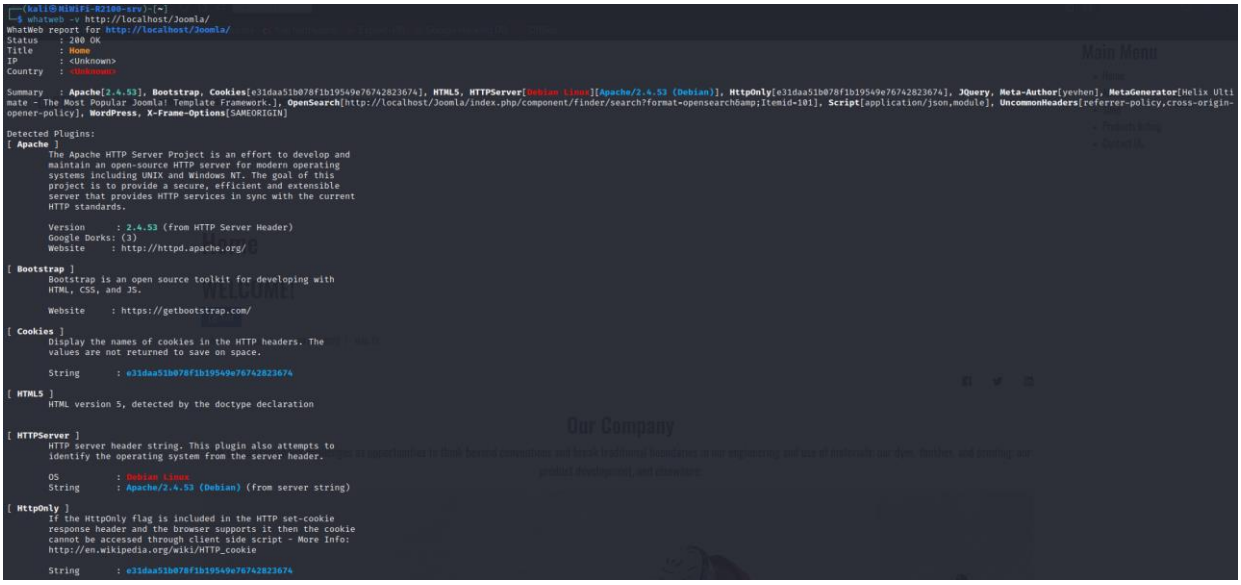


Рис. 3.5. Перша частина збору інформації про вебсайт без активованого захисного плагіну на базі CMS системи Joomla

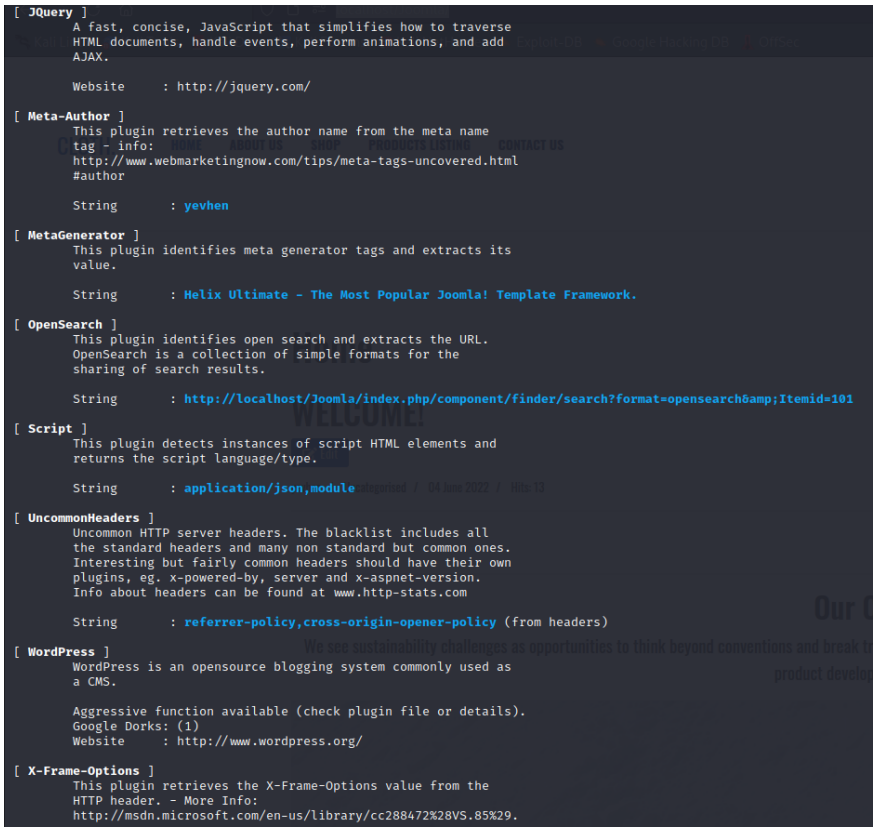


Рис. 3.6. Друга частина збору інформації про вебсайт без активованого захисного плагіну на базі CMS системи Joomla

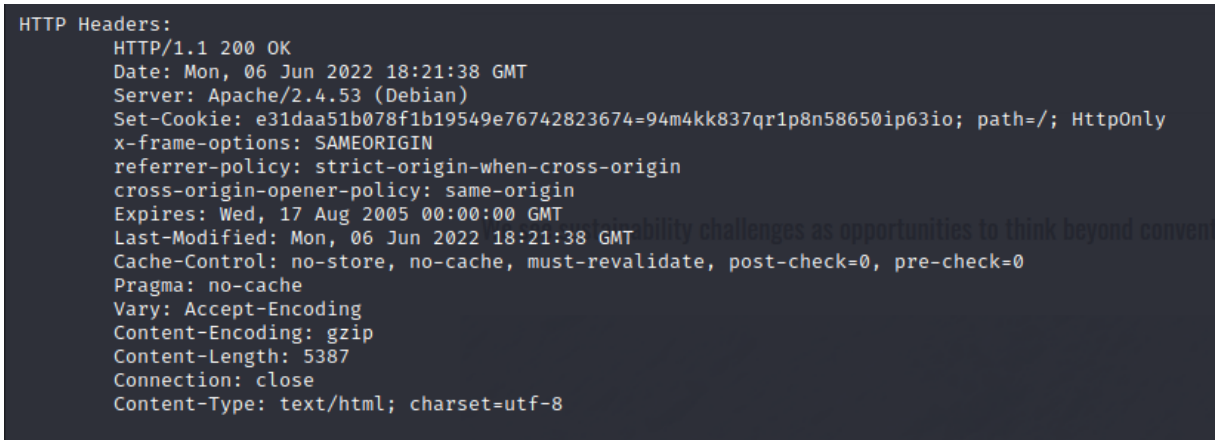


Рис. 3.7. Третя частина збору інформації про вебсайт без активованого захисного плагіну на базі CMS системи Joomla

Результати збору інформації з активним захисним плагіном зображено на рис. 3.8, 3.9. і 3.10.

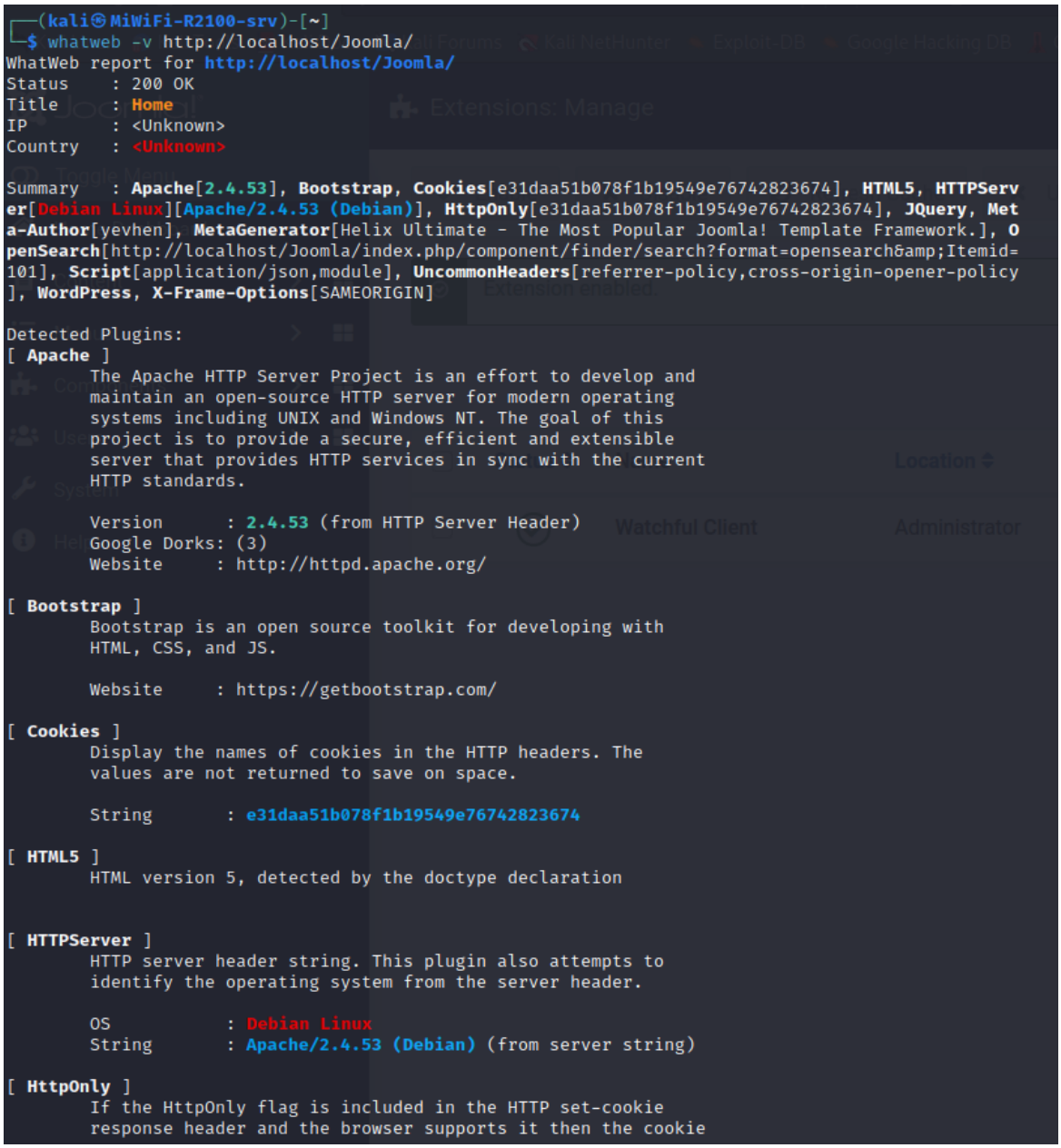


Рис. 3.8. Перша частина збору інформації про вебсайт з активованим захисним плагіном на базі CMS системи Joomla

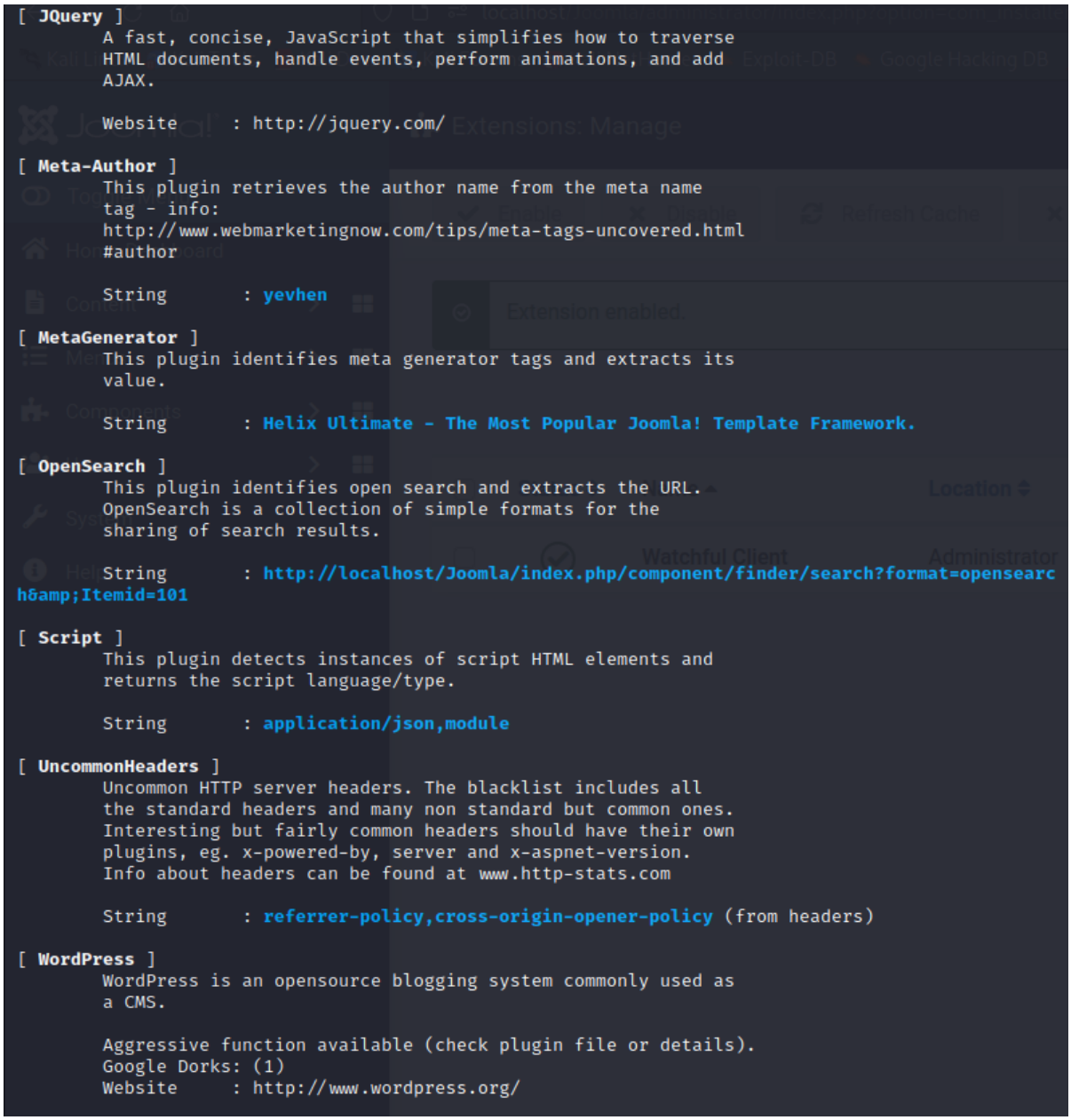


Рис. 3.9. Друга частина збору інформації про вебсайт з активованим захисним плагіном на базі CMS системи Joomla

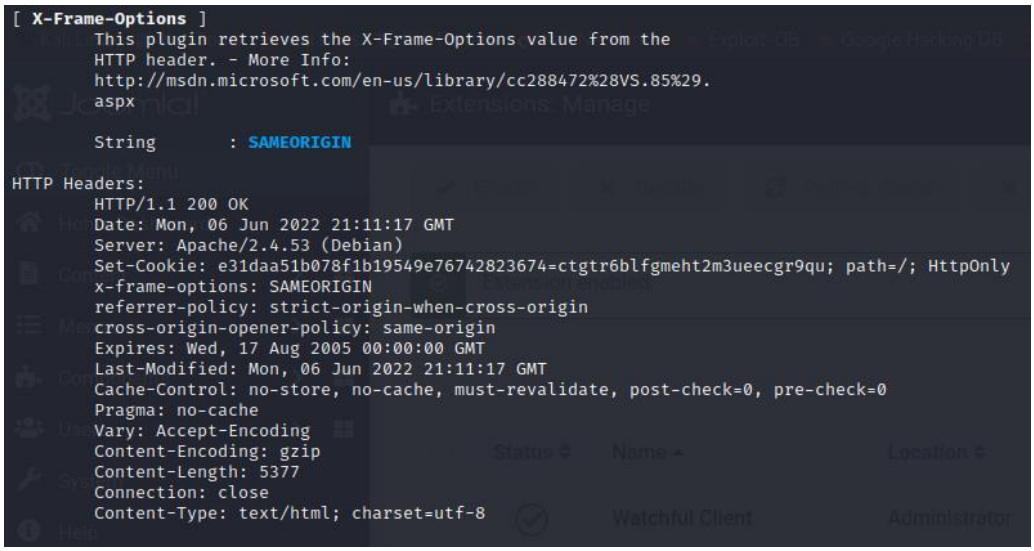


Рис. 3.10. Третя частина збору інформації про вебсайт з активованим захисним плагіном на базі CMS системи Joomla

Результат збору інформації щодо вебсайту, створеного на базі CMS системи Joomla, з активованим захисним плагіном та без нього дозволяє констатувати відсутність відмінностей. Тому можна зробити висновок, що встановлений захисний плагін ніяким чином не перешкоджає збору інформації про вебсайт.

Надалі необхідно провести аналз доступних HTTP методів. Для аналзу застосовуємо інструмент Nmap з використанням скрипта http-methods. При перевірці використаємо таку команду nmap -A -T4 --script http-methods --script-args http-methods.url-path='шлях до вебсайту' localhost, в якій вказано назву інструменту, опція використання скрипта та цільовий ресурс.

Першим проаналзуємо вебсайт, створений на базі CMS системи WordPress. Результати тестування наведено на рис. 3.11.

Результат аналзу вебсайту, створеного на базі CMS системи Joomla, наведено на рис. 3.12.

За результатами аналзу доступних HTTP методів вебсайтів, створених на базі CMS систем WordPress та Joomla, з активним захисним плагіном та без нього можна прийти до висновку, що плагін не впливає на доступність HTTP методів по відношенню до вебсайтів, що розгорнуті на локальному сервері.

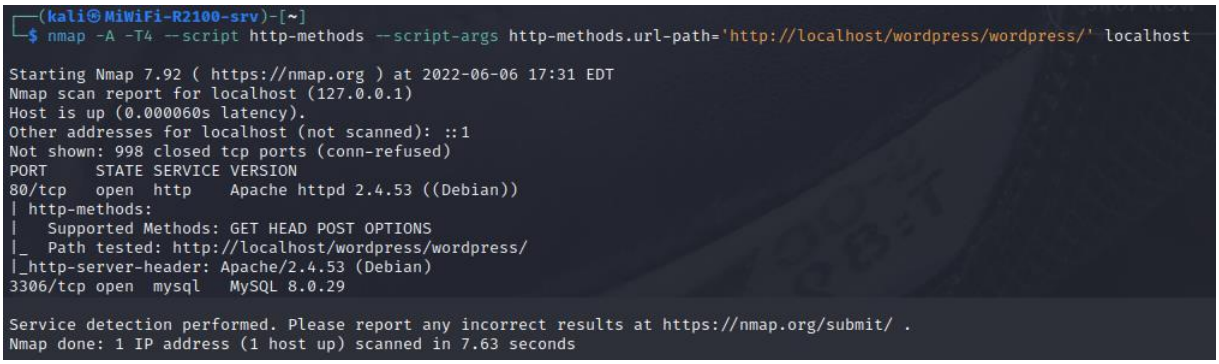


Рис. 3.11. Аналз доступних HTTP методів для вебсайту, створеного на базі CMS системи WordPress з активним захисним плагіном

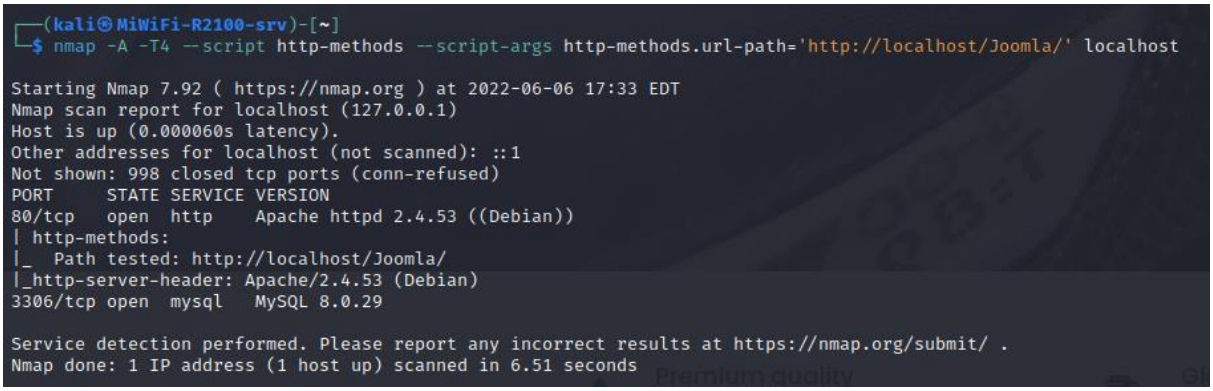


Рис. 3.12. Аналз доступних HTTP методів для вебсайту, створеного на базі CMS системи Joomla з активним захисним плагіном

Результати аналзу доступних HTTP методів збігаються для вебсайтів, які створені на рзних CMS системах, через те, що ці вебсайти розгорнуті на одному локальному сервері.

Наступним етапом буде тестування на впровадження сторонніх скриптів, яке виконується за допомогою інструмента OWASP ZAP, котрий містить в собі рзноманітний функціонал та має рзні варіанти сканування. Для початку необхідно вибрати вид сканування (автоматизований чи ручний). Обираємо автоматизований та вводимо посилання на вебсайт. На рис. 3.13 і рис. 3.14 наведено результати сканування вебсайту, створеного на базі CMS системи WordPress без активного захисного плагіну, які свідчать про те, що вебсайт схильний до впровадження сторонніх скриптів.

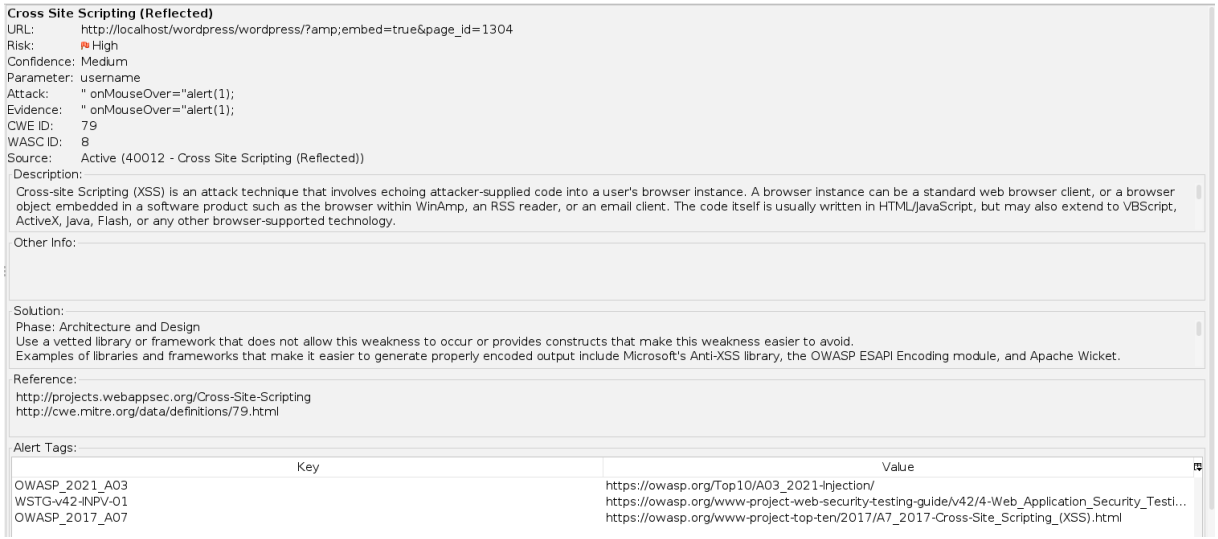


Рис. 3.13. Результат аналізу на можливість введення сторонніх скриптів без активного захисного плагіну

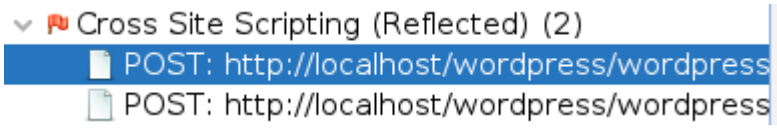


Рис. 3.14. Перелік загроз критичного рівня на можливість введення сторонніх скриптів без активного захисного плагіну

Проведений аналз вебсайту з активованим захисним плагіном за допомогою OWASP ZAP засвідчив, що цей плагін не покращує захищеність сайту від впровадження сторонніх скриптів (рис. 3.15 і рис. 3.16).



Рис. 3.15. Результат аналізу на можливість введення сторонніх скриптів з активним захисним плагіном

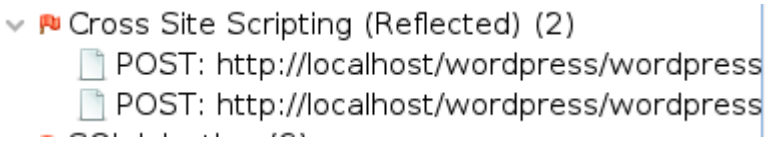


Рис. 3.16. Перелік загроз критичного рівня на можливість введення сторонніх скриптів активним захисним плагіном

Проведено аналогічні дії по відношенню до вебсайту, створеного на базі CMS системи Joomla з активним захисним плагіном. Результати перевірки продемонстровано на рис. 3.17 і рис. 3.18.

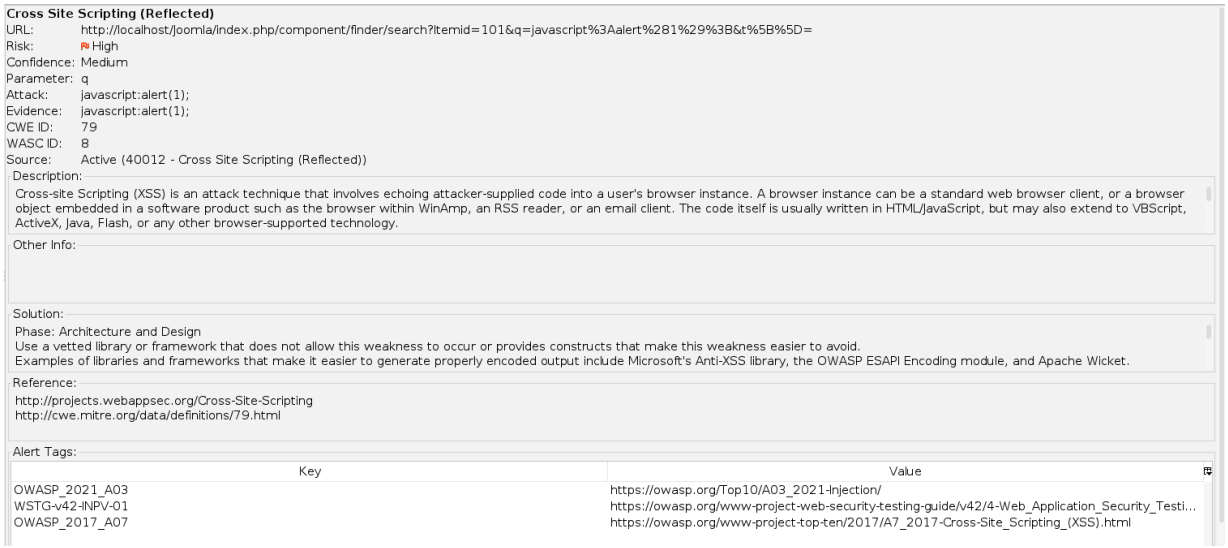


Рис. 3.17. Результат аналізу на можливість введення сторонніх скриптів з активним захисним плагіном

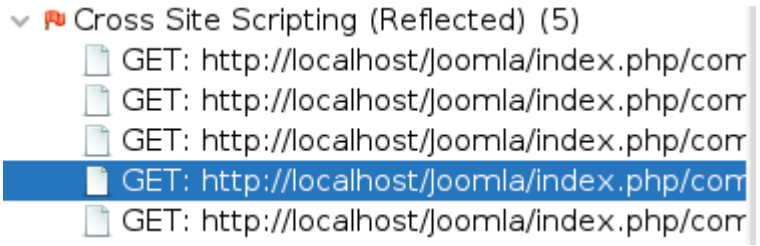


Рис. 3.18. Перелік загроз критичного рівня на можливість введення сторонніх скриптів активним захисним плагіном

Результати аналізу вебсайту без активного захисного плагіну наведено на рис. 3.19 і рис. 3.20.



Рис. 3.19. Результат аналізу на можливість введення сторонніх скриптів без активного захисного плагіну

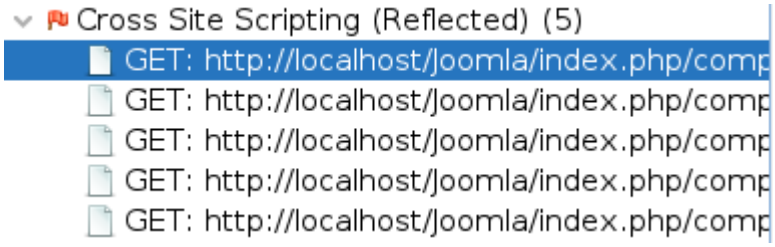


Рис. 3.20. Перелік загроз високого рівня на можливість введення сторонніх скриптів без активного захисного плагіну

При порівняння результатів перевірки з використанням захисного плагіну та без нього, поліпшення безпеки у відношенні до загрози введення стороннього скрипта, не було відзначено.

Тепер проведемо аналз на можливість впровадження SQL ін’єкцій за допомогою OWASP ZAP. Під час сканування вебсайтів на захищеність від введення сторонніх скриптів також було проведено аналз на можливість SQL ін’єкцій, було проаналзовано вебсайти з активним захисним плагіном та без нього. Результати аналзу вебересурсу, створеного на базі CMS системи WordPress без активного захисного плагіну, наведено на рис. 3.21 і рис. 3.22.

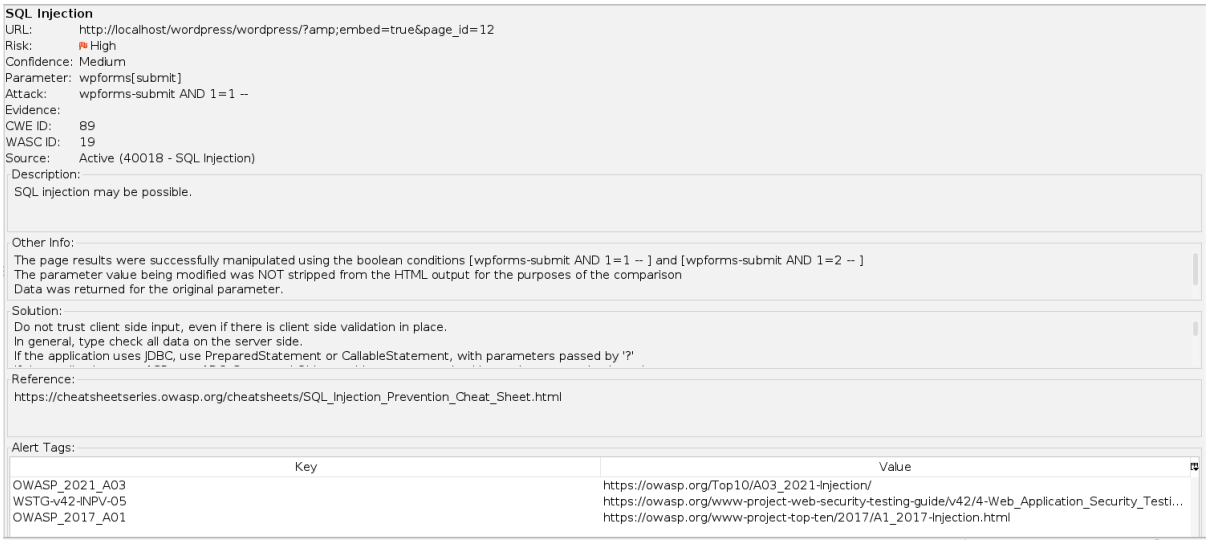


Рис. 3.21. Результат аналізу на можливість SQL ін’єкцій скриптів без активного захисного плагіну

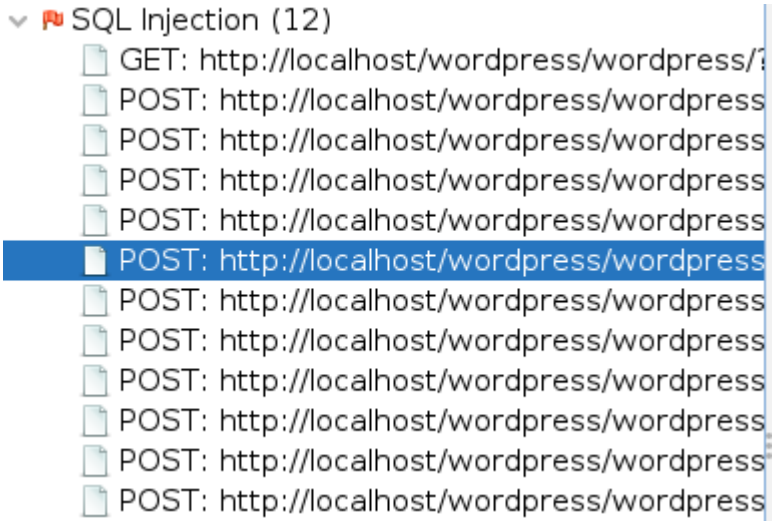


Рис. 3.22. Перелік загроз критичного рівня на можливість SQL ін’єкцій без активного захисного плагіну

Результати аналізу вебересурсу, створеного на базі CMS системи WordPress з активним захисним плагіном, зображено на рис. 3.23 і 3.24.

Результати аналізу вебересурсу, створеного на базі CMS системи Joomla з активним захисним плагіном, наведено на рис. 3.25 і рис. 3.26.



Рис. 3.23. Результат аналізу на можливість SQL ін’єкцій з активним захисним плагіном

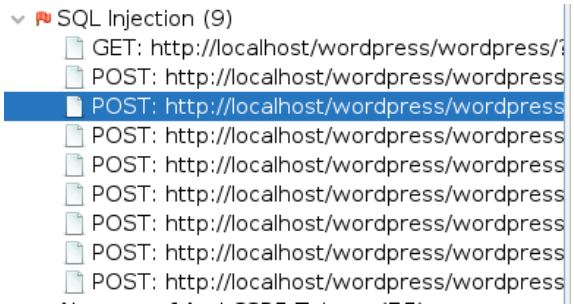


Рис. 3.24. Перелік загроз критичного рівня на можливість SQL ін’єкцій з активним захисним плагіном

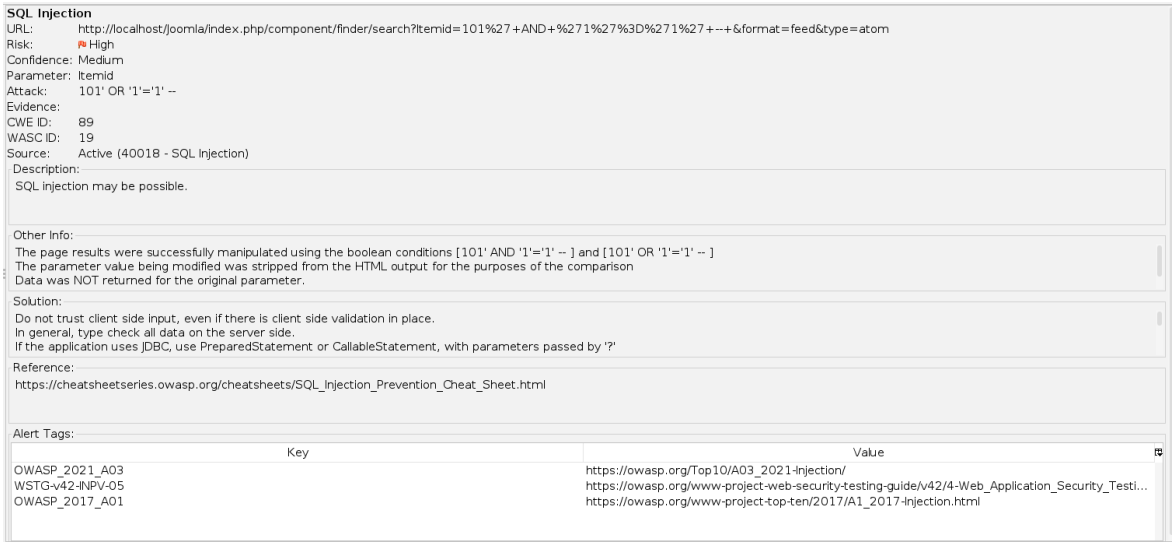


Рис. 3.25. Результат аналізу на можливість SQL ін’єкцій з активним захисним плагіном

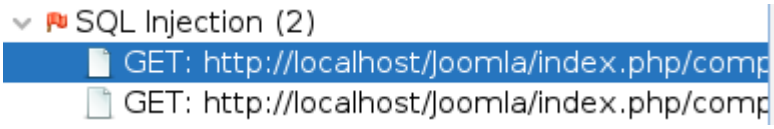


Рис. 3.26.Перелік загроз критичного рівня на можливість SQL ін’єкцій з активним захисним плагіном

Результати аналзу вебересурсу, створеного на базі CMS системи Joomla без активного захисного плагіну, наведено на рис. 3.27 і рис. 3.28.

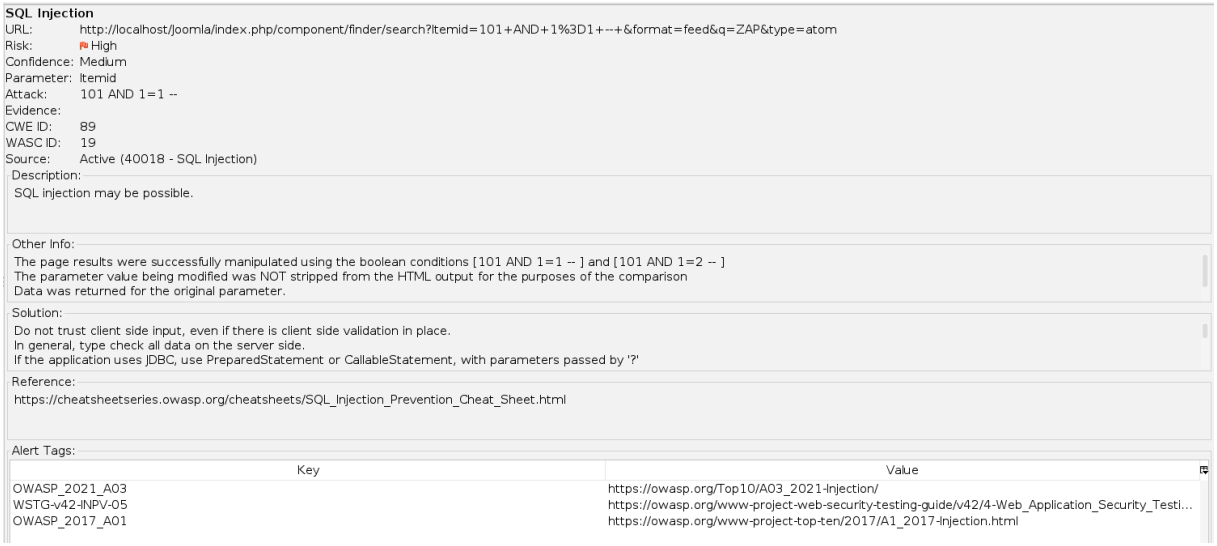


Рис. 3.27. Результат аналзу на можливість введення SQL ін’єкцій без активного захисного плагіну



Рис. 3.28. Перелік загроз критичного рівня на можливість SQL ін’єкцій без активного захисного плагіну

Порівнюючи результати аналзу двох вебсайтів, можна констатувати, що захисний плагін дозволяє знизити кількість вразливостей, пов’язаних з SQL ін’єкціями, лише у випадку вебсайту, створеного на базі Wordpress. У той же час, як у випадку вебсайту, створеного на базі Joomla, захисний плагін не дав ніяких результатів. Для перевірки раніше виявлених вразливостей використаємо інструмент SQLmap. Для початку аналзу вводимо команду sqlmap -u “посилання на вебсайт” --batch –banner/--password, після чого вебсайт атакується з використанням SQL ін’єкцій (рис. 3.29 і рис. 3.30).



Рис. 3.29. Результат аналізу на SQL ін’єкції вебсайту на базі WordPress

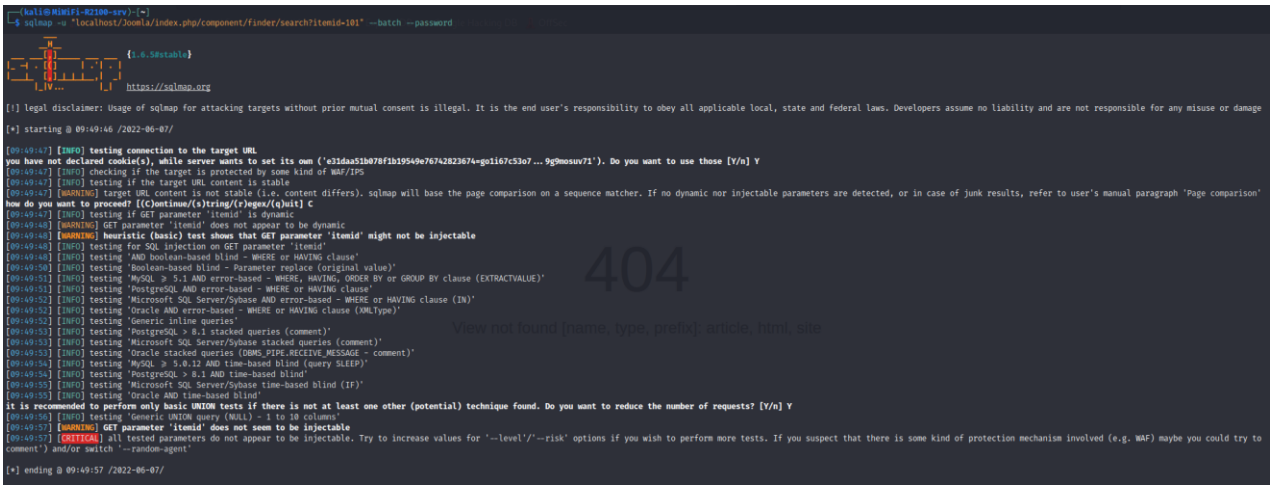


Рис. 3.30. Результат аналізу на SQL ін’єкції вебсайту на базі Joomla

Аналізуючи отримані результати, SQLmap не виявив можливості на введення SQL ін’єкцій до вебсайтів, хоча при скануванні OWASP ZAP такі вразливості були ідентифіковані. Попри такі результати було проведено ще один аналз. Була здійснена спроба атакувати знайдені вразливості за допомогою введення запропонованих ін’єкцій у вразливі місця, які було знайдено за допомогою OWASP ZAP. При виконанні запиту «http://localhost/Joomla/index.php/component/finder/search?itemid=101+AND+1%3D1+--&format=feed&q=ZAP&type=atom», у створеному вебсайті на базі CMS системи Joomla було отримано такі результати (рис. 3.31).

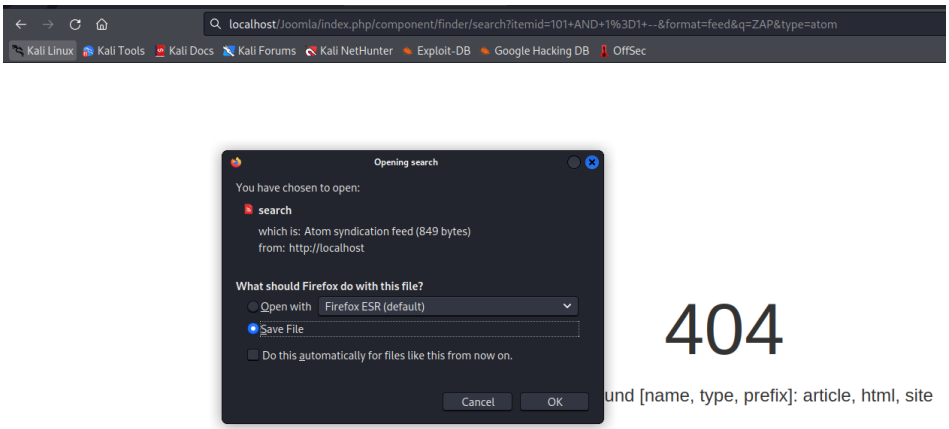


Рис. 3.31. Результат аналзу на SQL ін’єкції вебсайту на базі Joomla за допомогою ручного аналзу

У результаті отримано запит на завантаження файлу під назвою «search», який містить в собі такі дані, зображені на рис. 3.32.

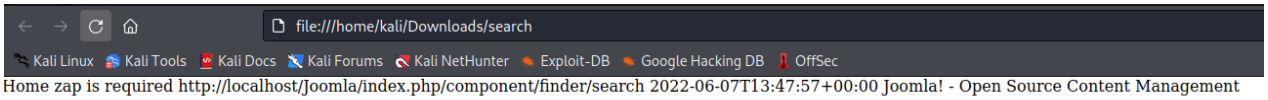


Рис. 3.32. Вміст файлу під назвою «search»

Як можна переконатися, жодний чутливих даних не було виявлено. Проте це підводить нас до того, що можливість проникнення дуже висока.

При спробі ручного введення SQL ін’єкцій у вебсайт, створений на базі CMS системи WordPress, таких результатів досягнуто не було.

За підсумками аналзу захищеності можна відзначити, що збір інформації про вебсайти вдалося здійснити просто та без перешкод. У результаті з’ясовано який сервер був використаний та його версію, інформацію про використані технології та CMS систему при створенні, її версію. Тестування HTTP методів показало, що доступними є такі методи: GET, POST, OPTIONS та GET. Аналз на можливість введення сторонніх скриптів засвідчив, що така загроза існує в незалежності від увімкненого захисного плагіну. Аналз на можливість введення SQL ін’єкцій показав, що така загроза існує, хоча і не вдалося це підтвердити за допомогою автоматизованої системи на обох вебсайтах. Проте використовуючи

ручне тестування, було підтверджено одну з виявлених вразливостей у вебсайті, створеному на базі CMS системи Joomla.

Загальні дані аналзу відображені у Табл. 3.1.

Таблиця 3.1

Порівняння результатів аналзу

CMS система	WordPress з захисним плагіном	Joomla з захисним плагіном	WordPress без захисного плагіну	Joomla без захисного плагіну
Введення сторонніх скриптів	4	3	4	3
SQL ін’єкції	4	3	3	3
Ручний аналз SQL ін’єкції	5	3	5	3

Оцінки виставлені у діапазоні від 1 до 5: 1 – не захищена, 2 - майже не захищена, 3 – середня захищеність, 4 – майже захищена, 5 - захищена.

Загалом аналз захищеності вебсайтів, створених за допомогою CMS систем WordPress та Joomla, показав, що у цих вебсайтах доступний збір інформації, а також вдалося дзнатися, які порти використовуються та доступні методи HTTP. Аналз на вразливість до XSS засвідчив, що створені вебсайти мають такі вразливості, а захисні плагіни не допомагають їх позбутися. Проте деякі вразливості, які пов’язані з SQL ін’єкціями, плагіни Jetpack та Watchful змогли заблокувати.

ВИСНОВКИ

У даній роботі було досліджено основні вразливості систем управління вмістом та розроблено методику аналізу захищеності вебсайту, створеного за допомогою CMS систем. Для проведення аналізу захищеності створено вебсай на базі CMS систем WordPress та Joomla. Основою для розробки методики дослідження стала існуюча методологія тестування безпеки вебсайтів OWASP Web Security Testing Guide, з якої обрано відповідні окремі етапи. Для обраних етапів здійснено аналіз та опис автоматизованих і ручних методів аналізу захищеності, розглянуто спеціалізований інструментарій, за допомогою якого можна проаналізувати захищеність вебсайтів. Зокрема, для аналізу було обрано такі інструменти, як WhatWeb, Nmap, OWASP ZAP та SQLmap. Кожен із таких інструментів має свої функціональні можливості і відповідає за конкретні етапи.

Проведений аналіз захищеності вебсайтів, створених на базі CMS систем WordPress та Joomla, показав, що за достатнього рівня захисту певні недоліки у захищеності ресурсів все ж існують. Зважаючи на результат аналізу, встановлено, що вебсайти, які розроблено з використанням відповідних плагінів (Starter Template, WooCommerce, Jetpack) для WordPress та (HikaShop, Watchful) для Joomla, мають середній рівень захищеності. Визначено основні вразливості вебсайтів, що створені на базі CMS систем WordPress та Joomla. При цьому варто враховувати, що аналіз здійснюється у «стерильних» умовах на локальному сервері, що вплинуло на результати аналізу. Наразі виявлені загрози не є критичними, тож створені вебсайти можна вважати безпечними для користувача.

Аналіз підтвердив факти того, що використання системи управління вмістом є доцільними через простоту створення вебсайту і вбудований захист для користувачів без спеціальних навичок розроблення вебсайту.

СПИСОК ЛІТЕРАТУРИ

1. This CMS cyberattack has affected thousands of sites worldwide. URL: https://www.techradar.com/news/this-cms-cyberattack-has-affected-thousands-of-sites-worldwide
2. Злам API Drupal. URL: https://apisecurity.io/issue-20-drupal-apis-hacked-eu-iot-standards/
3. How to Secure Your Content Management System (CMS) URL: https://securityboulevard.com/2019/08/how-to-secure-your-content-management-system-cms/
4. Что такое WordPress. URL: https://www.hostinger.com.ua/rukovodstva/chto-takoe-wordpress-obzor-populjarnoj-cms/
5. Основи: Drupal як система управління сайтом (CMS). URL: https://www.drupal.org/ru/docs/user_guide/ru/understanding-drupal.html
6. LAMP. URL: https://uk.wikipedia.org/wiki/LAMP
7. Joomla. URL: https://joomla.ru/docs/articles/2676-five-reasons
8. CMS Security: How to Keep Your Website Safe. URL: https://www.brandextract.com/Insights/Articles/CMS-Security-How-to-Keep-Your-Website-Safe/
9. CMS Vulnerabilities: Why Are CMS Platforms Common Hacking Targets URL: https://beaglesecurity.com/blog/article/cms-vulnerabilities.html
10. How Hackers Exploited your WordPress Website in 2018. URL: https://www.getastra.com/blog/cms/wordpress
11. 5 Most Critical Vulnerabilities Ever Found on Drupal. URL: https://www.getastra.com/blog/cms/drupal-vulnerability-5-most-critical-vulnerabilities-ever-found-on-drupal/

12. Most Common Vulnerabilities Found in Joomla. URL: https://www.getastra.com/blog/cms/joomla-security/joomla-vulnerabilities-3-most-common-vulnerabilities-found/
13. Starter Templates — Elementor, WordPress & Beaver Builder Templates: URL: https://wordpress.org/plugins/astra-sites/
14. WooCommerce. URL: https://wordpress.org/plugins/woocommerce/
15. Jetpack. URL: https://wordpress.org/plugins/jetpack/
16. HikaShop. URL : https://www.hikashop.com/
17. Watchful. URL: https://watchful.net/
18. Nikto. URL: https://cirt.net/Nikto2
19. OWASP ZAP. URL: https://www.zaproxy.org/docs/
20. SQLmap. URL : https://sqlmap.org/
21. Nmap. URL : https://nmap.org/
22. Vega Scanner. URL: https://subgraph.com/vega/
23. WPScan. URL: wpscan.com
24. WhatWeb. URL: https://github.com/urbanadventurer/WhatWeb
25. OWASP WSTG - v4.2. URL: https://owasp.org/www-project-web-security-testing-guide/v42/