

## АНОТАЦІЯ

«Розробка CMS та методів захисту web-сайтів на її основі» // Дипломна робота ОР «Магістр» // Тригубець Богдан Іванович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра кібербезпеки, група СБм-61 // Тернопіль, 2019 // С. 97, табл. – 3, рис. – 18, додат. – 1.

Ключові слова: CMS, WEB-САЙТ, ІНТЕРНЕТ-МАГАЗИН, ВРАЗЛИВОСТІ, АТАКИ, МЕТОДИ ЗАХИСТУ, УКРПОШТА API, LIQPAY API, SENDGRID API, ONESIGNAL API, TELEGRAM API, CLOUDFLARE, DDOS, XSS, SQL ІН'ЄКЦІЇ, SSL, FTP, SSH, SAMESITE COOKIE.

Метою магістерської роботи є дослідження існуючих інструментів для спрощення взаємодії користувача з інтернет-магазинами та для спрощення взаємодії інтернет-магазинів з службами доставки, розробка на їх основі власної CMS для українського ринку, а також аналіз основних загроз роботи web-сайтів та їх врахування при розробці захищеної CMS та web-сайтів, розроблених на її основі.

У першому розділі було проаналізовано методи створення сучасних web-сайтів, проаналізовано функціонал деяких CMS для інтернет-магазинів, популярних на українському ринку. Також було досліджено найбільші проблеми захисту web-сайтів, розглянуто найбільш популярні атаки на сервер web-сайту та програмну частину web-сайту.

У другому розділі описуються міркування на рахунок необхідного функціоналу сучасної CMS для інтернет-магазину, описуються наявні на ринку технологічні інструменти та процес їх інтеграції в N83 CMS, та описуються методи захисту серверу web-сайту та його програмної частини від найбільш популярних атак.

У третьому розділі описується робота розробленої CMS, показується робота модуля авторизації та механізм роботи двохфакторної аутентифікації,

робота з розділом замовлень та публікацій, інструменти для автоматичної генерації накладних відправлень для поштових служб.

У четвертому розділі відбувається тестування розробленої CMS на наявність деяких популярних вразливостей за допомогою утиліт, наявних у середовищі тестування Kali Linux.

У п'ятому розділі відбувається розрахунок норм часу на виконання науково-дослідної роботи та розрахунок усіх витрат, визначається термін окупності та обґрунтовується економічна ефективність.

У шостому розділі описуються основні правила та нормативи, яких було дотримано при проведенні даної розробки.

У сьомому розділі описуються організаційні форми, види та способи статистичного спостереження в екології.

У результаті підготовки магістерської роботи розроблено сучасну захищену CMS для web-сайтів, зокрема інтернет-магазинів, у яку через API інтегровано програмні рішення від популярних поштових служб, платіжних систем, та інших сервісів, які найбільше підходять для роботи на українському ринку.

## ANNOTATION

«Development of CMS and web site security methods» // Master's Thesis // Bohdan Trygubets // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information System and Software Engineering, Department of Cybersecurity // Ternopil, 2019 // P. 97, Tables – 3 , Fig. – 18, Annexes. – 1.

Keywords: CMS, WEB SITE, ONLINE SHOPPING, VULNERABILITY, ATTACK, PROTECTION METHODS, UKRPOSHTA API, LIQPAY API, SENDGRID API, ONESIGNAL API, TELEGRAM API, CLOUDFLARE, DDOS, XSS, SQL INJECTION, SSL, FTP, SSH, SAMESITE COOKIE.

The aim of the master's thesis is to research existing tools to facilitate user interaction with online stores and to facilitate interaction of online stores with delivery services, to develop new own CMS for the Ukrainian market, as well as to analyze the main threats to the websites and their consideration in the development secure CMS and web sites based on it.

The first section was analyzed methods for creating advanced web-sites, analyzed some functional CMS for online stores popular in the Ukrainian market. Also investigated was the biggest problem of protection of web-sites, considered the most popular attacks on web-site server and software of web-site.

The second section describes the considerations for the required functionality of a modern CMS for the online store, describes the commercially available technological tools and the process of integrating them into the H83 CMS, and describes methods for protecting the web site server and its software against the most popular attacks.

The third section describes the work of the developed CMS, shows the work of the authorization module and the mechanism of work of two-factor authentication, work with the section of orders and publications, tools for automatic generation of invoices for mail services.

The fourth section occurs CMS developed test for the presence of certain popular vulnerabilities using tools available in the test environment Kali Linux.

The fifth section calculates the standards of time spent on the research work and the calculation of all costs, defines the payback period and substantiates economic efficiency.

The sixth section describes the basic rules and regulations that were followed during this development.

The seventh section describes the organizational forms, types and methods of statistical observation in ecology.

As a result of master's work, a modern secure CMS was developed for web sites, in particular online stores, which through API integrates software solutions from popular mail services, payment systems and other services that are most suitable for work in the Ukrainian market.

## ЗМІСТ

<b>ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ .....</b>	<b>9</b>
<b>ВСТУП.....</b>	<b>10</b>
<b>РОЗДІЛ 1 СТВОРЕННЯ WEB-САЙТІВ ТА ПРОБЛЕМИ ЇХ ЗАХИСТУ ....</b>	<b>12</b>
1.1 Створення web-сайтів .....	12
1.1.1 Підхід до створення web-сайту .....	12
1.1.2 Огляд існуючих CMS та їх порівняння .....	14
1.2 Атаки на сервер .....	15
1.2.1 DDoS - атаки .....	15
1.2.2 Атака через FTP/SSH .....	17
1.2.3 Відсутність SSL-сертифікату .....	18
1.3 Атаки на програмну частину .....	18
1.3.1 SQL-ін'єкції .....	18
1.3.2 Атаки типу Cross Site Scripting (XSS) .....	20
1.3.3 Недостатня аутентифікація та авторизація .....	21
1.3.4 Вразливості плагінів популярних CMS .....	23
1.4 Висновки до розділу .....	24
<b>РОЗДІЛ 2 РОЗРОБКА CMS ТА МЕТОДІВ ЗАХИСТУ WEB-САЙТУ .....</b>	<b>25</b>
2.1 Визначення функціоналу нової CMS .....	25
2.1.1 Вимоги до розробки нової CMS .....	25
2.1.2 Вибір інструментів для розробки CMS .....	26
2.2 Розробка CMS .....	26
2.2.1 Інтеграція API Укрпошти .....	26
2.2.4 Авторизація через API соцмережі Facebook .....	34
2.2.5 Інтеграція з платіжною системою LiqPay .....	35

	6
2.2.6 СМС та Email-сповіщення .....	37
2.2.7 Push-сповіщення через API OneSignal .....	38
2.2.8 Інтеграція з API Telegram .....	39
2.2 Захист серверу .....	41
2.2.1 Використання сервісу Cloudflare .....	41
2.2.2 Використання SSH-ключа замість паролю .....	42
2.2.3 Використання протоколу HTTP/3 .....	44
2.2.4 Безпечне з'єднання через криптографічні протоколи TLS/SSL .....	46
2.2.5 Використання Kernel-based Virtual Machine (KVM) .....	48
2.2.6 Правильне налаштування .htaccess та .htpasswd .....	49
2.3 Захист програмної частини .....	51
2.3.1 Безпечне програмування та перевірка вхідних даних .....	51
2.3.2 Захист від передачі сторонніх cookie і прихованої ідентифікації.....	54
2.3.3 Двохфакторна аутентифікація через Google Authenticator .....	55
2.3.4 Розмежування прав доступу користувачів до функцій адмін-панелі ..	56
2.3.5 Переваги використання власної розробки над популярними рішеннями .....	57
2.4 Висновки до розділу .....	57
<b>РОЗДІЛ 3 РОБОТА CMS.....</b>	<b>58</b>
3.1 Модуль авторизації та аутентифікації .....	58
3.1.1 Авторизація за допомогою паролю .....	58
3.1.2 Другий фактор аутентифікації .....	59
3.1.3 Генерація токена сесії та його перевірка .....	60
3.2 Функції адмін-панелі .....	61
3.2.1 Створення публікації .....	61
3.2.2 Каталог товарів .....	62
3.2.3 Розділ замовлень .....	63
3.3 Журнал подій .....	64

	7
3.3.1 Звіт про події в адмін-панелі .....	64
3.3.2 Логування з використанням LogDNA .....	64
3.4 Розподіл прав користувачів .....	66
3.4.1 Видача прав користувачу .....	66
3.4.2 Спроба доступу до розділу без наявності прав .....	66
3.5 Висновки до розділу .....	67
<b>РОЗДІЛ 4 ТЕСТУВАННЯ АТАК НА WEB-САЙТ .....</b>	<b>68</b>
4.1 Вибір середовища тестування .....	68
4.2 Тестування на SQL-ін'єкції .....	68
4.3 Тестування на атаку XSS .....	69
4.4 Тестування на brute-force-атаку на SSH .....	70
4.5 Висновки до розділу .....	71
<b>РОЗДІЛ 5 ОБҐРУНТУВАННЯ ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ.....</b>	<b>72</b>
5.1 Розрахунок норм часу на виконання науково-дослідної роботи.....	72
5.2 Визначення витрат на оплату праці та відрахувань на соціальні заходи ...	73
5.3 Розрахунок матеріальних витрат .....	75
5.4 Розрахунок витрат на електроенергію .....	76
5.5 Розрахунок суми амортизаційних відрахувань.....	76
5.6 Обчислення накладних витрат .....	77
5.7 Кошторис та визначення собівартості науково-дослідницької роботи .....	77
5.8 Розрахунок ціни науково-дослідної роботи .....	78
5.9 Визначення економічної ефективності і терміну окупності .....	79
5.10 Висновки до розділу .....	80
<b>РОЗДІЛ 6 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ.....</b>	<b>81</b>
6.1 Охорона праці .....	81
6.2 Фактори ризику та можливі порушення здоров'я користувачів комп'ютерної мережі .....	84
6.3 Висновки до розділу .....	88

<b>РОЗДІЛ 7 ЕКОЛОГІЯ.....</b>	<b>89</b>
7.1 Організаційні форми, види і способи статистичного спостереження в екології .....	89
7.2 Джерела електромагнітних полів, іонізуючих випромінювань і методи їх знешкодження .....	91
7.3 Висновки до розділу .....	93
<b>ВИСНОВКИ .....</b>	<b>94</b>
<b>БІБЛІОГРАФІЯ .....</b>	<b>95</b>



## **ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ**

CMS - Content management system  
DDoS - Distributed Denial-of-service attack  
IP - Internet Protocol  
FTP - File Transfer Protocol  
SSH - Secure Shell  
SSL - Secure Sockets Layer  
HTTP - HyperText Transfer Protocol  
SQL - Structured Query Language  
PHP - Hypertext Preprocessor  
XSS - Cross-Site Scripting  
API - Application Programming Interface  
AJAX - Asynchronous JavaScript And XML  
URL - Uniform Resource Locator  
AES - Advanced Encryption Standard  
JSON - JavaScript Object Notation  
TLS - Transport Layer Security  
TCP - Transmission Control Protocol  
CDN - Content Delivery Network  
QUIC - Quick UDP Internet Connections  
OSI - Open Systems Interconnection model  
KVM - Kernel-based Virtual Machine  
HOTP - HMAC-Based One-Time Password Algorithm  
TOTP - Time-based One-Time Password Algorithm  
WYSIWYG - What You See Is What You Get  
XML - Extensible Markup Language

## ВСТУП

*Актуальність дослідження.* У сучасному технологічному світі web-сервіси стали невід’ємною частиною життя багатьох людей та інструментом для вирішення багатьох повсякденних проблем, а поняття безпеки відіграє одну з передових ролей у сучасному інформаційному просторі в цілому. На сьогодні їх створення значно спрощується у зв’язку з наявністю готових CMS, які дозволяють не вдаватися до послуг web-розробників.

Проте попри дешевизну та швидкість розробки web-сайтів на основі готових CMS, цей варіант має певні недоліки у функціоналі та безпеці. Адже при розробці власної системи для керування сайтом з’являється можливість детального налаштування функціоналу web-сайту під потреби конкретного проекту, а гнучкість системи дозволяє інтегровувати будь-який складний функціонал та сторонні сервіси, які можуть бути не представлені у вигляді додаткових модулів для існуючих CMS, що надає суттєві переваги при розробці нестандартних рішень.

З точки зору безпеки головною перевагою є відсутність готових рішень для реалізації популярних атак, які можна з легкістю знайти у мережі під більшість популярних CMS. Необхідність персоналізованої розробки для атаки на CMS власної розробки суттєво збільшує вартість цих атак.

Саме тому аналіз існуючих інструментів для спрощення взаємодії користувача з інтернет-магазинами, які відсутні на ринку, розробка на їх основі власної CMS, а також аналіз основних загроз роботи web-сайтів, та їх врахування при розробці захищеної CMS, є актуальним.

*Мета і завдання дослідження.* Метою магістерської роботи є дослідження розробка власної CMS для українського ринку для спрощення взаємодії користувача з інтернет-магазинами та для спрощення взаємодії інтернет-магазинів з службами доставки, з врахуванням основних загроз роботи web-сайтів. Мета дослідження обумовила поставлення та розв’язання наступних завдань:

- проаналізувати популярні CMS для інтернет-магазинів;

- дослідити функції, інтеграція яких спростить взаємодію користувача з інтернет-магазином, та спростить взаємодію менеджерів з службами доставки при відправці замовлень;

- проаналізувати популярні вразливості та атаки на web-сайти;
- дослідити методи захисту CMS від можливих атак;
- обґрунтувати використання вибраних технологій;
- розробити сучасну та захищену CMS для інтернет-магазину.

*Об'єктом* дослідження магістерської роботи є функціонал та система захисту CMS.

*Предметом* дослідження є моделі, структури та методи створення CMS.

*Методи дослідження.* В процесі цього дослідження було використано такі загальнонаукові методи пізнання як порівняння, системний аналіз та пошуку методів для забезпечення інформаційної безпеки web-ресурсів.

*Наукова новизна роботи:* було розроблено нову захищену CMS для інтернет-магазинів з використанням сучасних інтеграцій для ефективної роботи на українському ринку та забезпечення безпечної та функціональної роботи CMS;

*Практичне значення* дослідження полягає у застосуванні розробленої CMS для створення захищених web-сайтів з інтернет-магазинами на українському ринку.

*Апробація результатів дослідження.* Основні положення та результати дослідження обговорювалися на VII науково-технічній конференції Тернопільського Національного Технічного Університету імені Івана Пулюя “Інформаційні моделі, системи та технології” (Тернопіль, 2019).

## **РОЗДІЛ 1**

### **СТВОРЕННЯ WEB-САЙТІВ ТА ПРОБЛЕМИ ЇХ ЗАХИСТУ**

#### **1.1 Створення web-сайтів**

##### **1.1.1 Підхід до створення web-сайту**

Доволі важко переоцінити роль web-сайтів та мобільних технологій у сучасному світі. У наш час web-сервіси стали невід’ємною частиною життя багатьох людей, і в найбільш розвнених містах світу повсякденна взаємодія людини з звичними речима найчастіше відбувається за участі технологічних сервісів, web-сайтів та мобільних додатків. Виклик таксі, замовлення їжі, навіть ввімкнення світла у кімнаті, все це тепер відбувається за участю спеціальних посередників, які намагаються не лише максимально професійно виконувати свою роботу, але й спрощувати життя та допомагати не витратити на звичні дії багато часу. Частка таких активних користувачів щороку суттєво збільшується, технології поширюються все далі за межі міст та країн, у яких їх було створено, та інтенсивно розвиваються.

Одним з найпопулярніших типів сервісів у цифрову епоху є web-сайти. Їх функціонал може бути дуже різним — звичайні новинні блоги мають мінімальний функціонал для читання новин та статей, сучасні інтернет-магазини дозволяють переглядати та купувати найрізноманітніші товари з зручною доставкою, а унікальні web-сервіси на основі власних розробок надають самі різні унікальні послуги, та задають нові користувацькі тренди.

Першим етапом розробки будь якого web-сайту чи web-сервісу є чітке визначення завдань, які він має вирішити. Саме від функціоналу та послуг, які повинен надавати сервіс, і залежить вибір технологій, які будуть використовуватись для його розробки.

Розробка будь-яких web-сайтів являє собою написання індивідуального програмного коду під кожен з них, проте, в зв’язку з тим, що усі web-сайти мають досить схожу структуру, та в більшості випадків діляться на чіткі групи, розробниками було створено багато готових рішень, які можуть допомогти

користувачу створити необхідний сервіс з базовими функціями, використовуючи готові напрацювання розробників у попередніх проектах. На основі цих розробок було створено різноманітні платформи, які відомі під назвою CMS - Content management system, тобто система керування вмістом web-сайту.

При мінімальному функціоналі, наприклад для створення блогу, зазвичай немає необхідності вдаватися до послуг web-розробників. На сьогодні існує вже багато спеціальних сервісів, які пропонують готові CMS, що дають змогу звичайному користувачу, без навиків програмування та розробки, створити власний інтернет-блог чи сайт-візитку для онлайн-представлення власного бізнесу. Для таких цілей це найкращий варіант, адже він не вимагає багато фінансових та часових затрат, та дозволяє швидко розробити необхідний функціонал, використовуючи вже готові сторонні розробки. Наразі багато сервісів надають платформи під створення і набагато складніших за функціоналом сайтів, таких як інтернет-магазини. Вони надають базові функції з можливістю додавання каталогу товарів та їх оплати за оптимальну ціну, проте для подальшого вдосконалення їх функціоналу всерівно вимагають втручання спеціалістів, які зможуть розробити додаткові модулі для даної платформи під потреби клієнта. Такі рішення є оптимальними для малого бізнесу, який не має достатніх ресурсів для виходу на ринок з унікальним продуктом.

Проте якщо задачі, які були поставлені перед розробником, вимагають креативних та унікальних рішень, які на даний момент не представлені на ринку, або вимагають внесення суттєвих корективів у існуючі рішення, найкращим варіантом буде розробка власної CMS-платформи. Це дозволить максимально гнучко налаштувати web-сайт, врахувавши найдетальніші побажання як у дизайні, так і в нюансах функціоналу. Тому для створення якісного сервісу та для вирішення поставлених завдань на найкращому рівні web-сайт має використовувати лише найтехнологічніші інструменти, розробку яких найоптимальніше здійснювати на платформі власної розробки.

### 1.1.2 Огляд існуючих CMS та їх порівняння

Основними перевагами сучасного web-сайту для інтернет-магазину є максимальна зручність замовлення товару для користувача, та максимальна зручність обробки замовлення та його передача для відправки у поштову службу. Деякі з цих задач неможливо вирішити оптимальним технологічним шляхом (такі як пакування товару), тому основною задачею CMS є оптимізація, де це можливо, усіх кроків замовлення на сайті. Після вивчення можливих вдосконалень було визначено, що профільні сервіси платіжних систем та служб доставки надають доступ до необхідних функцій через API, тому подальшим кроком є або використання CMS з готовим продуктом, або ж створення нової CMS з використанням цих технологій. Для пошуку найоптимальнішого вирішення поставлених завдань було проаналізовано популярні CMS, результати порівняння доступні у таблиці 1.1.

Таблиця. 1.1 - Таблиця порівняння деяких функцій популярних CMS у модулях замовлення та оплати

Назва	Автодоповнення адреси доставки	Інтеграція з українською платіжною системою	Автоматизація генерації накладних відправлень
WordPress	-	+	-
Хорошоп	+	+	-
OpenCart	-	-	-
Shopify	-	-	-
1С Бітрікс	+	+	-
H83 CMS	+	+	+

За результатами проведеного порівняння було виявлено, що ніодна з представлених на українському ринку CMS не містить інтеграції усіх

доступних на ринку функцій для оптимізації роботи інтернет-магазину, тому для їх повноцінної інтеграції необхідно використовувати власну розробку.

В списку першочергових переваг власної CMS над шаблонними є легкість підтримки та максимальна гнучкість у розробці нових модулів. Серед основних функціональних недоліків усіх шаблонних CMS для інтернет-магазинів є:

- наявність великої кількості зайвого коду, що сповільнює роботу сайту та створює велике навантаження на сервер;
- відсутність технологічного функціоналу, адаптованого під український ринок, у більшості проаналізованих систем;
- наявність великої кількості вразливостей, які було розроблено для атак на популярні CMS, та які легко знайти у вільному доступі в мережі інтернет.

## **1.2 Атаки на сервер**

### **1.2.1 DDoS - атаки**

Сучасні веб-сервери щосекундно приймають на себе величезну кількість трафіку, тому пропускна здатність будь-якого сервера завжди є вразливим місцем, а один з найпопулярніших видів атаки - перенавантаження сервера штучно згенерованими запитами. Використовуючи потужні DDoS-атаки можна не лише завадити нормальній роботі певного сайту чи web-сервісу, але й порушити нормальну роботу цілого сегменту мережі.

Суть DDoS-атаки, повна назва якої distributed denial-of-service (розподілена атака на відмову в обслуговуванні), полягає в направленні величезної кількості запитів на певний сервер, кількість яких перевищує його пропускну здатність, в результаті чого від перенавантаження сервер перестає відповідати на запити реальних користувачів.

Щоб проілюструвати це на простому прикладі, можна уявити сервер, який генерує веб-сторінку для відвідувачів за пів секунди. При такій кількості запитів цей сервер може нормально функціонувати. Відповідно, якщо кількість запитів до веб-ресурсу буде більше, ніж два запита в секунду, усі наступні

запити будуть виставлятися в чергу, і сервер буде відповідати на них, як тільки відповідь на попередні. При чотирьох запитах в секунду це зумовить невеликі затримки при завантаженні сторінки, які можуть бути непомітними користувачу. Проте якщо кількість запитів буде постійно зростати, і черга на обробку буде складатися лише з запитів, згенерованих DDoS-атакою, це сильно вплине на час відповіді серверу. Відповідно, коли кількість запитів буде збільшуватись у десятки разів, усі нові запити від користувачів фактично перестануть оброблятися, що зупинить нормальне функціонування сервісу. Саме це і є метою будь-якої DDoS-атаки [27].



Рисунок 1.1 Схема сучасної DDoS - атаки

На початках розвитку мережі інтернет атаки такого типу здійснювались зазвичай з одного пристрою за допомогою спеціальних програм. Проте так як потік запитів відправлявся з однієї IP-адреси, ці атаки можна було легко виявити та знешкодити, додавши IP-адресу пристрою, чи цілий діапазон IP-адрес, в список блокування. З розвитком технологій подібні атаки почали здійснюватись з використанням відразу великої кількості пристроїв.



Ця мережа, створена зловмисниками, може складатися з відеокамер, холодильників, ротуретів чи інших пристроїв. Зазвичай доступ до них отримується підбором стандартних паролів, які власники пристроїв не змінили після покупки, або через вразливості старого програмного забезпечення. Саме недбалість та низька інтернет-грамотність більшості користувачів є основою для DDoS-атак. Ці пристрої зазвичай не мають швидкого підключення до мережі інтернет, проте “ботнет” може складатися з десятків та тисяч пристроїв, які знаходяться в різних куточках світу [4].

Використовуючи один з вищенаведених методів DDoS-атаки, та відсилаючи одночасно запити на один сервер у великій кількості, їх загальна швидкість може досягати 10 Тб/с, що є надскладним для обробки об’ємом інформації для більшості серверів.

### **1.2.2 Атака через FTP/SSH**

Наступний різновид атаки на web-сайт відбувається через FTP (File Transfer Protocol) — протокол передачі файлів, який використовують для обміну файлами з сервером.

Найчастіше причиною успішного несанкціонованого доступу до web-сайту через атаку на FTP є наявність на комп’ютері користувача, який завантажує чи оновлює файли через цей протокол, вірусів. Скрипти зловмисників викрадають логіни та паролі для доступу до FTP, завдяки чому отримують повний доступ до файлів вашого сервера.

Небезпека починається з того, що сама по собі технологія FTP не є достатньо безпечною та захищеною. Файли, якими розробник обмінюється з сервером, можуть з легкістю перехопити зловмисники, просто підключившись до вашого з’єднання, адже данні, які передаються через протокол FTP, не шифруються. Для шифрування даних використовується окремий протокол прикладного рівня SSH (Secure Shell). Завдяки використанню ключа для шифрування даних на стороні користувача, і ключа розшифрування даних на стороні сервера, підключившись до вашого з’єднання зловмисник не отримає ніякої інформації, придатної для читання.

### **1.2.3 Відсутність SSL-сертифікату**

HTTP (Hypertext Transfer Protocol) — це дуже поширений протокол передачі гіпертекстових даних, який встановлює порядок та синтаксис передачі даних по мережі. Проте в стандартній версії протоколу ці дані також не шифруються, тому вони не є захищеними. Розвиток технологій та політика великих ІТ-компаній майже не залишає шансів на повноцінне функціонування web-сайтам, які не використовують безпечне з'єднання між клієнтом та сервером через SSL-сертифікат. Використання незахищеного протоколу HTTP несе велику кількість небезпек і вразливостей, проте він все ще займає вагомe місце в мережі інтернет.

Різниця між HTTP і HTTPS полягає саме в шифруванні, адже HTTP та HTTPS протоколи відрізняються тим, що в другому випадку дані по HTTP передаються з використанням додаткового шифрування SSL або TLS. Для більшості інформації, що надсилається через Інтернет, включаючи вміст майже усіх web-сайтів, використовується протокол HTTP або HTTPS.

Існує два основних види HTTP-повідомлень: запити та відповіді. HTTP-запити генеруються браузером користувача під час його взаємодії з веб-ресурсами. Наприклад, коли користувач натискає на гіперпосилання, браузер надішле серію HTTP-запитів для того, щоб отримати інформацію необхідну для візуалізації сторінки, яку завантажує користувач.

Усі ці HTTP-запити відправляються на сервер, на якому розміщений веб-ресурс, і відповідаючи на цей запит сервер генерує HTTP-відповідь на HTTP-запити.

## **1.3 Атаки на програмну частину**

### **1.3.1 SQL-ін'єкції**

Більшість web-сайтів в Інтернеті використовують для роботи зв'язку з технологій PHP + MySQL. Це допомагає коду web-додатків зручно взаємодіяти з базою даних для зберігання, виводу та обробки інформації. Саме в цих місцях і виникають вразливості. Якщо в місцях входу/виходу даних є

вразливості, та якщо вони не перевіряються на достовірність і відповідність тому стандарту, який у данному місці має використовуватись, зловмисники легко можуть цим скористатися. SQL-ін'єкція ніщо інше як “троянський кінь”, у якому разом з необхідними для звичайної роботи даними передається шкідливий код.

Якщо інтерпретувати це зрозумілою мовою, SQL-ін'єкція — це атака на базу даних, яка дозволяє зловмиснику виконати певну дію, яка не планувалася розробником. Розглянемо приклад SQL-ін'єкції, у якій серверна обробка, отримуючи параметр `id`, не перевіряє його формат та не перетворює в (`int`).

#### Лістинг 1.1 - Приклад коду для запиту

```
SELECT * FROM news WHERE id_news = 5
```

Якщо в структуру коду, зображеного у лістингу 1.1 потрапляє значення параметру `id` рівне 5, виконується стандартний запит типу <https://h83cms.website/news.php?id=5>, який видасть користувачу згенеровану сторінку з інформацією з комірки таблиці бази даних, яка міститься під `id = 5`.

Проте це відбувається лише при правильному введенні вхідних даних. Якщо зловмисник в якості параметру `id` передасть стрічку `-1 OR 1=1` ( <https://h83cms.website/news.php?id=5=-1+OR+1=1>), то виконається запит, зображений у лістингу 1.2.

#### Лістинг 1.2 - Приклад коду для запиту

```
SELECT * FROM news WHERE id_news = -1 OR 1=1
```

Тобто зловмисник передасть в якості параметру стрічку, в яку додано конструкцію на мові SQL, яка інтерпретується як частина запиту та внесе зміни в логіку виконання SQL-запиту. В данному прикладі сервер, відповідаючи на запит, вибере усі наявні в таблиці бази даних `news` новини, оскільки логічний вираз `1=1` завжди істинний.

На даний момент SQL-ін'єкція є не лише однією з найпоширеніших вразливостей, а ще й однією з найнебезпечніших за версією Open Web Application Security Project (OWASP), очолюючи цей рейтинг як у 2013, так і у 2017 роках [12]. Реалізація цієї атаки є дуже простою, так як SQL-сценарії доступні для завантаження на багатьох інтернет-ресурсах. Не дивлячись що це одна з найпоширеніших вразливостей, і більшості розробників про неї відомо, нові атаки все ж таки регулярно відбуваються. Ця проста у реалізації атака несе за собою дуже небезпечні наслідки, адже вона взаємодіє з базою даних web-сайту. Це може призвести як до витоку даних, що несе за собою великі репутаційні ризики, з якими регулярно стикаються великі інтернет-компанії, так і до повної втрати даних, що є непоправною втратою для web-сервісів.

### **1.3.2 Атаки типу Cross Site Scripting (XSS)**

Cross Site Scripting (XSS), з англійської “міжсайтовий скриптинг”, також є однією з поширених вразливостей web-ресурсів. В основі атаки типу XSS лежить вразливість програмного коду, яка дозволяє зловмиснику додавати власні скрипти на сторінки сайту, які генеруються сервером.

Незважаючи на те, що захист від цієї вразливості, як і від SQL-ін'єкції, є дуже простим в реалізації, часто саме ця вразливість у методах обробки залишається відкритою, що може мати критичні наслідки.

Атаки типу XSS зазвичай поділяють на два основних типи. Атаки першого типу, або пасивні атаки, відбуваються при натисканні користувачем посилання, в яке було інтегровано шкідливий JS - код. Приклад реалізації атаки першого типу зображено у лістингу 1.3

#### **Лістинг 1.3 - Приклад коду для запиту**

```
http://h83cms.website/index.php?word=<script src=http://evilhost/xss_code.js></script>
```

Атаки другого типу передбачають зберігання фрагментів шкідливого коду на сервері, який спрацьовує при завантаженні сторінки сайту користувачем. Найбільш вразливими до цього типу XSS-атаки є сайти категорії

Web 2.0, які дають можливість користувачам свого ресурсу генерувати певні фрагменти контенту на сторінках сайту (створення публікацій, коментарів, відгуків). Зловмисник додає шкідливий код до тексту у полях, доступних йому, після чого сервер, не перевіряючи цю інформацію, зберігає її в своїй базі даних. Так як ця інформація не обробляється належним чином, код стає активним при кожному завантаженні сторінки, яка використовує для своєї роботи данні з комірок в базі даних, у які був збережений шкідливий код [1].

XSS-атаки дозволяють завдати серйозної шкоди web-сайту, включаючи крадіжку конфіденційної інформації, спам, і навіть переадресацію всього сайту вцілому. Оскільки цей скрипт виконується в браузері у користувача, він має доступ до інформації в його cookie, тому може виконувати дії на сайті від імені користувача, який авторизувався на вашому сайті, наприклад, писати, читати чи видаляти повідомлення.

### **1.3.3 Недостатня аутентифікація та авторизація**

Наступною найбільш популярною та небезпечною загрозою функціонування захищеної системи є атаки, спрямовані на отримання доступу до функцій адмін-панелі сайту від імені адміністратора (або іншого користувача, який наділений великою кількістю прав). Суть цих атак полягає у використанні методів, які підміняють ідентифікатори користувача.

Ця вразливість виникає тоді, коли сервер може надає доступ до стратегічних функцій системи без належної аутентифікації, і найчастіше це відбувається через користувацький інтерфейс.

Перший крок до атаки — можливість відкриття адмін-панелі по передбачливому посиланні, наприклад, “/admin/”. Якщо ж посилання має інший уніфікований вигляд, існує загроза його наявності на інших сторінках сервера або інших загальнодоступних ресурсах. URL-адреса для входу у адмін-панель може бути знайдена шляхом пошуку на сторінках web-сайту, у повідомленнях про помилки, або шляхом читання іншої документації.

Якщо ж адреса адмін-панелі захищена від виявлення найпростішими методами, або якщо зловмисник все ж такий її дізнався, наступним етапом

захисту є введення логіну та паролю доступу. У цьому випадку загрозою є використання загальноприйнятих імен користувачів (таких як root, admin, user чи інших) та загальноприйнятих стандартних паролів (таких як admin, pass, 12345 чи інших). В цьому випадку зловмисник може з легкістю підібрати пароль доступу лише за декілька спроб.

Якщо для доступу було створено складний пароль, з використанням багатьох символів нижнього регістру, верхнього регістру, цифр та символів підкреслення, вгадати його буде складно. В цьому випадку зловмисники використовують метод Brute-force, що в перекладі “метод грубої сили”. За його допомогою алгоритми повного перебору паролів підставляють по чергові спочатку найпопулярніші пароль, а потім усі можливі варіанти. У деяких випадках для роботи цього методу потрібно надто багато часу, проте зловмисник отримує шанс на позитивний результат. Щоб запобігти цьому, необхідно використовувати обмеження на кількість спроб неправильного введення паролю та журналу логування. Перший метод завадить використанню стандартизованих інструментів brute-force-перебору, а другий метод сповістить адміністратора про поточну атаку.

Завдяки використанню усіх вищенаведених методів можна зменшити вірогідність успішної атаки на адмін-панель. Проте щоб виключити можливість проведення будь-яких успішних атак через недостатню аутентифікацію, на даний момент одним з найпопулярніших та найзручніших методів захисту доступу до особистого кабінету користувача (та адмін-панелі зокрема) є двофакторна аутентифікація.

Двофакторна аутентифікація — це наявність додаткового рівня захисту облікового запису на випадок, якщо пароль доступу до профілю було скомпрометовано. Окрім звичайного паролю, при вході в обліковий запис система буде вимагати ще одне підтвердження вашої ідентичності. Існує декілька типів другого фактора, таких як SMS-сповіщення з одноразовим паролем (які часто використовується в банківських додатках), додаткове секретне запитання, фізичний USB-ключ безпеки, або ж спеціальні сервіси для генерації одноразових паролів, такі як Google Authenticator.

Тому навіть якщо пароль було скомпроментовано, маючи пароль та не маючи другого фактора, зломисник не зможе зайти у ваш обліковий запис, як і навпаки: при наявності лише другого фактору для входу до вашого облікового запису необхідно дізнатись ще й ваш пароль. Саме така комбінація робить використання двофакторної аутентифікації необхідним явищем в сучасних реаліях.

Наступна проблема недостатньої авторизації полягає в тому, що деякі CMS не мають розмежування прав доступу до певних розділів та функцій адмін-панелі між різними користувачами. Тобто, сервіс надає однакові права на внесення змін в функціонал та контент сайту як для головного адміністратора, так і для звичайного редактора.

Проте, якщо користувач пройшов аутентифікацію (ввів логін і пароль, що відповідають зареєстрованому в адмін-панелі користувачу), це ще не означає, що він повинен мати доступ до усіх функцій. Тому для того, щоб уникнути проблеми людського фактору, у CMS має бути реалізований модуль розмежування прав доступу, який при правильному налаштуванні повинен не надавати звичайному користувачу доступ до важливих функцій для керування web-сайтом.

### **1.3.4 Вразливості плагінів популярних CMS**

З практики роботи з інформаційними системами відомо, що усі вони мають вразливості, пошуком яких щодня займаються як спеціалісти по кібербезпеці, так і зломисники. І основною причиною появи більшості вразливостей є людський фактор. Деякі вразливості з'являються на стиці розробок програмних блоків, адже різними модулями займаються люди з різних сфер, що деколи грає критичну роль саме в питаннях безпеки.

Також важко заперечити, що сумлінність роботи сторонніх розробників завжди потрапляє під питання. Для прикладу, CMS WordPress дозволяє встановлення на сайт зовнішніх плагінів, розробкою яких займаються не працівники компанії WordPress, а сторонні особи. І навіть якщо розробники компанії відповідально відносяться до підбору плагінів та їх перевірки, це не

виключає можливої наявності у них “бекдорів”, доступ до яких може бути доступним лише для зловмисників, і використаний ними пізніше в їх же злочинних цілях. Саме тому атаки через зовнішні плагіни — це одна з головних проблем для користувачів не тільки CMS WordPress, але я інших CMS, які підтримують встановлення зовнішніх плагінів.

Наступна проблема популярних CMS — наявність готових стандартизованих методів атаки. Чим популярніше програмне забезпечення — тим пристальніша увага різного роду спеціалістів до нього. Наявність автоматизованих рішень для brute-force-атак, вразливість через використання стандартних логінів та паролів, загальновідома адреса адмін-панелі — все це робить ймовірність легкого підбору методів атаки дуже високою.

#### **1.4 Висновки до розділу**

У результаті аналізу підходів до створення web-сайтів, аналізу функціоналу існуючих CMS та аналізу функціоналу, доступного на ринку у вигляді окремих сервісів, було прийнято рішення розробити власну CMS та поставлено чіткі завдання, які вона має вирішувати.

Проаналізувавши популярні атаки на сервер web-сайту та на його програмну частину було сформовано список основних загроз, захист від яких необхідно реалізувати у новій системі.



## РОЗДІЛ 2 РОЗРОБКА CMS ТА МЕТОДІВ ЗАХИСТУ WEB-САЙТУ

### 2.1 Визначення функціоналу нової CMS

#### 2.1.1 Вимоги до розробки нової CMS

З огляду на порівняння існуючих програмних рішень для розробки інтернет-магазину, наведених у пункті 1.1.2, та на те, що більшість сучасних розробок, які мають найбільш наближений функціонал, що відповідав би поставленим вимогам, не мають повноцінної інтеграції на український ринок (наприклад, налаштування місцевих сервісів оплати, інтеграція з місцевими поштовими операторами), було прийнято рішення розробити власну платформу, яка б вирішувала поставлені задачі. Основна задача — розробити зручну платформу для інтернет-магазину, яка б максимально використовувала існуючі технології та можливості інтеграції для виконання поставлених задач в реаліях місцевого українського ринку, а також давала можливість подальшого розширення функціоналу web-сайту без суттєвих труднощів.

З огляду на поставлені задачі, нова CMS для інтернет-магазину має містити наступні базові функції:

- можливість групування товарів по розміру
- можливість групування та перелінкування схожих товарів;
- можливість фільтрації товарів у розділі по багатьом параметрам;
- скрізний пошук по товарам на сайті (з використанням суміжних назв товарів, які не вказані в назві товару, як ключових слів);
- можливість встановлення знижок на товари;
- можливість групування товарів з різних категорії у одну колекцію;
- інтеграція з українськими поштовими операторами;
- можливість як кур'єрської доставки, так і отримання на відділенні;
- інтеграція з зручною українською платіжною системою;
- можливість вибору накладеного платежу для оплати при отриманні;
- аналіз доданих у кошик товарів та пропонування суміжних до них товарів у кошині;

- СМС та email-сповіщення про статус замовлення;
- автоматизація генерування накладних товару для поштового оператора;
- можливість відслідковування поточного статусу відправлення користувачем під час перегляду свого замовлення;

Наявність усіх вищеописаних функцій у початковій версії інтернет-магазину, та можливість швидкого додавання нового функціоналу, дозволить продуктам, розробленим на основі H83 CMS бути конкурентоспроможними на українському ринку в порівнянні з іншими CMS для інтернет-магазинів.

### **2.1.2 Вибір інструментів для розробки CMS**

Основною платформою для розміщення програмної розробки було обрано провайдер хмарних інфраструктур DigitalOcean. Компанія надає професійні послуги датацентру та центру обробки даних, надає сучасний функціонал для роботи з ними, та авторитетну позицію на ринку у своїй сфері.

Для написання програмного коду було вибрано доволі звичний для web-розробок інструмент — мову програмування PHP. Її функціонал задовільняє потреби розробки, а остання її модифікація відбулася у кінці 2019 року, що підтверджує її актуальність та постійну роботу спільноти над виправленням минулих помилок, вдосконаленням поточних рішень та розробкою нових.

Для взаємодії з зовнішніми сервісами та сторонніми API-платформами зазвичай використовується cURL. cURL — це назва крос-платформового програмного засобу, який дає можливість взаємодіяти з файлами, які знаходяться на сторонніх серверах за допомогою параметрів, які можуть передаватися у рядку URL. cURL-запити підтримують протоколи безпеки, які дозволяють шифрувати з'єднання.

## **2.2 Розробка CMS**

### **2.2.1 Інтеграція API Укрпошти**

Для доставки замовлень з інтернет-магазинів, які працюють на розробленій CMS-системі, було вибрано транспортну компанію Укрпошта.

Головною перевагою Укрпошти перед конкурентами є її мережа відділень: наявність більше 12 000 відділень забезпечують покриття 100% населених пунктів України [20]. Це дозволяє максимально розширити географію покупців, що дуже актуально для деяких типів бізнесу, і відповідно максимально збільшити кількість потенційних клієнтів.

Для того щоб максимально спростити взаємодію користувача з корзиною покупця в інтернет-магазині було вирішено використовувати інтеграцію з базою даних населених пунктів та вулиць від Укрпошти, доступ до якої надається партнерам через спеціальне API. Користувачу не потрібно самостійно повністю вводити свою адресу, адже спеціальна форма пропонує йому готові варіанти на основі введених ним перших букв міста або вулиці.

Використання співставлення введених користувачем даних з базою існуючих населених пунктів та вулиць допомагає уникнути опечаток та неправильного написання їх назв, адже дані введені через вибір варіантів, згенерованих та запропонованих API на основі даних користувача, додаються у базу даних замовлень у однаково структуризованому форматі, тому їх можна відразу використовувати для генерації накладних поштового відправлення, що допоможе уникнути введення неіснуючих адрес, і створених на основі цього проблем. Вартість доставки також автоматично обраховується, в залежності від ваги усіх обраних користувачем продуктів (попередньо при розробці бази даних товарів вказується вага кожного товару).

Автодоповнення міст здійснюється з використанням jQuery-плагіну Select2 [21], дані в форму, взяті з API Укрпошти, підвантажуються з використанням технології AJAX.

Системою передбачено два варіанти доставки — адресна кур'єрська доставка на вказану адресу, яка вибирається при умові оплати замовлення відразу на сайті, та доставка до обраного користувачем відділення Укрпошти, для того, щоб замовник міг оплатити відправлення під час отримання.

Першим кроком оформлення замовлення користувачем є введення ним свого імені та прізвища, та контактного номеру телефону. Процес заповнення адреси для кур'єрської доставки відбувається наступним чином: при введенні

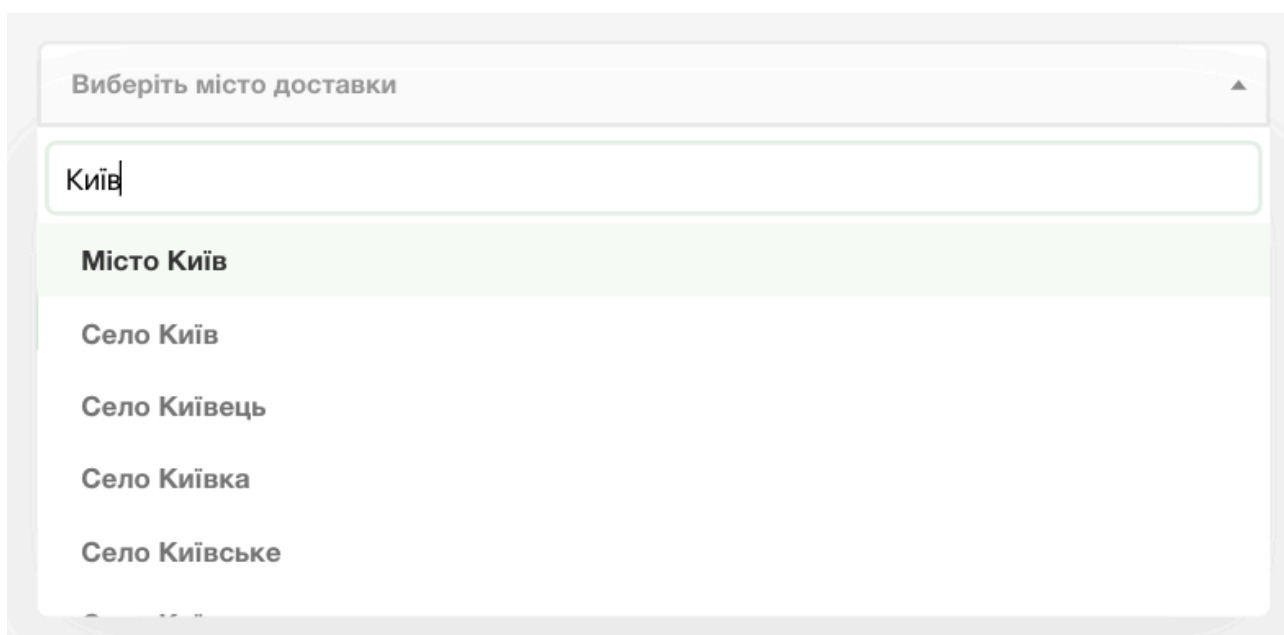
користувачем більше трьох символів у полі “місто доставки” йому пропонуються варіанти населених пунктів, назви яких розпочинаються з введених символів. Для отримання переліку відправляється запит API Укрпошти, показаний у лістингу 2.1.

#### Лістинг 2.1 - GET Request URI:

```
/get_city_by_region_id_and_district_id_and_city_ua?  
city_ua={cityUa}
```

де {cityUa} у лістингу 2.1 — це введені користувачем символи у поле “місто отримання”;

Після обробки отримані дані виводяться у форму, з запропонованого переліку користувач вибирає необхідний населений пункт.



Виберіть місто доставки ▲

Київ

- Місто Київ
- Село Київ
- Село Київець
- Село Київка
- Село Київське

Рисунок 2.1 Поле для введення назви міста

Далі користувачу пропонується спеціальне поле для введення вулиці, в якому після введення перших символів, аналогічно до запитів для визначення міста, відбувається звернення до API Укрпошти, показане у лістингу 2.2, яке пропонує користувачу найбільш схожі варіанти.

### Лістинг 2.2 - GET Request URI:

```
/
get_street_by_region_id_and_district_id_and_city_id_and_street_ua?
    city_id={cityId}&street_ua={street}
```

де `city_id={cityId}` у лістингу 2.2 — це `id` вибраного користувачем міста, а `street_ua={street}` у лістингу 2.2 — це перші символи вулиці, введені користувачем.

На наступному кроці користувачу пропонується ввести номер будинку, та при наявності, номер квартири. При виборі методу доставки у відділення, після вибору міста за аналогічним алгоритмом користувачу пропонується вибір відділення у формі, аналогічній до вибору вулиці при кур'єрській доставці.

Приклад структурованих даних, які записуються у базу даних для автоматичної генерації накладної відправлення, при адресній доставці, показані у лістингу 2.3.

Лістинг 2.3 - Приклад структурованих даних адреси отримувача при кур'єрській доставці на адресу отримувача.

```
{ "city_id": "1", "house_id": "34", "city_text": "Місто Київ", "street_id": "35656", "house_text": "34", "street_text": "провулок Ломоносова", "apartment_id": "50", "warehouse_id": "", "apartment_text": "50", "warehouse_text": "" }
```

Приклад структурованих даних, які записуються у базу даних для автоматичної генерації накладної відправлення, при доставці у відділення, показані у лістингу 2.4.

Лістинг 2.4 - Приклад структурованих даних адреси отримувача при доставці у відділення.

```
{ "city_id": "10761", "house_id": "", "city_text": "Місто Тернопіль", "street_id": "", "house_text": "", "street_text": "", "apartment_id": "", "warehouse_id": "46011", "apartment_text": "", "warehouse_text": "46011 №11 вул. Слівенська,15" }
```

Наступним кроком автоматизації роботи з відправленнями є створення розділу в адмін-панелі з можливістю перегляду усіх замовлень та з функцією автоматичної генерації накладної відправлення. Це набагато пришвидшує процес обробки відправлень, адже не потребує ні ручної генерації накладної відправлення у поштовому відділені, ані генерації через особистий кабінет користувача на сайті Укрпошти, де генерація накладної потребує заповнення великої форми, що при великій кількості замовлень сильно сповільнює роботу. Для спрощення роботи у цьому напрямку було розроблено поштовий модуль H83 CMS, який автоматизовує вказані вище ручні процеси.

Поштовий модуль H83 CMS автоматично генерує PDF-файл зі штрихкодом, використовуючи збережені дані відправника (інтернет-магазину), та дані отримувача, який зробив замовлення.

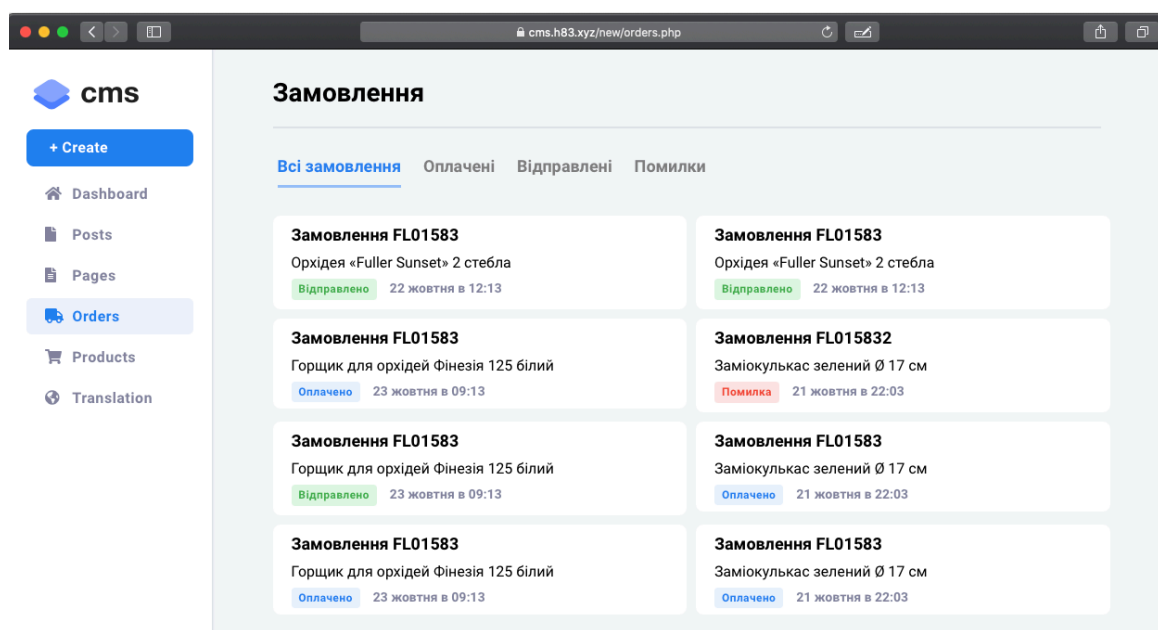


Рисунок 2.2 Вигляд адмін-панелі у розділі “замовлення”

Для цього послідовно використовується чотири методи з документації API Укрпошти. Першим кроком є генерація id адреси отримувача з використанням введених користувачем даних через метод /addresses. Приклад коду для запиту наведено в листінгу 2.5.

## Лістинг 2.5 - Приклад коду запиту до API Укрпошти через метод /addresses

```
curl -X POST "https://dev.ukrposhta.ua/ecom/0.0.1/addresses" \
  -H "accept: */*" \
  -H "Authorization: Bearer ec601f35-7be1-3aef-b283-1bb4931423ec" \
  -H "Content-Type: application/json" \
  -d "postcode":"46011","country":"UA","region":"Тернопільська область","city":"Тернопіль"}
```

де змінна \$postcode у лістингу 2.5 - це поштовий індекс адреси отримувача, який отримується з бази даних Укрпошти шляхом попередніх запитів користувачем при введенні своїх даних у формі,

а \$region, \$city, \$street\_text, \$house\_text, \$apartment\_text у лістингу 2.5 — дані про адресу отримувача, введені ним у формі.

Новоствореній адресі присвоюється персональний id формату:

0009483993844012

На основі отриманої id-адреси генерується наступний запит до API Укрпошти на реєстрацію отримувача методом /clients. За допомогою технології cURL, аналогічно до попереднього, відбувається запит до API Укрпошти, показаний у лістингу 2.6.

## Лістинг 2.6 - Приклад коду для запиту до API Укрпошти методом /clients

```
curl -X POST "https://dev.ukrposhta.ua/ecom/0.0.1/clients?token=4e64b7f0-434d-45bb-addd-835c57202f54" \
  -H "accept: */*" \
  -H "Authorization: Bearer ec60df35-7be5-3ae1-b283-1bb4931423ec" \
  -H "Content-Type: application/json" \
  -d "firstName":"Богдан", "lastName":"Тригубець", "addressId":47505588, "phoneNumber":380681050808, "type":"COMPANY"}
```

де \$firstName, \$lastName та \$phone у лістингу 2.6 — це особисті дані отримувача,

\$api\_addressId у лістингу 2.6 — згенерована методом /addresses id-адреса отримувача.

Результатом запиту `/clients` є отримання `uuid`, тобто `id` відправника. Після отримання `uuid` в наявності є усі необхідні дані для генерації накладної відправлення. Для її генерації використовується метод `/shipments`, приклад запиту зображено у лістингу 2.7.

Лістинг 2.7 - Приклад коду для запиту до API Укрпошти методом `/shipments` для генерації накладної відправлення

```
curl -X POST "https://dev.ukrposhta.ua/ecom/0.0.1/shipments?
token=4e64b7f0-434d-45bb-addd-835c57202f54" \
-H "accept: */*" \
-H "Authorization: Bearer ec60df35-7be5-3aef-
b283-1bb4931423ec" \
-H "Content-Type: application/json" \
-d '{"sender":{"uuid":"91d22fa5-0d1d-4712-bcf4-
b01f8ac1f865"},"recipient":
{"uuid":"aecb1a92-3b43-40ec-95fd-59a5f4a1396e"},"deliveryType":
"W2W","parcels":[{"weight":400,"declaredPrice":300,
"length":50}]}'
```

де `$sender_uuid` та `$api_uuid` у лістингу 2.7 — унікальні ідентифікатори, отримані в результаті роботи методів `/addresses` та `/clients`,

а `$weight`, `$length` у лістингу 2.7 - це вага та найбільша сторона посилки, утворена шляхом сумування ваги кожного товару (які попередньо прораховані та додані у відповідне поле у базі даних), та розмір найбільшого товару, який буде використовуватися для модуля автоматичного визначення необхідної для відправлення коробки, що також спрощує обробку замовлення.

Результатом роботи методу `/shipments` є унікальне значення `$barcode`, у якому знаходиться номер накладної. Номер накладної є результатом аналогічного заповнення з працівником відділення усіх необхідних для реєстрації відправлення даних вручну, що зазвичай займає досить багато часу, а у випадку роботи скрипта відбувається майже миттєво (при умові перевірки правильності введених користувачем даних).

Після генерування накладної, останнім кроком у цьому ланцюжку є генерації PDF-файлу для друку та розміщення його на коробці відправлення, що спростить обробку нового відправлення у відділенні Укрпошти.



Лістинг 2.8 - Приклад коду для запиту до API Укрпошти для отримання PDF-файлу накладної відправлення.

```
curl -X GET 'https://www.ukrposhta.ua/forms/ecom/0.0.1/
shipments/0500180935765/sticker?token=4e64b7f1-434d-45bb-
addd-835c57202f54&size=SIZE_A4' \
--header 'Content-Type: application/json' \
--header 'Authorization: Bearer ec60df35-7be5-3aef-
b283-1bb4931423ec' \
--output sticker.pdf
```

Результатом роботи методу shipments/sticker є готовий до друку PDF-файл, який друкується та приклеюється на коробці відправлення.

Останнім методом, який використовує Н83 CMS у свої роботі для обміну даними з API Укрпошти, є метод /statuses, який використовується для відстежування поточного статусу замовлення прямо на сайті інтернет-магазину. Приклад коду запиту зображено у лістингу 2.9.

Лістинг 2.9 - Приклад коду для запиту

```
curl -X GET 'https://www.ukrposhta.ua/status-tracking/0.0.1/
statuses?barcode=05
00053912667&lang=en' \
--header 'Content-Type: application/json' \
--header 'Authorization: Bearer
958a49fa-77aa-4888-81ab-47f536256a59'
```

В результаті цього запиту сервер API Укрпошти повертає масив даних, який містить:

\$value->eventName - текстову назву статусу відправки;

\$value->date - дату створення запису про новий статус;

\$value->name - назва місця оновлення статусу відправлення.

Використання інтеграції з поштовим сервісом Укрпошта за допомогою наданого ним API дозволяє спростити взаємодію користувача з модулем доставки, запобігти введенню помилкових даних про адресу доставки, або даних, чия структура заважає швидкій обробці замовлення, дозволяє користувачу відслідковувати статус свого замовлення та допомагає менеджменту інтернет-магазину швидше виконувати свою роботу.

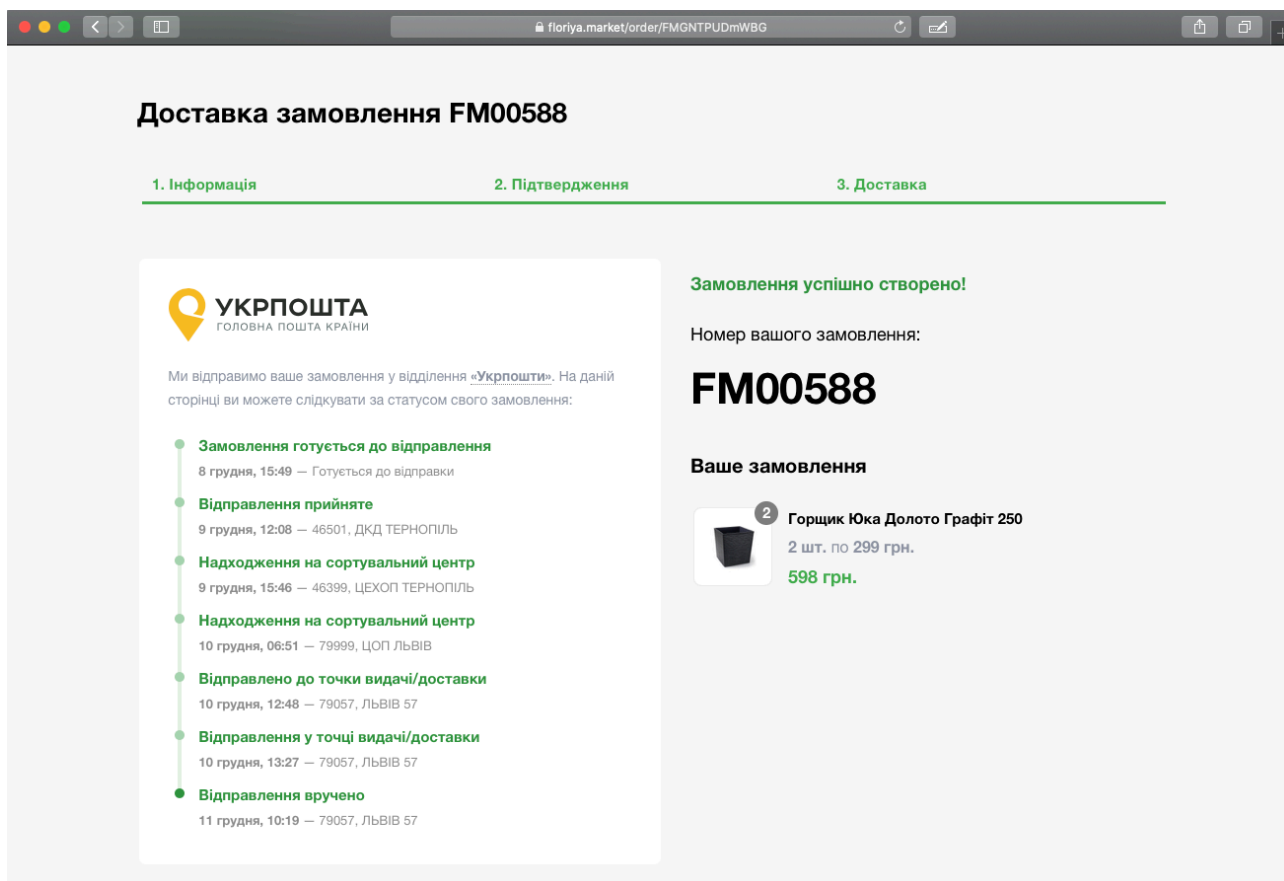


Рисунок 2.3 Вигляд сторінки відслідковування зміни статусу замовлення

## 2.2.4 Авторизація через API соцмережі Facebook

Для створення особистого кабінету та можливості користувачем зберігати особисту інформацію про доставку, зроблені замовлення та збережені товари під єдиним профілем, було обрано соціальну мережу Facebook. Facebook — це найбільша у світі соціальна мережа, яка також є найпопулярнішою соціальною мережею України.

Використання авторизації через соціальну мережу Facebook дозволяє спростити процес реєстрації на web-сайті, та подальші процеси входу користувача. Вхід через API Facebook відбувається наступним чином: замість введення логіну та паролю, електронної пошти та підтвердження реєстрації шляхом переходу по посиланню на електронній скринці, користувачу вистачає підтвердити отримання web-сайтом базових даних про користувача (імені та прізвища користувача, а також URL-адреси головної фотографії його профілю)

від Facebook, що значно пришвидшує як процес реєстрації, так і довіру користувача до нового web-сайту.

### 2.2.5 Інтеграція з платіжною системою LiqPay

Для оплати замовлень у інтернет-магазинах, працюючих на основі розробленої H83 CMS, було обрано українську платіжну систему LiqPay, яка є дочірньою структурою найбільшого українського банку ПриватБанк. За допомогою цього сервісу клієнт має змогу оплатити своє замовлення банківською картою Visa або MasterCard, через Приват24, готівкою через термінали самообслуговування ПриватБанку, за допомогою електронних сервісів оплати MasterPass, Visa Checkout, а також через Apple Pay. Це дозволяє надати користувачу максимально широкий вибір варіантів, для того щоб він зміг обрати найбільш комфортний для себе.

Першим етапом інтеграції є створення кабінету компанії для сайту інтернет-магазину, прив'язка реквізитів ФОП, на які будуть відправлятися платежі за оплачені товари, та відправка web-сайту на перевірку на відповідність до умов роботи платіжної системи.

Оплата замовлення з використанням API платіжної системи LiqPay відбувається наступним чином: використовуючи PHP-бібліотеку, надану LiqPay, генерується посилання на оплату. Для виклику API LiqPay необхідно передати значення параметру data і signature (Server - Server) POST методом [23], приклад коду якого показано у лістингу 2.10.

#### Лістинг 2.10 - Приклад запиту до API LiqPay

```
$array = array('version' => '3', 'public_key' =>
$public_key, 'action' => 'pay', 'amount' => $sum, 'currency' =>
'UAH', 'description' => "Оплата замовлення:\n ".$description,
'order_id' => $type, 'language' => "uk", 'server_url' => "https://
floriya.market/paystatus.php", 'result_url' => "https://
floriya.market/order/".$hpid_encrypt);
$liqpay = new LiqPay($public_key, $private_key);
$data = base64_encode(json_encode($array));
$signature = $liqpay->cnb_signature($array);
$liqpay_url = addslashes("https://www.liqpay.ua/api/3/
checkout?data=".$data."&signature=".$signature."");
```

де \$data — це json рядок з параметрами APIs, які закодовані функцією base64, base64\_encode(json\_string),

signature — унікальний підпис запиту base64\_encode(sha1(private\_key+data+private\_key)),

base64\_encode — значення рядку, закодованого методом base64,

sha1 — значення хешу у вигляді бінарного рядку довжиною 20 символів,

sum — вартість замовлення, яке формується додаванням усіх товарів, доданих користувачем у корзину, та вартості доставки.

Далі користувач переходить по згенерованому посиланні на сторінку оплати платіжної системи LiqPay, на якій бачить перелік товарів та суму, яку необхідно оплатити. Для оплати користувачу пропонуються варіанти методів оплати, які адміністратор може самостійно налаштувати в особистому кабінеті магазину LiqPay.

При зміні статусу оплати сервер платіжної системи надсилає оновлені дані з id статусу оплати на адресу server\_url. Алгоритмом, який знаходиться за адресою, вказаною у server\_url, з використанням PHP-бібліотеки LiqPay, проводиться перевірка отриманої сигнатури за допомогою приватного ключа на відповідність сигнатурі, яка паралельно генерується на сервері LiqPay. Після підтвердження відповідності сигнатур, та наявності нового статусу, оновлені дані зберігаються у базі даних.

Після вибору методу оплати та підтвердження платіжних даних користувач перенаправляється на сторінку result\_url. Відповідно до статусу оплати, на сторінці замовлення користувачу відображаються поточний етап його замовлення. Якщо оплата пройшла успішно, відобрадається останній етап, на якому користувач може відслідкувати статус пакування та відправки його замовлення. Після генерації накладної відправлення, на сторінці через API Укрпошти відображаються етапи доставки. Відповідно адреса result\_url, на яку користувач перенаправляється після оплати, та яку отримує у листі на свою електронну скриньку, є місцем, де він може у будь який час отримати найсвіжішу інформацію про будь-які зміни в статусі свого замовлення.

## 2.2.6 СМС та Email-сповіщення

Для інформування клієнтів про створення/оплату/відправку замовлення було обрано класичний метод надсилання email-листів на електронну скриньку, вказану клієнтом при замовленні. Цей метод є популярним у сфері інтернет-комерції, тому дозволить користувачу відчувати себе в звичному середовищі при роботі з сервісом. Для відправлення email-сповіщень і аналізу їх ефективності було обрано популярний сервіс SendGrid. Він дозволяє автоматизувати процес відправлення листів через інтеграцію сайту з сервісом, використовуючи спеціальне SendGrid API, приклад запиту зображено у лістингу 2.11.

### Лістинг 2.11 - Приклад коду для відправки електронного листа через API SendGrid

```
https://sendgrid.com/docs/API_Reference/Web_API_v3/Mail/index.html
$data = '{"personalizations": [{"to": [{"email": "'.
$email.'"}]}], "from": {"email": "mail@floriya.market"}, "subject":
"'.$subject.'", "content": [{"type": "text/html", "value": "'.
$text.'"}]}';
$ch = curl_init('https://api.sendgrid.com/v3/mail/send');
$options = array(
    CURLOPT_RETURNTRANSFER => true,
    CURLOPT_HEADER          => false,
    CURLOPT_FOLLOWLOCATION   => false,
    CURLOPT_POST            => 1,
    CURLOPT_POSTFIELDS      => $data,
    CURLOPT_SSL_VERIFYHOST  => 0,
    CURLOPT_SSL_VERIFYPEER => false,
    CURLOPT_VERBOSE         => 1,
    CURLOPT_HTTPHEADER      => array(
        "Authorization: Bearer
SG.dqQDMA59T_y4MBC_zRb1cg.8JzZ5Rb7Fr69RLaki2bI8ST7bH5ualmuwH1uKgjm
PbI",
        "Content-Type: application/json"
    )
);
curl_setopt_array($ch, $options);
$data = curl_exec($ch);
$curl_errno = curl_errno($ch);
$curl_error = curl_error($ch);
curl_close($ch);
```

В доповнення до звичайних Email-сповіщень, для того, щоб інформувати клієнтів про важливі зміни статусу обробки їх замовлень, було вибрано сповіщення через СМС-повідомлення. Такий тип мобільних сповіщень зазвичай не губиться серед потоку інших сповіщень, та виглядає авторитетно перед користувачами, так як найчастіше цей спосіб підтвердження та сповіщення використовують банківські сайти та мобільні додатки. Серед усіх сервісів, які надають такі послуги, було обрано SMSC.ua, який є одним з найпопулярніших постачальників даних послуг на території України, та надає можливість автоматичної відправки СМС-повідомлень використовуючи власне SMSC.ua API, аналогічно до сервісу email-розсилок SendGrid.

### 2.2.7 Push-сповіщення через API OneSignal

Щоб збільшити залученість клієнтів та підтримки з ними постійного контакту було інтегровано браузерні push-повідомлення, відправка яких здійснюється через API сервісу OneSignal, приклад надсилання push-повідомлення зображено у лістингу 2.12.

Лістинг 2.12 - Приклад відправки push-повідомлення через API OneSignal

```
$fields = array(
    'app_id' => "6d484312-ff9c-4b37-a1d4-22a55b503dce",
    'included_segments' => array('All'),
    'data' => array("foo" => "bar"),
    'url' => $_POST["url"],
    'contents' => $content
);

$fields = json_encode($fields);
print("\n");
print($fields);

$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, "https://onesignal.com/
api/v1/notifications");
curl_setopt($ch, CURLOPT_HTTPHEADER, array('Content-
Type: application/json; charset=utf-8',

'Authorization:                                     Basic
OGFiNGJkyjUtNGRhMC0NGYwLWE1YmYtYzAwZDdiYTFjYTJh'));
```

```

curl_setopt($ch, CURLOPT_RETURNTRANSFER, TRUE);
curl_setopt($ch, CURLOPT_HEADER, FALSE);
curl_setopt($ch, CURLOPT_POST, TRUE);
curl_setopt($ch, CURLOPT_POSTFIELDS, $fields);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, FALSE);

$response = curl_exec($ch);
curl_close($ch);

```

## 2.2.8 Інтеграція з API Telegram

Для сповіщення менеджерів та власників проектів про надходження нових замовлень з інтернет-магазину було обрано платформу месенджеру Telegram. Telegram — захищений месенджер, який використовує для роботи криптографічний протокол власної розробки MTProto [24]. В основі цього протоколу лежить розроблена комбінація симетричного алгоритму шифрування AES (в режимі IGE) та протокол Діффі-Хеллмана для обміну 2048-бітними RSA-ключами між клієнтом та сервером, а також ряд інших власних хеш-функцій. Протокол дозволяє використання end-to-end шифрування з опціональною можливістю зв'язки ключів. Месенджер має клієнти під більшість популярних мобільних та веб-платформ, тому є зручним у використанні. Приоритетним рішенням при виборі платформи є наявність на ній інструментів для створення телеграм-ботів, роботу яких можна налаштувати під власні потреби. Взаємодія з ними відбувається через API Telegram. Для підтримки постійного з'єднання та обміну новими подіями між web-серверами використовується технологія Webhook [25]. Встановлення з'єднання відбувається за прикладом, зображеним у лістингу 2.13.

Лістинг 2.13 - Приклад коду для встановлення webhook-з'єднання з API Telegram

```

curl https://api.telegram.org/bot<token>/setWebhook \
-F "url=<yoururl>"

```

Webhook у web-розробці — це метод збільшення функціональної поведінки веб-сторінки за допомогою зворотніх користувацьких викликів

(callbacks). Ці зворотні користувацькі виклики можуть обслуговуватись сторонніми користувачами, розробниками чи web-сервісами, надаючи відповіді на отримані запити.

Для обміну даними між сервісом та користувачем зазвичай використовується формат JSON, а запити виконуються як HTTP POST-запити. Кінцева точка, яка приймає запити, може обмежувати список IP-адрес, які можуть надсилати запити, створюючи умовний “білий список” джерел. Інформація отримана через Webhook може містити інформацію про тип події, її значення, а також зазвичай містить секретний ключ або підпис для підтвердження джерела. Telegram встановлює з'єднання, використовуючи сертифікат безпеки TLS, який підтверджує достовірність сервера-відправника.

В результаті створення Webhook-з'єднання сервер CMS буде миттєво сповіщений про будь-яку взаємодію користувачів з ботом (наприклад, підтвердження виконання замовлення натисканням спеціальної кнопки під повідомленням у месенджері, після чого зміна статусу замовлення буде відображена у базі даних на сервері).

Зворотні запити на публікацію нового замовлення здійснюються звичайними запитами до API Telegram, приклад яких зображено у лістингу 2.14.

Лістинг 2.14 - Приклад коду запиту до API Telegram для публікації повідомлення в чат

```
curl -X POST \
-H 'Content-Type: application/json' \
-d '{"chat_id": "123456789", "text": "This is a test from
curl", "disable_notification": true}' \
https://api.telegram.org/bot$TELEGRAM_BOT_TOKEN/sendMessage
```

де \$chatId у лістингу 2.14 — це id створеного каналу, у якому публікуються сповіщення.

Для роботи модуля сповіщень було створено спеціальний закритий Telegram-канал, у який було додано усіх працівників, які повинні отримувати інформацію, пов'язану з новими замовленнями. Для публікації нових повідомлень у цей канал було створено Telegram-bot, який також було додано у



Telegram-канал. Саме він публікує усі нові сповіщення, згенеровані через API Telegram.

## 2.2 Захист серверу

### 2.2.1 Використання сервісу Cloudflare

Cloudflare — американська компанія, що надає мережеві послуги для роботи з web-сайтами. Сервіс надає послуги для вирішення багатьох питань, пов'язаних з перевіркою, аналізом трафіку та доставкою контенту [8].

Система фільтрів захисту від DDoS-атак на Cloudflare побудована за принципом перенаправлення трафіку через власну систему CDN (Content Delivery Network). CDN — це географічно розподілена мережева інфраструктура, що дозволяє оптимізувати доставку і дистрибуцію контенту кінцевим користувачам [10].

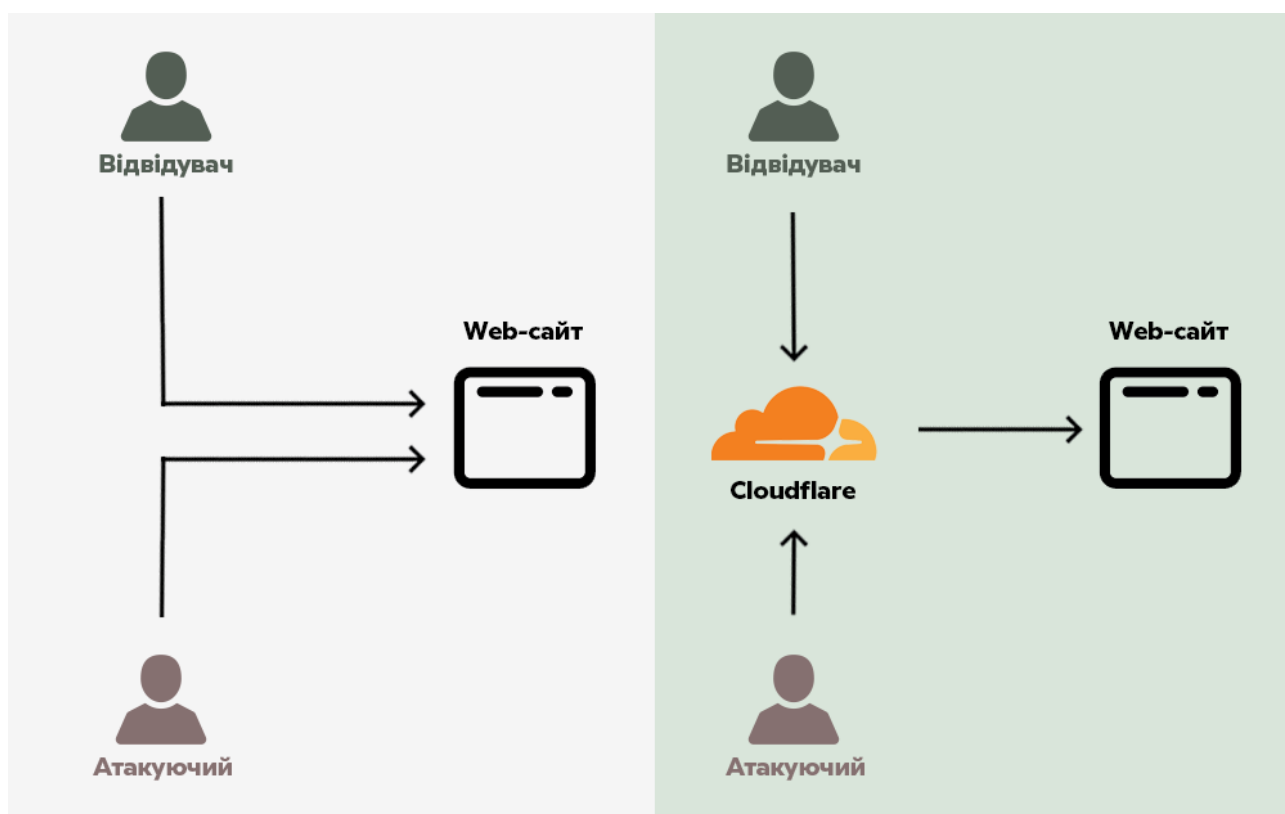


Рисунок 2.4 Загальний принцип доступу до web-сайту без та з використанням Cloudflare

Використовуючи Cloudflare як проміжний етап при запиті користувача до вашого сайту, ви дозволяєте сервісу аналізувати ваш трафік на предмет аномальної активності. Великий досвід компанії в аналізі DDoS-атак дозволяє їй з легкістю розрізняти запити користувачів та запити з IP-адрес “ботнетів”.

За інформацією сервісу, він має можливість пом'якшити розширені DDoS-атаки 7-го рівня, які вважаються найскладнішими, надаючи обчислювальні послуги JavaScript, які необхідно завершити браузером користувача, перш ніж користувач зможе отримати доступ до web-сайту. Також сервіс стверджує, що захистив компанію SpamHaus від DDoS-атаки, потужність якої що перевищувала 300 Гбіт/с, що ще раз підтверджує раціональність його використання та високий рівень захисту в цьому напрямку [8].

### 2.2.2 Використання SSH-ключа замість паролю

Дебати між раціональністю використання SSH-ключа замість паролю тривають десятиріччями [2].

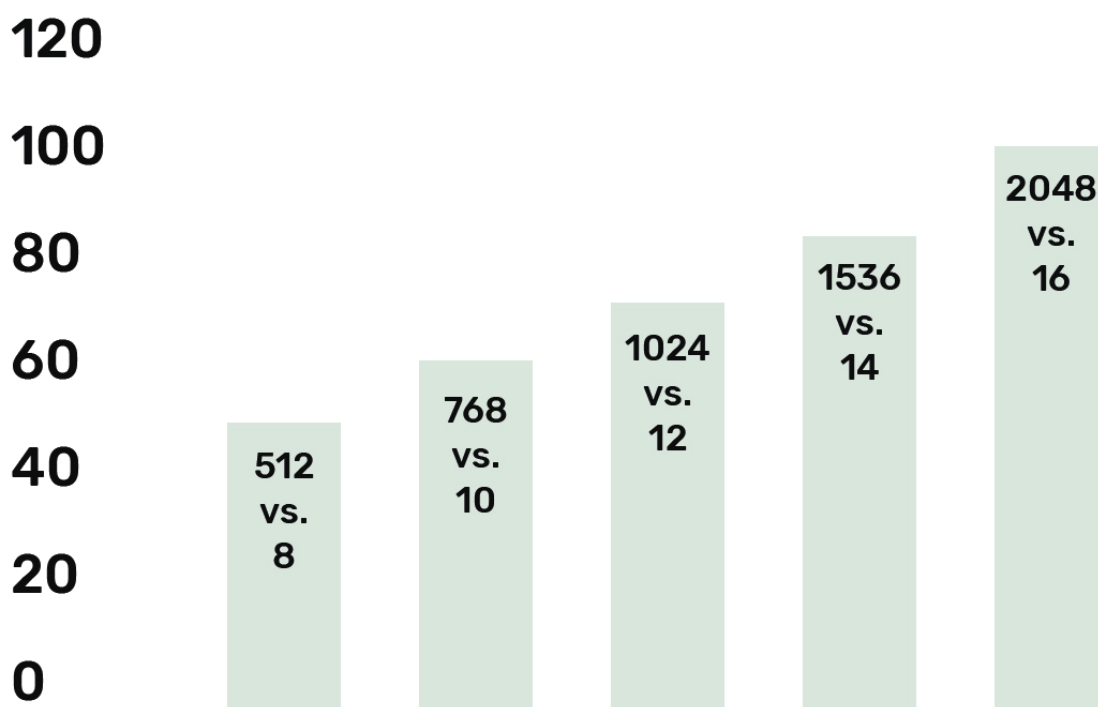


Рисунок 2.5 Оцінка надійності паролю та SSH-ключа через ентропію [6]

Оцінюючи рівень безпеки SSH-ключа та паролю через ентропію можна визначити, що формально перевага першого варіанту досить не суттєва. Проте є декілька нюансів. Оцінка є правильною при умові абсолютно випадкового набору символів у паролі, що зустрічається дуже рідко. Відомо [9], що більшість користувачів використовують у своїх паролях загальновідомі слова, додають до них цифри першого десятку, або пробують імітувати випадковий набір символів, використовуючи клавіші на клавіатурі, що знаходяться поряд. На практиці усе це значно спрощує підбір паролю зломисником. Результати порівняння зображено у таблиці 2.1.

Таблиця 2.1 - Схема оцінки вразливостей конфігурацій використання портів та паролів/SSH-ключів

Конфігурація	Ймовірність взлому*	Втрати від флуду**
22 порт, авторизація за допомогою паролю, без обмеження кількості спроб	висока	високі
22 порт, авторизація по ключам, без обмеження кількості спроб	середня***	високі
22 порт, авторизація по ключам, обмеження кількості невдалих спроб	низька	середні****
Нестандартний порт, авторизація за допомогою паролю, без обмеження кількості спроб	висока	низькі
Нестандартний порт, авторизація по ключам, без обмеження кількості спроб	середня***	низькі
Нестандартний порт, авторизація по ключам, обмеження кількості невдалих спроб	низька	низькі

Перевага використання SSH-ключа в тому, що це абсолютно випадковий набір символів, який не передається по каналах зв'язку. В цьому випадку його компрометація можлива лише шляхом повного перебору, тоді як пароль може бути перехоплений та розшифрований [3].

Також додатковим варіантом є захист з використанням нетипових портів для протоколу SSH, що описано у таблиці 2.1.

\* значення параметрів (висока, середня, низька) носять відносний характер і використовуються лише для порівняння показників.

\*\* береться до уваги витрата ресурсів сервера (процесор, диск, мережевий канал і т.п.) на обробку потоку запитів, які зазвичай йдуть на стандартний 22-й порт.

\*\*\* здійснити успішну атаку, якщо для авторизації використовуються RSA-ключі, дуже складно, проте необмежену кількість спроб авторизації робить це можливим.

\*\*\*\* кількість спроб авторизації обмежена, проте серверу все одно доводиться обробляти їх у великій кількості.

### 2.2.3 Використання протоколу HTTP/3

HTTP/3 і QUIC — це нові стандарти, які обіцяють усунути багато недоліків попередніх версій та запровадити безпечні та швидкі з'єднання у Інтернеті. QUIC (Quick UDP Internet Connections) — транспортний мережевий протокол, який було створено як альтернативу зв'язці TCP + TLS для web-сайтів у 2013 році компанією Google. HTTP/3 — це нова версія протоколу передачі гіпертексту HTTP для обміну бінарною інформацією у Інтернеті.

За інформацією розробників [5], використання протоколу дозволяє вирішити такі проблеми, наявні у минулих версіях:

- контроль за цілісністю потоку, щоб запобігти втрати пакетів;
- поділ на потоки, при якому втрата пакета впливає лише на свій потік;
- збільшення продуктивності і пропускну здатності, в порівнянні з TCP.

Застосування QUIC на відеосервісі YouTube показало скорочення операцій повторної буферизації при перегляді відео на 30% [7];

- вирішення проблеми з блокуванням черги TCP;
- підтримка ідентифікатора з'єднання, який дозволить скоротити час на встановлення повторного з'єднання для мобільних пристроїв;
- можливість використання розширених механізмів контролю за перевантаженням з'єднання;

Принципову різницю у роботі старої і нової версії можна побачити на Рисунок 2.6 та рис 2.7.

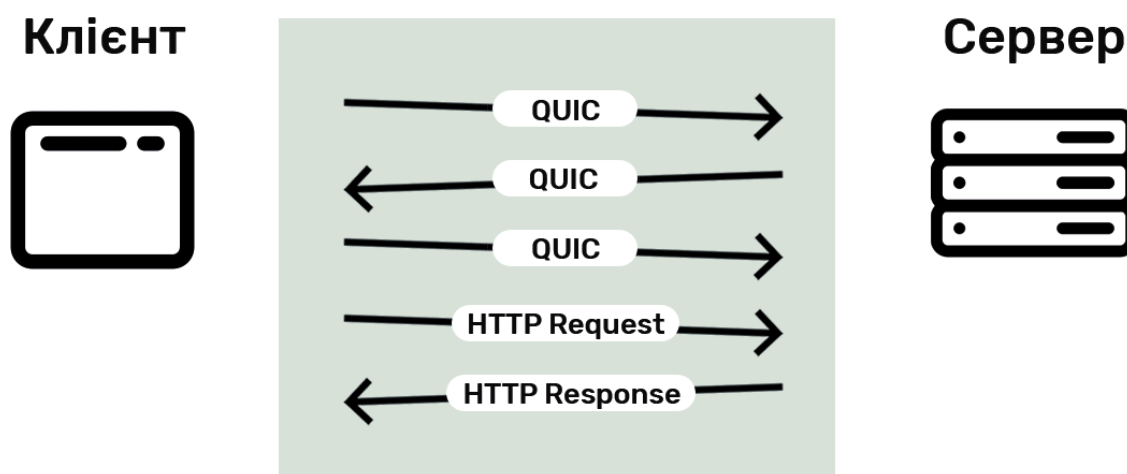


Рисунок 2.6 Схема роботи HTTP - запиту QUIC

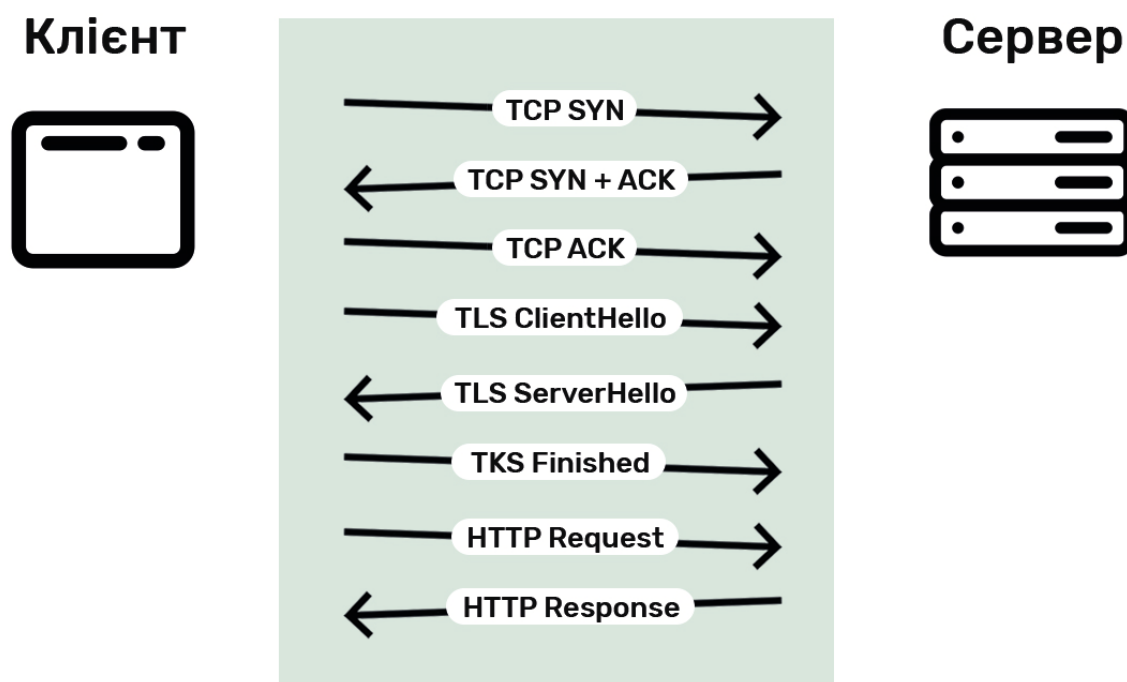


Рисунок 2.7 Схема роботи HTTP - запиту TCP + TLC

## 2.2.4 Безпечне з'єднання через криптографічні протоколи TLS/SSL

Протокол HTTPS — це модифікація протоколу HTTP з використанням шифрування. Єдина відмінність між цими протоколами полягає в тому, що HTTPS використовує TLS (SSL) для шифрування тих самих HTTP-запитів та відповідей. Як результат, HTTPS набагато безпечніший ніж HTTP.

HTTP-запити генеруються браузером користувача під час його взаємодії з web-ресурсами. Наприклад, коли користувач натискає на гіперпосилання, браузер надсилає серію запитів HTTP GET для отримання вмісту, який після виконання запиту відобразиться на цій сторінці. Тобто, під час натискання на посилання браузер створює та надсилає ряд HTTP-запитів для отримати інформації, необхідної для візуалізації сторінки. Приклад HTTP GET-запиту показано у лістингу 2.15.

### Лістинг 2.15 - Приклад коду для запиту

```
GET /index.php HTTP/1.1
User-Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 13_2_3 like
Mac OS X) AppleWebKit/605.1.15 (KHTML) Version/13.0.3 Mobile/
15E148 Safari/604.1
Host: www.h83cms.website
Accept-Language: uk
```

Цей блок тексту, згенерований браузером користувача, надсилається ним через мережу Інтернет. Проблема полягає в тому, що надсилається він саме у такому вигляді, як на прикладі вище. Тобто у форматі відкритого тексту, який у читабельному вигляді може перехопити та прочитати кожен, хто контролює з'єднання.

Якщо це звичайний блог, який містить лише текстову інформацію, перехоплюючи з'єднання злоумисник не отримає ніякої додаткової інформації в порівнянні з тією, яку він може отримати завантаживши цю сторінку в своєму браузері самостійно. Проте особливо гостро це питання постає тоді, коли користувач надсилає якісь конфіденційні дані через форми на web-сайті або у web-додатку. Це може бути як логін і пароль, номер кредитної картки, так і будь-які інші дані, які вводяться користувачем у форму, і через HTTP-запит

надсилаються у відкритому вигляді. Коли сервер отримує цей HTTP-запит, він надсилає HTTP-відповідь, вигляд якої можна побачити у лістингу 2.16.

### Лістинг 2.16 - Приклад коду HTTP-запиту

```
HTTP/1.1 200 OK
Date: Wed, 4 Dec 2019 12:34:56 GMT
Server: Apache
Last-Modified: Mon, 2 Dec 2019 11:11:11 GMT
Accept-Ranges: bytes
Content-Length: 12
Vary: Accept-Encoding
Content-Type: text/html

Hello World!
```

Тобто якщо веб-сайт використовує HTTP замість HTTPS, усі його запити та відповіді може прочитати кожен, хто приєднається до з'єднання. Зловмисник може з легкістю прочитати тексти запитів та відповідей і точно знати, яка інформація надсилається чи отримується.

В цьому випадку потрібен HTTPS — безпечна версія протоколу HTTP. HTTPS використовує TLS (або SSL) для шифрування HTTP-запитів та HTTP-відповідей, тому перехопивши з'єднання, замість читабельного тексту зловмисник побачить набір випадкових символів, які є зашифрованою версією сторінки. Відповідно замість звичайного зрозумілого тексту, який мав би надіслати веб-браузер, який зображено у лістингу 2.17, при перехопленні даних зловмисник побачить зашифрований код, який зображено у лістингу 2.18.

### Лістинг 2.17 - Приклад коду HTTP-відповіді

```
GET /index.php HTTP/1.1
User-Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 13_2_3 like
Mac OS X) AppleWebKit/605.1.15 (KHTML) Version/13.0.3 Mobile/
15E148 Safari/604.1
Host: www.h83cms.website
Accept-Language: uk
```

### Лістинг 2.18 - Приклад коду HTTPS-відповіді

```
t8Fw6T8UV81pQfyhDkhebbz7+oiwldrlj2gHBB3L3RFTRsQCpaSnSBZ78Vme
+DpDVJPvZdZUZHpzbbcqmSW1+3xXGseRHg9YDmpYk0VVDiRvw1h5miNieJeJ/
FNUjgH0BmVRWII6+T4MnDwmCMZUI/orxP3HGwYCSiVyzS3MpmSe4iaWKCONQ==
```

TLS використовує технологію, яка називається “шифруванням відкритого ключа”: є два ключі, відкритий та приватний. Відкритий ключ надається спільно з клієнтськими пристроями через SSL-сертифікат сервера. Коли клієнт відкриває з'єднання з сервером, ці два пристрої використовують обмін відкритим та приватним ключами для узгодження нових ключів, званих сеансовими ключами, та для шифрування подальших комунікацій між ними.

Усі запити та відповіді HTTP шифруються ключами сеансу (це відбувається на 6 рівні в Мережевій моделі OSI), тому зломисник, який перехоплює мережеві повідомлення, зможе побачити лише випадковий набір символів, а не читабельний текст, як показано у лістингу 2.18.

Ще одним використанням HTTPS є підтвердження автентичності. Це означає що людина чи машина — саме ті, за кого себе видають. У протоколі HTTP немає перевірки ідентичності — вона була заснована на принципі довіри. У першій версії протоколу HTTP його архітектори не ставили під питання довіру web-серверам, адже у ті часи були інші пріоритети безпеки. Але в сучасному Інтернеті наявність автентифікації є невід'ємним компонентом для створення безпечного з'єднання. Так само як особистий документ підтверджує особу людини, приватний ключ підтверджує особу сервера. Коли клієнт відкриває канал із початковим сервером (наприклад, коли користувач переходить на web-сайт), приватний ключ, що збігається із відкритим ключем у сертифікаті SSL web-сайту, доводить, що сервер насправді є законним хостом цього web-сайту.

### 2.2.5 Використання Kernel-based Virtual Machine (KVM)

Використання звичайного хостингу для розміщення web-сайту несе за собою наявність вразливостей через використання однієї операційної системи



для декількох клієнтів хостинг-серверу. Якщо зломисники отримують доступ до одного з сайтів цим методом — під атаку потрапляють усі інші.

При використанні віртуальної машини на одному апаратному забезпеченні запускається декілька незалежних операційних систем, що дозволяє обмежити web-сайт від стороннього впливу.

Одне з найкращих програмних рішень для цих цілей — Kernel-based Virtual Machine (KVM). KVM використовує комбінацію посиленої безпеки Linux (SELinux) та безпечної віртуалізації (sVirt) для посилення безпеки та ізоляції віртуальної машини [15].

### 2.2.6 Правильне налаштування .htaccess та .htpasswd

Файл .htaccess — це конфігураційний файл веб-сервера Apache, який дозволяє керувати роботою web-сервера та налаштовувати роботу web-сайту за допомогою спеціальних параметрів без внесення змін в основний файл конфігурації web-сервера.

Директиви кожного .htaccess-файлу діють лише для каталогу, в якому цей файл був розміщений, а також для всіх його підкаталогів. Для того щоб за допомогою .htaccess-файлу налаштувати директиви для web-сайту в цілому, необхідно розмістити його в кореновому каталозі web-сайту. Розглянемо приклади безпечного налаштування .htaccess-файлу.

Можливість доступ до web-сайту лише з певних IP-адрес. Для прикладу лише 123.4.5.6 и 123.5.4.3, що зображено у лістингу 2.19.

#### Лістинг 2.19 - .htaccess-обмеження доступу

```
Order Allow,Deny
Allow from all
Deny from 123.4.5.6 123.5.4.3
```

Наступним налаштуванням безпеки є заборона лістингу каталогу. В стандартних налаштуваннях web-серверів на Apache у разі відсутності у папці індексного файлу (тобто головної сторінки), при зверненні користувача до директорії без вказування запиту до конкретного файлу в результаті виконання

цього запиту буде виданий повний список файлів, які знаходяться у даному каталозі.

Таке налаштування є небезпечним, адже користуючись цією вразливістю злоумисник може отримати детальну інформацію про структуру каталогів, у який можуть міститися важливі для функціонування сайту файли, або ж, як найчастіше трапляється, отримати доступ до серверного каталогу, у якому відбувається збереження завантажуваних користувачами персональних файлів, у яких може міститися конфіденційна інформація. Для того щоб заборонити відображення лістингу каталогу, достатньо додати в файл `.htaccess` лише один рядок: `Options -Indexes`.

Наступним кроком у правильному налаштуванні файлу `.htaccess` є керування відображенням PHP-помилки. Вивід помилок це безумовно корисна річ, яку можна ефективно використовувати при налагодженні програми. Проте після завершення розробки та налагодження директиви `error_reporting` і `display_errors` варто відключати, адже злоумисник може дізнатися з них багато інформації про структуру роботи скрипта. Для того щоб вимкнути відображення помилок використовуються директиви, описані у лістингу 2.20.

#### Лістинг 2.20 - `.htaccess`-обмеження доступу

```
php_flag display_errors off
php_value error_reporting 0
```

Для технічних директорій, доступ до яких не потрібний користувачу, можна використовувати обмеження доступу до директорії. Для цього необхідно створити у цільовій директорії файл `.htaccess`, з кодом, який зображено у лістингу 2.21.

#### Лістинг 2.21 - `.htaccess`-обмеження доступу до технічних директорій

```
Order allow,deny
Deny from all
```

Заборона завантаження файлів, розміщених на вашому web-сервері, на зовнішніх сайтах (для прикладу, зображень чи документів) допоможе

заощадити трафік web-серверу та запобігти непотрібному навантаженні на нього, як зображено у лістингу 2.22.

#### Лістинг 2.22 - .htaccess-заборона завантаження файлів

```
RewriteEngine On
RewriteCond %{HTTP_REFERER} !^$
RewriteCond  %{HTTP_REFERER}  !^http://(www.)?h83cms.website/
[nc]
RewriteRule  *.*.(gif|jpg|png)$ https://h83cms.website/img/
offimg.gif[nc]
```

Також за допомогою файлу можна налаштувати базову HTTP-аутентифікацію, як зображено у лістингу 2.23.

#### Лістинг 2.23 - HTTP-аутентифікація через .htaccess

```
AuthName 'Stuff only'
AuthType Basic
AuthUserFile '/home/users/catalog/.htpasswd'
Require user admin
```

де файл .htpasswd - це файл який містить імена користувачів та хеш їхніх паролів (зазвичай викривується хешування MD5).

Після налаштування файлів при спробі входу в захищений каталог користувач побачить поле входу, у якому для доступу необхідно ввести логін і пароль, перевірка правильності яких буде відбуватися у файлі .htpasswd.

## 2.3 Захист програмної частини

### 2.3.1 Безпечне програмування та перевірка вхідних даних

Основа безпечного програмування — це розробка програмного забезпечення, що запобігає випадковому впровадженню вразливостей і забезпечує стійкість до впливу шкідливих програм та несанкціонованому доступу. Адже зазвичай різноманітні баги і логічні помилки є основною причиною появи вразливостей програмного забезпечення.

Захист від таких вразливостей, як XSS-атаки та SQL-ін'єкції, є досить простим у реалізації. Для безпечної роботи коду просто необхідно проводити фільтрацію даних, що надходять на сайт. Зазвичай PHP-розробники використовують для цього функцію `htmlspecialchars()` [13]. Ця функція перетворює усі теги HTML, а відповідно і фрагменти JS-коду, в їх символічні еквіваленти у стандарті ISO-8859-1 [14], що дозволяє знешкодити будь-які скрипти, та запобігти їх неправомірному використанні на вашому web-сайті.

Розглянемо приклад перетворення тегів на спеціальні символи: еквівалентом амперсанду `&` є `"&amp;";`, `<` замінюється на `"<lt;";`, `>` на `">gt;";`. На практиці, примінивши функцію `htmlspecialchars()` на фрагмент коду, введений користувачем в формі, наприклад код `<a href='url'>Hacked text</a>` буде замінений на код `&lt;a href='url'&gt;Hacked text&lt;/a&gt;`;

На практиці зустрічаються досить витончені XSS-атаки, у яких використовується різні кодування символів, які не завжди підпадають під стандартну фільтрацію. Для роботи з такими запитами було створено спеціальний PHP-клас — `AntiXSS` [37], який допомагає фільтрувати велику кількість інтерпретацій символів, які утворюють відкриваючі та закриваючі теги у скриптах зловмисників.

Відповідно, посилання на web-сторінку або скрипт, який зловмисники розмістили для збереження у базі та подальшого виводу на сайті, перетвориться з активного посилання на звичайний набір символів, які вже ніяк не будуть загрожувати коректній роботі web-сайту.

Для захисту від SQL-ін'єкцій та інших вразливостей, які можуть виникнути у випадку неправильного формулювання SQL-запитів, було обрано PHP-клас `safemysql.class.php` [22]. Використання цього додаткового класу виконує як функцію безпеки, так і функцію спрощення взаємодії з базою даних, адже синтаксис класу дозволяє оптимізувати кількість стрічок коду в порівнянні з традиційним форматом запитів.

Для прикладу: для роботи нам необхідно отримати інформацію про користувача по значенню `url`, отриманого з GET-запиту. Класичний запис PHP-коду та SQL-синтаксису для виконання цієї функції описаний у лістингу 2.24.

Лістинг 2.24 - Приклад коду доступу до бази даних через звичайні SQL-запити

```
$url = mysql_real_escape_string($_GET['url']);
$result = mysql_query("SELECT * FROM users WHERE url='$url'"
);
$user = mysql_fetch_array($result);
```

З використанням `safemysql` аналогічні дані можна отримати за допомогою функції значно меншого розміру, як зображено у лістингу 2.25.

Лістинг 2.25 - Приклад використання `safemysql`

```
$user = $db->getRow("SELECT * FROM users where name=?s",
$_GET['name']);
```

І чим складніший буде запит, тим більшою буде вигода від відмови використання стандартних запитів та використання `safemysql`.

Наступною вразливістю є Include-баг — ще одна загальновідома, проте популярна помилка, яку допускають програмісти розробляючи скрипти на PHP. Функція `include()` служить для того, щоб прикріплювати до PHP-коду інші готові модулі на PHP. Помилкою є написання коду формату: `include ("$file")`, де змінна `$file` береться з вхідного параметру скрипта (наприклад, `page.php?file=about.php`). В цьому випадку зловмиснику просто залишається створити на своєму web-сервері PHP-файл з web-шелом (<https://www.hacker.com/shell.php>) і передати в параметрі до скрипту адресу свого web-шела (наприклад, `page.php?file=shell.php`).

Тому вразливості неправильної обробки вхідних даних відкривають можливість не лише для XSS-атак та SQL-ін'єкцій. Використання загальновідомих функцій для екранування вхідного коду, а також перевірка усіх вхідних даних через форми та GET-запити допомагає уникнути будь-якого несанкціонованого доступу до web-сайту. Так як будь-який контекст отриманий сервером з браузера, і який взаємодіє з базою даних сайту, може бути

потенційно небезпечним і містити шкідливі скрипти, слід уважно та відповідально ставитися до нього.

Також у CMS відбувається перевірка форматів завантажуваних файлів у розділах, які дозволяють завантаження фотографій чи документів. Зловмисники можуть скористатися тим, що формат файлів, які завантажуються, не перевіряється на відповідність до формату документів, необхідних у даному полі. Відповідно зловмисник може завантажити замість \*.png файлу файл \*.js з шкідливим скриптом, який почне завантажуватись у користувачів при відкритті ними сайту. Ця перевірка відбувається як на стороні клієнта (за допомогою JS), так і на стороні серверу (за допомогою PHP), та дозволяє завантажувати у кожне поле лише файли певних форматів. Наприклад, у поле з фотографією можна додати лише файли з розширенням .png та .jpg, та аналогічно відбувається з документами — лише файли .doc, .docx, .pdf [18].

### **2.3.2 Захист від передачі сторонніх cookie і прихованої ідентифікації**

Розвиток безпеки даних не стоїть на місці, і великі корпорації продовжують впроваджувати нові інструменти в своїх сервісах. Відтак, в нових версіях браузера Google Chrome від корпорації Google продовжується акцент на захист персональної інформації. Зокрема, компанія змінить підхід до обробки Cookie і підтримки атрибута SameSite [11]. Уже в 76 версії браузера Google Chrome буде за замовчуванням включений прапор «same-site-by-default-cookies».

Тобто, якщо в отриманому HTTP-заголовку не буде вказано атрибут SameSite, браузер автоматично виставить значення "SameSite = Lax", яке обмежує відправку cookie для вставок зі сторонніх сайтів.

Атрибут SameSite дає змогу визначити можливість передачі Cookie під час отримання запиту зі стороннього ресурсу. Наразі Cookie передаються на будь-який запит до сайту, для якого виявлені Cookie, навіть у випадку, якщо було відкрито один сайт, а звернення здійснюється опосередковано за допомогою завантаження зображення або через блок iframe.

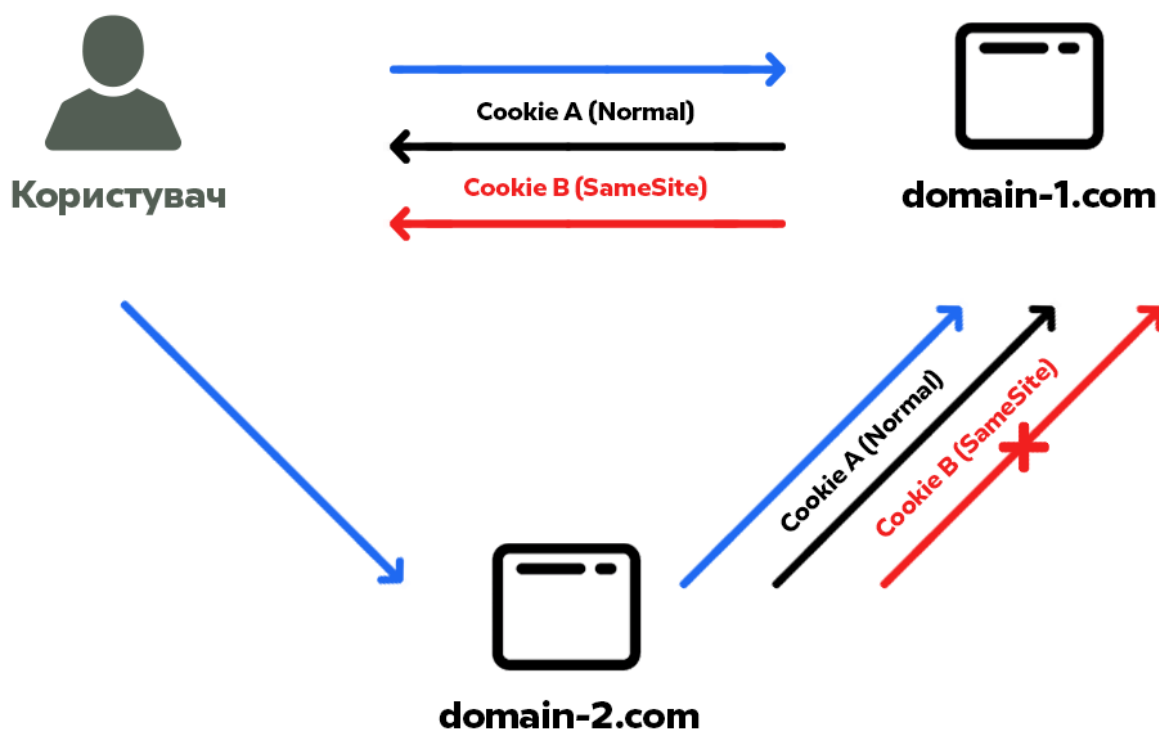


Рисунок 2.8 Схеми взаємодії користувача та web-сайтів з Cookie по старим та новим стандартам

Цю можливість використовують рекламні мережі, щоб відстежувати переміщення користувача між сайтами. Крім того, часто цю вразливість використовують і зломисники для проведення CSRF-атак. Так називають атаки, в ході яких при відкритті скомпрометованого сайту з його сторінок відправляється запит на інший сайт, де поточний користувач вже був авторизований. При цьому браузер користувача виставляє для зазначеного запиту сесійні-файли.

### 2.3.3 Двохфакторна аутентифікація через Google Authenticator

Одним з найскладніших для компрометації методів другого фактору є використання одноразових паролів, які генеруються за алгоритмом на основі часу TOTP (Time-based One-time Password Algorithm). Найпопулярнішим представником цього методу є розроблений корпорацією Google додаток Google Authenticator.

Підключаючи другий фактор аутентифікації, користувач встановлює на свій мобільний пристрій iOS/Android додаток Google Authenticator, після чого сайт надає користувачу секретний ключ у вигляді QR-коду, який скануванням на камеру мобільного пристрою зберігається у додаток.

Під час наступного входу на web-сайт користувач повинен після введення імені користувача і пароля, на наступному кроці аутентифікації, ввести 6-ти або 8-ми значний одноразовий пароль, який генерується на короткий проміжок часу (наприклад, 15 секунд).

Алгоритм TOTP (Time-based One-time Password Algorithm, RFC 6238 [16]), що використовується у додатку, це алгоритм OATH, створений для захищеної аутентифікації та являє собою вдосконалений алгоритм HOTP (HMAC-Based One-Time Password Algorithm). У 2004 році OATH (Ініціативи відкритої автентифікації) працювала над проектом одноразових паролів (OTP). Першим результатом їх роботи був HOTP (алгоритм OTP на основі Hash-коду аутентифікації повідомлень (HMAC)), опублікований в грудні 2005 року. Подальше робота OATH була спрямована на вдосконалення HOTP і в 2008 році був представлений алгоритм TOTP. Головна їх відмінність — це генерація паролю на основі часу, який є основним параметром. При цьому використовується не точний поточний час, а поточний інтервал, межі якого було встановлено заздалегідь (наприклад, 15 секунд) [17].

#### **2.3.4 Розмежування прав доступу користувачів до функцій адмін-панелі**

У захищеній CMS необхідно передбачити можливість чіткого розмежування доступу до функцій адмін-панелі між декількома користувачами відповідно до їх потреб.

Аутентифікований користувач не повинен отримувати доступ до усього функціоналу, ці привілеї необхідно надавати лише головному адміністратору, як і перегляд детального журналу подій. Саме він і повинен розпоряджатися наданням доступу до певних функцій іншим користувачам.



Відповідно, редактори отримують доступ лише до додавання та редагування статей, модератори до перегляду та видалення коментарів, а дизайнери та перекладачі доступ до редагування елементів інтерфейсу. Це допоможе уникнути проблеми людського фактору, коли випадковим, або навмисним чином системі наноситься завдається удар через дії користувачів.

### **2.3.5 Переваги використання власної розробки над популярними рішеннями**

Основною перевагою власної розробки є можливість детального налаштування функціоналу web-сайту під потреби конкретного проекту. Гнучкість системи дозволяє інтегрувати будь-який складний функціонал, та сторонні сервіси, які не представлені у вигляді додаткових модулів для існуючих CMS, що надає суттєві переваги при розробці нестандартних рішень.

З точки зору безпеки перевагою є відсутність готових рішень для реалізації популярних атак, які з легкістю можна знайти у мережі під більшість популярних CMS. Необхідність персоналізованої розробки для атаки на CMS власної розробки суттєво збільшує вартість цих атак.

## **2.4 Висновки до розділу**

У результаті роботи було визначено основні вимоги до функціоналу нової CMS та методів її захисту, проведено інтеграцію з API Укрпошти, API Liqpay, API SendGrid, API SMSC.ua, та розроблено новий функціонал на основі цих інтеграцій. Також було проведено інтеграцію з сервісом Cloudflare та впроваджено програмні алгоритми захисту від атак, описаних у попередньому розділі.

## РОЗДІЛ 3 РОБОТА CMS

### 3.1 Модуль авторизації та аутентифікації

#### 3.1.1 Авторизація за допомогою паролю

Перший етап авторизації користувача відбувається на сторінці входу в адмін-панель за допомогою звичної комбінації з логіну та паролю.

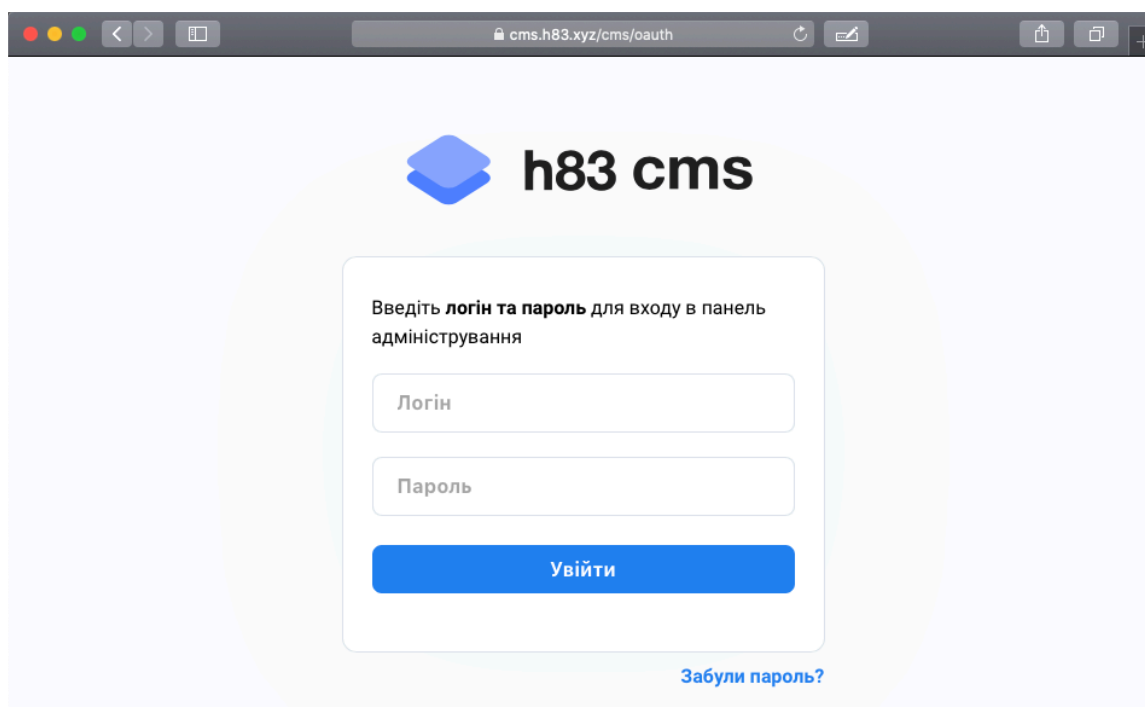


Рисунок 3.1 Сторінка авторизації користувача в адмін-панелі

Введений користувачем логін перевіряється у базі даних на наявність такого користувача, а введений пароль хешується алгоритмом MD5[19], та перевіряється з захешованим паролем в базі даних у відповідності до введенного логіну.

Пароль у базі даних не зберігається в відкритому вигляді, зберігається лише його хеш. Приклад хешу який зберігається у базі даних H83 CMS — 70e09a00b616fb87774dac504de31bfb

Якщо хеші співпадають, та пароль вірний, перший етап аутентифікації проходить успішно, і користувач переадресовується на другий фактор аутентифікації.

### 3.1.2 Другий фактор аутентифікації

При першому вході для користувача генерується QR-код, за допомогою якого він отримує токен для додавання у мобільний додаток Google Authenticator, та проходження другого фактору аутентифікації з його використанням майбутньому.

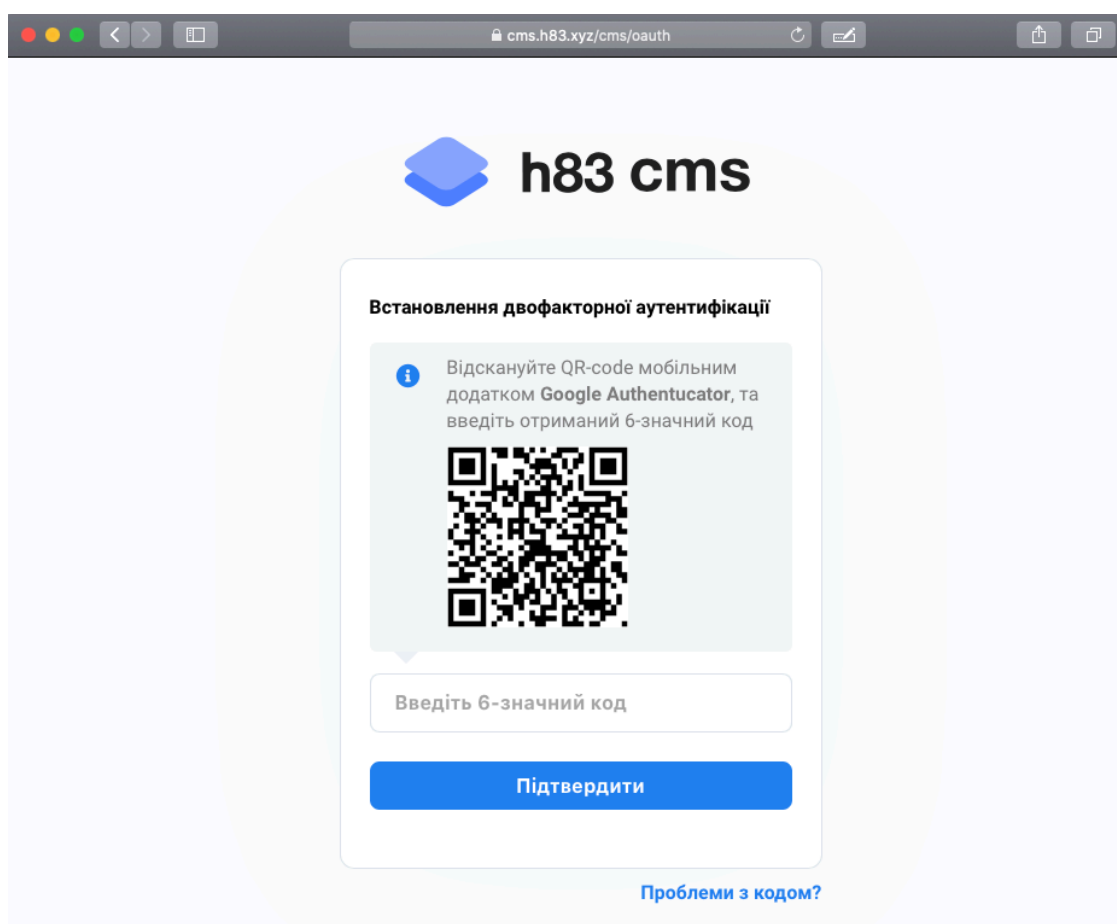


Рисунок 3.2 Сторінка авторизації користувача в адмін-панелі при першій аутентифікації

Після успішного приєднання до профілю у CMS коду синхронізації додатком Google Authenticator, додаток показує користувачу 6-значний код, який за алгоритмом у пункті 2.3.3 змінюється кожні 15 секунд. Користувач вводить

отриманий код протягом короткого проміжку часу його актуальності для підтвердження другого фактору аутентифікації.

При другому і наступних входах процедура другого фактору аутентифікації користувача проводиться звичайним введенням 6-ти значного одноразового паролю з мобільного додатку у спеціальне поле. У випадку правильного введення паролю, користувач проводить успішний вхід у адмін-панель.

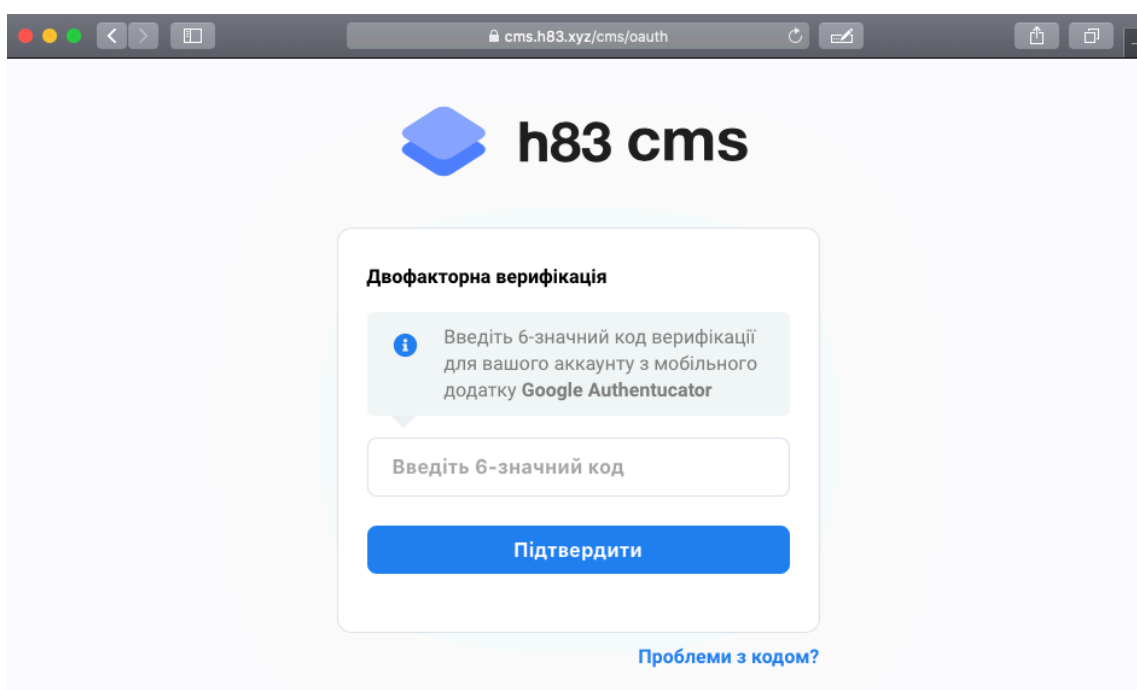


Рисунок 3.3 Сторінка авторизації користувача в адмін-панелі при другій та наступних аутентифікаціях

### 3.1.3 Генерація токена сесії та його перевірка

При успішній аутентифікації користувача проводиться генерування юзер-токену сесії, та сервер-токену сесії, які необхідні для ідентифікації поточної сесії користувача.

Унікальність токенів зумовлена алгоритмами їх генерації. Юзер-токен утворюється складанням випадкових чисел, поточного часу та хешуванням отриманого значення алгоритмом MD5, і зберігається у Cookie користувача. Алгоритм генерації юзер-токену зображено у лістингу 3.1.

### Лістинг 3.1 - Алгоритм генерації юзер-токену

```
$user_token = substr(md5(md5(rand(0,100^100)).time()).rand(0,100^100)), 0, 31);
```

Приклад згенерованого юзер-токену: b94a56b2d4dd69fe4e4821d00a59ef

Після генерації юзер-токену відбувається генерація сервер-токену, який буде зберігатися у базі даних. Він генерується шляхом хешування алгоритмом MD5 значення, отриманого сумуванням токену, записаного в Cookie, IP-адреси користувача та UserAgent користувача, що зображено у лістингу 3.2.

### Лістинг 3.2 - Алгоритм генерації сервер-токену:

```
$server_token = md5($_COOKIE["token"].$_SERVER['REMOTE_ADDR'].$_SERVER['HTTP_USER_AGENT']);
```

Перевірка актуальності сесії користувача відбувається перед завантаженням кожної сторінки адмін-панелі. Так як сесія прив'язана до IP-адреси користувача та його пристрою (через UserAgent), можлива лише одна активна сесія авторизації. При успішному логуванні на іншому пристрої у базу даних записується новий сервер-токен, тому перша сесія стає автоматично недійсною. Використання цього алгоритму унеможливорює перехоплення Cookie, та використання сесії зломисником на своєму пристрої.

## 3.2 Функції адмін-панелі

### 3.2.1 Створення публікації

Однією з базових функцій будь-якого web-сайту є можливість публікації новин, чи іншого типу матеріалів, та ведення блогу. У H83 CMS передбачена можливість публікації записів у декількох рубриках та на декількох мовах, публікації яких прив'язуються до мовної версії сайту, яка переглядається користувачем. Також можливе створення чернеток, що дозволяє зберігати певні редакторські напрацювання, та публікувати їх згодом.

Для редагування тексту, додавання зображень, таблиць, чи іншої базової розмітки, на платформі використовується текстовий редактор CKEditor 5 версії.

Це популярний кросплатформовий WYSIWYG-редактор, який дозволяє швидко та зручно оперувати з текстом через адмін-панель.

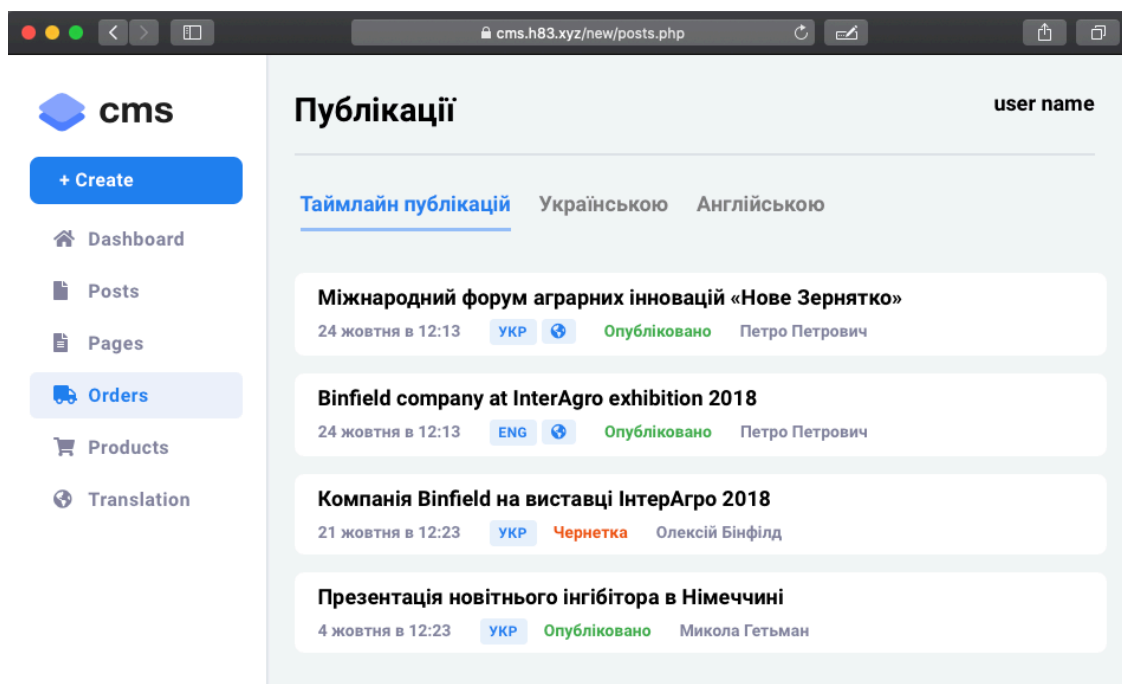


Рисунок 3.4 Сторінка перегляду списку поточних публікацій

### 3.2.2 Каталог товарів

Функціонал адмін-панелі дозволяє менеджерам інтернет-магазину створювати, редагувати та тимчасово приховувати товари, яких немає у наявності. Через модуль каталогу товарів можна виконувати наступні дії з асортиментом товарів:

- вказувати наявність товару;
- вказувати декілька розмірів одного товару;
- завантажувати декілька фотографій товарів;
- групувати схожі товари;
- вказувати знижку на товари
- додавати товари з різних категорій у одну колекцію;
- вказувати висоту та ширину товару, а також його вагу;
- вказувати параметри товару, такі як бренд, колір, та інші параметри, які

можна попередньо налаштовувати для використання в пошуку по фільтрам.

Використовуючи список доданих товарів CMS автоматично генерує Sitemap-файл у форматі XML для зручної індексації пошуковими системами, який також називають “мапа” сайту, та фід даних про товарів у спеціальному форматі Google Merchant, який допомагає автоматично налаштовувати товарну рекламу у рекламних сервісах Google. Ці файли оновлюються автоматично, та враховують усі зміни ціни/наявності товарів, та додавання нових товарів.

### 3.2.3 Розділ замовлень

У цьому розділі відбувається перегляд списку списку всіх замовлень та взаємодія з ними. Менеджер може переглянути як успішні оплачені та підтверджені замовлення, готові до відправки, так і невдалі замовлення з неуспішною оплатою та замовлення, які користувачі не підтвердили. Маючи особисті дані користувачів замовлень, які стикнулись з певними труднощами при оплаті, адміністратор може зв’язатись з ними для вирішення виникнутих проблем.

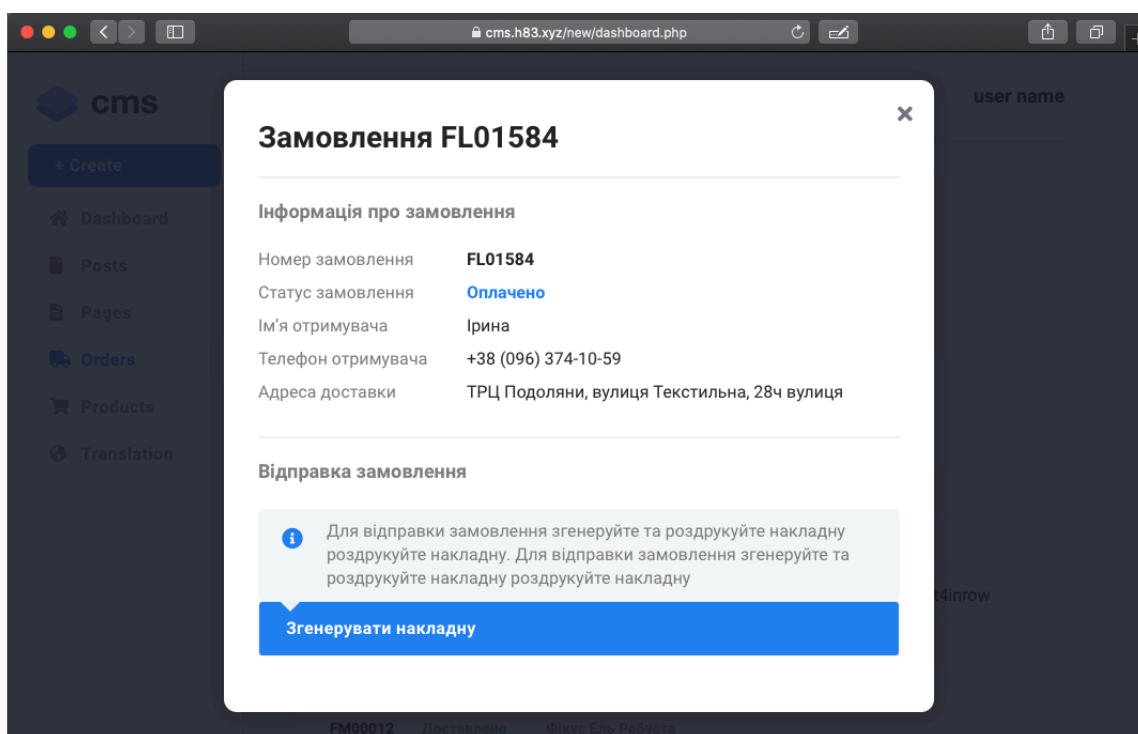


Рисунок 3.5 Вспливаюче вікно з переглядом замовлення

У випадку взаємодії з оплаченими та підтвердженими замовленнями, менеджеру надається доступ до функціоналу генерації накладних відправлення для поштового сервісу Укрпошта, яким відбувається відправка.

### **3.3 Журнал подій**

#### **3.3.1 Звіт про події в адмін-панелі**

Для того, щоб головний адміністратор мав змогу контролювати роботу усіх менеджерів та редакторів, які задіяні в проекті, відбувається автоматичне ведення журналу усіх дій, які виконуються ними. Створення нових публікацій, редагування товарів чи генерація накладних — усі ці дії формують звіт, доступ до якого має лише головний адміністратор web-сервісу або адміністратор, якому були надані такі права. Також відбувається запис усіх завантажень сторінок адмін-панелі, що дозволяє використовувати ці дані в подальшому аналізі.

З точки зору безпеки логування допомагає вичисляти навіть невдалі атаки на web-сайт, адже у журналі подій фіксуються як неправильні входи, що допомагає виявити brute-force-атаку, так і спроби завантаження шкідливих файлів через інструменти для завантаження фотографій. Для прикладу, при спробі завантаження JS-скрипта через форму для завантаження зображення, зломисник побачить помилку, та не зможе завантажити файл, проте ця спроба буде записана у журналі подій.

#### **3.3.2 Логування з використанням LogDNA**

Журнал подій є лише частиною модуля логування подій на web-сайті та на web-сервері. Для обширного аналізу активності сервера використовується сервіс LogDNA. LogDNA — це спеціальна система управління журналу подій, яка дозволяє розробникам та системним адміністраторам збирати та аналізувати дані про звіти системи та додатку в одному місці.

Після встановлення програми на сервер, вона починає збирати всі системні дані для аналітики у зручному форматі: файли access.log (логування доступу до серверу), syslog (системні логування, наприклад, перезавантаження



серверу, додавання нових користувачів), error.log (файл виникнутих на сервері помилок) та багато інших системних журнальних файлів [26].

В доповнення до агрегації усіх стандартних журналів подій, які генеруються на сервері, сервіс дозволяє додатково налаштувати запис власних подій через API. Це дозволяє дублювати записи з журналу подій у адмін-панелі у LogDNA, та дозволити перегляд та обробку усіх логувань в одному місці.

Лістинг 3.3 - Приклад коду для запису рядка логування з власними параметрами через API сервісу LogDNA

```
curl "https://logs.logdna.com/logs/ingest?
hostname=h83&mac=C0:FF:EE:C0:FF:EE&ip=10.0.1.101&now=$(date +%s)" \
-u INSERT_INGESTION_KEY: \
-H "Content-Type: application/json; charset=UTF-8" \
-d \
'{
  "lines": [{
    "line": "This is an awesome log statement",
    "app": "myapp",
    "level": "INFO",
    "env": "production",
    "meta": {
      "customfield": {
        "nestedfield": "nestedvalue"
      }
    }
  ]
}
```

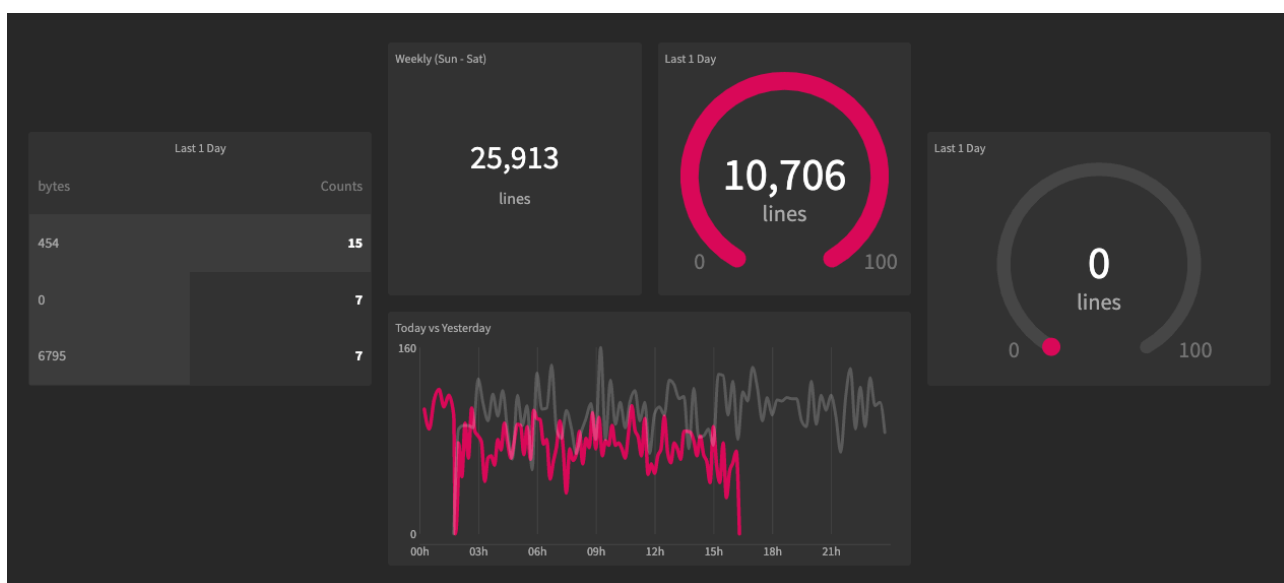


Рисунок 3.6 Вигляд екрану з статистикою записів логування LogDNA

### 3.4 Розподіл прав користувачів

#### 3.4.1 Видача прав користувачу

У розробленій CMS є можливість налаштовувати обмеження доступу та можливість видачі додаткових прав певним користувачам в залежності від їх потреб, та в залежності від необхідного для їх роботи функціоналу.

Права доступу до кожного розділу поділяються на категорії за рівнем доступу. У головного адміністратора є можливість дозволити лише перегляд, або перегляд та редагування та створення нових записів кожному користувачу для кожного модуля, встановленого в CMS для даного сайту.

#### 3.4.2 Спроба доступу до розділу без наявності прав

Акцент на конфіденційності даних та доступу до них займає у H83 CMS одне з пріоритетних місць. Тому на основі конфігурацій налаштування прав доступу до розділів адмін-панелі відбувається перевірка наявності даних прав у поточного користувача при доступі до певного розділу. Ця перевірка відбувається при кожному завантаженні сторінки користувачем. Приклад функцій цієї перевірки можна побачити в лістингу 3.4.

Лістинг 3.4 - Функції перевірки наявності прав доступу користувача до поточного розділу

```
public function permissions($section){
    if(isset($this->config["pages"][$section])){
        $section = $this->config["pages"][$section];
        return $this->permissions->$section;
    }
    else{
        return false;
    }
}

public function permissions_verification(){
    if($this->permissions($this->active_tab()) == false or
    $this->permissions($this->active_tab()) == "no"){
        header("location: /cms/dashboard");exit();
    }
}
```

### **3.5 Висновки до розділу**

У даному розділі було показано механізм роботи деяких розділів та функції нової CMS. Було проілюстровано роботу модуля авторизації, показано механізм роботи другого фактору аутентифікації, генерації юзер-токенів та сервер-токенів, показано розділи для створення нових публікацій, каталог товарів, розділ замовлень та журнал подій, а також показано інтеграцію логування з використанням сервісу LogDNA та модуль аналізу наявності прав користувача при доступу до кожного з розділів, та описано алгоритм видачі прав доступу до розділів новим користувачам.

## РОЗДІЛ 4 ТЕСТУВАННЯ АТАК НА WEB-САЙТ

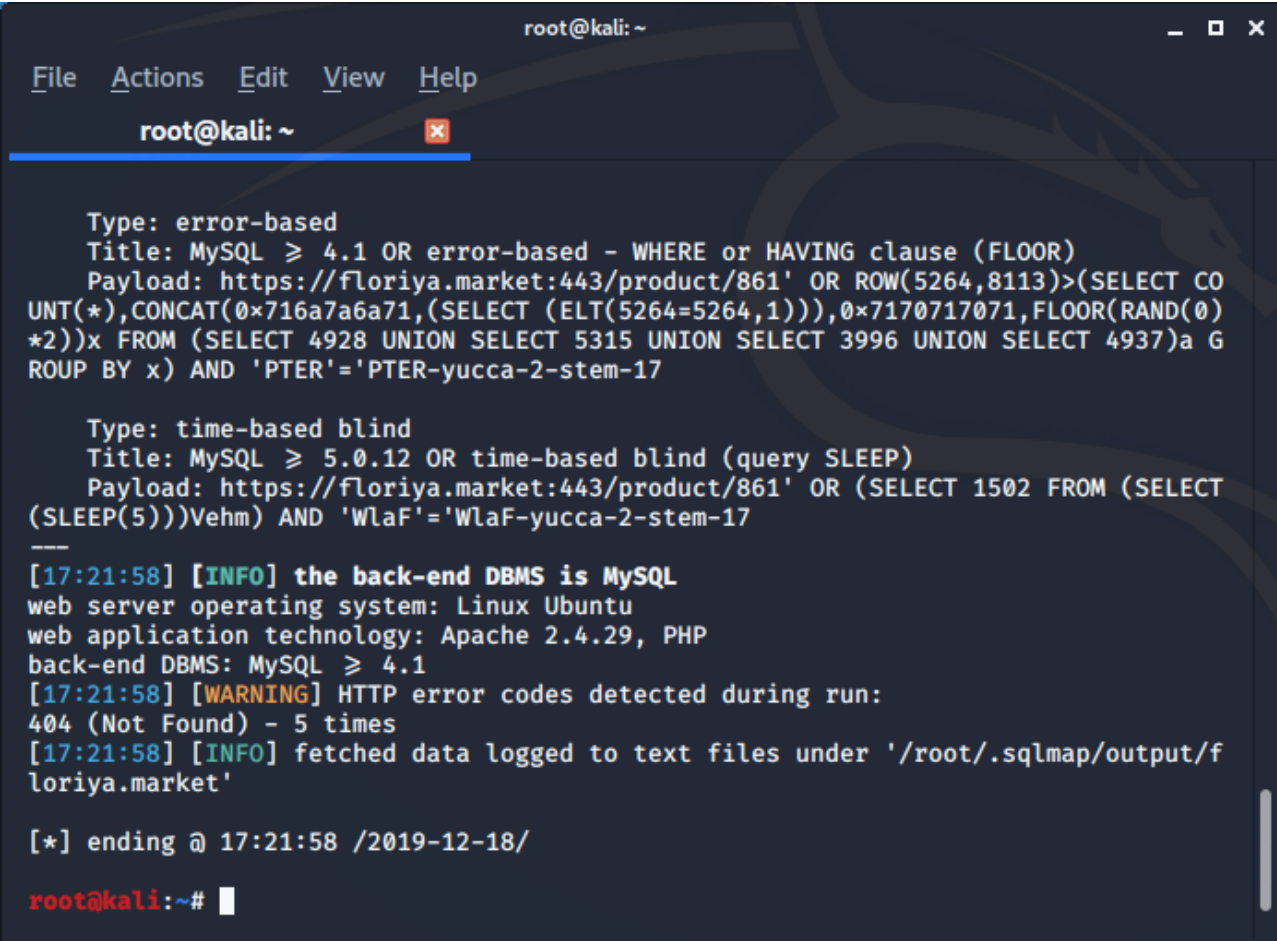
### 4.1 Вибір середовища тестування

Для того щоб переконатися у захищеності web-сайтів, розроблених на основі N83 CMS, було вирішено провести тестування на популярні вразливості. Для проведення цих тестувань було використано середовище Kali Linux.

Kali Linux — це дистрибутив Linux, який використовують для цифрової криміналістики та тестувань на проникнення. Цей дистрибутив включає в себе обширний набір утиліт для оцінки рівня захищеності комп'ютерних систем та web-сайтів шляхом моделювання дій зловмисників для проникнення у них.

### 4.2 Тестування на SQL-ін'єкції

У ході тестування було проведено тестування на вразливість через SQL-ін'єкції за допомогою утиліти sqlmap у середовищі Kali Linux.



```

root@kali: ~
File Actions Edit View Help
root@kali: ~

Type: error-based
Title: MySQL ≥ 4.1 OR error-based - WHERE or HAVING clause (FLOOR)
Payload: https://floriya.market:443/product/861' OR ROW(5264,8113)>(SELECT COUNT(*),CONCAT(0x716a7a6a71,(SELECT (ELT(5264=5264,1))),0x7170717071,FLOOR(RAND(0)*2))x FROM (SELECT 4928 UNION SELECT 5315 UNION SELECT 3996 UNION SELECT 4937)a GROUP BY x) AND 'PTER'='PTER-yucca-2-stem-17

Type: time-based blind
Title: MySQL ≥ 5.0.12 OR time-based blind (query SLEEP)
Payload: https://floriya.market:443/product/861' OR (SELECT 1502 FROM (SELECT (SLEEP(5)))Vehm) AND 'WlaF'='WlaF-yucca-2-stem-17

[17:21:58] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.29, PHP
back-end DBMS: MySQL ≥ 4.1
[17:21:58] [WARNING] HTTP error codes detected during run:
404 (Not Found) - 5 times
[17:21:58] [INFO] fetched data logged to text files under '/root/.sqlmap/output/floriya.market'

[*] ending @ 17:21:58 /2019-12-18/
root@kali:~#

```

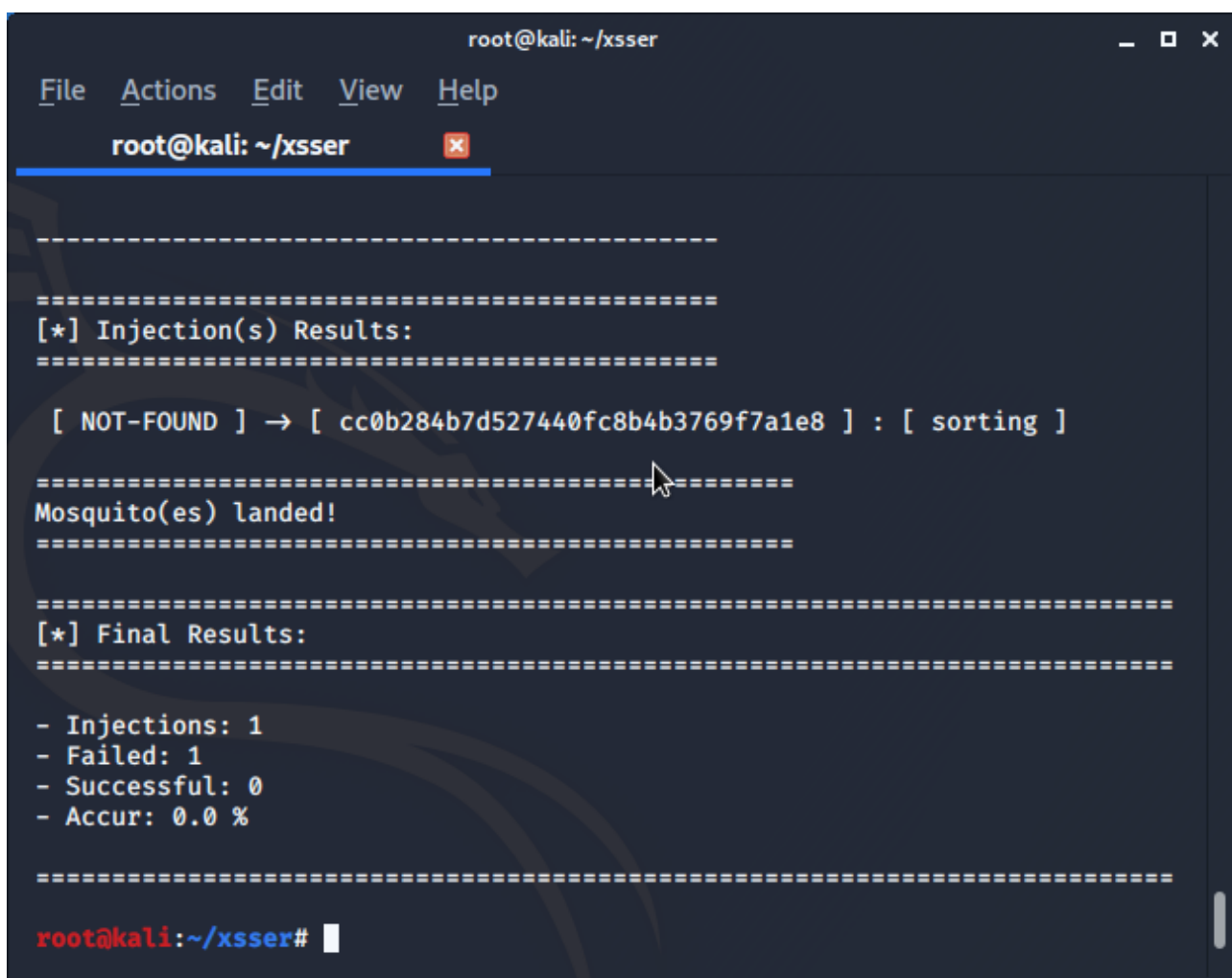
Рисунок 4.1 Результат тестування на SQL-ін'єкції через утиліту sqlmap

Sqlmap — це відкрите програмне забезпечення, яке використовують для виявлення вразливих у базах даних SQL та підбору варіантів введення в них шкідливих фрагментів коду [36].

Як зображено на Рисунок 4.1, у результаті тестування web-сайту на наявність вразливостей до SQL-ін'єкцій у місцях, де передаються змінні для доступу до рядка бази даних, завдяки обробці усіх вхідних значень змінних спеціальними функціями, вразливостей до цього типу атак не було виявлено.

### 4.3 Тестування на атаку XSS

Наступним кроком тестування була перевірка web-сайту на вразливості типу XSS за допомогою утиліти XSSer — автоматичного фреймворку для виявлення, експлуатації та повідомлення про XSS-вразливості у web-додатках.



```
root@kali: ~/xsser
File Actions Edit View Help
root@kali: ~/xsser x

-----
=====
[*] Injection(s) Results:
=====

[ NOT-FOUND ] -> [ cc0b284b7d527440fc8b4b3769f7a1e8 ] : [ sorting ]
=====
Mosquito(es) landed!
=====

[*] Final Results:
=====

- Injections: 1
- Failed: 1
- Successful: 0
- Accur: 0.0 %

=====
root@kali:~/xsser#
```

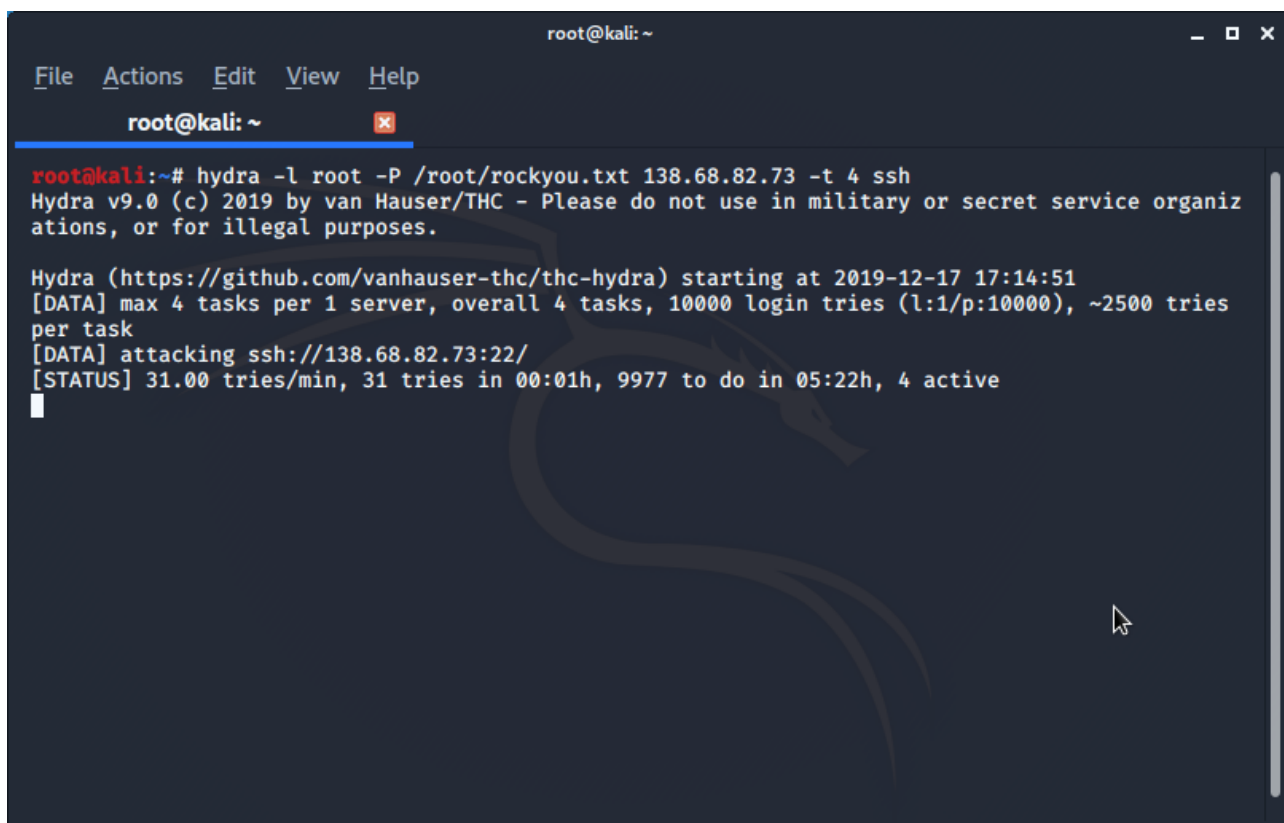
Рисунок 4.2 Результат тестування утилітою XSSer

Цей інструмент містить декілька опцій для обходу фільтрів екранування вхідного коду, та використовує різноманітні техніки вставки шкідливого коду.

Як зображено на Рисунок 4.2, у результаті тестування утилітою XSSer на наявність вразливостей до XSS-атак, завдяки використанню стандартних функцій екранування та обробки вхідних даних, а також спеціальних класів обробки коду, які враховують методи обходу фільтрування, вразливостей до цього типу атак не було виявлено.

#### 4.4 Тестування на brute-force-атаку на SSH

Для наступного тесту на проникнення було обрано утиліту hydra, яка входить до складу Kali Linux, та є популярною серед багатьох тестувальників завдяки своїй швидкості та надійності роботи.

A screenshot of a terminal window titled 'root@kali: ~'. The terminal shows the command 'hydra -l root -P /root/rockyou.txt 138.68.82.73 -t 4 ssh' being executed. The output includes the Hydra version (v9.0), a warning not to use it for illegal purposes, the start time (2019-12-17 17:14:51), and attack statistics: 'max 4 tasks per 1 server, overall 4 tasks, 10000 login tries (l:1/p:10000), ~2500 tries per task', 'attacking ssh://138.68.82.73:22/', and '[STATUS] 31.00 tries/min, 31 tries in 00:01h, 9977 to do in 05:22h, 4 active'. A progress bar is visible at the bottom of the output.

```
root@kali: ~  
File Actions Edit View Help  
root@kali: ~  
root@kali:~# hydra -l root -P /root/rockyou.txt 138.68.82.73 -t 4 ssh  
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organiz  
ations, or for illegal purposes.  
  
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2019-12-17 17:14:51  
[DATA] max 4 tasks per 1 server, overall 4 tasks, 10000 login tries (l:1/p:10000), ~2500 tries  
per task  
[DATA] attacking ssh://138.68.82.73:22/  
[STATUS] 31.00 tries/min, 31 tries in 00:01h, 9977 to do in 05:22h, 4 active  
█
```

Рисунок 4.3 Процес тестування на brute-force-атаку через SSH за допомогою утиліти hydra

Для проведення brute-force-атаки на SSH було обрано атаку на суперкористувача root, який є стандартним та головним користувачем у Linux-

системах. Було використано файл `gookyou.txt`, у якому знаходилась вибірка найпопулярніших паролів. Швидкість підбору цією утилітою складає близько 31-го пароля на хвилину, що при довжині паролю в 16 символів складає  $62^{16}$  варіацій, а отже час на перебір паролю довжиною 16 випадкових символів буде складати десятки років, що дозволяє вважати атаку неуспішною.

#### **4.5 Висновки до розділу**

У даному розділі було вибрано середовище для тестування — Kali Linux. Web-сайт на основі CMS було протестовано на вразливість через SQL-ін'єкції за допомогою утиліти `sqlmap`, перевірено на вразливості типу XSS за допомогою утиліти `XSSer`, та за допомогою утиліти `hydra` було проведено brute-force атаку на SSH для суперкористувача `root`. Усі ці атаки були невдалими, що ще раз підтвердило правильність вибору методів захисту.

## РОЗДІЛ 5 ОБҐРУНТУВАННЯ ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ

### 5.1 Розрахунок норм часу на виконання науково-дослідної роботи

Ефективне використання часу має велике значення, адже коефіцієнт корисної дії наряду залежить від оптимального використання часу. Метою магістерської роботи є розробка нової CMS та методів захисту web-сайтів на її основі. Робота над проектом поділена на декілька етапів, що дозволяє полегшити і структурувати виконання поставленого завдання.

Основні етапи такі:

1. Дослідження популярних CMS для роботи інтернет-магазинів на українському ринку та пошук нових сервісів для інтеграції;
2. Дослідження популярних вразливостей та типів атак на web-сайти;
3. Інтеграція нової CMS з необхідними сторонніми сервісами;
4. Розробка методів захисту від популярних типів атак.

Таблиця 5.1 – Операції технологічного процесу та їх час виконання

№ п/п	Назва операції (стадії)	Виконавець	Середній час виконання операції, год.
1.	Дослідження популярних CMS для роботи інтернет-магазинів на українському ринку та пошук нових сервісів для інтеграції	розробник	32
2.	Дослідження популярних вразливостей web-сайтів та типів атак на них	розробник	22
3.	Інтеграція нової CMS з необхідними сторонніми сервісами	розробник	48
4.	Розробка методів захисту від популярних типів атак	розробник	28
Разом			130



Для оцінки тривалості виконання окремих робіт використовують нормативи часу. Виконавцем усіх вказаних вище етапів є розробник. Витрати часу по окремих операціях технологічного процесу відображені в таблиці 5.1.

Загальні затрати часу на реалізацію даної роботи становить 130 годин, найбільш трудомістким є інтеграція сторонніх сервісів у нову CMS – 48 годин.

## **5.2 Визначення витрат на оплату праці та відрахувань на соціальні заходи**

Відповідно до Закону України “Про оплату праці” заробітна плата – це “винагорода, обчислена, як правило, у грошовому виразі, яку власник або уповноважений ним орган виплачує працівникові за виконану ним роботу”.

Розмір заробітної плати залежить від складності та умов виконуваної роботи, професійно-ділових якостей працівника, результатів його. Заробітна плата складається з основної та додаткової оплати праці. Основна заробітна плата нараховується за виконану роботу за тарифними ставками, відрядними розцінками чи посадовими окладами.

Додаткова заробітна плата – це складова заробітної плати працівників, до якої включають витрати на оплату праці, не пов’язані з виплатами за фактично відпрацьований час. Нараховують додаткову заробітну плату залежно від досягнутих і запланованих показників, кваліфікації виконавців. Джерелом додаткової оплати праці є фонд матеріального стимулювання, який створюється за рахунок прибутку.

При розрахунку заробітної плати кількість робочих днів у місяці слід в середньому приймати – 24,5 дні/міс., або ж 196 год./міс. (тривалість робочого дня – 8 год.).

Місячний оклад кожного працівника слід враховувати згідно існуючих на даний час тарифних окладів. Згідно закону України «Про Державний бюджет України на 2019 рік», зокрема статтею восьмою мінімальна заробітна плата у погодинному розмірі становить 25,13 грн.

Рекомендовані тарифні ставки: керівник магістерської роботи – 30,00...50,00 грн./год., розробник – 25,13...30,00 грн./год., консультант –

25,13...30,00 грн./год., технік – 25,13...30,00 грн./год., лаборант – 25,13...26,00 грн./год.

Основна заробітна плата розраховується за формулою:

$$Z_{осн.} = T_c \cdot K_z, \quad (5.1)$$

де  $T_c$  – тарифна ставка, грн.;  $K_z$  – кількість відпрацьованих годин.

Оскільки всі види робіт в виконує розробник, то основна заробітна плата буде розраховуватись тільки за однією формулою

$$Z_{осн.} = 25,13 \cdot 126 = 3266,9 \text{ грн.}$$

Додаткова заробітна плата становить 10–15 % від суми основної заробітної плати.

$$Z_{дод.} = Z_{осн.} \cdot K_{дод.}, \quad (5.2)$$

де  $K_{дод.}$  – коефіцієнт додаткових виплат працівникам, 0,1–0,15 (візьмемо його рівним 0,15).

$$Z_{дод.} = 3266,9 \cdot 0,15 = 490,03 \text{ грн.}$$

Звідси загальні витрати на оплату праці ( $B_{о.п.}$ ) визначаються за формулою:

$$B_{о.п.} = Z_{осн.} + Z_{дод.} \quad (5.3)$$

$$B_{о.п.} = 3266,9 + 490,03 = 3756,93 \text{ грн.}$$

Крім того, слід визначити відрахування на соціальні заходи:

- єдиний соціальний внесок ЄСВ (прибутковий податок) – 22%;
- військовий збір – 1,5%.

У сумі зазначені відрахування становлять 23,5 %.

Отже, сума відрахувань на соціальні заходи буде становити:

$$B_{с.з.} = \Phi_{оп} \cdot 0,235 \quad (5.4)$$

де  $\Phi_{оп}$  – фонд оплати праці, грн.

$$B_{с.з.} = 3756,93 \cdot 0,235 = 882,87 \text{ грн.}$$

Проведені розрахунки витрат на оплату праці наведено у таблицю 5.2.

З таблиці розрахунки витрат на оплату праці видно що всього витрати на плату праці становить 4639,8 грн.

Таблиця 5.2 – Розрахунки витрат на оплату праці

з/п	Категорія працівників	Основна заробітна плата, грн.			Додаткова заробітна плата, грн.	Відрахування $\Phi_{OP}$ , грн.	Всього витрати на плату праці, грн. (6=3+4+5)
		Тарифна ставка, грн.	Кількість відпрацьованих год.	Фактично нарах. з/пл., грн.			
А	Б	1	2	3	4	5	6
1.	Розробник	25,13	130	3266,9	490,03	882,87	4639,8

### 5.3 Розрахунок матеріальних витрат

Матеріальні витрати визначаються як добуток кількості витрачених матеріалів та їх ціни:

$$M_{vi} = q_i \cdot p_i, \quad (5.5)$$

де:  $q_i$  – кількість витраченого матеріалу  $i$ -го виду;  $p_i$  – ціна матеріалу  $i$ -го виду.

Звідси, загальні матеріальні витрати можна визначити:

$$Z_{м.в.} = \sum M_{vi}. \quad (5.6)$$

Таблиця 5.3 – Розрахунки матеріальних витрат

Найменування матеріальних ресурсів	Один. Виміру	Норма витрат	Ціна за один., грн.	Затрати матер., грн.	Транспортно-заготівельні витрати, грн.	Загальна сума витрат на матер., грн.
1. Основні матеріали						
Використання мережі Internet	Години	120	–	120	–	120
2. Допоміжні витрати						
Папір формату А4	шт.	160	0,3	48	–	48
Разом:						168

Розрахунки занесено у таблицю 5.3. Отже загальні матеріальні витрати на Internet і Папір формату А4 становить 168 грн.

#### 5.4 Розрахунок витрат на електроенергію

Затрати на електроенергію 1-ці обладнання визначаються за формулою:

$$Z_e = W \cdot T \cdot S, \quad (5.7)$$

де  $W$  – необхідна потужність, кВт;  $T$  – кількість годин на реалізацію розробки;  $S$  – вартість кіловат-години електроенергії.

Вартість кіловат-години електроенергії слід приймати згідно існуючих на даний час тарифів. Отже, 1 кВт з ПДВ коштує 1,68 грн.

Потужність комп'ютера для створення дипломної роботи – 90 Вт, кількість годин роботи обладнання згідно таблиці 5.1 – 130 години.

Тоді,

$$Z_e = 0,09 \cdot 126 \cdot 1,68 = 19,65 \text{ грн.}$$

Згідно формули затрати на електроенергію де необхідна потужність множиться на кількість годин на реалізацію розробки і множиться на вартість кіловат-години електроенергії що в висновку дорівнює 19,65 грн.

#### 5.5 Розрахунок суми амортизаційних відрахувань

Характерною особливістю застосування основних фондів у процесі виробництва є їх відновлення. Для відновлення засобів праці у натуральному виразі необхідне їх відшкодування у вартісній формі, яке здійснюється шляхом амортизації.

Амортизація – це процес перенесення вартості основних фондів на вартість новоствореної продукції з метою їхнього повного відновлення.

Для визначення амортизаційних використовується формула:

$$A = \frac{B_B \cdot H_A}{100\%}, \quad (5.8)$$

де  $A$  – амортизаційні відрахування за звітний період, грн.;  $B_B$  – балансова вартість групи основних фондів на початок звітного періоду, грн.;  $H_A$  – норма амортизації.

Комп'ютери та оргтехніка належать до четвертої групи основних фондів. Для цієї групи річна норма амортизації дорівнює 60 % (квартальна – 15 %).

Для даної магістерської роботи засобом розробки є комп'ютер. Його сума становить 12480 грн. Отже, амортизаційні відрахування будуть рівні:

$$A = 12480 \cdot 5\% / 100\% = 624,00 \text{ грн.}$$

Оскільки робота виконувалась 126 години, то амортизаційні відрахування будуть становити:

$$A = 624,00 \cdot 126 / 126 = 624,00 \text{ грн.}$$

Згідно формули для визначення амортизаційних де  $B_B$  множиться  $H_A$  і ділиться на 100% амортизація розробки становить 624,00 грн.

### 5.6 Обчислення накладних витрат

Накладні витрати пов'язані з обслуговуванням виробництва, утриманням апарату управління спілкою та створення необхідних умов праці.

В залежності від організаційно-правової форми діяльності господарюючого суб'єкта, накладні витрати можуть становити 20–60 % від суми основної та додаткової заробітної плати працівників.

$$H_g = B_{o.n.} \cdot 0,2 \dots 0,6, \quad (5.9)$$

де  $H_g$  – накладні витрати.

Отже, накладні витрати:

$$H_g = 4639,8 \cdot 0,2 = 927,96 \text{ грн.}$$

Накладні витрати згідно розрахунку формули, становить 927,96 грн.

### 5.7 Кошторис та визначення собівартості науково-дослідницької роботи

Результати проведених вище розрахунків зведемо у таблицю 5.4.

Таблиця 5.4 – Кошторис витрат на НДР

Зміст витрат	Сума, грн.	В % до загальної суми
Витрати на оплату праці $B_{o.n.}$	3756,93	58,68
Відрахування на соціальні заходи $B_{c.z.}$	882,87	13,79
Матеріальні витрати $З_{м.в.}$	168,00	2,71
Витрати на електроенергію $З_{\epsilon}$	19,65	0,30
Амортизаційні відрахування $A$	624,00	10,06
Накладні витрати $H_{\epsilon}$	927,96	14,46
Собівартість $C_{\epsilon}$	6379,41	100,00

Собівартість ( $C_{\epsilon}$ ) роботи розраховуємо за формулою:

$$C_{\epsilon} = B_{o.n.} + B_{c.z.} + З_{м.в.} + З_{\epsilon} + A + H_{\epsilon} . \quad (5.10)$$

Отже, собівартість роботи дорівнює:

$$C_{\epsilon} = 3756,93 + 882,87 + 168 + 19,65 + 624,00 + 927,96 = 6379,41 \text{ грн.}$$

Загальний кошторис витрат та визначення собівартості науково-дослідницької роботи становить 6379,41 грн.

### 5.8 Розрахунок ціни науково-дослідної роботи

Ціну науково-дослідної роботи можна визначити за формулою:

$$Ц = C_{\epsilon} \cdot (1 + P_{рен.}) (1 + ПДВ) \quad (5.11)$$

де  $P_{рен.}$  – рівень рентабельності, 30 %,  $ПДВ$  – ставка податку на додану вартість, (20 %).

Звідси ціна на роботу складе:

$$Ц = 6379,41 \cdot (1 + 0,3) \cdot (1 + 0,2) = 9951,87 \text{ грн.}$$

Загальний розрахунок ціни програмного продукту становить 9951,87 грн.

### 5.9 Визначення економічної ефективності і терміну окупності

Ефективність виробництва – це узагальнене і повне відображення кінцевих результатів використання робочої сили, засобів та предметів праці на підприємстві за певний проміжок часу.

Економічна ефективність ( $E_p$ ) полягає у відношенні результату виробництва до затрачених ресурсів:

$$E_p = \frac{\Pi}{C_B}, \quad (5.12)$$

де  $\Pi$  – прибуток;  $C_B$  – собівартість.

Плановий прибуток ( $\Pi_{пл}$ ) знаходимо за формулою:

$$\Pi_{пл} = Ц - C_{\varepsilon}. \quad (5.13)$$

Розраховуємо плановий прибуток:

$$\Pi_{пл} = 9951,87 - 6379,41 = 3572,46 \text{ грн.}$$

Отже, формула для визначення економічної ефективності набуде вигляду:

$$E_p = \frac{\Pi_{пл}}{C_{\varepsilon}}. \quad (5.14)$$

Тоді,

$$E_p = 3572,46 / 6379,41 = 0,56.$$

Поряд із економічною ефективністю розраховують термін окупності капітальних вкладень ( $T_p$ ):

$$T_p = \frac{1}{E_p}, \quad (5.15)$$

Термін окупності дорівнює:

$$T_p = 1 / 0,56 = 1,8 \text{ р.}$$

Згідно формул плановий прибуток від розробки становить 3572,46 грн., економічна ефективність дорівнює 0,56 а термін окупності становить 1,8 роки що вважається доцільним та економічно вигідним.

### **5.10 Висновки до розділу**

У даному розділі було проведено розрахунок норм часу на виконання науково-дослідної роботи, визначено витрати на оплату праці на відрахування на соціальні заходи, проведено розрахунок матеріальних витрат, витрат на електроенергію, амортизаційні відрахування, обчислено накладні витрат, складено кошторис на визначено собівартість науково-дослідницької роботи, визначено термін окупності та обґрунтовано економічну ефективність.



## **РОЗДІЛ 6 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ**

### **6.1 Охорона праці**

Проведені дослідження та розробка програмного забезпечення відбувалося з використанням персональних комп'ютерів. Тому важливим фактором безпеки праці є дотримання основних правил користування комп'ютерною технікою, основних норм та правил охорони праці. Для цього потрібно забезпечити користувачам максимально комфортні та безпечні умови для їх перебування в приміщенні для того, щоб вони якісно та ефективно виконували поставлені завдання.

Працівники підприємств, де буде проводитися використання розробленої CMS для інтернет-магазину, повинні дотримуватися правил внутрішнього трудового розпорядку, виконувати правила особистої гігієни та гігієни приміщення, володіти знаннями з техніки безпеки користування технікою, електробезпеки, пожежної безпеки та не виконувати дії, які суперечать правилам охорони праці.

Будівлі та приміщення, де розміщені робочі місця для працівників, які будуть користуватися розробленою для інтернет-магазину CMS, повинні відповідати вимогам нормативно-технічної та експлуатаційної документації виробника персональних комп'ютерів ДСанПіН 3.3.2-007-98. []

Для внутрішнього оздоблення приміщень з персональними комп'ютерами слід обирати світлі нейтральні кольори стін. Покриття підлоги та поверхня має бути рівною, неслизькою, з антистатичними властивостями.

Відповідно до встановлених санітарно-гігієнічних норм (ГОСТ 12.1.005-88, СН 4088-86) [] регламентуються вимоги до приміщення, де буде проводитись захист персональної інформації:

- площа, відведена на одне робоче місце має становити не менше 6 кв.м, а об'єм – не менше 20 куб.м;

- для облаштування потрібно використовувати лише ті матеріали, які пройшли відповідну сертифікацію, не містять шкідливих речовин та дозволені для використання в приміщеннях;
- ергономічне розташування робочого місця, виробничих меблів з урахуванням антропометричних характеристик людини; раціональне компонування обладнання на робочих місцях;
- достатня освітленість робочих місць;
- гарантії електро- та пожежо- безпеки;
- приміщення не повинно межувати з джерелами шуму і вібрації, що перевищують допустимі норми;
- щодня має здійснюватися вологе прибирання та регулярне провітрювання приміщення.

Основним нормативним документом, що регулює забезпечення охорони праці користувачів комп'ютерної техніки є «Державні санітарні норми електронно-обчислювальних машин» ДСанПіН 3.3.2.007-98. У виробничих приміщеннях та на робочих місцях з ВДТ та ПК мають бути забезпечені оптимальні значення параметрів мікроклімату – температури повітря, відносної вологості, швидкості руху повітря. Для цього приміщення, в яких розташовані комп'ютеризовані робочі місця повинні бути обладнані системами опалення, кондиціонування, які автоматично підтримують задані параметри мікроклімату.

Згідно ГОСТ 12.1.005-88 “Загальні санітарно-гігієнічні вимоги до повітря робочої зони”, температура навколишнього середовища повинна бути в межах від +18 до + 22 °С, відносна вологість повітря близько 55% швидкість руху повітря – 0,1-0,2 м/сек. Рівні позитивних і негативних іонів у повітрі мають відповідати санітарно-гігієнічним нормам №2152-80. Допустима інтенсивність шуму на робочих місцях з ЕОМ має відповідати вимогам ДСанПіН 3.3.2-007-98: оптимальна — до 45 дБ, гранична — до 60 дБ.

Наступним нормативом створення сприятливих умов для зорової роботи, які б мінімізували втому очей, виникнення професійних захворювань та сприяли підвищенню продуктивності праці. Тому освітлення повинне відповідати вимогам ДБН В.2.5-28:2018 «Природне і штучне освітлення» [].

Основною вимогою є необхідність створення на робочій поверхні освітленості, що відповідає характеру зорової роботи і знаходиться в межах встановлених норм. Освітлення у приміщенні має бути суміщеним. Відтак, недостача денного природнього освітлення компенсується необхідною для приміщення кількістю штучного освітлення. Як джерело штучного освітлення в приміщеннях, де встановлено комп'ютерну техніку, бажано використовувати люмінесцентні лампи. Освітленість робочого місця у горизонтальній площині на висоті 0,8 м від рівня підлоги повинна бути не менше 400 лк. Для захисту від прямих сонячних променів повинні бути передбачені сонцезахисні пристрої, жалюзі, штори. Безпосередньо при виконанні роботи з обладнанням можливе використання місцевого освітлення в комбінації з загальним.

Також при роботі з електронними пристроями ЕОМ, потрібно дотримуватися наступних правил:

- перед початком роботи потрібно оглянути робоче місце на наявність пошкоджень, диму, неприємного запаху; перевірити правильність під'єднання обладнання до електричної мережі;
- виконувати роботу лише передбачену регламентом робіт;
- підтримувати порядок і чистоту робочого місця;
- устаткування, що використовується та працює від електромережі, повинне бути заземленим;
- при будь-яких випадках порушень роботи технічного обладнання або програмного забезпечення негайно викликати представника технічної служби з питань експлуатації обчислювальної техніки.

При дослідженні методів захисту від атак на web-сайти в задачах аналізу та обробки великих даних користуються лініями електромереж. Персональні комп'ютери і периферійні пристрої повинні підключатися до електромережі тільки за допомогою справних штепсельних з'єднань і електророзеток заводського виготовлення. В них, окрім контактів фазового та нульового робочого провідників, мають бути спеціальні контакти для підключення нульового захисного провідника. Конструкція вилки має бути такою, щоб приєднання нульового захисного провідника відбувалося раніше, ніж

приєднання фазового та нульового робочого провідників. Усі електроприлади, згідно з ДНАОП 0.00-1.21-98, повинні бути заземлені за допомогою нульового захисного провідника.

Заземлені конструкції, що знаходяться в приміщеннях, де розміщені робочі місця (батареї опалення, водопровідні труби, кабелі із заземленим відкритим екраном), мають бути надійно захищені діелектричними щитками або сітками з метою недопущення потрапляння працівника під напругу.

Під час монтажу та експлуатації ліній електромережі необхідно повністю унеможливити виникнення електричного джерела загоряння внаслідок короткого замикання та перевантаження проводів, обмежувати застосування проводів з легкозаймистою ізоляцією.

Приміщення мають бути оснащені системою автоматичної пожежної сигналізації і вогнегасниками відповідно до «Правил експлуатації та типових норм належності вогнегасників» від 23 лютого 2018 р. за N 225/31677. Проходи до засобів пожежогасіння мають бути вільними. Згідно техніки пожежної безпеки пристрої повинні розташовуватися не ближче одного метра від джерел тепла. Також на них не повинні падати прямі сонячні промені, щоб виключити можливість перегріву компонентів та вбудованих акумуляторів. Адже уже відомі приклади небезпек, які спричинили нагріті до критичних температур акумуляторні батареї.

## **6.2 Фактори ризику та можливі порушення здоров'я користувачів комп'ютерної мережі**

У зв'язку з бурхливим розвитком ІТ - технологій, засобів комп'ютерної техніки, мобільного зв'язку та електронного захисту інформації різко зросла, й надалі буде зростати, кількість областей і сфер діяльності людини, у яких використовуються інформаційні технології. Будь-яка професія характеризується певними професійними або обумовленими виробничими факторами захворюваннями. Професії, які пов'язані з застосуванням технічних засобів в ІТ– технологіях, не виключення.

«Комп'ютерні» захворювання, як і галузь, яка викликала їхню появу, ще досить молоді й мало вивчені. Так, при використанні комп'ютерів найбільшого ризику зазнають органи зору, скелетно-м'язова система, репродуктивна функція, центральна нервова система. Крім того, на користувачів комп'ютерів впливає цілий комплекс факторів малої інтенсивності, негативна дія яких проявляється поступово й потай. Тому захворювання проявляються лише після багатьох місяців або навіть років роботи, коли боротися з ними вже вкрай важко. У багатьох випадках важко навіть установити головну причину захворювань. За таких умов усе більш значимим і необхідним стає потреба формування в майбутніх фахівців відповідних знань щодо особливостей впливу несприятливих виробничих факторів на робочих місцях з інформаційними технологіями, заходів і засобів, спрямованих на мінімізацію такого впливу, збереження здоров'я й професійного довголіття користувачів.

Сукупність змін, що спостерігаються в стані здоров'я користувачів комп'ютерів, включає у себе захворювання органів зору, опорно-рухового апарату, центрально-нервової і серцево-судинної систем, шлунково-кишкового тракту, алергійні розлади. Виконання робіт із застосуванням комп'ютера, також може спричинити розвиток професійних захворювань периферичної нервової системи; захворювання кістково-м'язової системи та сполучної тканини, інші захворювання.

У вагітних відзначається ускладнення вагітності та пологів. У 80% працівників, які виконують роботу, що вимагає постійного напруження зору (а саме такою є робота на комп'ютері), відзначається прогресуюче зниження працездатності, що настає через 45-60 хв після початку роботи і поступово призводить до перевтоми, розладів центральної нервової та інших систем організму. У другій половині дня (іноді раніше) з'являється втома, головний біль, біль в очах та інші скарги, а швидкість переробки інформації зменшується на 25-34%, знижується стійкість ясного бачення на 40-52%. До кінця робочого дня збільшується частота серцевих скорочень і показники систолічного і діастолічного артеріального тиску.

Вже в перші роки комп'ютеризації було відзначено специфічне зорове стомлення у користувачів ЕОМ, яке дістало назву астенопія. За різними даними, частота випадків астенопії коливається від 40-92% (періодично) до 10-40% (щодня). Деякі симптоми астенопії можуть зберігатися і наступного дня після роботи з комп'ютером. Жінки частіше, ніж чоловіки, скаржаться на зоровий дискомфорт при роботі за комп'ютером. У жінок у віці 31-45 років астенопія виникає частіше, ніж у жінок 18-30 років, що свідчить про вплив стажу. Астенопія часто зустрічається у операторів з введення даних, прийому даних і діалогової роботи.

До 47% користувачів скаржаться на зорове стомлення, якщо робота без перерви триває менше 30 хв, і 66% — при безперервній роботі більше 30 хв. Очні симптоми більш виражені у тих осіб, що менше контролюють свою роботу, працюють з великим напруженням і відчують незадоволеність своєю працею. Більш чутливими до виникнення астенопії є люди з порушеннями зору.

Тривала робота у вимушеній робочій позі сидячи призводить до довготривалого статичного напруження одних і тих самих груп м'язів. Часто користувачі ЕОМ змушені сидіти у сутулій позі, яка характеризується максимальним згинанням у поперековому і грудному відділах хребта та нахилом тулуба вперед. Ця поза має низку несприятливих біомеханічних і фізіологічних наслідків, що призводить до виникнення болю у шийних і поперекових відділах хребта, остеохондрозу хребта, геморою, простатиту та інших захворювань. У підлітків під час тривалої роботи з комп'ютером може виникати сколіоз.

Усі ці негативні чинники під час роботи за комп'ютером можна поділити на такі основні групи:

- Чинники, що впливають на опорно-руховий апарат;
- Чинники, що впливають на сенсорні органи;
- Чинники, що впливають на психологічний стан [22].

Докладно про правильний вибір і конструкції обладнання, яке запобігає розвитку хронічних захворювань, займається окрема наука - ергономіка. Історія розвитку цієї науки починається з 1920. Передумовою для появи науки було

зростання кількість травматичних випадків при експлуатації техніки, але якщо раніше ці випадки списували на особистісні недоліки людини (халатність, погану пам'ять, концентрація) то з розвитком і ускладненням техніки почали замислюватися що причиною може бути неправильна організація трудових умов, що й прищепило до появи нової науки. Зараз ергономіка має кілька підрозділів, такі як соціальна інженерія, технічна естетика, мікроергономіці (розділом якої є «Юзабіліті») [22].

Найчастіше в користувачів ЕОМ виникають наступні порушення:

Неправильна постава — це призводить до подальшого розвитку викривлення хребта сколіозу, лордозу, кіфозу, а як результат головні болі, болі в області шиї і всього хребта, болі в області таза. Неправильно положення ніг може привести до артрити (запалення суглобів), артрозу (деформації).

Тунельний синдром - найвідоміше захворювання людей працюють за комп'ютером. Воно ж синдром зап'ястного каналу. Тунельний синдром проявляється після кількох годин напруженої роботи за комп'ютером. Синдроми - почуття «бігають мурашки" на кисті, біль пронизує кисть, оніміння кисті, важкість у руці, знесилення кисті. Тунельний синдром надає собою травму зап'ястя. Через зап'ястний канал між кістками (тунель) проходять серединний нерв і 9 сухожилів м'язів кисті. Серединний нерв забезпечує чутливість пальців, а також управляє м'язами, що забезпечують руху великого, вказівного і середнього пальців. Сам тунельний канал дуже вузький. Він стискається, защемляється серединний нерв. Під час частих, повторюваних рухів кистей рук в незручному положенні сухожилля труться об кістки зап'ястя і зв'язки. Постійно повторювані дрібні рухи пальцями призводять до внутрішніх мікротравм. Накопичуючись, вони і дають про себе знати в початковій стадії хвороби тремтінням, сверблячкою набряком і поколювання в пальцях [22].

Розвиток короткозорості — через те, що екран монітора за контрастом вище, ніж навколишні об'єкти розвивається короткозорість.

Порушення фокусування — наслідком напруженої роботи за монітором є порушення фокусування, яка може бути викликана перенапруженням очних м'язів.

Сухість очей — через рефлекс, заснований на тому, що при погляді на джерело світла очей починає менше моргати, виникає «обсушування» рогівки ока, яке призводить до очних болів.

Отже висновком є те, що комп'ютерні захворювання проявляються не відразу, тому їх початкові стадії важко діагностувати, а через їх особливості часто навіть з часом важко визначити точну причину захворювання. Проте дотримання трудових умов та правил роботи за ЕОМ допомагає знизити ризики цих захворювань, тому знання користувачами особливостей впливу несприятливих виробничих умов на робочих місцях допоможуть мінімізувати цей вплив та допомогти зберегти здоров'я й професійного довголіття користувачів.

робить ймовірність легкого підбору методів атаки дуже високою.

### **6.3 Висновки до розділу**

У даному розділі було описано, що проведені дослідження та розробка програмного забезпечення відбувалась з дотриманням основних правил користування комп'ютерною технікою, правил внутрішнього трудового розпорядку, правил особистої гігієни та гігієни приміщення, а працівники підприємств, де буде проводитися використання розробленої CMS для інтернет-магазину володіють знаннями з техніки безпеки, електробезпеки та пожежної безпеки, та проаналізовано загальні фактори ризику та можливі порушення здоров'я користувачів комп'ютерної мережі.



## РОЗДІЛ 7 ЕКОЛОГІЯ

### 7.1 Організаційні форми, види і способи статистичного спостереження в екології

Основою для будь-якої статистичної обробки та оцінки екологічної інформації є збір інформації, що здійснюється методом статистичного спостереження. Статичне спостереження - це одна з найважливіших стадій статистичного дослідження. Саме вона вважається фундаментом статистичного дослідження, тому що в процесі ститистичного дослідження здійснюється формується первинної статистичної інформації, яка на наступних етапах дослідження підлягає обробленню і аналізу.

Статичне спостереження в екології - це планомірний, науково-організований збір масових даних про екологічні явища і процеси. Здійснюється шляхом реєстрації за заздалегідь розробленою програмою спостереження [30].

Об'єктом спостереження є стан забруднення навколишнього середовища (природних об'єктів) атмосферного повітря, природних водних об'єктів, земель та ґрунтів. Збір даних проводиться не стихійно, а регулярно, що дає змогу вивчити тенденції, напрями, закономірності розвитку екологічних явищ і процесів. План статистичного спостереження передбачає широке коло питань методики та організації збору статистичної інформації, контролю її якості та вірогідності. Для об'єкта статистичного спостереження характерне те, що його не можна вивчати безпосередньо в цілому, для цього потрібно виділити в його складі окремі одиниці.

Одиниця статистичного спостереження - це складовий елемент об'єкта дослідження, який є основою рахунку і носієм істотних ознак та властивостей, які підлягають реєстрації. Це первинний елемент об'єкта дослідження. Одиницю спостереження встановлюють, виходячи із завдань спостереження і складності об'єкта дослідження.

Правильне визначення одиниці спостереження має істотне значення для організації і проведення статистичного дослідження. Цим значною мірою зумовлюється об'єктивність одержаних результатів [31].

Основне завдання статистичного спостереження - отримання вірогідних статистичних даних, які об'єктивно характеризують явища і процеси суспільного життя. Інформація статистичного спостереження повинна бути об'єктивною і якісною, а отже, забезпечуватись правильною науковою організацією її одержання, належним виконанням самого спостереження. Завдання статистичного спостереження зумовлюється завданнями, які ставляться перед дослідженням певних екологічних процесів і явищ і впливають з потреб управління ними [32].

Форми спостереження. У статистичній практиці застосовують дві організаційні форми спостереження: звітність і спеціально організовані статистичні спостереження.

Звітність — це форма статистичного спостереження, при якій статистичні дані надходять у статистичні органи від підприємств і установ у вигляді обов'язкових і таких, що мають юридичну силу звітів про їх роботу.

Звітність підприємств, установ та організацій є поки що основним джерелом статистичної інформації. У ній передбачається система твердо регламентованих показників, які характеризують діяльність підприємств, установ та організацій. Зміст звіту, форма і термін подання також встановлюється вищим статистичним органом. Звітність складають на основі документів первинного оперативно-технічного і бухгалтерського обліку. Вірогідність гарантується також юридичною відповідальністю керівників підзвітних підприємств та організацій. Перелік усіх форм із зазначенням їх реквізитів називають табелем звітності [33].

Другою за значенням організаційною формою спостереження є спеціально організоване статистичне спостереження. Застосовують його у випадках, коли не можна застосувати звітність або складати звітність нераціонально; коли необхідно детально вивчити явище поряд з вивченням його у формі звітності або потрібно перевірити вірогідність даних звітності.

За способом одержання статистичних даних виділяють: безпосередній облік фактів, документальний облік і опитування респондентів:

Різноманітність екологічних явищ, їх специфіка, особливості статистичного вимірювання потребують поєднання зазначених способів і видів спостереження.

## **7.2 Джерела електромагнітних полів, іонізуючих випромінювань і методи їх знешкодження**

Іонізуючим випромінюванням називається випромінювання, взаємодія якого з речовиною призводить до утворення у цій речовині іонів різного знаку. Іонізуюче випромінювання складається із заряджених та незаряджених частинок, до яких відносяться також фотони. Енергію частинок іонізуючого випромінювання вимірюють у позасистемних одиницях – електрон-вольтах.

Джерело іонізуючого випромінювання – це об'єкт, що містить радіоактивну речовину, або технічний пристрій, який створює або в певних умовах здатний створювати іонізуюче випромінювання. Пристрій для генерування іонізуючого випромінювання (нерадіонуклідне джерело) - це технічний пристрій (рентгенівська трубка, прискорювач, генератор і т. п.), в якому іонізуюче випромінювання виникає за рахунок зміни швидкості заряджених часток, їх анігіляції або ядерних реакцій.

Штучними джерелами електромагнітних випромінювань (ЕМВ) радіочастотного діапазону є потужні радіостанції, генератори надвисоких частот, установки індукційного і діелектричного нагрівання, радари, вимірювальні і контролюючі пристрої, дослідницькі установки, високочастотні прилади і пристрої, персональні комп'ютери, мобільники та інші гаджети.

Контроль рівнів ЕМП проводиться розрахунковими методами і (або) проведенням вимірів на робочих місцях. Розрахункові методи можуть використовуватися при проектуванні нових або реконструкції діючих об'єктів, що є джерелами ЕМВ. Для вже діючих джерел контроль рівнів ЕМВ здійснюється за допомогою інструментальних вимірювань з використанням засобів вимірювальної техніки спрямованого прийому (однокоординатний) і

приладів ненаправленого прийому, оснащених ізотропними (трикоординатними) антенними перетворювачами (антенами, датчиками). Допускається для діючих джерел ЕМВ також здійснювати контроль і розрахунковими методами. Вимірювання виконуються при роботі джерела ЕМВ з найбільшою використовуваною потужністю, в тому числі з максимальною. Контроль рівнів ЕМП повинен здійснюватися засобами вимірювань, які пройшли метрологічну атестацію та мають свідоцтво про метрологічну атестацію. Гігієнічна оцінка результатів вимірювань повинна здійснюватися з урахуванням похибки використовуваного засобу вимірювання. Забороняється проведення вимірювань при наявності атмосферних опадів, а також при температурі і вологості повітря, що виходять за граничні робочі параметри засобів вимірювань. Результати вимірювань слід оформляти у вигляді протоколу і (або) карти розподілу рівнів ЕМВ, суміщеної з планом розміщення обладнання або приміщення [34].

Оцінка результатів проводиться по санітарним нормам, правилам і гігієнічним нормативам. Вимірювання показників ЕМВ радіочастотного діапазону (30 кГц - 300 ГГц) на виробництві повинно проводитися при роботі всіх джерел випромінювання в характерному режимі. Якщо у виробничому приміщенні знаходяться і одночасно працюють джерела ЕМВ радіочастот, що випромінюють в діапазоні частот з різними гігієнічними нормативами, то вимір проводиться окремо в кожному діапазоні частот. Якщо при роботі джерел ЕМВ застосовуються захисні екрани, то вимірювання також повинні проводитися із застосуванням захисних екранів.

При проведенні вимірювань між джерелом випромінювання і приймальною антеною приладу не повинні знаходитися люди. Вимірювання нормованих параметрів проводяться на постійних робочих місцях. Вимірювання також проводяться в місцях можливого перебування персоналу в процесі роботи. При відсутності постійних робочих місць для проведення замірів вибирають кілька точок в межах робочої зони, де працівник проводить не менше 50% робочого часу. При проведенні вимірювань в діапазоні від 30 кГц до 300 МГц застосовують вимірювачі ближнього поля типу NFM-1, ПЗ-16, що

дозволяють виміряти напруженість електричного поля і напруженість магнітного поля. При проведенні даних вимірювань на частотах від 300 МГц до 300 ГГц застосовуються вимірювачі ГПЕ типу ПЗ-13, ПЗ-9Г. Прилади повинні бути повірені (мати чинне свідоцтво про повірку) і відкалібровані. Прилади використовуються відповідно до інструкцій по їх експлуатації [35].

Методи захисту від іонізуючого випромінювань являють собою комплекс заходів, що спрямовані на обмеження опромінення населення та запобігання виникнення як ранніх, так і віддалених наслідків опромінення.

Головними принципами захисту є:

- захист кількістю — розрахунок допустимої активності джерела випромінювання;
- захист відстанню — розрахунок допустимої відстані до джерела випромінювання;
- захист часом — розрахунок допустимого часу роботи із джерелом іонізуючого випромінювання;
- захист за допомогою екранування — розрахунок необхідної товщини захисного екрану;
- хімічні методи захисту — використання спеціальних фармацевтичних препаратів і сполук: радіопротекторів та радіоінгібіторів;
- захист культурою праці — дотримування правил техніки безпеки та особистої гігієни.

### **7.3 Висновки до розділу**

У даному розділі було описано організаційні форми, види і способи статистичного спостереження в екології, джерела електромагнітних полів, іонізуючих випромінювань та методи їх знешкодження.

## ВИСНОВКИ

У дипломній роботі магістра було досліджено популярні на українському ринку CMS для інтернет-магазинів, проаналізовано недоліки їх функціоналу та розроблено власну CMS. Проаналізовано актуальні вразливості web-сайтів та розроблено методи захисту від них. Отже в результаті роботи:

- Досліджено популярні CMS для роботи інтернет-магазинів на українському ринку, досліджено їх недоліки та розглянуто найбільш актуальні для вдосконалення модулі;

- Виконано інтеграцію сторонніх сервісів через API, які спрощують як взаємодію користувача з інтернет-магазином, так і взаємодію інтернет-магазину з службами доставки. Було розроблено сучасний модуль доставки з використанням API Укрпошти, для спрощення реєстрації було використано API Facebook, для модулю оплати було обрано платіжну систему LiqPay, для E-mail та СМС сповіщень сервіси SendGrid та SMSC.ua відповідно, API OneSignal для сповіщень користувачів у браузері та API Telegram для сповіщення про нові замовлення для менеджерів проекту.

- Проаналізовано популярні вразливості web-сайтів та атаки на них, та інтегровано програмні методи захисту, після чого було проведено тестування, які показали, що web-сайт захищений від даних типів атак;

- Проведено розрахунок норм часу на виконання науково-дослідної роботи, обґрунтовано економічну ефективність, описано проведення дослідження та розробки програмного забезпечення з дотриманням основних норм та правил, та описано організаційні форми, види і способи статистичного спостереження в екології, джерела електромагнітних полів, іонізуючих випромінювань та методи їх знешкодження.;

Розроблене програмне забезпечення у вигляді CMS для інтернет-магазину є сучасним, захищеним, конкурентноспроможним на українському ринку та активно використовується на практиці. Дана розробка буде актуальною для компаній, які займаються, або планують займатися, онлайн торгівлею у власному місті або за допомогою поштових відправлень по всій Україні.

## БІБЛІОГРАФІЯ

1. Seth F., Jeremiah G. XSS Attacks: Cross Site Scripting Exploits and Defense. –Syngress, 2011. – 480 с. - <http://univd.edu.ua/science-issue/issue/3346>
2. Mike Chan, Thor Technologies, Passwords vs. SSH keys – what’s better for authentication? <https://www.thorntech.com/2018/12/passwords-vs-ssh/>
3. SSH, The Secure Shell: The Definitive Guide 2nd (second) Edition by Daniel J. Barrett, Richard E. Silverman, Robert G. Byrnes published by O'Reilly Media (2005) Paperback – 1994 - 49 сторінка
4. Imperva cybersecurity - Distributed denial of service attack (DDoS) definition - <https://www.imperva.com/learn/application-security/ddos-attacks/>
5. The Cloudflare Blog - Alessandro Ghedini, Rustam Lalkaka - HTTP/3: the past, the present, and the future - <https://blog.cloudflare.com/http3-the-past-present-and-future/>
6. Passwords vs. Private Keys, Johannes Weber - <https://blog.webernetz.net/passwords-vs-private-keys/>
7. The Chromium Projects - QUIC, a multiplexed stream transport over UDP - <https://www.chromium.org/quic>
8. Cloudflare Security Solutions - <https://www.cloudflare.com/en-au/security/>
9. WP Engin digital experience platform - What 10 million passwords reveal about the people who choose them - <https://wpengine.com/unmasked/>
10. Cloudflare Security Solutions - Why is HTTP not secure? HTTP vs. HTTPS - <https://www.cloudflare.com/learning/ssl/why-is-http-not-secure/>
11. SameSite cookies explained, Rowan Merewood - <https://web.dev/samesite-cookies-explained/>
12. OWASP Top 10 2017 – Application Security Risks, Arden Rubens - <https://www.checkmarx.com/2017/12/03/closer-look-owasp-top-10-application-security-risks/>
13. PHP: htmlspecialchars() function - <https://www.w3resource.com/php/function-reference/htmlspecialchars.php>
14. ISO-8859-1 - <https://kb.iu.edu/d/aequ>

15. What is KVM?, Red Hat - <https://www.redhat.com/en/topics/virtualization/what-is-KVM>
16. HOTP: An HMAC-Based One-Time Password Algorithm, D. M'Raihi, M. Bellare, UCSD, F. Hoornaert, Vasco, D. Naccache, Gemplus, O. Ranen, Aladdin, December 2005 - <https://tools.ietf.org/html/rfc4226>
17. TOTP: Time-Based One-Time Password Algorithm, D. M'Raihi, Verisign, Inc., S. Machani, Diversinet Corp., M. Pei, Symantec, J. Rydell, Portwise, Inc., May 2011 - <https://tools.ietf.org/html/rfc6238>
18. Unrestricted File Upload, OWASP™ Foundation the free and open software security community - [https://www.owasp.org/index.php/Unrestricted\\_File\\_Upload](https://www.owasp.org/index.php/Unrestricted_File_Upload)
19. The MD5 Message-Digest Algorithm - MIT Laboratory for Computer Science and RSA Data Security, Inc., April 1992 <https://tools.ietf.org/html/rfc1321>
20. AT «Укрпошта» — єдиний національний оператор поштового зв'язку України - Про Укрпошту - <https://www.ukrposhta.ua/ua/pro-ukrposhtu>
21. Select2 - Select2 jQuery-based replacement for select boxes - <https://github.com/select2/select2>
22. SafeMySQL - PHP class for safe and convenient handling of MySQL queries - <https://github.com/colshrapnel/safemysql>
23. LiqPay - Правила формування запиту на проведення платежу - [https://www.liqpay.ua/documentation/uk/data\\_signature](https://www.liqpay.ua/documentation/uk/data_signature)
24. Telegram FAQ - <https://telegram.org/faq>
25. About webhooks - GitHub - <https://help.github.com/en/github/extending-github/about-webhooks>
26. LogDNA is a log management company - <https://logdna.com/about/>
27. DDoS Protection Attack Analytics and rapid response - Anupam Vij Senior Product Manager, Azure Networking - <https://azure.microsoft.com/en-us/blog/ddos-protection-attack-analytics-rapid-response/>
28. Pentestit Information Security — Brute-force attacks with Kali Linux - [https://medium.com/@Pentestit\\_ru/brute-force-attacks-using-kali-linux-49e57bb89259](https://medium.com/@Pentestit_ru/brute-force-attacks-using-kali-linux-49e57bb89259)



29. XSSer – Automated Web Pentesting Framework Tool to Detect and Exploit XSS vulnerabilities - Gurubaran - <https://gbhackers.com/xsser-automated-framework-detectexploit-report-xss-vulnerabilities/>
30. Тарасова В.В. - Екологічна статистика, 2008 – С.77 – 87.
31. Основи охорони праці: Навч. посіб. / Воронов І.О., Коваленко І.Д., Афанасьєв П.В., Булгач Т.В. – К.: Генеза, 2004. – С.96 – 116.
32. Гандзюк М.П., Желібо Є.П., Халімовський М.О. Основи охорони праці.- Київ: Вища освіта в Україні, 2013. – С.200 – 244.
33. Ильин Л.А., Кириллов В.Ф., Коренков И.П. Радиационная гигиена. – М., Медицина, 1999. – С.157-175.
34. Основні санітарні правила забезпечення радіаційної безпеки України. Наказ МОЗ України 02.02.2005 № 54.
35. Катренко Л.А., Кіт Ю.В., Пістун І.П. Охорона праці.- Суми.- 2009.- С.107 - 123.
36. SQL Injection with Kali Linux - Bima Fajar Ramadhan - <https://linuxhint.com/sql-injection-kali-linux/>
37. Protection against Cross-site scripting (XSS) via PHP - <https://github.com/voku/anti-xss>

## ДОДАТКИ

УДК 004.418  
**Б. Тригубець**

(Тернопільський національний технічний університет імені Івана Пулюя)

## **РОЗРОБКА CMS ТА МЕТОДІВ ЗАХИСТУ WEB-САЙТІВ НА ЇЇ ОСНОВІ**

UDC 004.418  
**B. Tryhubets**

(Ternopil Ivan Puluj National Technical University, Ukraine)

## **DEVELOPMENT OF CMS AND WEB SITE SECURITY METHODS**

Основну частину сучасного інформаційного простору та мережі інтернет вцілому складають web-сайти, саме на них генерується інформаційний контент та зберігаються дані про користувачів. Якщо доступ до даних, які зберігаються на web-ресурсах, та доступ до інструментів для їх генерації, отримують зловмисники, вони можуть бути використані в корисливих цілях та завдати великих фінансових та репутаційних збитків компанії, яка потрапила під атаку.

В цій магістерській роботі було проаналізовано основні загрози безпечному функціонуванню web-сайтів, розглянуто правила, яких потрібно дотримуватись при їх розробці, та при розробці інструментів, за допомогою яких web-сайт буде адмініструватися. Захист web-сайту можна умовно поділити на захист сервера, на якому він знаходиться, та захист програмної частини, яку використовує як рядовий користувач, так і адміністратор.

Було розглянуто приклади основних груп атак на серверну частину та запропоновано методи захисту від них:

- DDoS-атакам в реаліях сьогодення масштабів можна успішно протистояти лише за допомогою використання спеціалізованих сервісів, таких як Cloudflare. Висока пропускна здатність сервісу та детальний аналіз усіх джерел трафіку, який надходить на сайт, на основі використання власних алгоритмів допомагає вистояти навіть при потужних DDoS-атаках;

- Уникнути несанкціонованого доступу через FTP/SSH можливо використовуючи SSH-ключі, доступ через нестандартні порти в налаштуваннях серверу, налаштуваючи обмеження кількості невдалих спроб авторизації та блокуванню доступу до певних директорій, у яких зберігаються важливі файли системи, або ж дозволити доступ до них тільки з певного IP;

- Запобігти атаці «людина посередині» та витоку придатних для читання даних можна використовуючи для HTTP-з'єднання SSL-сертифікат, та нової версії протоколу HTTP/3;

Розглянуто основні груп атак на програмну частину та методи захисту від них:

- Вразливості неправильної обробки вхідних даних відкривають можливість для SQL-ін'єкцій та XSS-атак. Використання популярних функцій для екранування вхідного коду, перевірка усіх вхідних даних через форми та GET-запити, а також правильна робота з доступом до файлів Cookies допомагає ліквідувати ці вразливості.

- Недостатня аутентифікація та авторизація на web-сайті призводить до виконання критичних дій від імені адміністратора, та доступу рядових користувачів до важливих функцій у адмін-панелі. Використовуючи двохфакторну аутентифікацію з використанням додатку Google Authenticator з тимчасовим паролем, цю вразливість можна нейтралізувати, а розмежування прав доступу між користувачами допоможе уникнути проблеми людського фактору. Власна розробка допоможе уникнути використання стандартизованих методів атак, які доступні для інших популярних CMS.

Використовуючи розроблену CMS, у яку інтегровані вищенаведені методи захисту, можна суттєво зменшити загрозу несанкціонованого доступу до інформації, яка знаходиться на сайті, та забезпечити його стабільну роботу.