**Міністерство освіти і науки України**
**Національний технічний університет**
**«Дніпровська політехніка»**

**Інститут електроенергетики**
**Факультет інформаційних технологій**
**Кафедра безпеки інформації та телекомунікацій**

# ПОЯСНЮВАЛЬНА ЗАПИСКА
## кваліфікаційної роботи ступеню бакалавра

**студента** *Каліневича Данііла Олександровича*

**академічної групи** *125-20-1*

**спеціальності** *125 Кібербезпека*

**спеціалізації**[1]

**за освітньо-професійною програмою** *Кібербезпека*

**на тему** *Аналіз типів вразливостей вебсайту та обгрунтування способів захисту від них*

| **Керівники** | **Прізвище, ініціали** | **Оцінка за шкалою** | | **Підпис** |
|---|---|---|---|---|
| | | **рейтинговою** | **інституційною** | |
| кваліфікаційної роботи | доц. Ковальова Ю.В. | | | |
| розділів: | | | | |
| спеціальний | доц. Ковальова Ю.В. | | | |
| економічний | к.е.н., доц. Пілова Д.П. | | | |
| **Рецензент** | | | | |
| **Нормоконтролер** | ст. викл. Мєшков В.І. | | | |

**Дніпро**
**2024**

**ЗАТВЕРДЖЕНО:**
завідувач кафедри
безпеки інформації та телекомунікацій
_____ д.т.н., проф. Корнієнко В.І.

«_____»_____20___року

**ЗАВДАННЯ**
**на кваліфікаційну роботу**
**ступеня бакалавра**

**студенту** ***Каліневичу Данііlу Олександровичу*** **академічної групи** ***125-20-1***
(прізвище ім'я по-батькові) (шифр)

**спеціальності** *125 Кібербезпека*
(код і назва спеціальності)

**на тему** ***Аналіз типів вразливостей вебсайту та обгрунтування способів захисту від них***

затверджену наказом ректора НТУ «Дніпровська політехніка» від 23.05.2024 № 469-с

| Розділ | Зміст | Термін виконання |
|---|---|---|
| Розділ 1 | ***Розробка вебресурсу «Kalina Foundation» для підтримки благодійних внесків*** | 15.03.2024 |
| Розділ 2 | ***Етичний аудит безпеки вебресурсу «Kalina Foundation». Аналіз виявлення вразливостей та обгрунтування ефективних способів захисту від них*** | 10.05.2024 |
| Розділ 3 | ***Техніко-економічне обґрунтування проекту*** | 11.06.2024 |

**Завдання видано** _____ _____
(підпис керівника) (ім'я, прізвище)

**Дата видачі: 15.01.2024р.**

**Дата подання до екзаменаційної комісії: 28.06.2024р.**

**Прийнято до виконання** _____ **Данііл КАЛІНЕВИЧ**
(підпис студента) (ім'я, прізвище)

# РЕФЕРАТ

Пояснювальна записка: 78 с., 33 рис., 6 табл., 5 додатків, 19 джерел.

Об'єкт досліджень: вебсайт донатного фонду «Kalina Foundation», розроблений з використанням Django framework (бекенд) та React.js (фронтенд).

Предмет дослідження: типи вразливостей, характерні для сучасних вебсайтів, та методи захисту від них.

Мета кваліфікаційної роботи: практична розробка вебсайту з використанням актуальних технологій, проведення етичного хакінгу для виявлення потенційних вразливостей, обгрунтування ефектичних способів захисту від них.

В першому розділі (практична частина) детально описано процес створення вебсайту "Kalina Foundation". Розглянуто такі аспекти: вибір технологічного стеку, архітектура вебсайту, функціональні можливості, дизайн та інтерфейс користувача.

Другий розділ присвячено виявленню потенційних вразливостей сайту шляхом етичного хакінгу: методологія тестування, тестування на проникнення, документування результатів. проводиться глибокий аналіз виявлених вразливостей та пропонуються рішення для їх усунення: класифікація вразливостей, аналіз причин, розробка заходів безпеки, обґрунтування вибору.

Економічний розділ присвячено оцінці економічної доцільності розробки та захисту вебсайту «Kalina Foundation».

ВЕББЕЗПЕКА, ВРАЗЛИВОСТІ ВЕБСАЙТІВ, ЕТИЧНИЙ ХАКІНГ, SQL-INJECTION, XSS-АТАКИ, CSRF-АТАКИ, DJANGO, REACT.JS, ЗАХИСТ ІНФОРМАЦІЇ, КІБЕРБЕЗПЕКА.

# ABSTRACT

Explanatory note: 78 pp., 33 pic., 6 table, 5 app, 19 sources.

Object of research: website of the donation fund "Kalina Foundation", developed using the Django framework (backend) and React.js (frontend).

Research subject: types of vulnerabilities typical of modern websites and methods of protection against them.

The goal of the diploma project: practical development of a website using current technologies, ethical hacking to identify potential vulnerabilities, justification of effective ways to protect against them.

The first section (practical part) describes in detail the process of creating the "Kalina Foundation" website. Aspects covered include technology stack selection, website architecture, functionality, design, and user interface.

The second section is devoted to identifying potential site vulnerabilities through ethical hacking: testing methodology, penetration testing, documenting the results. an in-depth analysis of detected vulnerabilities is carried out and solutions are proposed for their elimination: classification of vulnerabilities, analysis of causes, development of security measures, justification of choices.

The economic section is devoted to the assessment of the economic feasibility of developing and protecting the Kalina Foundation website.

WEB SECURITY, WEBSITE VULNERABILITIES, ETHICAL HACKING, SQL INJECTION, XSS ATTACKS, CSRF ATTACKS, DJANGO, REACT.JS, INFORMATION PROTECTION, CYBER SECURITY.

# СПИСОК УМОВНИХ СКОРОЧЕНЬ

ІД – інформаційна діяльність;

API – Application Programming Interface (інтерфейс програмування застосунків);

CSRF – Cross-Site Request Forgery (підробка міжсайтових запитів);

DOM – Document Object Model (об'єктна модель документа);

HTML – HyperText Markup Language (мова розмітки гіпертексту);

HTTP – HyperText Transfer Protocol (протокол передачі гіпертексту);

OWASP – Open Web Application Security Project (проект з безпеки вебзастосунків);

REST – Representational State Transfer (передача репрезентативного стану);

SQL – Structured Query Language (структурована мова запитів);

XSS – Cross-Site Scripting (міжсайтовий скриптинг).

ЗМІСТ

с.

# ВСТУП

Сучасна епоха характеризується безпрецедентним зростанням ролі інформаційних технологій у всіх сферах людської діяльності. Вебсайти стали невід'ємною частиною бізнесу, державного управління, освіти, охорони здоров'я та багатьох інших галузей. Вони забезпечують доступ до інформації, комунікацію, електронну комерцію, соціальну взаємодію та багато інших можливостей. Однак, разом з розвитком вебтехнологій зростає і кількість загроз безпеці вебресурсів.

Вразливості вебсайтів можуть призвести до витоку конфіденційних даних, фінансових втрат, порушення роботи систем та інших негативних наслідків. Зловмисники використовують різноманітні методи атак, такі як міжсайтовий скриптинг (XSS), підробка міжсайтових запитів (CSRF), SQL-ін'єкції та інші, щоб отримати несанкціонований доступ до даних або порушити функціонування вебсайту.

Особливо актуальним є питання безпеки вебсайтів благодійних організацій, які збирають та обробляють персональні дані донорів, а також фінансову інформацію. Незахищеність таких сайтів може призвести до витоку даних, що може завдати шкоди репутації організації та підірвати довіру донорів.

У зв'язку з цим, розробка та підтримка безпечних вебсайтів є важливим завданням для сучасних організацій. Для забезпечення безпеки вебресурсів необхідно використовувати комплексний підхід, який включає як технічні, так і організаційні заходи.

Актуальність теми дослідження обумовлена необхідністю розробки ефективних методів захисту вебсайтів від різноманітних загроз. Особливо важливо дослідити вразливості, характерні для сучасних вебтехнологій, та розробити методи їх виявлення та усунення.

Наукова новизна одержаних результатів полягає у наступному:

- проведено комплексний аналіз вразливостей вебсайту, розробленого з використанням сучасних технологій Django та React.js.

- розроблено рекомендації щодо усунення виявлених вразливостей, які враховують специфіку благодійних організацій.
- проведено економічне обґрунтування запропонованих заходів захисту, що дозволяє оцінити їх ефективність та доцільність впровадження.

Практичне значення одержаних результатів полягає у можливості використання розроблених рекомендацій для підвищення безпеки вебсайтів благодійних організацій та інших вебресурсів, що обробляють конфіденційні дані.

# РОЗДІЛ 1. СТАН ПИТАННЯ. ПОСТАНОВКА ЗАДАЧІ

## 1.1 Вибір та обґрунтування технологічного стеку

Для розробки вебресурсу "Kalina Foundation" було обрано наступний технологічний стек:

– Django (бекенд): Високорівневий Python-фреймворк, що сприяє швидкій розробці складних вебзастосунків. Django надає готові компоненти для роботи з базами даних, автентифікацією, адмініструванням та іншими типовими задачами, що суттєво скорочує час розробки та підвищує надійність системи. Django – це високорівневий вебфреймворк з відкритим кодом, написаний на мові програмування Python і його головні перваги – це швидкість розробки, безпека, масштабованість, зрілість та велика спільнота, що забезпечує постійний розвиток фреймворку та наявність великої кількості документації.

– React.js (фронтенд): декларативна JavaScript-бібліотека з відкритим кодом, розроблена компанією Facebook, для створення користувацьких інтерфейсів. React дозволяє створювати динамічні та інтерактивні компоненти, що легко масштабуються та підтримуються. Використання компонентного підходу спрощує розробку складних інтерфейсів та покращує їх читабельність. Вибір React.js для розробки фронтенду вебресурсу "Kalina Fond" обумовлений його перевагами у продуктивності, гнучкості та масштабованості. React дозволив створити сучасний, інтерактивний та зручний користувацький інтерфейс, що забезпечує позитивний досвід взаємодії з сайтом.

– REST API: Архітектурний стиль для побудови вебсервісів, що забезпечує взаємодію між фронтендом та бекендом. REST API дозволяє розділити клієнтську та серверну логіку, що сприяє гнучкості та масштабованості системи. Клієнт відповідає за відображення даних та взаємодію з користувачем, а сервер – за зберігання даних, обробку логіки та надання ресурсів. Переваги використання REST API:

– Масштабованість: Відсутність стану дозволяє легко масштабувати REST API для обробки великої кількості запитів.

– Гнучкість: REST API не залежить від конкретної технології реалізації клієнта або сервера, що дозволяє використовувати різні мови програмування та платформи.

– Простота: REST API використовує стандартні методи HTTP та формати даних, що спрощує його розуміння та використання.

– Кешування: Підтримка кешування дозволяє зменшити навантаження на сервер та прискорити роботу застосунку.

– PostgreSQL (база даних): Потужна реляційна система керування базами даних (СКБД) з відкритим кодом. У вебресурсі "Kalina Foundation" PostgreSQL використовується як основна база даних для зберігання інформації про користувачів, благодійні внески, проекти та інші дані. Вибір PostgreSQL обумовлений її високою продуктивністю, надійністю, масштабованістю та підтримкою широкого спектру функцій, необхідних для роботи вебзастосунку.

– JWT Authentication (автентифікація): JSON Web Token – стандарт для створення токенів доступу, що використовуються для автентифікації та авторизації користувачів. JWT забезпечує безпечний та зручний механізм автентифікації, особливо в розподілених системах.

– Environment Variables (змінні середовища): Механізм для зберігання конфігураційних параметрів застосунку, таких як ключі доступу, паролі, URL бази даних тощо. Використання змінних середовища дозволяє відокремити конфігурацію від коду, що підвищує безпеку та спрощує розгортання застосунку на різних середовищах.

## 1.2 Архітектура вебресурсу

Архітектура вебресурсу "Kalina Foundation" побудована за принципом клієнт-серверної взаємодії з використанням REST API.

Бекенд (Django):

– Моделі: Опис структур даних, що зберігаються в базі даних PostgreSQL (користувачі, категорії речей (дрони, БПЛА, транспорт і т.д.), речі на які збираються гроші тощо).

– Серіалізатори: Перетворення моделей у формати, придатні для передачі через мережу (JSON).

– Представлення: Обробка запитів від клієнта, виконання бізнес-логіки, взаємодія з базою даних та повернення відповідей у вигляді серіалізованих даних.

– URL-маршрутизація: Визначення відповідності між URL-адресами та представленнями.

Фронтенд (React.js):

– Компоненти: Модулі, що описують частини інтерфейсу користувача (форми, кнопки, списки тощо).

– Стан: Зберігання даних, що відображаються в інтерфейсі, та їх оновлення у відповідь на дії користувача.

– Життєвий цикл: Керування процесом створення, оновлення та видалення компонентів.

– Взаємодія з API: Надсилання запитів до бекенду для отримання та оновлення даних.

REST API:

– Ендпоїнти: URL-адреси, що відповідають певним діям (отримання списку проектів, створення благодійного внеску тощо).

– Методи HTTP наведені у таблиці 1.1. Визначення типу операції:

Таблиця 1.1 - HTTP Методи Запиту

| № | Метод | Використання |
|---|---|---|
| 1 | GET | для запиту даних від вказаного ресурсу. |
| 2 | POST | для надсилання даних на сервер для створення/оновлення ресурсу. |
| 3 | HEAD | майже ідентичний GET, але без тіла відповіді. |
| 4 | DELETE | видаляє вказаний ресурс. |
| 5 | OPTIONS | описує параметри зв'язку для цільового ресурсу. |
| 6 | PUT | використовується для надсилання даних на сервер для створення/оновлення ресурсу. (змінює усі значення) |
| 7 | PATCH | використовується для надсилання даних на сервер для створення/оновлення ресурсу. (змінює частину значень) |

У проекті будуть вискористовуватись лише GET, POST, PATCH, HEAD та OPTIONS методи.

– Формати даних: Використання JSON для представлення даних, що передаються між клієнтом та сервером.

## 1.3 Проектування бази даних

Для зберігання даних вебресурсу "Kalina Foundation" використовується реляційна база даних PostgreSQL.

Основні таблиці:

– Користувачі: Зберігає інформацію про користувачів системи (логін, пароль, ім'я, електронна пошта тощо).

– Категорія проекта: Містить дані виду різних проектів (повна назва та коротка назва).

– Проекти: Зберігає інформацію про благодійні проекти (назва, опис, мета, категорія, зібрані кошти тощо).

Зв'язки між таблицями, рис. 1.1:

– Проекти можуть бути пов’язані з однією категорією (багато-до-одного).

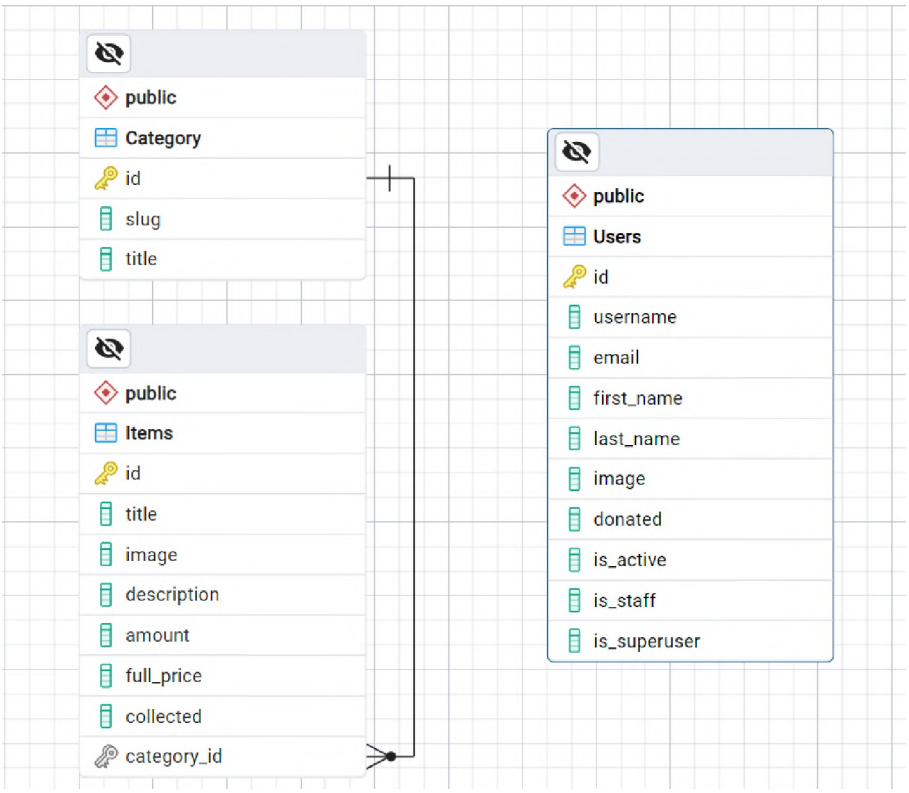



Рисунок 1.1 – Відображення зв’язків між таблицями

## 1.4 Автентифікація та авторизація

Для забезпечення безпеки вебресурсу "Kalina Foundation" використовується JWT-автентифікація.

Процес автентифікації:

– Користувач вводить логін та пароль.

– Бекенд перевіряє дані та генерує JWT-токен, що містить інформацію про користувача, яку можете побачити на рис. 1.2.

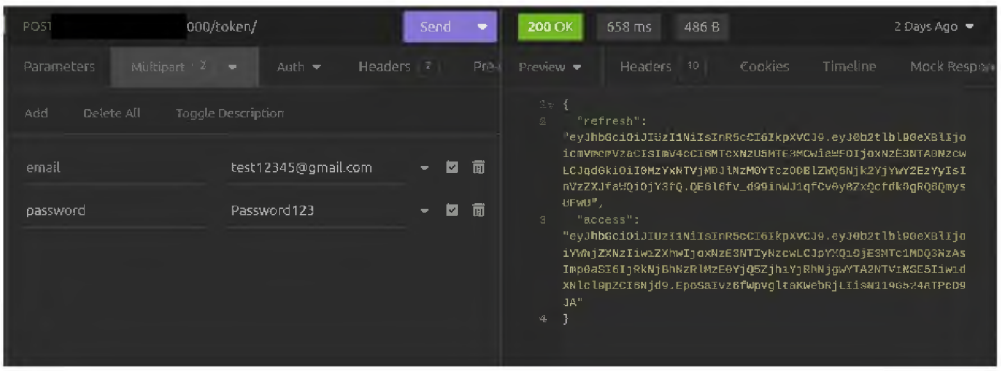



Рисунок 1.2 – Відповідь на POST запрос з параметрами email та password у додатку Insomnia

– Токен надсилається клієнту та зберігається у браузері.

– При кожному запиті до API клієнт надсилає токен у заголовку Authorization.

– Бекенд перевіряє токен та надає доступ до ресурсів, якщо користувач авторизований.

## 1.5 Використання змінних середовища

Змінні середовища використовуються для зберігання конфіденційних даних, таких як:

– SECRET_KEY: Секретний ключ Django, що використовується для підпису JWT-токенів та інших криптографічних операцій.

– DB_NAME: Назва бази даних PostgreSQL.

– DB_USER: Ім’я користувача, яке ваша програма Django використовуватиме для підключення до бази даних PostgreSQL.

– DB_PASSWORD: Пароль для облікового запису DB_USER.

– DB_HOST: Ім’я хоста або IP-адреса, де працює ваш сервер бази даних PostgreSQL.

– DB_PORT: Номер порту, який прослуховує ваш сервер бази даних PostgreSQL. Стандартним портом для PostgreSQL зазвичай є 5432.

Змінні середовища зберігаються в окремому файлі .env, який не відстежується системою контролю версій, що забезпечує безпеку конфіденційних даних.

## 1.6 Додаткові функціональні можливості бекенду

Окрім базових функцій, бекенд вебресурсу "Kalina Foundation реалізує ряд додаткових механізмів:

– Throttling (обмеження частоти запитів): Для запобігання зловживанням та захисту від DoS-атак впроваджено механізм обмеження частоти запитів до API. Користувачам дозволено здійснювати певну кількість запитів протягом певного інтервалу часу. Перевищення ліміту призводить до тимчасової блокування.

– Захист сторінки "Профіль": Доступ до сторінки "Профіль" надається лише авторизованим користувачам. Це забезпечується перевіркою наявності та валідності JWT-токена у кожному запиті до відповідного представлення.

Розглянемо більш детально функцію ProtectedRoute у Додатку А. Вона приймає у собі параметр children – це інша функція, у данному випадку це <Profile />. На початку створює змінну isAuthorized і setter для цієї змінної.

Коли прогружається цей компонент на сайті, вона викликає функцію useEffect, де встановлює що юзер не авторизований (у параметраї передається false).

Після цього створюється дві функції refreshToken та auth для оновлення токену та створення токену доступу (access_token) і токену оновлення (refresh_token). У функції auth перевіряємо чи є створений токен у Локальному

сховище, якщо немає, записуємо значення false у змінну isAuthorized, далі перевіряємо чи не закінчився термін дії токену доступу. Якщо дата закінчення роботи токена менша за сьогоднішню дату у форматі Unicode, то викликається функція оновлення токену – refreshToken.

У кінці перевірка, змінної isAuthorized, якщо вона дорівнює значенню true – то викликається параметр children, а якщо false – переходить на Логін сторінку.

Далі створюємо Route з Профіль сторінкою у головному файлі App.js з параметрами path – URL шлях на якому буде відображатись сторінка, та element – передаємо функцію ProtectedRoute з параметром – функція Profile.

```
<Route
        path={PROFILE_PAGE}
        element={
          <ProtectedRoute>
            <Profile />
          </ProtectedRoute>
        }
/>
```

– Pagination (пагінація): Для зручності роботи з великими обсягами даних (наприклад, списками благодійних внесків чи проектів) реалізовано механізм пагінації. Він дозволяє розділити дані на сторінки та відображати їх поступово, що зменшує навантаження на сервер та покращує користувацький досвід. У проекті на головній сторінці відображаються по три речі, на які можна задонатити.

## 1.7 Структура фронтенду

Фронтенд вебресурсу "Kalina Foundation", розроблений з використанням React.js, складається з наступних сторінок:

– Головна: Містить загальну інформацію про фонд, поточні проекти та заклик до дії (здійснити благодійний внесок), топ 5 донатерів, які зареєструвались на сайті, рис. 1.3-1.5.

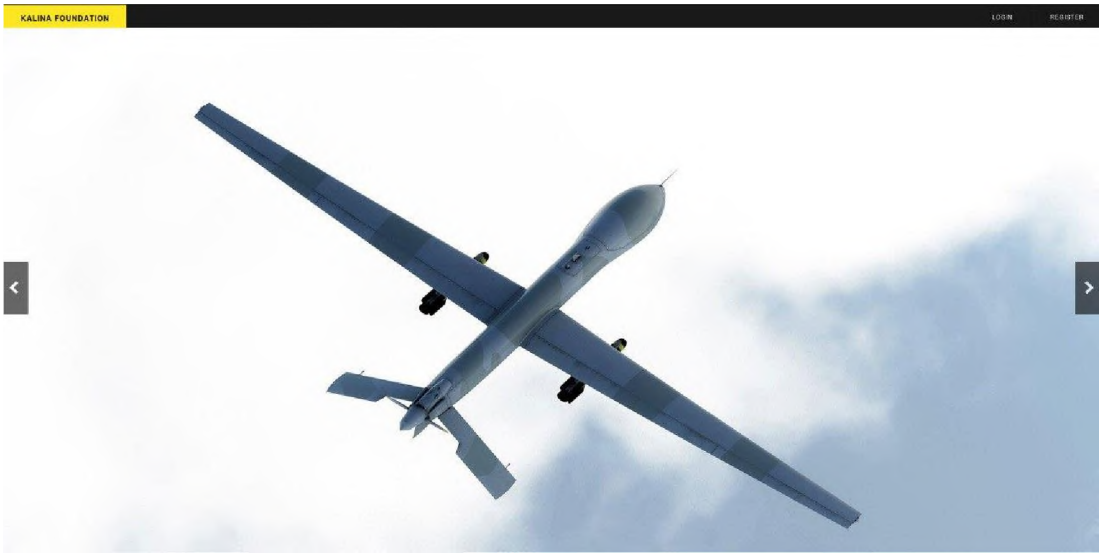



Рисунок 1.3 – Головна сторінка. Перемикання сторінок

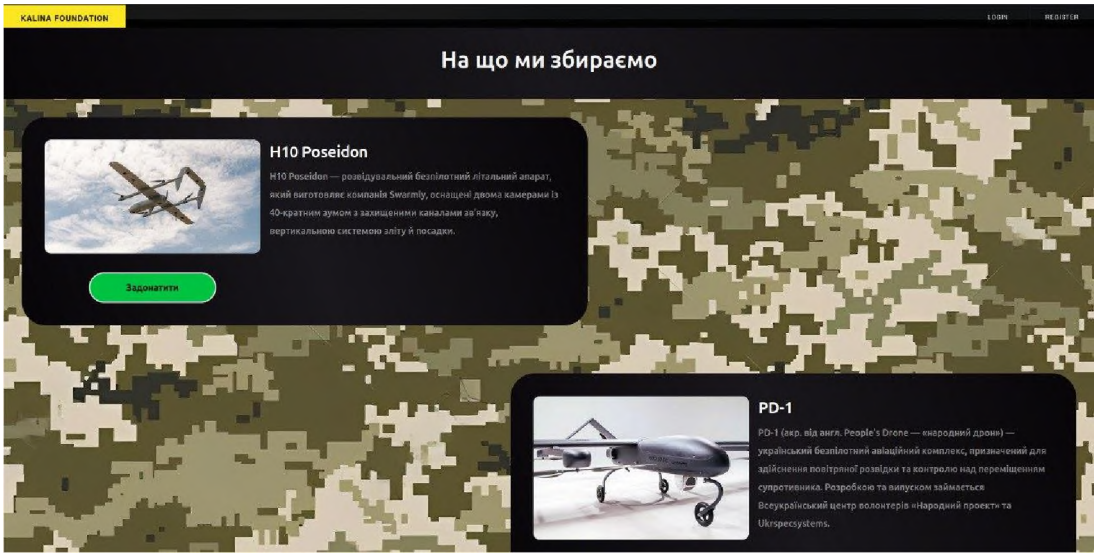



Рисунок 1.4 – Головна сторінка. Предмети на які можна задонатити

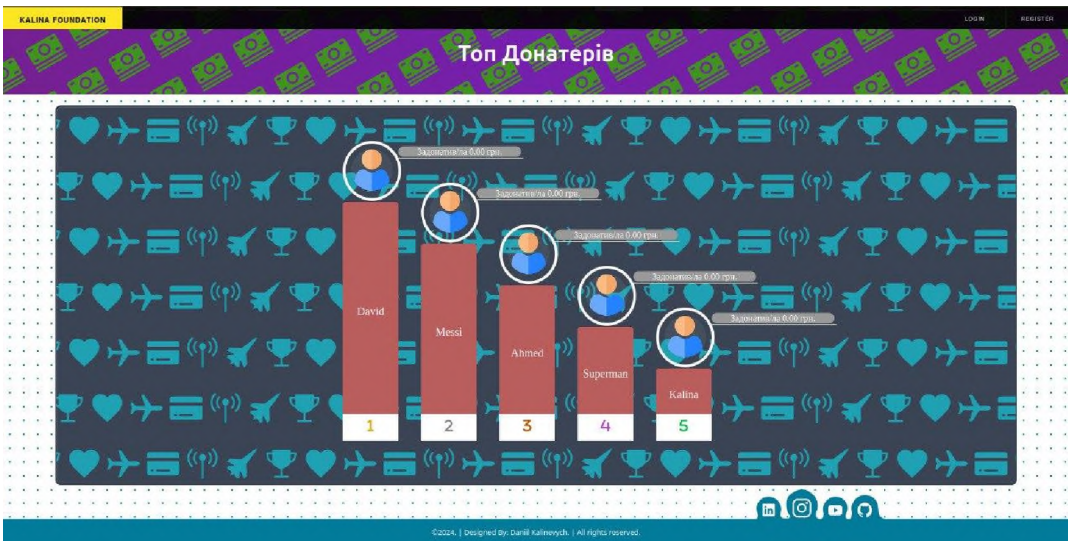



Рисунок 1.5 – Головна сторінка. Топ донатерів частина

– Донатна: Сторінка з детальною інформацією карток та посилання для здійснення благодійного внеску на рис. 1.6.

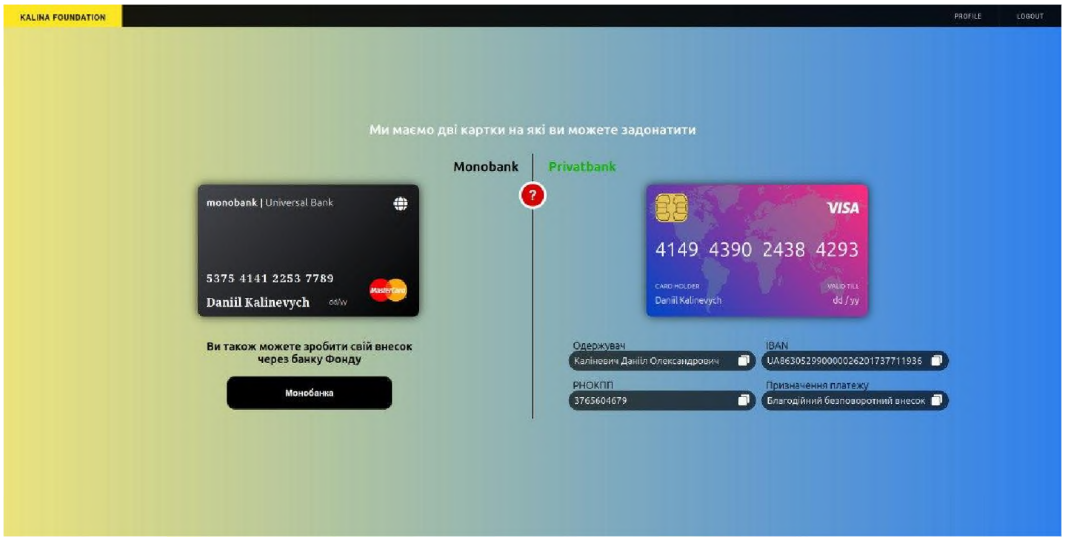



Рисунок 1.6 – Сторінка донату

– Логін: Форма для входу існуючих користувачів, рис. 1.7.

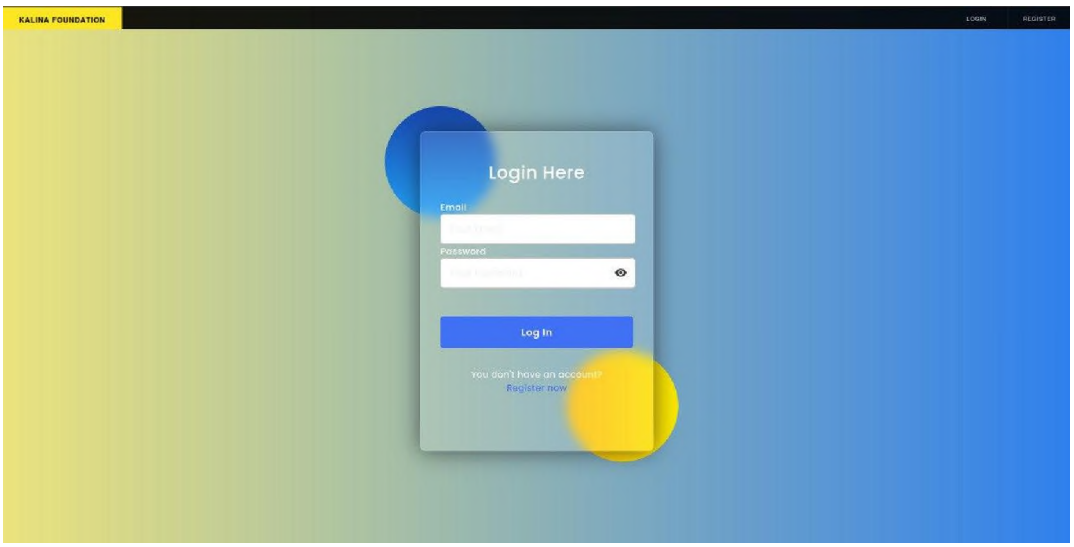



Рисунок 1.7 – Логін сторінка

– Реєстрація: Форма для створення нового облікового запису на рис. 1.8.

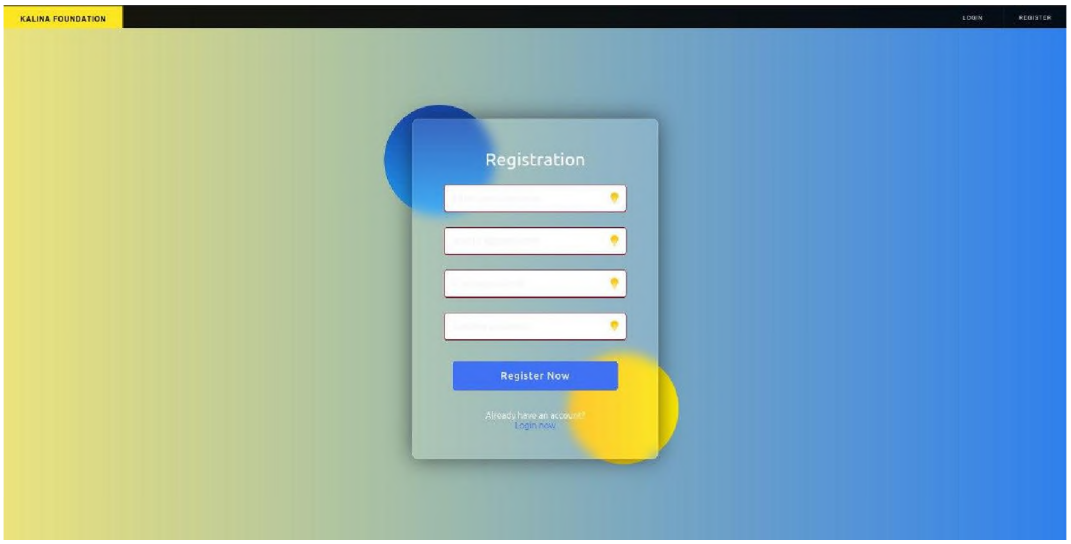



Рисунок 1.8 – Сторінка реєстрації

– Профіль: Особистий кабінет користувача, де він може редагувати профільні дані, рис. 1.9.

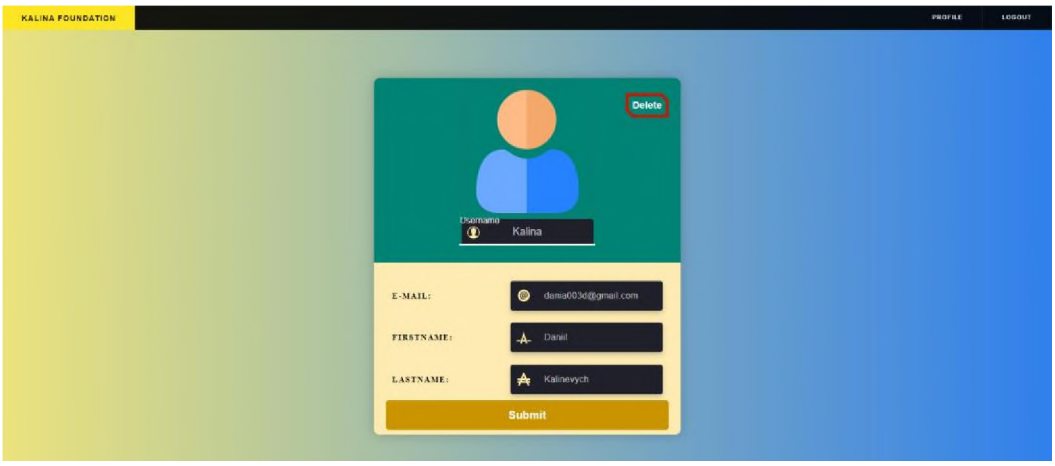



Рисунок 1.9 – Профіль сторінка

– Error 404: Кастомна сторінка на рис. 1.10 для обробки помилок 404 (сторінка не знайдена).

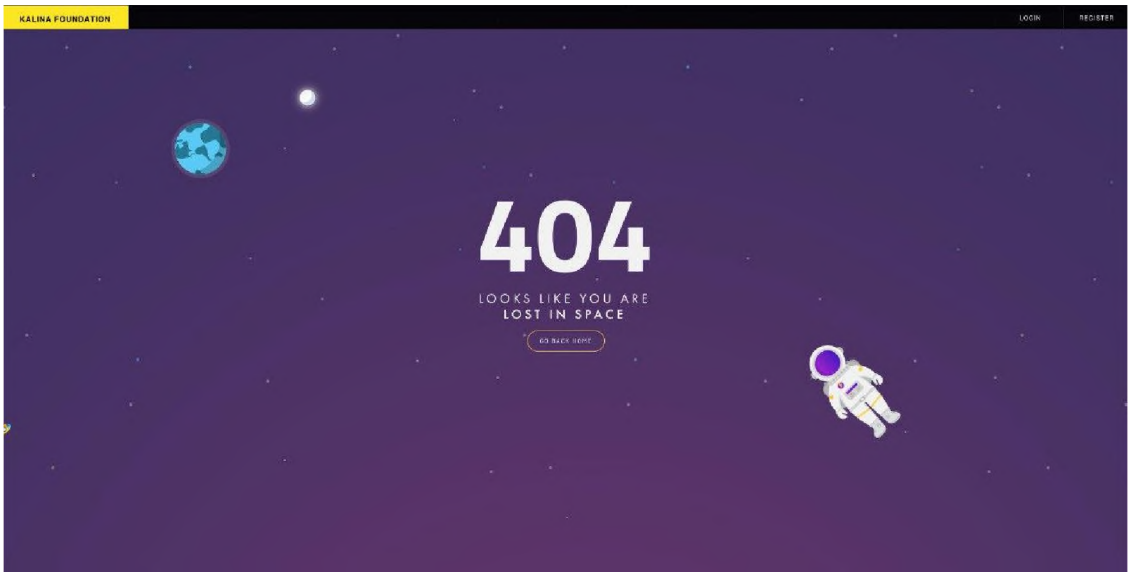



Рисунок 1.10 – Сторінка error 404

Для організації навігації між сторінками використовується бібліотека React Router, що надає компоненти Router, Route та Routes для визначення маршрутів та відображення відповідних компонентів.

## 1.8 Розробка та управління проектом

Для ефективної організації розробки та контролю версій вебресурсу "Kalina Fond" було використано систему управління версіями Git та платформу GitHub.

GitHub: Це вебсервіс для хостингу IT-проектів та їх спільної розробки. Він надає зручний інтерфейс для роботи з репозиторіями Git, дозволяє відстежувати зміни в коді, створювати гілки, об'єднувати зміни, вирішувати конфлікти та багато іншого. Крім того, GitHub надає інструменти для спілкування між розробниками, такі як issue tracker, pull requests та wiki.

Choreo: Для розгортання та тестування вебресурсу було обрано Choreo – хмарну платформу, що надає безкоштовний хостинг для невеликих проектів. Choreo дозволяє швидко та легко розгорнути застосунок, автоматизувати процес розгортання, моніторити його роботу та масштабувати ресурси за потреби.

Процес розробки:

1. Створення репозиторію на GitHub: Було створено публічний репозиторій на GitHub для зберігання коду вебресурсу "Kalina Foundation", рис. 1.9.

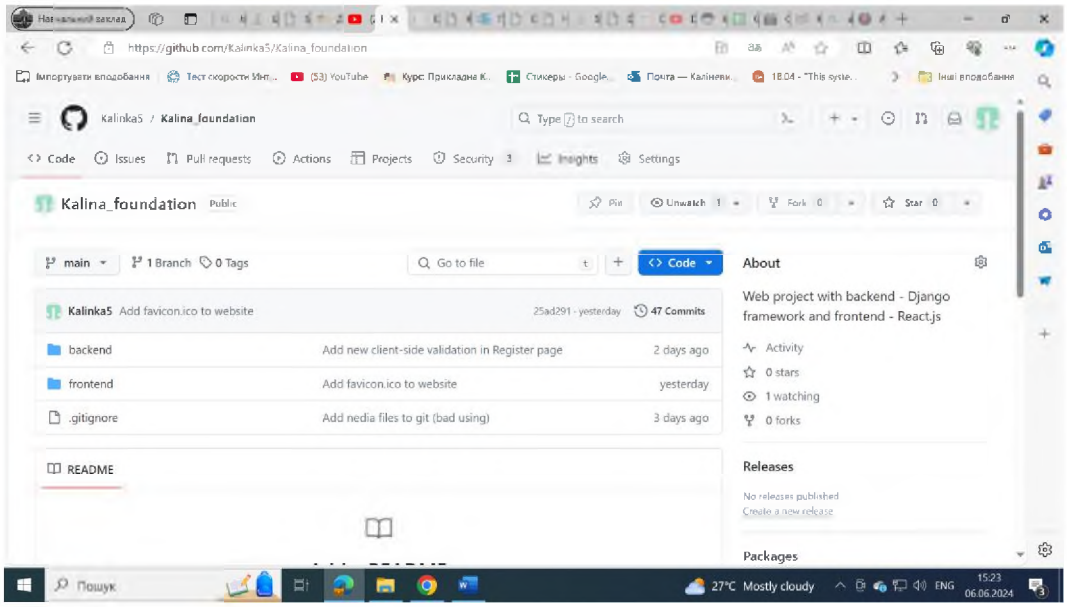



Рисунок 1.9 – GitHub репозиторій фондового застосунку

2. Розробка та тестування: Розробка вебресурсу велася локально, з використанням Git для контролю версій. Після завершення кожного етапу розробки зміни фіксувалися в репозиторії Git та відправлялися на GitHub.

3. Розгортання на Choreo: За допомогою Choreo було налаштовано автоматичне розгортання застосунку з репозиторію GitHub. При кожному оновленні коду в гілці main Choreo автоматично розгортає нову версію застосунку на своєму сервері.

4. Тестування на Choreo: Після розгортання застосунок тестувався на середовищі Choreo для перевірки його працездатності та відповідності вимогам.

5. Виправлення помилок та оновлення: Виявлені помилки виправлялися, а код оновлювався у репозиторії GitHub. Після цього Choreo автоматично розгортав оновлену версію застосунку.

6. Придбання доменного імені: На платформі Namecheap було придбано доменне ім'я `www.kalina-fond.com` на один рік. Це дозволило створити унікальну адресу для вебсайту та забезпечити його легку запам'ятовуваність для користувачів.

7. Налаштування SSL-сертифіката: Для забезпечення безпеки передачі даних між браузером користувача та сервером було встановлено SSL-сертифікат (Secure Sockets Layer). Це дозволяє шифрувати трафік та захищати його від перехоплення та підміни.

Використання Git та GitHub дозволило ефективно організувати процес розробки, відстежувати зміни в коді, співпрацювати з іншими розробниками та зберігати історію розробки. Завдяки використанню Namecheap та Choreo вдалося швидко та ефективно розгорнути вебресурс "Kalina Fond" у мережі Інтернет, забезпечивши його доступність для користувачів з усього світу та високий рівень безпеки передачі даних.

## 1.9 Висновки до розділу 1

У цьому розділі було визначено практичну реалізацію вебресурсу "Kalina Fond". Було описано вибір та обґрунтування технологічного стеку, архітектуру вебресурсу, проектування бази даних, механізми автентифікації та авторизації, а також використання змінних середовища для зберігання конфіденційних даних. Крім того, було розглянуто додаткові функціональні можливості бекенду, такі як throttling, захист сторінки "Профіль" та пагінація, а також структуру фронтенду з використанням React Router.

# РОЗДІЛ 2. СПЕЦІАЛЬНИЙ РОЗДІЛ

## 2.1. Збір інформації (розвідка)

Збір інформації, також відомий як розвідка, є першим критичним етапом етичного хакерства. Цей етап передбачає збір якомога більшої кількості інформації про цільову систему для виявлення потенційних векторів атак і вразливостей. Мета полягає в тому, щоб збирати дані без безпосередньої взаємодії з ціллю, таким чином мінімізуючи ризик виявлення та гарантуючи, що розвідувальні дії не заважатимуть операціям цілі.

Збір інформації можна розділити на дві основні категорії: пасивна розвідка та активна розвідка. Кожна категорія використовує різні методи та інструменти для збору цінних даних про цільову систему.

### 2.1.1 Пасивна розвідка

1. WHOIS Lookup

WHOIS – це протокол, який використовується для запитів до баз даних, у яких зберігаються зареєстровані користувачі або правонаступники Інтернет-ресурсу, наприклад доменного імені, блоку IP-адреси або автономної системи. База даних WHOIS містить інформацію про право власності на домен, реєстраційні дані та контактну інформацію.

Було проведено пошук WHOIS для домену www.kalina-fond.com, і результати представлені в чотирьох ключових розділах: інформація про реєстратора (Рис. 2.1), важливі дати та назви серверів (Рис. 2.2), дані реєстратора (Рис. 2.3), а також Записи DNS (Рис. 2.4).

Перший розділ результатів пошуку WHOIS, можете побачити на Рис. 2.1, містить відомості про реєстратора, відповідального за керування реєстрацією домену. Для www.kalina-fond.com інформація про реєстратора наступна:

Реєстратор: NAMECHEAP INC.

Реферальне посилання: http://www.namecheap.com

NAMECHEAP INC. є відомим реєстратором доменів, що надає послуги реєстрації доменів і хостингу. Ця інформація підтверджує, що домен www.kalina-fond.com був придбаний через Namecheap.

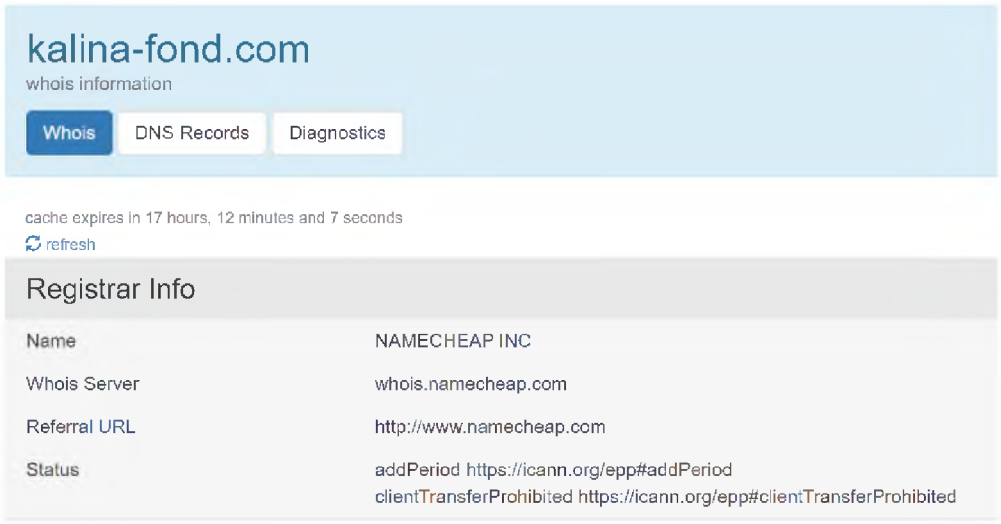



Рисунок 2.1 – Інформація про реєстратора

У другому розділі детально описано важливі дати, пов’язані з реєстрацією домену, і сервери імен, які розпізнають домен, Рис. 2.2:

Термін придатності: 2025-06-11

Дата реєстрації: 2024-06-11

Дата оновлення: 0001-01-01

Назви серверів:

1. dns1.registrar-servers.com
2. dns2.registrar-servers.com

Ці дати вказують, коли було зареєстровано домен, коли його термін дії закінчується, а також час останнього оновлення реєстраційних даних. Дата реєстрації має вирішальне значення для розуміння віку та легітимності домену, тоді як дата закінчення терміну дії має важливе значення для планування

поновлення, щоб уникнути збоїв у обслуговуванні. Сервери імен вказують на сервери, відповідальні за перетворення доменного імені на відповідну IP-адресу.

| Important Dates | |
|---|---|
| Expires On | 2025-06-11 |
| Registered On | 2024-06-11 |
| Updated On | 0001-01-01 |
| **Name Servers** | |
| dns1.registrar-servers.com | 156.154.132.200 |
| dns2.registrar-servers.com | 156.154.133.200 |

Рисунок 2.2 – Важливі дати та назви серверів

Третій розділ пошуку WHOIS показує дані реєстратора, які зазвичай включають адресу, контактну інформацію, e-mail та ін. Для www.kalina-fond.com ці дані захищені конфіденційністю, Рис. 2.3, та відображаються таким чином:

Ім’я реєстранта: Redacted for Privacy

Організація: Privacy service provided by Withheld for Privacy ehf

Адреса: Kalkofnsvegur 2

Місто: Reykjavik

Штат/провінція: Capital Region

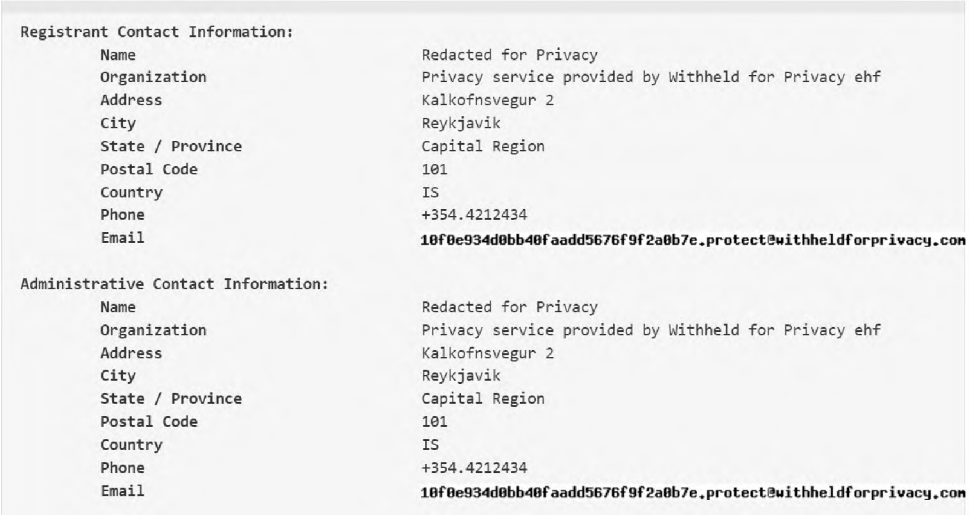
Поштовий індекс реєстранта: 101

Країна: IS

Телефон: +354.4212434

Електронна адреса:
10f8e934d0bb40bb48faadf4d5s4fd5sf46d.protect@withheldforprivacy.com

Використання реєстратором послуг із захисту конфіденційності гарантує, що фактичні контактні дані власника домену не є загальнодоступними, що підвищує безпеку та конфіденційність власника домену.



```
Registrant Contact Information:
        Name                    Redacted for Privacy
        Organization            Privacy service provided by Withheld for Privacy ehf
        Address                 Kalkofnsvegur 2
        City                    Reykjavik
        State / Province        Capital Region
        Postal Code             101
        Country                 IS
        Phone                   +354.4212434
        Email                   10f0e934d0bb40faadd5676f9f2a0b7e.protect@withheldforprivacy.com

Administrative Contact Information:
        Name                    Redacted for Privacy
        Organization            Privacy service provided by Withheld for Privacy ehf
        Address                 Kalkofnsvegur 2
        City                    Reykjavik
        State / Province        Capital Region
        Postal Code             101
        Country                 IS
        Phone                   +354.4212434
        Email                   10f0e934d0bb40faadd5676f9f2a0b7e.protect@withheldforprivacy.com
```

Рисунок 2.3 – Дані реєстратора

У четвертому розділі пошуку WHOIS (Рис. 2.4) детально описуються записи DNS, пов'язані з доменом. Записи DNS надають важливу інформацію про те, як домен зіставляється та керується в Інтернеті. Для www.kalina-fond.com виявлено наступні записи DNS:

– Запис A (запис адреси) зіставляє домен із відповідною IP-адресою: www.kalina-fond.com – 20.166.183.113

– Запис CNAME (запис канонічного імені) – псевдонім одного імені до іншого: пошук DNS продовжиться шляхом повторної спроби пошуку з новим ім'ям: www.kalina-fond.com

CNAME customdns.e1-eu-north-azure.choreoapps.dev

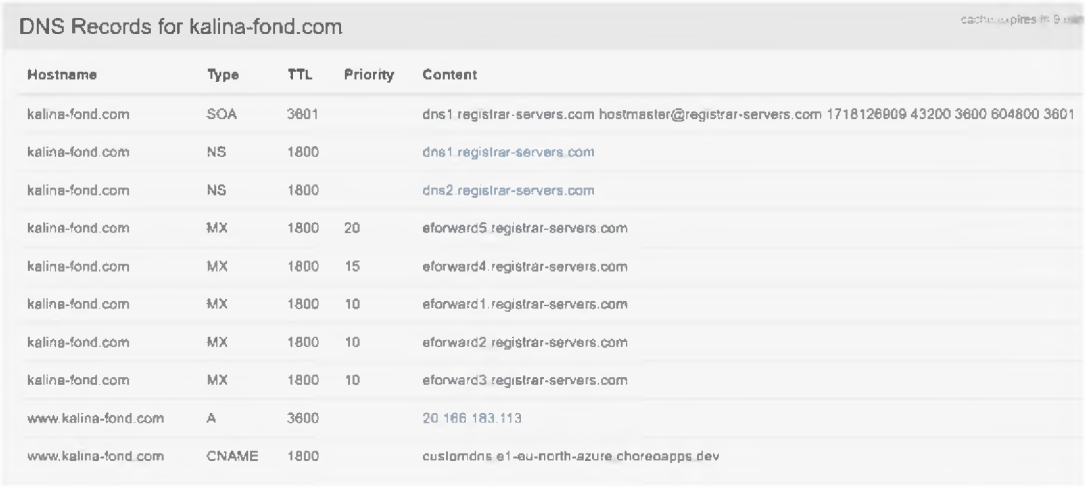
– Запис SOA (початок повноважень) містить адміністративну інформацію про домен, включаючи основний сервер імен, електронну адресу відповідальної сторони та різні таймери: SOA dns1.registrar-servers.com hostmaster@registrar-servers.com 1718126909 43200 3600 604800 3601

– Запис NS (запис сервера імен) делегує зону DNS для використання вказаних авторитетних серверів імен:

dns1.registrar-servers.com dns2.registrar-servers.com

– Запис MX (запис обміну поштою) направляє електронну пошту на сервери домену: MX eforward5.registrar-servers.com

Ці записи DNS надають повне уявлення про те, як домен www.kalina-fond.com налаштований і керований ним. Значення CNAME показує, що сайт зібраний на платформі Choreo.



DNS Records for kalina-fond.com

| Hostname | Type | TTL | Priority | Content |
|---|---|---|---|---|
| kalina-fond.com | SOA | 3601 | | dns1.registrar-servers.com hostmaster@registrar-servers.com 1718126909 43200 3600 604800 3601 |
| kalina-fond.com | NS | 1800 | | dns1.registrar-servers.com |
| kalina-fond.com | NS | 1800 | | dns2.registrar-servers.com |
| kalina-fond.com | MX | 1800 | 20 | eforward5.registrar-servers.com |
| kalina-fond.com | MX | 1800 | 15 | eforward4.registrar-servers.com |
| kalina-fond.com | MX | 1800 | 10 | eforward1.registrar-servers.com |
| kalina-fond.com | MX | 1800 | 10 | eforward2.registrar-servers.com |
| kalina-fond.com | MX | 1800 | 10 | eforward3.registrar-servers.com |
| www.kalina-fond.com | A | 3600 | | 20.166.183.113 |
| www.kalina-fond.com | CNAME | 1800 | | customdns.e1-eu-north-azure.choreoapps.dev |

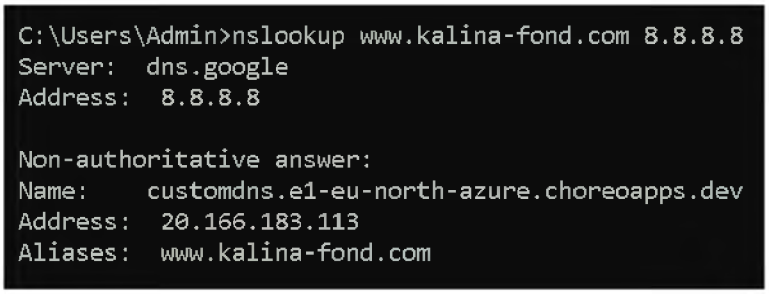
Рисунок 2.4 – Записи DNS для kalina-fond.com

2. Nslookup

nslookup – це інструмент командного рядка адміністрування мережі, який використовується для запиту до системи доменних імен (DNS) для отримання інформації про відображення імені домену або IP-адреси. Тут розглядаємо використання двох конкретних команд nslookup на www.kalina-fond.com.

– nslookup www.kalina-fond.com 8.8.8.8

Ця команда запитує DNS-сервер за IP-адресою 8.8.8.8 (Google Public DNS) для отримання інформації про домен www.kalina-fond.com. Результати можете побачити на Рис. 2.5:



```
C:\Users\Admin>nslookup www.kalina-fond.com 8.8.8.8
Server:  dns.google
Address:  8.8.8.8

Non-authoritative answer:
Name:    customdns.e1-eu-north-azure.choreoapps.dev
Address:  20.166.183.113
Aliases:  www.kalina-fond.com
```

Рисунок 2.5 – Значення CNAME та IP адреси

– nslookup -type=any www.kalina-fond.com 8.8.8.8

Ця команда запитує DNS-сервер за IP-адресою 8.8.8.8 щодо всіх типів DNS-записів, пов'язаних із www.kalina-fond.com. Результат можете побачити на Рис. 2.6:

```
C:\Users\Admin>nslookup -type=any  www.kalina-fond.com 8.8.8.8
Server:  dns.google
Address:  8.8.8.8

Non-authoritative answer:
www.kalina-fond.com     canonical name = customdns.e1-eu-north-azure.choreoapps.dev
www.kalina-fond.com     internet address = 162.255.119.34
```

Рисунок 2.6 – Значення CNAME та інтернет адреси

Різниця IP-адрес у результатах лише у тому, що перша адреса – IP-адреса, пов'язана з CNAME, а друга – IP-адреса, пов'язана з доменом.

3. Whatweb.net

Використовуючи такий онлайн-сервісів, як Whatweb.net, знвйшли додаткову інформацію про цільовий вебсайт, включаючи деталі його хостингу та розташування сервера. Whatweb.net визначив, що країною розміщення є Сполучені Штати, підтверджуючи місцезнаходження сервера та надаючи додатковий контекст щодо географічних аспектів інфраструктури домену, Рис. 2.7.

```
http://www.kalina-fond.com/ [308 Permanent Redirect] Country[UNITED STATES][US],
 IP[20.166.183.113],
 RedirectLocation[https://www.kalina-fond.com],
 Title[308 Permanent Redirect]
https://www.kalina-fond.com [200 OK] Cookies[route],
 Country[UNITED STATES][US],
 HTML5, HttpOnly[route],
 IP[20.166.183.113],
 Script, Strict-Transport-Security[max-age=31536000; includeSubDomains],
 Title[Kalina Foundation],
 UncommonHeaders[x-envoy-upstream-service-time,x-keda-http-cold-start]
```

Рисунок 2.7 – Знаходження країни хоста

### 2.1.2 Активна розвідка

Активна розвідка передбачає безпосередню взаємодію з цільовою системою для збору детальної інформації. Цей метод несе більший ризик виявлення порівняно з пасивною розвідкою, але може надати більш точні та обширні дані. Під час цієї фази використовуються різні інструменти та методи для дослідження цільової системи, виявлення відкритих портів і збору інформації про служби, що працюють на цих портах.

Сканування портів за допомогою Zenmap

Zenmap – це графічний інтерфейс користувача (GUI) для Nmap, потужного та широко використовуваного інструменту мережевого сканування. За допомогою Zenmap можна проводити кілька типів сканування, щоб зібрати повну інформацію про ціль. Далі зосередимося на інтерпретації результатів інтенсивного сканування цільового сайту www.kalina-fond.com.

Інтерпретація результатів інтенсивного сканування:

- Підсумок сканування:
  - Версія Nmap: 7.95
  - Час початку сканування: 2024-06-13 10:22
  - Хост: www.kalina-fond.com (20.166.183.113)
  - Статус хоста: підвищився (затримка 0,060 с)
  - Відстань мережі: 24 хопи
  - Загальна кількість сканованих портів: 1000
  - Відкриті порти: 2 (80/tcp, 443/tcp)
  - Відфільтровані порти: 998 (немає відповіді)
- Відкриті порти та служби:
  - Порт 80/tcp:
    - Стан: відкритий

- Сервіс: HTTP
- Версія служби: nginx (зворотний проксі)
- Заголовок HTTP: 308 Постійне перенаправлення
- Підтримувані методи: GET, HEAD, POST, OPTIONS

- Порт 443/tcp:
  - Стан: відкритий
  - Сервіс: SSL/HTTP
  - Версія служби: nginx (зворотний проксі)
  - HTTP-заголовок: Фонд Калина
  - Підтримувані методи: GET, HEAD
  - Сертифікат SSL:
    - Загальна назва: www.kalina-fond.com
    - Емітент: Загальна назва: R10, Організація: Let's Encrypt, Країна: США
    - Діє з: 2024-06-11
    - Дійсний до: 2024-09-09
    - Тип відкритого ключа: RSA
    - Біти відкритого ключа: 2048
    - Алгоритм підпису: sha256WithRSAEncryption
    - Сертифікат MD5 Відбиток: 9ac8:e083:3aa9:4141:4aca:d2a5:1ba0:7d56
    - Відбиток SHA-1 сертифіката: c4e2:ec9b:69ff:b3d3:2cb5:8349:eeda:8260:6f40:294a

- Traceroute:

- Traceroute вказує на те, що ціль знаходиться на відстані 24 стрибків, показуючи шлях, який пройшли пакети, щоб досягти мети.

- Виявлення ОС:

  - Статус виявлення ОС: Ненадійний

  - Причина: відсутній закритий TCP-порт, що робить результати неповними.

  - Примітка. Збігів ОС для хоста не знайдено через відсутність достатньої кількості даних.

- Результати Nmap Script Engine (NSE):

  - Сценарії до та після сканування: завантажено та виконано кілька сценаріїв для збору додаткової інформації.

  - Підтримувані методи HTTP: визначено підтримувані методи HTTP (GET, HEAD, POST, OPTIONS для HTTP та GET, HEAD для HTTPS).

- Інформація SSL/TLS:

  - Деталі сертифіката SSL надають інформацію про використане шифрування та термін дії сертифіката, що важливо для розуміння стану безпеки вебсервера.

Результати сканування показують, що на цільовому домені www.kalina-fond.com працює вебсервер nginx із відкритими службами HTTP та HTTPS. Ось кілька ключових висновків:

- Наявність відкритих портів 80 і 443 свідчить про те, що вебсервер доступний як через HTTP, так і через HTTPS.

- Сервер nginx діє як зворотний проксі, що може означати, що він пересилає запити на інший внутрішній сервер.

- Сертифікат SSL від Let's Encrypt показує, що HTTPS налаштовано належним чином, але важливо регулярно перевіряти та оновлювати сертифікат для підтримки безпеки.

– Служба HTTP переспрямовує на іншу сторінку (постійне перенаправлення 308), що може означати перехід на HTTPS або іншу URL-адресу.

– Назва служби HTTPS «Kalina Foundation» передбачає основний вміст, який обслуговує вебсайт.

– Деталі сертифіката SSL дають уявлення про заходи безпеки, які застосовуються для з'єднань HTTPS. Використання Let's Encrypt вказує на прагнення захищати комунікації.

## 2.2 Оцінка вразливості

Оцінка вразливості є вирішальним кроком у виявленні та зменшенні потенційних ризиків безпеки в системі. Цей процес передбачає систематичне дослідження цільової системи на відомі вразливості за допомогою спеціальних інструментів і методів. Мета полягає в тому, щоб виявити слабкі сторони, якими можуть скористатися зловмисники, дозволяючи реалізувати відповідні заходи безпеки для захисту системи.

У цьому розділі обговоримо методологію та інструменти, що використовуються для оцінки вразливостей, зосереджуючись на результатах, отриманих за допомогою OpenVAS, комплексний інструмент сканування вразливостей і керування ними.

### 2.2.1 Автоматичне сканування

– OpenVAS

OpenVAS (Open Vulnerability Assessment System) – це платформа з відкритим вихідним кодом, яка надає повний набір інструментів для сканування вразливостей і керування ними. Він широко використовується для виявлення

проблем із безпекою в різних системах і програмах, пропонуючи докладні звіти та вказівки щодо виправлення.

Під час оцінки вразливості www.kalina-fond.com OpenVAS виявив загалом 25 уразливостей. Сканер не виявив високих та середніх рівней вразливостей, що далі можемо побачити у звіті на Рис. 2.8.

| Date | Status | Task | Severity | Scan Results | | | |
|---|---|---|---|---|---|---|---|
| | | | | High | Medium | Low | Log |
| Fri Jun 14 08:57:58 2024 | Done | Scan host www.kalina-fond.com | 2.6 (Low) | 0 | 0 | 1 | 24 |

Рисунок 2.8 – Звіт по усім вразливостям хосту www.kalina-fond.com

Серед усіх результатів, 1 вразливість низького рівня та 24 проблеми на рівні журналу. Візуалізацію результатів можете побачити на Рис. 2.9.

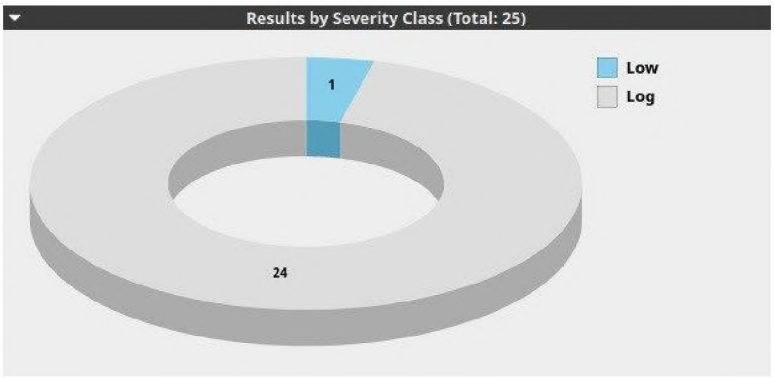



Рисунок 2.9 – Результати за класом серйозності хосту www.kalina-fond.com

OpenVAS також надає таблицю з усіма вразливостями, які знайшла за період сканування, Рис. 2.10. Можна дізнатись більше інформацію про певну уразливість, клікнувши на її назву. Нижче розглянемо деталі виявленої вразливості низького рівня серйозності.

| Vulnerability | Severity | QoD | Host | Location |
|---|---|---|---|---|
| CPE Inventory | 0.0 (Log) | 80% | 20.166.183.113 | general/CPE-T |
| wapiti (NASL wrapper) | 0.0 (Log) | 98% | 20.166.183.113 | 443/tcp |
| wapiti (NASL wrapper) | 0.0 (Log) | 98% | 20.166.183.113 | 80/tcp |
| Nikto (NASL wrapper) | 0.0 (Log) | 98% | 20.166.183.113 | 443/tcp |
| CGI Scanning Consolidation | 0.0 (Log) | 80% | 20.166.183.113 | 443/tcp |
| CGI Scanning Consolidation | 0.0 (Log) | 80% | 20.166.183.113 | 80/tcp |
| OS Detection Consolidation and Reporting | 0.0 (Log) | 80% | 20.166.183.113 | general/tcp |
| nginx Detection | 0.0 (Log) | 80% | 20.166.183.113 | 80/tcp |
| TCP timestamps | 2.6 (Low) | 80% | 20.166.183.113 | general/tcp |
| SSL/TLS: Report Supported Cipher Suites | 0.0 (Log) | 98% | 20.166.183.113 | 443/tcp |

Рисунок 2.10 – Таблиця перших десяти вразливостей знайдених сканером

Детальний огляд уразливісті низького рівня: TCP Timestamps, яку ви можете побачити на Рис. 2.11:

| Vulnerability | Severity | QoD | Host | Location |
|---|---|---|---|---|
| TCP timestamps | 2.6 (Low) | 80% | 20.166.183.113 | general/tcp |

Рисунок 2.11 – Детальний огляд вразливості

- Рівень серйозності: низький (2,6)
- Якість виявлення (QoD): 80%
- Хост: 20.166.183.113
- Розташування: general/tcp

Опис:

- Головна частина:

Віддалений хост реалізує мітки часу TCP, що дозволяє обчислити час безвідмовної роботи системи.

- Результат виявлення вразливості:

Хост реалізує RFC1323, як зазначено в отриманих мітках часу із затримкою 1 секунда між пакетами:

- Пакет 1: 370651190
- Пакет 2: 2473132679

– Вплив:

Наявність позначок часу TCP може виявити час безперебійної роботи віддаленого хоста, який може бути використаний зловмисниками для визначення шаблонів або періодів активності системи.

Рішення:

– Для Linux: додайте рядок net.ipv4.tcp_timestamps = 0 до /etc/sysctl.conf і виконайте sysctl -p, щоб застосувати налаштування.

– Для Windows: виконайте netsh int tcp set global timestamps=disabled. Зверніть увагу, що, починаючи з Windows Server 2008 і Vista, мітку часу не можна повністю вимкнути, але вона не використовуватиметься під час ініціювання з'єднань TCP, якщо одноранговий вузол не включить їх у свій сегмент SYN.

Уражене програмне забезпечення/ОС:

– Реалізації TCP/IPv4, які відповідають RFC1323.

Відомості про вразливість:

– Реалізація позначок часу TCP на віддаленому хості, як визначено в RFC1323, дозволяє обчислити час безвідмовної роботи системи.

Метод виявлення вразливості:

– Спеціально створені IP-пакети надсилаються з невеликою затримкою між ними. Відповіді аналізуються на часові позначки, які, якщо вони є, повідомляються.

Подробиці:

– OID: 1.3.6.1.4.1.25623.1.0.80091

– Використана версія: $Редакція: 14310 $

### 2.2.2 Ручний аналіз

Аналіз вручну є критичним етапом оцінки вразливості, на якому експерти з безпеки використовують свої навички та знання для виявлення та використання

потенційних слабких місць у системі. На відміну від автоматизованих інструментів, ручне тестування дає змогу глибше зрозуміти логіку програми, спеціальні функції та потенційні недоліки безпеки, які автоматизовані інструменти можуть пропустити. У цьому розділі буде зосереджено на ручному тестуванні поширених уразливостей вебдодатків, таких як впровадження SQL, міжсайтовий сценарій (XSS), підробка міжсайтового запиту (CSRF) і недоліки завантаження файлів.

1. Міжсайтовий сценарій (XSS)

Міжсайтовий сценарій (XSS) – це вразливість безпеки, яка зазвичай зустрічається у вебдодатках, що дозволяє зловмисникам впроваджувати шкідливі сценарії у вміст, який потім надається іншим користувачам. Це може призвести до несанкціонованих дій від імені користувачів, викрадення даних або інших зловмисних дій. Також цю атаку називають JavaScript-ін'єкція.

React.js, як і багато інших сучасних фреймворків, надає механізми захисту від XSS. Однак уразливості все одно можуть бути введені, якщо використовуються небезпечні методи кодування. Однією з таких небезпечних практик є використання opasnoSetInnerHTML для відтворення наданих користувачами даних без належної обробки.

Розглянемо наступний приклад, коли дані користувача відображаються безпосередньо за допомогою dangerouslySetInnerHTML:

```
<div
   className="username"
   dangerouslySetInnerHTML={{ __html: user.username }}
/>
```

Цей код відповідає за відображення username в ствопчастій діаграмі Донатерів. Тут змінна user.username містить наданий користувачем вміст, який відображається як HTML. Якщо цей вміст не очищається належним чином, це може призвести до вразливості XSS.

Для демонстрації JavaScript-ін'єкції змінемо username у профілі користувача на тег зображення з атрибутом onerror для того, щоб вивкликати скрипт з відображенням певного тексту:

<img onerror="document.body.innerHTML = '<h1>Цей сайт був зламаний!</h1>';" src="invalid-image" />

Цим скриптом зловмисник змінив вміст сторінки, щоб зіпсувати вебсайт. Тепер замість відображення головної сторінки, як на Рис. 1.3-1.5, відображається текст "Цей сайт був зламаний!" на білому фоні, що ви можете побачити на Рис. 2.12.

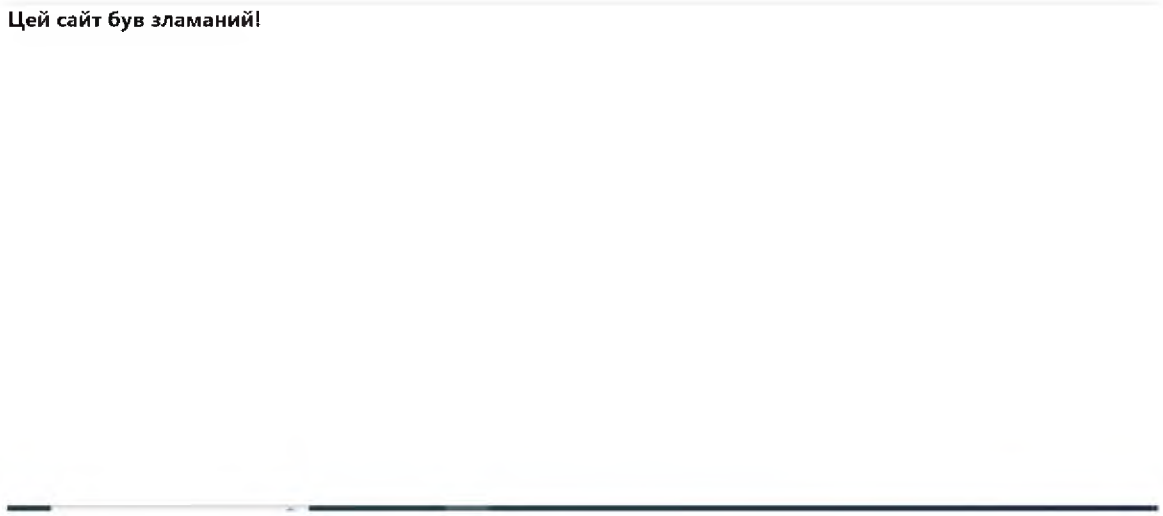



Рисунок 2.12 – Повідомлення про загрозу

Це відбувається коли обліковий запис зі "зламаним" username відображається на сторінці. Так як цей акакунт задонатив гроші, він відображається першим серед усіх донатерів (Рис. 2.13), тому кожний користувач, хто переходить на цю сторінку побачить це повідомлення.

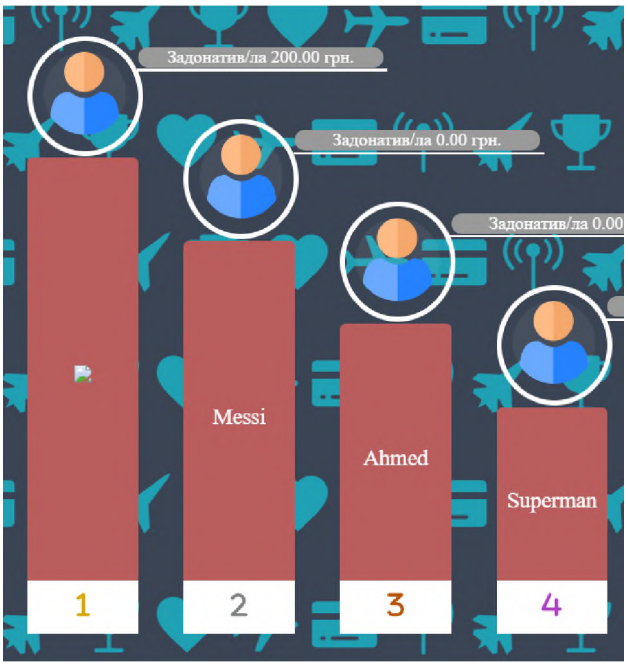



Рисунок 2.13 – Стовпчаста діаграма донатерів

Також атаку можна здійснити за допомогою шкідливого посилання. Зміна імені користувача для включення попередження JavaScript у тег прив'язки:

<a href="javascript:alert('Вас взламали!');">Мільонер</a>

Коли користувач натискає на посилання серед усіх донатерів, як на Рис. 2.14, відобразиться сповіщення "Вас взламали!" (Рис. 2.15).

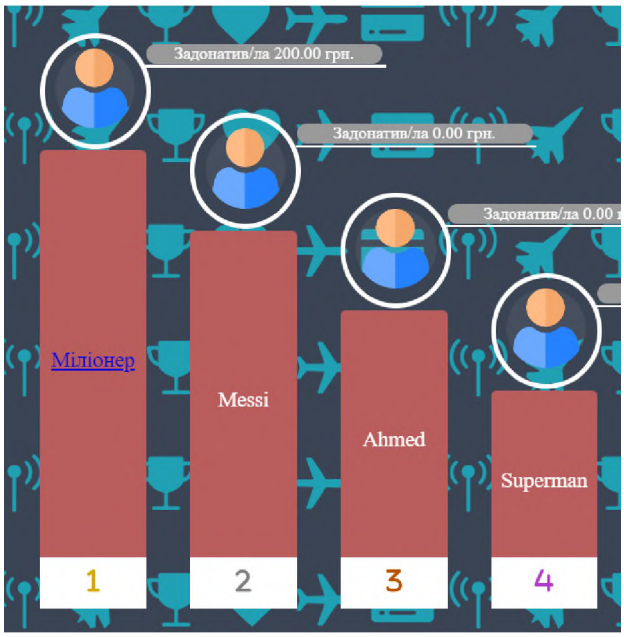



Рисунок 2.14 – Відображення шкідливого посилання серед донатерів


Рисунок 2.15 – Повідомлення після натискання на username шкідливого користувача

2. Підробка міжсайтового запиту (CSRF) із використанням XSS

Міжсайтова підробка запитів (CSRF) – це тип атаки, яка змушує користувача виконувати небажані дії у вебпрограмі, у якій він пройшов автентифікацію. Використовуючи автентифікований сеанс користувача, зловмисник може надсилати неавторизовані запити до вебдодатку від імені користувача без його відома.

Здійснення атаки.

Зловмисник повинен змусити жертву відвідати шкідливий вебсайт. Це можна зробити різними методами:

– Фішинговий електронний лист: зловмисник надсилає фішинговий електронний лист із посиланням на шкідливий вебсайт. Електронний лист може використовувати методи соціальної інженерії, щоб переконати користувача натиснути посилання.

– Соціальні медіа: зловмисник розміщує зловмисне посилання на платформах соціальних мереж, заохочуючи користувачів натискати на нього.

– Зловмисна реклама: зловмисник розміщує зловмисну рекламу на вебсайті, який, ймовірно, відвідає жертва. Коли оголошення завантажується, воно автоматично надсилає форму CSRF.

– Вбудоване посилання: зловмисник вставляє зловмисне посилання в повідомлення на форумі, розділ коментарів або будь-яке інше місце, де жертва може натиснути на нього.

Перегляд профілю у серверній частині вебпрограми обробляє запити GET, PATCH і DELETE і він захищений автентифікацією JWT:

```
@api_view(['GET', 'PATCH', 'DELETE'])
@permission_classes([IsAuthenticated])
def profile(request):
    user = request.user
    if request.method == 'GET':
        serializer = UserSerializer(user, many=False)
        return Response(serializer.data)
    elif request.method == 'PATCH':
        serializer = UserSerializer(user, data=request.data, partial=True)
        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data)
        return Response(serializer.errors,status=status.HTTP_400_BAD_REQUEST)
    elif request.method == 'DELETE':
        user.delete()
        return Response(status=status.HTTP_204_NO_CONTENT)
```

Метод DELETE видаляє обліковий запис користувача. Але він захищається автентифікацією JWT. Однак, якщо належний захист CSRF не реалізовано, зловмисник може використати цю кінцеву точку за допомогою атаки CSRF.

Щоб здійснити CSRF-атаку з метою видалення облікового запису користувача, зловмисник повинен обманом змусити автентифікованого користувача надіслати запит DELETE до кінцевої точки /profile. Ось приклад того, як може бути створена така атака:

Знову оновлюємо поле username, але вже з іншим скриптом:

```
<a href="javascript:fetch('https://www.kalina-fond.com/choreo-apis/kalinafond/backend/v1/profile', {method: 'DELETE',headers: {'Authorization': 'Bearer '+localStorage.getItem('access')}});"> Account</a>
```

Коли інший користувач, який увійшов у систему, натискає посилання « Account», виконується вбудований JavaScript, надсилаючи запит DELETE до кінцевої точки /profile із маркером JWT користувача, включеним у заголовок авторизації.

Сервер обробляє запит DELETE, вважаючи, що це законний запит від користувача, який увійшов у систему, і видаляє обліковий запис користувача.

Якщо перейдемо у вікно розробника і натиснемо на вкладку “Мережа”, то побачимо надісланий запит “DELETE” із відповіддю – “204 Немає Вмісту” (Рис. 2.16) – це значить, що більше немає даних за цього користувача.

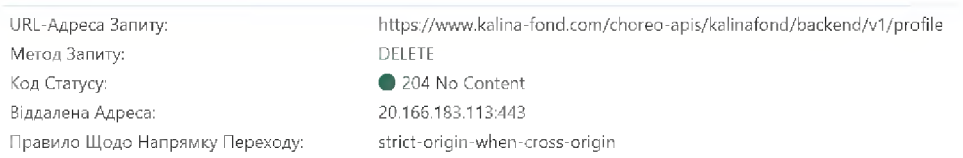



Рисунок 2.16 – Вкладка “Мережа” у вікні розробника

Ця комбінація XSS, для введення не захищених значень, та CSRF, для виконання несанкціонованої дії, може бути дуже ефективною для використання вразливостей у вебдодатках.

## 2.3 Використання сторонніх програм для повного аналізу

Що таке Metasploit? Metasploit – це потужний інструмент, який використовують професіонали з кібербезпеки для перевірки безпеки комп’ютерних систем. Подумайте про це як про «швейцарський ніж» для хакерів (як хороших, так і поганих). Це дозволяє користувачам знаходити вразливі місця в системах, використовувати ці вразливості та бачити, як зловмисник може отримати доступ. Його часто використовують етичні хакери (називаються тестувальниками проникнення), щоб допомогти покращити безпеку шляхом пошуку та усунення недоліків, перш ніж зловмисники зможуть ними скористатися.

Що таке testssl.sh? Це інструмент командного рядка, який перевіряє безпеку конфігурацій SSL/TLS на сервері. SSL/TLS – це протоколи, які захищають з'єднання між вашим браузером і вебсайтами (наприклад, маленький значок замка, який ви бачите в адресному рядку браузера). testssl.sh допоможе зрозуміти, наскільки добре сервер налаштовано для захисту цього з'єднання.

### 2.3.1 Metasploit Auxiliary Module: SSL Version Scanner

Ми використовували модуль auxiliary/scanner/ssl/ssl_version для визначення версій SSL/TLS, які підтримуються сервером на www.kalina-fond.com (20.166.183.113).

Використовувані команди:

1. use auxiliary/scanner/ssl/ssl_version
2. set RHOSTS www.kalina-fond.com
3. set THREADS 10
4. run

Опис результату:

- Інформація про версію SSL і шифр:
  - Шифри TLSv1.2:
    - ECDHE-RSA-AES256-GCM-SHA384
    - ECDHE-RSA-CHACHA20-POLY1305
    - ECDHE-RSA-AES128-GCM-SHA256
  - Шифри TLSv1.3:
    - TLS_AES_256_GCM_SHA384
    - TLS_CHACHA20_POLY1305_SHA256
    - TLS_AES_128_GCM_SHA256
- Інформація про сертифікат:

- Тема: /CN=*.choreoapis.dev
- Емітент: /C=US/O=Let's Encrypt/CN=R3
- Алгоритм підпису: sha256WithRSAEncryption
- Розмір відкритого ключа: 2048 біт
- Термін дії:
  - Не діє раніше: 2024-05-16
  - Не діє після: 14.08.2024
- Емітент CA: http://r3.i.lencr.org/

Аналіз:

- Сервер підтримує TLS 1.2 і TLS 1.3, які є безпечними та рекомендованими протоколами.
- Використовувані шифри містять надійні шифри AEAD і підтримують пряму секретність.
- Сертифікат виданий компанією Let's Encrypt і є дійсним, що вказує на те, що сервер правильно налаштовано для безпечного зв'язку.

### 2.3.2 Testssl.sh Results

Ми використовували testssl.sh для виконання комплексного аналізу конфігурації SSL/TLS того самого сервера.

Використовувані команди:

- testssl.sh www.kalina-fond.com

Опис результату:

Виявлення служби:

- Не вдалося визначити службу, що працює на порту 443.

Підтримка протоколу:

- SSLv2 і SSLv3: не пропонується (OK)

- TLS 1.0 і TLS 1.1: не пропонується (OK)
- TLS 1.2 і TLS 1.3: пропонується (OK)

Категорії шифру:

- Жодних слабких або застарілих шифрів не пропонується.
- Підтримуються лише надійні шифри AEAD із прямою секретністю.
- Сервер надає перевагу порядку шифрування, що підвищує безпеку.

Надійна передня секретність:

- Підтримувані шифри:
  - TLS_AES_256_GCM_SHA384
  - TLS_CHACHA20_POLY1305_SHA256
  - ECDHE-RSA-AES256-GCM-SHA384
  - ECDHE-RSA-CHACHA20-POLY1305
  - TLS_AES_128_GCM_SHA256
  - ECDHE-RSA-AES128-GCM-SHA256

Еліптичні криві:

- Пропоновані криві: prime256v1, secp384r1, secp521r1, X25519, X448
- Кінцеві групи полів: ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192

Алгоритми підпису:

- TLS 1.2 sig_algs:
  - RSA-PSS-RSAE+SHA256
  - RSA-PSS-RSAE+SHA384
  - RSA-PSS-RSAE+SHA512
  - RSA+SHA256
  - RSA+SHA384
  - RSA+SHA512

- RSA+SHA224

- TLS 1.3 sig_algs:
  - RSA-PSS-RSAE+SHA256
  - RSA-PSS-RSAE+SHA384
  - RSA-PSS-RSAE+SHA512

Стандартні параметри сервера:

- Для підвищення безпеки підтримуються різні розширення TLS.

Деталі сертифіката:

- Розмір ключа: RSA 2048 біт
- Використання ключа: цифровий підпис, шифрування ключа
- Розширене використання ключа: автентифікація вебсервера TLS, автентифікація вебклієнта TLS

Аналіз:

- Конфігурація SSL/TLS сервера надійна, із надійними протоколами та шифрами, які забезпечують конфіденційність і цілісність даних, що передаються.
- Відсутність слабких протоколів і шифрів свідчить про хорошу практику безпеки.
- Детальна інформація про підтримувані алгоритми та еліптичні криві допомагає зрозуміти можливості та стійкість сервера до певних типів атак.

Однак є сфери, де можна покращити:

- OCSP Stapling:

Сервер не підтримує зшивання OCSP, що може допомогти зменшити накладні витрати на перевірку сертифіката та покращити безпеку.

- Відновлення сесії:

Сервер не підтримує відновлення сеансу. Реалізація відновлення сеансу може покращити продуктивність і зменшити ризик певних типів атак.

## 2.4 Висновки до розділу етичного аудиту вебдодатку

У цьому розділі проаналізовано роботу вебсервера, проблеми безпеки та методи злому на різних прикладах.

Етап оцінки вразливості виявив кілька проблем, серед яких уразливість позначок часу TCP. Усуваючи ці вразливості за допомогою відповідних заходів пом'якшення та усунення, можна значно покращити рівень безпеки www.kalina-fond.com. Постійний моніторинг і регулярна оцінка вразливості є важливими для забезпечення безпеки системи від нових загроз.

Використання 'dangerouslySetInnerHTML' може бути небезпечним, оскільки воно обходить вбудовані механізми безпеки React, розроблені для захисту від атак XSS. Бажано не використовувати цей елемент коду в вашому вебдодатку, але все ж таки, якщо ви вставите ненадійний вміст у DOM за допомогою цієї властивості, ви можете ненавмисно наразити своїх користувачів на шкідливі сценарії. Щоб запобігти цій вразливості, нам потрібно вжити кількох заходів безпеки в наших програмах:

– Усі введені користувачем дані мають мати екрановані об'єктами HTML. Коли ви використовуєте React, він запобігає більшості вразливостей XSS завдяки тому, як він створює вузли DOM і текстовий вміст.

– Під час серіалізації стану на сервері для надсилання клієнту вам потрібно серіалізувати таким чином, щоб уникнути сутностей HTML.

З результатів Metasploit і testssl.sh видно, що сервер за адресою www.kalina-fond.com (20.166.183.113) безпечно налаштовано для зв'язку SSL/TLS. Використання надійних шифрів, підтримка прямої секретності та дійсний сертифікат від надійного центру сертифікації – все це сприяє безпечному налаштуванню.

## 2.5 Вирішення різних вразливостей вебдадатку “Kalina Foundation”

### 2.5.1 Вразливість dangerouslySetInnerHTML

Ви повинні використовувати dangerouslySetInnerHTML помірковано і лише в особливих ситуаціях, коли вам абсолютно необхідно вставити HTML-контент, який безпосередньо не створено вашими компонентами React. Поширені сценарії включають:

1. Вміст, відтворений на стороні сервера: коли вам потрібно вставити вміст HTML, який надходить із надійного джерела, як-от відтворення на стороні сервера Markdown або іншого попередньо відрендереного HTML.
2. Бібліотеки сторонніх розробників: під час інтеграції зі сторонніми бібліотеками, які надають вміст HTML, який потрібно безпосередньо вставляти в DOM.
3. Застарілий код: під час роботи зі застарілим кодом, який генерує рядки HTML, які потрібно вставити на сторінку.

Кращі практики:

– Розумне використання: уникайте використання dangerouslySetInnerHTML, коли це можливо. Якщо вам потрібно його використовувати, переконайтеся, що вміст належним чином оброблено. У багатьох випадках ви можете досягти того самого результату, використовуючи синтаксис React JSX, який безпечніший.

– Очищення введення: завжди очищайте вміст HTML перед його вставленням за допомогою dangerouslySetInnerHTML. Використовуйте такі бібліотеки, як DOMPurify, щоб очистити HTML і видалити будь-які шкідливі сценарії.

```
import DOMPurify from 'dompurify';

const sanitizedHTML = DOMPurify.sanitize(user.username);
```

<div dangerouslySetInnerHTML={{ __html: sanitizedHTML }} />

– Безпечне використання стану та реквізитів: переконайтеся, що будь-які дані, які надходять із введених користувачами чи зовнішніх джерел, перевірені та продезінфіковані перед використанням.

– Перевірка: регулярно перевіряйте місця, де у вашій кодовій базі використовується dangerouslySetInnerHTML, щоб переконатися, що він не створює вразливих місць у безпеці.

– Політика безпеки вмісту (CSP): запровадьте надійну політику безпеки вмісту, щоб пом'якшити вплив XSS.

Використовуючи JSX, React автоматично усуває будь-який HTML, перетворюючи його на текстовий вміст. Це гарантує, що будь-які дані, надані користувачем, безпечно відтворюються без виконання будь-яких сценаріїв. Можете використовувати наступний код:

{user.username}

Після внесення цієї зміни ми протестували програму, щоб переконатися, що шкідливий сценарій більше не виконується. На сторінці жертводавців ім'я користувача тепер відображається як звичайний текст, можете побачити на Рис. 3.1.

Замість виконання всього рядка, включаючи теги <a>, тепер ім'я користувача відображається як звичайний текст, що вказує на те, що браузер більше не інтерпретує сценарій:

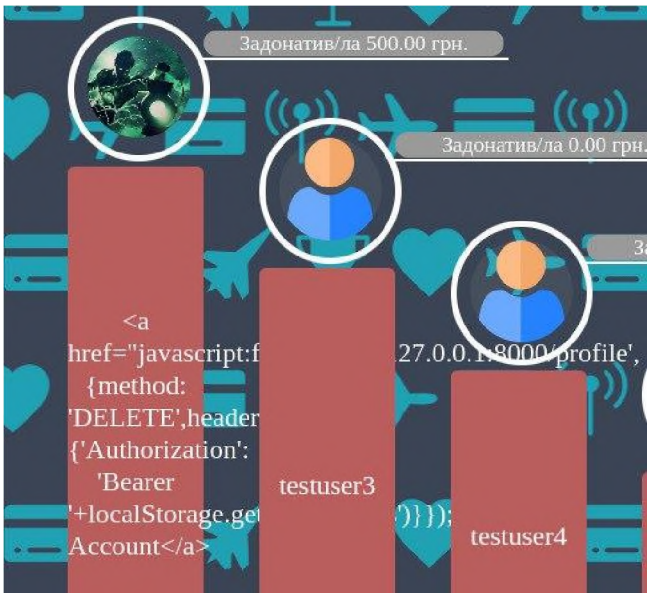



Рисунок 3.1 – Відображення шкідливого ім’я користувача

### 2.5.2 Вразливість CSRF-атаки

Для захисту від CSRF-атак серверна частина має реалізовувати маркери CSRF. Ці маркери є унікальними для кожного сеансу, і їх слід включати в усі запити на зміну стану (наприклад, POST, PATCH, DELETE). Сервер повинен перевірити наявність і дійсність цього маркера перед обробкою запиту.

Наприклад, у Django будемо використовувати декоратор @ensure_csrf_cookie, щоб переконатися, що маркер CSRF включено у відповідь:

```
from django.views.decorators.csrf
import ensure_csrf_cookie from rest_framework.decorators
import api_view, permission_classes from rest_framework.permissions
import IsAuthenticated from rest_framework.response
import Response from rest_framework
import status

@api_view(['GET', 'PATCH', 'DELETE'])
@permission_classes([IsAuthenticated])
@ensure_csrf_cookie
def profile(request):
```

```
        user = request.user
        if request.method == 'GET':
                serializer = UserSerializer(user, many=False)
                return Response(serializer.data)
        elif request.method == 'PATCH':
                serializer = UserSerializer(user, data=request.data, partial=True)
                if serializer.is_valid():
                        serializer.save()
                        return Response(serializer.data)
                return Response(serializer.errors, status=status.HTTP_400_BAD_REQUEST)
        elif request.method == 'DELETE':
                user.delete()
                return Response(status=status.HTTP_204_NO_CONTENT)
```

У React.js нам потрібно переконатися, що наші запити містять маркер CSRF. Будемо використовувати бібліотеку js-cookie для читання маркера CSRF із файлів cookie та включення його в заголовки наших запитів.

Спочатку нам потрібно встановити бібліотеку js-cookie. Вписуємо у командний рядок цю команду:

npm install js-cookie

Далі оновлюємо наш екземпляр api, щоб включити маркер CSRF у заголовки:

```
import axios from 'axios';

const api = axios.create({
        baseURL: 'http://victim-website.com/api',
        withCredentials: true, // Include credentials in requests
});

api.interceptors.request.use(config => {
        const token = getCsrfToken(); // Function to retrieve CSRF token
        if (token) {
                config.headers['X-CSRFToken'] = token;
        }
```

```
    return config;
});

function getCsrfToken() {
    // Implementation to retrieve CSRF token from cookies or meta tags
}

export default api;
```

Завдяки цим змінам наша програма тепер захищена від атак CSRF. Щоб перевірити це, можемо виконати такі кроки:

1. Переконайтеся, що маркер CSRF установлено. Переконайтеся, що маркер CSRF встановлено в файлах cookie, коли користувач отримує доступ до програми. Це можна перевірити за допомогою інструментів розробника браузера, як можете побачити на Рис. 3.2:


Рисунок 3.2 – csrftoken користувача включений у Cookies

2. Включайте маркер CSRF у запити: переконайтеся, що маркер CSRF включено в заголовки всіх запитів. Це можна перевірити, перевіривши мережеві запити в інструментах розробника браузера.

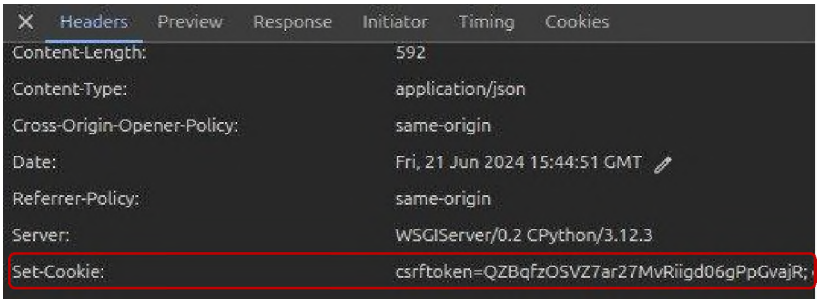



Рисунок 3.3 – csrftoken користувача включений у заголовки

3. Тестування захисту CSRF: спробуйте виконати дію, яка потребує автентифікації, як-от видалення облікового запису користувача, не включаючи маркер CSRF. Сервер має відповісти помилкою 401 Unauthorized.

Приклад сценарію: атака на посилання електронної пошти

Уявіть, що зловмисник надсилає зловмисне посилання користувачеві електронною поштою. Посилання намагається видалити обліковий запис користувача, надсилаючи запит DELETE до кінцевої точки /profile. За наявності захисту CSRF ця атака не вдасться, оскільки запит не включатиме дійсний маркер CSRF.

Ось можливе шкідливе посилання, як на Рис. 3.4:

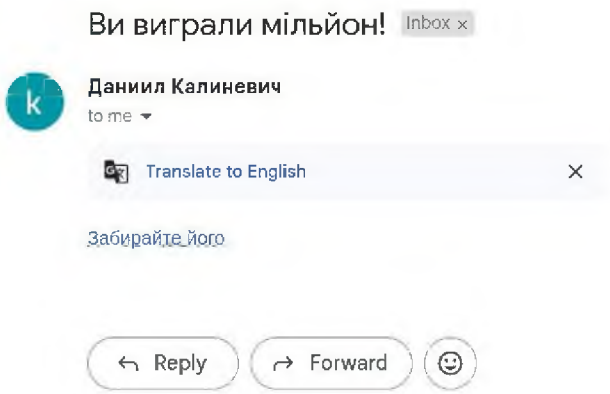



Рисунок 3.4 – Отримане шкідливе повідомлення

Коли користувач, який увійшов у систему вебдодатку “Kalina Foundation” і клацає це посилання, запит завершується помилкою 401 Unauthorized (Рис. 3.5), оскільки маркер CSRF не включено в заголовки і Cookies запиту, як на Рис. 3.6.

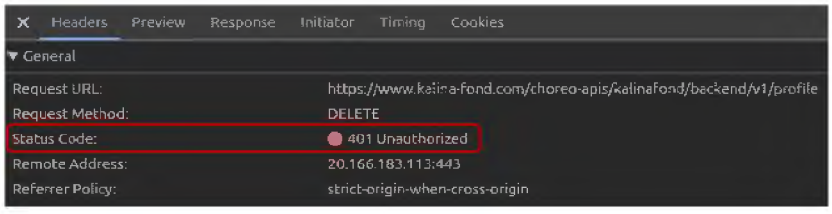



Рисунок 3.5 – Відповідь сервера на запит - 401 Unauthorized

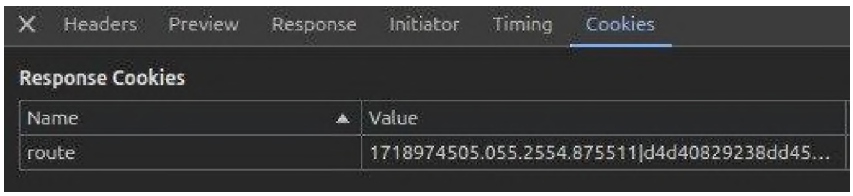



Рисунок 3.6 – Відсутність значення CSRF токена

Є ще декілька варіантів захисту від CSRF-атак:

– Використання файлів cookie SameSite: установіть атрибут SameSite для файлів cookie, щоб запобігти їх надсиланню в запитах між джерелами.

– Перевірка заголовків посилання: перевірте заголовок Referer, щоб переконатися, що запит надходить із надійного джерела.

Впроваджуючи механізми захисту CSRF, такі як маркери CSRF, програма може захищатися від атак CSRF і гарантувати, що запити на зміну стану виконуються лише авторизованими користувачами.

## 2.6 Рекомендації до захисту вебдодатку “Kalina Foundation”

У цьому розділі надаємо набір рекомендацій щодо подальшого підвищення безпеки вебпрограми. Дотримуючись цих найкращих практик, ви можете значно знизити ризик різноманітних атак і забезпечити безпеку та цілісність вашої програми та даних користувача.

1. Безпечна автентифікація та авторизація

– Використовуйте політику надійних паролів: дотримуйтеся політик надійних паролів, включаючи мінімальну довжину, вимоги до складності та періодичні зміни паролів.

– Запровадження багатофакторної автентифікації (MFA): використовуйте MFA, щоб додати додатковий рівень безпеки обліковим записам користувачів.

– Використовуйте OAuth і OpenID Connect: для автентифікації сторонніх розробників використовуйте безпечні стандартизовані протоколи, такі як OAuth 2.0 і OpenID Connect.

2. Автентифікація електронної пошти

– Перевірка електронної пошти: реалізуйте процес перевірки електронної пошти, коли користувачі створюють новий обліковий запис. Надішліть посилання для підтвердження на надану адресу електронної пошти та активуйте обліковий запис лише після натискання посилання.

– Повторно надіслати електронний лист із підтвердженням: надайте користувачам можливість повторно надіслати електронний лист із підтвердженням, якщо вони не отримали його спочатку.

– Захищений вміст електронної пошти: переконайтеся, що вміст електронної пошти для підтвердження є безпечним і не містить конфіденційної інформації.

– Відстежуйте доставку електронної пошти: відстежуйте стан доставки електронних листів із підтвердженням і вирішуйте будь-які проблеми, пов'язані з доставкою електронної пошти.

3. Безпечна передача даних

– Використовуйте HTTPS: завжди використовуйте HTTPS для шифрування даних, що передаються між клієнтом і сервером. Отримайте та встановіть сертифікати SSL/TLS від довіреного центру сертифікації (CA).

– HSTS: запровадьте HTTP Strict Transport Security (HSTS), щоб забезпечити використання HTTPS і захистити від атак на пониження версії.

4. Подальший аналіз:

– Виконуйте додаткові сканування, наприклад виявлення версій служби та сканування вразливостей, щоб виявити потенційні вразливості в службах, що працюють на портах 80 і 443.

5. Безпечне завантаження файлів

– Перевірте типи файлів: обмежте завантаження файлів дозволеними типами (наприклад, зображення) і перевірте тип файлу як на стороні клієнта, так і на стороні сервера.

– Обмеження розміру файлу: установіть обмеження на максимальний розмір файлу, щоб запобігти атакам виснаження ресурсів.

– Сканувати завантажені файли: Використовуйте антивірусне програмне забезпечення для перевірки завантажених файлів на наявність шкідливих програм.

– Безпечне зберігання файлів: зберігайте завантажені файли за межами кореневої вебсторінки та використовуйте безпечні угоди про іменування файлів, щоб запобігти атакам обходу каталогу.

6. Оцінка безпеки:

– Оцініть конфігурації безпеки сервера nginx і базової операційної системи, щоб переконатися, що вони актуальні та правильно налаштовані.

7. Регулярні перевірки та оновлення безпеки

– Перевірка коду: проводите регулярні перевірки коду, щоб виявити та усунути вразливі місця безпеки.

– Керування залежностями: оновлюйте всі залежності та бібліотеки за допомогою останніх виправлень безпеки.

– Тестування на проникнення: Виконуйте регулярне тестування на проникнення, щоб виявити та усунути недоліки безпеки.

8. Логування та моніторинг

– Увімкнути журналювання: увімкніть всебічне журналювання для відстеження дій користувачів і виявлення потенційних інцидентів безпеки.

– Контролюйте журнали: регулярно перевіряйте журнали на наявність підозрілих дій і налаштуйте сповіщення про можливі порушення безпеки.

– Безпечне зберігання журналів: безпечно зберігайте журнали та зберігайте їх протягом відповідного періоду для підтримки судових розслідувань.

9. Резервне копіювання та відновлення

– Регулярне резервне копіювання: виконуйте регулярне резервне копіювання програми та бази даних. Переконайтеся, що резервні копії зашифровані та надійно зберігаються.

– План аварійного відновлення: розробіть і протестуйте план аварійного відновлення, щоб гарантувати швидке відновлення програми та даних у разі інциденту безпеки або втрати даних.

Систематично аналізуючи результати сканування та вживаючи профілактичних заходів, етичні хакери можуть отримати повне розуміння стану безпеки цільової системи та визначити області, які потребують подальшого дослідження та тестування.

Наявність позначок часу TCP є проблемою невеликої серйозності, але все ще становить потенційний ризик. Розкриваючи час безвідмовної роботи системи, зловмисник може зібрати інформацію про шаблони перезавантаження та можливі періоди обслуговування. Щоб пом'якшити цю вразливість, доцільно вимкнути мітки часу TCP на уражених системах.

Для систем Linux рекомендованим підходом є зміна файлу /etc/sysctl.conf і застосування змін. У системах Windows, хоча параметр мітки часу не можна повністю вимкнути, починаючи з певних версій, його можна налаштувати, щоб мінімізувати його використання.

## 2.7 Висновки до розділу ефективних способів захисту від вразливостей

Усунувши вразливості XSS за допомогою цих методів, значно покращили безпеку нашої програми. Забезпечення безпечного відтворення створеного користувачами вмісту запобігає виконанню шкідливих сценаріїв і захищає наших користувачів від потенційних атак.

У цьому розділі підкреслено важливість захисту введених користувачами даних і процесів рендерингу, продемонстровано ефективні методи пом'якшення вразливостей XSS у вебдодатках.

Запровадивши захист CSRF за допомогою декоратора Django @ensure_csrf_cookie і включивши маркер CSRF у запити Axios на інтерфейсі, ми ефективно зменшили ризик атак CSRF. Це гарантує, що наша вебпрограма залишається в безпеці, а дії користувача захищені від несанкціонованого виконання.

Ці кроки підкреслюють важливість захисту від CSRF-атак і демонструють, як реалізувати надійні заходи безпеки у вебпрограмі.

Дотримуючись цих рекомендацій, ви можете значно підвищити безпеку вебдодатку. Безпека – це безперервний процес, який вимагає пильності, регулярних оновлень і проактивного підходу для виявлення та пом'якшення потенційних загроз. Застосування цих передових методів допоможе захистити вашу програму, дані користувача та репутацію від зловмисних атак.

# РОЗДІЛ 3. ЕКОНОМІЧНИЙ РОЗДІЛ

## 3.1 Обґрунтування вартості створення сайту "Kalina Foundation"

Економічний розділ кваліфікаційної роботи спрямований на комплексний аналіз фінансових аспектів створення та підтримки вебсайту «Kalina Foundation». Цей аналіз має вирішальне значення для розуміння економічної доцільності проекту та забезпечення того, що інвестовані ресурси принесуть значну віддачу з точки зору як операційної ефективності, так і підвищеної безпеки.

Розробка вебсайту «Kalina Foundation» передбачає як початкові капітальні витрати, так і поточні операційні витрати. У наступних розділах детально описано ці витрати та економічний вплив проекту.

Головна мета цієї частини – визначити та розрахувати початкові витрати, пов'язані з розробкою та розгортанням вебсайту, проаналізувати економічні вигоди, отримані від впровадження вебсайту та системи захисту інформації, визначити періодичні витрати, необхідні для підтримки та ефективної роботи вебсайту, надати докази того, що проект є економічно доцільним і стійким у довгостроковій перспективі.

## 3.2 Розрахунок вартості розробки сайту

У цьому розділі наведено детальну оцінку капітальних (постійних) витрат і річних поточних (операційних) витрат, пов'язаних із розробкою та обслуговуванням вебсайту «Kalina Foundation». На даному етапі необхідно визначити собівартість продукту, який розробляється та економічно обґрунтувати доцільність вибору однієї із стратегій.

3.2.1 Капітальні (постійні) витрати

Бюджет витрат матеріалів наведено у табл. 3.1.

Таблиця 3.1 – Бюджет витрат матеріалів та комплектуючих виробів

| Назва матеріалів та комплектуючих | Марка, тип, модель | Фактична кількість, шт. | Ціна за одиницю, грн. | Разом, грн. |
|---|---|---|---|---|
| Ноутбук | Asus Vivobook S 15 OLED M5506NA | 1 | 38 000,00 | 38 000,00 |
| Разом: | | | | 38 000,00 |

До розроблення вебсайту залучено одного розробника, який повинен бути забезпеченим відповідними матеріалами, а також оплатою його щоденної праці протягом усього часу розроблення продукту, яка вимагає його часу та навичок високого рівня.

Розрахунок витрат на оплату праці занесено у табл. 3.2.

Таблиця 3.2 – Бюджет витрат на оплату праці

| Посада, спеціальність | Кількість працівників, осіб | Час роботи, дні | Денна заробітна плата працівників, грн. | Сума витрат на оплату праці, грн. |
|---|---|---|---|---|
| Розробник | 1 | 21 | 1500,0 | 31500,0 |
| Разом: | | | | 31500,0 |

До обов’язкових відрахувань належать:

1. Військовий збір, у розмірі 1,5% від суми виплати найманому працівнику;

2. Єдиний соціальний внесок, що становить 22% суми заробітної плати;

3. Податок на дохід фізичної особи, у розмірі 18% від оплати праці робітника.

Суми у контексті даного продукту розраховані та описані у таблиці 3.3.

Таблиця 3.3 – Бюджет обов'язкових відрахувань та податків

| Посада, спеціальність | Сума основної заробітної плати | Сума додаткової заробітної плати | Разом витрат на оплату праці | Сума військового збору, грн. (1,5%) | Сума ЄСВ, грн (22%) | Сума податку з доходів фізичних осіб, грн. (18%) |
|---|---|---|---|---|---|---|
| Розробник | 31500,0 | 0,0 | 31500,0 | 472,5 | 6930,0 | 5670,0 |
| Разом: | 31500,0 | 0,0 | 31500,0 | 472,5 | 6930,0 | 5670,0 |

Доменне ім'я для вебсайту було придбано у namecheap.com, американська компанія, на один рік. Це постійні витрати, які виникають щорічно. Всі ціни на сайті у доларах, тому будемо переводити у гривні, як можете побачити у таблиці 3.4.

Таблиця 3.4 - Бюджет витрат на купівлю доменного ім'я

| Назва сервісу | Назва придбаного доменного ім'я | Сума за доменне ім'я ($) | Курс долара (на момент купівлі) | Сума за доменне ім'я (₴) |
|---|---|---|---|---|
| Namecheap | www.kalina-fond.com | 6,16 | 40,4 | 248,86 |
| Разом: | | 6,16 | 40,4 | 248,86 |

Програмне забезпечення та інструменти:

– Інструменти веброзробки (IDE, фреймворки): безкоштовно (з використанням інструментів з відкритим кодом)

– Інструменти безпеки: безкоштовно (з використанням інструментів безпеки з відкритим кодом)

Для того щоб вирахувати витрати на електроенергію, подивимось тарифи на сайті Yasno. Будемо вважати, що використовуємо тариф для населення та колективних побутових споживачів з електроопаленням і в результаті отримаємо таблицю 3.5.

Таблиця 3.5 - Бюджет витрат на електроенергію

| Намер тариф у | Енергоспоживання комп'ютера (кВт·год) | Час розробки (днів) | Час розробки (годин) | Енергоспоживання (кВт/год) | Вартість електроенергії (₴) |
|---|---|---|---|---|---|
| 1 | 0,2 | 21 | 168 | 33,6 | 145,15 |
| Разом: | | | | | 145,15 |

3.2.2 Річні поточні (операційні) витрати

Оскільки розробка сайту «Фонд Калина» передбачала придбання ноутбука, необхідно врахувати витрати на амортизацію цього обладнання. Витрати на амортизацію можна розрахувати за допомогою прямолінійного методу, який передбачає, що вартість активу зменшується однаково щороку протягом терміну його корисного використання. Для обладнання мінімальним терміном корисного використання зазвичай вважається 5 років. Ліквідаційна вартість оцінюється у 8000 грн за статистикою продажів у магазинах.

Формула прямолінійної амортизації:

$$\text{Н}_\text{а} = \frac{(C_\text{а} - V_\text{л})}{T} \quad (4.1)$$

де $C_\text{а}$ – вартість купленого товару;

$V_л$ – ліквідаційна вартість за мінімальний термін використання;

$T$ – мінімальний термін корисного використання.

$$Н_а = (38\ 000 – 8000) / 5 = 6000 \text{ гривень};$$

Отже, витрати на амортизацію становлять 6000 гривень.

Поновлення хостингу:

– Безкоштовно. Використовуючи підписку розробника, сервіс Choreo дає доступ до необмеженої кількості проектів, над проектами можуть працювати лише два розробника.

Поновлення домену:

– 248,86 грн на рік за розміщення на www.kalina-fond.com, якщо курс долара не зміниться.

Технічне обслуговування:

– Регулярні оновлення та виправлення: безкоштовно (самопідтримувані)

– Технічна підтримка: без додаткових витрат

## 3.3 Оцінка можливих збитків від нападу

Для оцінки економічної доцільності впровадження заходів безпеки вкрай важливо враховувати потенційну шкоду від конкретних загроз. Оцінюючи ці загрози та пов’язані з ними втрати, можемо визначити, чи виправдані витрати на впровадження політики безпеки. Для цієї оцінки зосередимося на потенційних витратах на відновлення бази даних, штрафах за втрату конфіденційної інформації, витратах на безпеку вебсайту та загальних збитках.

1. Витрати на відновлення бази даних

Відновлення бази даних передбачає відновлення після атаки або збою. Це включає технічну підтримку, адміністрування системи та перевірку цілісності даних.

– Погодинна оплата менеджера технічної підтримки: 68,75 грн

– Погодинна оплата системного адміністратора: 106,25 грн

– Орієнтовний час відновлення: 10 годин

– Ймовірність атаки: 2 відсотка

Розрахунок збитків:

$$В_1 = (ЗП_{мтп} + ЗП_{са}) \times t_в \quad (4.1)$$

де $ЗП_{мтп}$ – заробітна плата менеджера технічної підтримки;

$ЗП_{са}$ – заробітна плата системного адміністратора;

$t_в$ – час відновлення.

$$В_1 = (68,75 + 106,25) \times 10 = 1\ 750 \text{ гривень};$$

2. Штрафи за втрату конфіденційної інформації

Втрата конфіденційної інформації може призвести до юридичних санкцій і втрати довіри. Припустимо, що штрафні санкції залежать від кількості порушених записів і серйозності порушення.

– Кількість постраждалих записів: 100

– Штраф за запис: 200 грн

– Ймовірність атаки: 8%

Розрахунок збитків:

$$В_2 = N_{пз} \times Ш_{зз} \quad (4.2)$$

де $N_{пз}$ – кількість постраждалих записів;

$Ш_{зз}$ – штраф за запис.

$$В_2 = 100 \times 200 = 20\ 000 \text{ гривень};$$

3. Витрати на безпеку вебсайту

Впровадження та підтримка заходів безпеки вебсайту передбачає витрати на початкове налаштування та постійне обслуговування.

– Вартість початкового налаштування безпеки: 5 000 грн

– Річні витрати на обслуговування: 2 000 грн

– Ймовірність атаки: 5%

Розрахунок збитків:

$$В_З = V_{пнб} + С_{рно} \quad (4.3)$$

де $V_{пнб}$ – вартість початкового налаштування безпеки;

$С_{рно}$ – щорічні витрати на обслуговування.

$$В_З = 5\ 000 + 2\ 000 = 7\ 000 \text{ гривень;}$$

Загальні втрати включають витрати на відновлення бази даних, штрафи за втрату конфіденційної інформації та витрати на безпеку вебсайту:

$$В = В_1 + В_2 + В_3 \quad (4.4)$$

$$В = 1\ 750 + 20\ 000 + 7\ 000 = 28\ 750 \text{ гривень;}$$

В результаті отримаємо загальний збиток – 28 750 гривень.

## 3.4 Загальний ефект від впровадження вебдодатку

Економічний ефект від впровадження заходів безпеки розраховується шляхом порівняння потенційних втрат, яких вдалося уникнути, з витратами на впровадження:

$$Е = \sum В_i \cdot R_i - С, \quad (4.5)$$

де В – загальний збиток від атаки на вузол або сегмент корпоративної мережі, тис. грн;

R – очікувана імовірність атаки на вузол або сегмент корпоративної мережі, частки одиниці;

С – щорічні витрати на експлуатацію системи інформаційної безпеки, тис. грн.

$$Е = 1\ 750 \times 0{,}02 + 20\ 000 \times 0.08 + 7\ 000 \times 0.05 - 346{,}72 = 1\ 985 - 346{,}72 =$$

$$= 1\ 638 \text{ гривень } 28 \text{ копійок}$$

## 3.5 Визначення та аналіз показників економічної ефективності

$$ROSI = \frac{E}{K}, \quad \text{частки одиниці,} \tag{3.8}$$

де E – загальний ефект від впровадження системи інформаційної безпеки, тис. грн; К – капітальні інвестиції за варіантами, що забезпечили цей ефект, тис.

$$ROSI = 1\ 638{,}28 / 3{,}4672 \approx 472{,}51$$

## 3.6 Висновки до розділу 3

Виходячи з отриманих розрахунків та загального огляду – можна зазначити що розроблення вебдодатку “Kalina Foundation” і витрати на його захист було економічно доцільним.

Економічна доцільність вебсайту “Kalina Foundation” переконливо підтверджена аналізом. Потенційні збитки від незапровадження заходів безпеки (28 750 грн) значно перевищують витрати на розробку вебсайту та його захист (346,72 грн). Розрахована рентабельність інвестицій у безпеку (ROSI) приблизно 472,51% ще більше підкреслює значні переваги цих інвестицій у безпеку. Таким чином, проект є економічно життєздатним і демонструє значну фінансову обачність і передбачення заходів безпеки.

# ВИСНОВКИ

Проект сайту «Kalina Foundation» покликаний створити безпечну та ефективну платформу для підтримки благодійної діяльності. Протягом усього процесу розробки та впровадження було досягнуто кількох ключових цілей, забезпечивши досягнення цілей проекту як щодо функціональності, так і безпеки.

По-перше, проект успішно розробив зручний і адаптивний вебсайт. Використовуючи сучасні вебтехнології, такі як React для інтерфейсу та Django для серверу, вебсайт був розроблений, щоб запропонувати інтуїтивно зрозумілий досвід користувача. Вибір цих технологій забезпечив надійну продуктивність і масштабованість, що дозволило вебсайту ефективно обробляти зростаючу кількість користувачів і транзакцій.

По-друге, значний акцент був зроблений на безпеці. Завдяки застосуванню засобів захисту міжсайтових сценаріїв (XSS) і підробки міжсайтових запитів (CSRF) проект усунув загальні вразливості, які могли поставити під загрозу дані користувача та цілісність вебсайту. Використання декоратора @ensure_csrf_cookie у Django в поєднанні з обробкою токенів CSRF у JavaScript захищало вебсайт від несанкціонованих дій, таким чином захищаючи взаємодію користувачів і дані.

З економічної точки зору проект продемонстрував економічну ефективність. Капітальні витрати на купівлю та розробку домену були мінімальними, а операційні витрати були низькими завдяки безкоштовному хостингу на Choreo. Потенційні втрати від порушень безпеки були значно вищими, ніж витрати на впровадження заходів безпеки, що підкреслює економічну виваженість інвестування в надійні протоколи безпеки. Розрахована рентабельність інвестицій у безпеку (ROSI) приблизно 8,19% підтвердила фінансову вигоду від інвестицій у безпеку.

Крім того, проект включав комплексну модель загроз та оцінку ризиків, визначаючи загрози з найвищою ймовірністю та їхній потенційний вплив. Цей

проактивний підхід дозволив застосувати цілеспрямовані заходи безпеки, забезпечивши стійкість вебсайту до певних ризиків. Включення планів відновлення бази даних і штрафів за втрату конфіденційної інформації ще більше підкреслило ретельну стратегію управління ризиками проекту.

Підсумовуючи, проект вебсайту «Kalina Foundation» не лише досяг своїх основних цілей зі створення функціональної та безпечної платформи, але й продемонстрував сильну економічну доцільність та розумне управління ресурсами. Ретельне планування та впровадження заходів безпеки в поєднанні з ефективним використанням ресурсів призвели до стійкого та економічно ефективного рішення. Цей проект служить цінним прикладом того, як стратегічне планування, сучасні технології та надійні методи безпеки можуть бути інтегровані для розробки ефективних вебрішень для благодійних організацій. Результати створюють міцну основу для майбутніх удосконалень і масштабованості, гарантуючи, що вебсайт може й надалі ефективно підтримувати місію фонду.

# ПЕРЕЛІК ПОСИЛАНЬ

1. Державна служба спеціального зв'язку та захисту інформації України. (2023). Національна стратегія кібербезпеки України. URL: https://cip.gov.ua/ua (дата звернення: 02.06.2024)

2. ІНФОРМАЦІЙНА БЕЗПЕКА УКРАЇНИ В СУЧАСНИХ УМОВАХ. URL: https://zakon.rada.gov.ua/laws/show/2657-12#Text (дата звернення: 03.06.2024)

3. Захист від XSS, CSRF та інших типових атак. URL: https://blog-it.com.ua/zahyst-vid-xss-csrf-ta-inshyh-typovyh-atak/ (дата звернення: 05.06.2024)

4. ЗАХИСТ ВЕБДОДАТКУ ВІД XSS АТАК. URL: https://ir.lib.vntu.edu.ua/bitstream/handle/123456789/29408/9405.pdf?sequence=3&isAllowed=y (дата звернення: 05.06.2024)

5. Закон України «Про Державну службу спеціального зв'язку та захисту інформації України». URL: https://zakon.rada.gov.ua/laws/show/3475-15#Text (дата звернення: 03.06.2024)

6. Кудрявцев, С. М. (2018). Захист інформації в комп'ютерних системах.

7. Що таке XSS атака і які є методи захисту. URL: https://foxminded.ua/xss-ataka/ (дата звернення: 05.06.2024)

8. Як реалізувати CSRF-захист. URL: https://symfony.com.ua/doc/current/security/csrf.html (дата звернення: 05.06.2024)

9. Anderson, R. (2020). Security Engineering: A Guide to Building Dependable Distributed Systems. Wiley.

10. Bishop, M. (2018). Introduction to Computer Security. Addison-Wesley.

11. Choreo. (2023). Choreo Website. URL: https://wso2.com/choreo/ (дата звернення: 13.05.2024)

12. Django Software Foundation. (2023). Django Documentation. URL: https://docs.djangoproject.com/ (дата звернення: 13.05.2024)

13. Google. (2023). Google Cloud Security Best Practices. URL: https://cloud.google.com/security/best-practices (дата звернення: 16.05.2024)

14. Microsoft. (2023). Azure Security Documentation. URL: https://docs.microsoft.com/en-us/azure/security/ (дата звернення: 11.05.2024)

15. OWASP Foundation. (2023). OWASP Top Ten. URL: https://owasp.org/www-project-top-ten/ (дата звернення: 11.05.2024)

16. Postman. (2023). Postman API Platform. URL: https://web.postman.co/ (дата звернення: 11.05.2024)

17. React.js. (2023). React Documentation. URL: https://reactjs.org/docs/getting-started.html (дата звернення: 11.05.2024)

18. Schneier, B. (2015). Data and Goliath: The Hidden Battles to Collect Your Data and Control Your World. W.W. Norton & Company.

19. Stallings, W. (2019). Cryptography and Network Security: Principles and Practice. Pearson.

ДОДАТОК А. Відомість матеріалів кваліфікаційної роботи

| № | Формат | Найменування | Кількість листів | Примітка |
|---|---|---|---|---|
| 1 | А4 | Реферат | 1 | |
| 2 | А4 | Список умовних скорочень | 1 | |
| 3 | А4 | Зміст | 2 | |
| 4 | А4 | Вступ | 2 | |
| 5 | А4 | 1 Розділ | 15 | |
| 6 | А4 | 2 Розділ | 37 | |
| 7 | А4 | 3 Розділ | 7 | |
| 8 | А4 | Висновки | 2 | |
| 9 | А4 | Перелік посилань | 2 | |
| 10 | А4 | Додаток А | 1 | |
| 11 | А4 | Додаток Б | 2 | |
| 12 | А4 | Додаток В | 1 | |
| 13 | А4 | Додаток Г | 1 | |
| 14 | А4 | Додаток Ґ | 1 | |

# ДОДАТОК Б. Функція захищенної сторінки

```
function ProtectedRoute({ children }) {
  const [isAuthorized, setIsAuthorized] = useState(null);

  useEffect(() => {
    auth().catch(() => setIsAuthorized(false));
  });

  const refreshToken = async () => {
    const refreshToken = localStorage.getItem(REFRESH_TOKEN);
    try {
      const res = await api.post("/token/refresh/", { refresh: refreshToken });
      if (res.status === 200) {
        localStorage.setItem(ACCESS_TOKEN, res.data.access);
        setIsAuthorized(true);
      } else {
        setIsAuthorized(false);
      }
    } catch (error) {
      console.log(error);
      setIsAuthorized(false);
    }
  };

  const auth = async () => {
    const token = localStorage.getItem(ACCESS_TOKEN);
    if (!token) {
      setIsAuthorized(false);
      return;
    }
    const decoded = jwtDecode(token);
    const tokenExpiration = decoded.exp;
    const now = Date.now() / 1000;
```

```
    if (tokenExpiration < now) {
      await refreshToken();
    } else {
      setIsAuthorized(true);
    }
  };

  if (isAuthorized === null) {
    return <div>Loading...</div>;
  }

  return isAuthorized ? children : <Navigate to="/login" />;
}

export default ProtectedRoute;
```

## ДОДАТОК В. Перелік документів на оптичному носії

Пояснювальна записка.docx

Презентація.pptx

# ДОДАТОК Г. Відгуки керівників розділів

Відгук керівника економічного розділу:

Керівник розділу _____ _____

(підпис) (ініціали, прізвище)

# ДОДАТОК Ґ. ВІДГУК

на кваліфікаційну роботу бакалавра на тему:

студента групи 125-20-1
Каліневича Данііла Олександровича

Пояснювальна записка складається з титульного аркуша, завдання, реферату, списку умовних скорочень, змісту, вступу, трьох розділів, висновків, переліку посилань та додатків, розташованих на __ сторінках та містить __ рисунка, __ таблиці, __ джерел та __ додатка.

Студент показав достатній рівень володіння теоретичними положеннями з обраної теми, показав здатність формувати власну точку зору (теоретичну позицію).

Робота оформлена та написана грамотною мовою, відповідає вимогам положення про систему запобігання та виявлення плагіату у Національному технічному університеті «Дніпровська політехніка». Містить необхідний ілюстрований матеріал. Автор добре знає проблему, уміє формулювати наукові та практичні завдання і знаходить адекватні засоби для їх вирішення.

В цілому робота задовольняє усім вимогам і може бути допущена до захисту, а його автор заслуговує на оцінку «_____».

Керівник