МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА

ДЕНИСЮК ВЯЧЕСЛАВ ВОЛОДИМИРОВИЧ

Допускається до захисту:
в. о. завідувача кафедри
прикладної математики та
кібербезпеки,
д – р філософії з математики,
_____ А.В.Луценко
«__»_____20__ р.

АНАЛІЗ ТА РОЗРОБКА ПЛАГІНУ ДЛЯ ЗАХИСТУ ПРИВАТНОСТІ У ВЕБ-БРАУЗЕРАХ

Спеціальність 125 Кібербезпека

Кваліфікаційна (бакалаврська) робота

Керівник:
Д-р техн. наук, професор,
професор кафедри прикладної
математики та кібербезпеки,
Крижановський В.Г.

Оцінка:___/___/_____
(бали за шкалою ЄКТС/за національною шкалою)
Голова ЕК:_____
(підпис)

Вінниця - 2024

ЗМІСТ

ВСТУП

У сучасному цифровому світі, де значна частина життя людей переноситься до онлайн-простору, питання захисту приватності в Інтернеті стає все більш актуальним. Веб-браузери, як ключові інструменти доступу до інформації, фактично перетворюються на середовище з високими ризиками, де користувачі змушені ділитися великим обсягом особистих даних. Це робить їх вразливими до онлайн-стеження, таргетованої реклами та інших форм зловживання [13].

Сучасні браузери здатні збирати широкий спектр інформації про користувачів, включаючи:

- Історію пошукових запитів та веб-переглядів;
- Введені дані (логіни, паролі, адреси, номери телефонів);
- Файли cookie та інші веб-маяки;
- Геолокаційні дані;
- Інформацію про пристрій (IP-адреса, тип операційної системи, браузер);
- Дані про форми (імена, адреси електронної пошти, номери телефонів);
- Відбитки пальців пристроїв (унікальні ідентифікатори, що генеруються на основі характеристик пристрою).

Збір особистих даних веб-браузерами може мати негативні наслідки, такі як:

- Втрата конфіденційності;
- Онлайн-стеження;
- Таргетована реклама;
- Зловживання особистими даними.

Метою дослідження є розробка функціонального плагіну для веб-браузерів, який буде забезпечувати надійний захист приватності користувачів.

Плагін буде:

- Блокувати онлайн-стеження;
- Забезпечувати анонімність в Інтернеті;
- Шифрувати дані користувачів;

- Надавати користувачам контроль над збором та використанням їхніх особистих даних [14].

Завдання дослідження:

- **Теоретичний аналіз** літературних джерел з питань приватності в Інтернеті, методів її захисту та розробки плагінів для веб-браузерів.
- **Аналіз методів онлайн-стеження**, таких як файли cookie, веб-маяки, відбитки пальців пристроїв.
- **Розробка архітектури та функціональних можливостей плагіну**, включаючи:
  - Алгоритми блокування онлайн-стеження;
  - Методи забезпечення анонімності;
  - Функції шифрування даних;
  - Інструменти контролю за збором та використанням особистих даних.
- **Розробка плагіну** з використанням відповідних технологій, таких як JavaScript, HTML, CSS, API веб-браузерів.
- **Тестування плагіну** на різних веб-браузерах та веб-сайтах для оцінки його ефективності та виявлення помилок.

РОЗДІЛ 1. ПРИВАТНІСТЬ В ІНТЕРНЕТІ ТА РОЛЬ ФАЙЛІВ COOKIE

1.1. Поняття приватності в інтернеті

Приватність в інтернеті – це багатогранне поняття, що включає право користувачів на:

- **Контроль над своїми особистими даними:** Користувачі мають право на доступ до своїх особистих даних, їх виправлення, видалення та обмеження обробки.
- **Анонімність:** Користувачі мають право залишатися анонімними в інтернеті, приховуючи свою особистість та дані про пристрій.
- **Безпека:** Особисті дані користувачів повинні бути захищені від несанкціонованого доступу, крадіжки, використання та модифікації.
- **Конфіденційність:** Особисті дані користувачів не повинні розголошуватися без їх згоди [13].

Важливість приватності в інтернеті:

- **Захист особистої інформації від зловживання:** Захист від шахрайства, крадіжки особистих даних, спаму та інших форм зловживання.
- **Збереження автономії та свободи дій в інтернеті:** Можливість вільно висловлювати свої думки та ідеї, без цензури та стеження.
- **Забезпечення конфіденційності спілкування та обміну інформацією:** Захист особистих даних та приватного спілкування від несанкціонованого доступу.
- **Протидія цензурі та стеження з боку урядів та корпорацій:** Захист свободи слова та права на приватне життя [5].

Виклики приватності в інтернеті:

- **Зростання кількості даних:** З кожним днем в інтернеті генерується все більше даних, що робить їх збір, аналіз та використання більш доступними.

- **Розвиток технологій:** Нові технології, такі як штучний інтелект, машинне навчання та біометрія, створюють нові можливості для відстеження та моніторингу користувачів.
- **Відсутність єдиного регулювання:** Не існує єдиного міжнародного закону, який би регулював питання приватності в інтернеті, що робить користувачів вразливими до різних юрисдикцій [8].

1.2. Загрози для приватності в веб-браузерах

Веб-браузери, як ключові інструменти для доступу до інформації, стають провідним джерелом ризиків для приватності [12].

Основні загрози:

- **Збір даних:** Веб-сайти та онлайн-сервіси збирають величезний масив особистих даних про користувачів, включаючи:
  - Історію пошуку
  - Історію веб-переглядів
  - Введені дані
  - Файли cookie
  - Геолокаційні дані
  - Інформацію про пристрій
- **Онлайн-стеження:** Відстеження активності користувача на різних веб-сайтах для створення його профілю та таргетованої реклами.
- **Таргетована реклама:** Нав'язлива реклама, що не відповідає інтересам користувача.
- **Продаж особистих даних:** Передача даних третім особам без відома та згоди користувача.
- **Несанкціонований доступ:** Крадіжка особистих даних зловмисниками.
- **Урядове стеження:** Відстеження активності користувачів з боку урядів та правоохоронних органів [10].

- **Кіберзлочинність:** Зростання кіберзлочинності та кібератак робить захист особистих даних в інтернеті ще більш актуальним завданням.
- **Аналітичні інструменти:** Такі як Google Analytics, які збирають дані про трафік веб-сайту та поведінку користувачів.
- **Вбудовані соціальні мережеві віджети:** Кнопки "Подобається" та "Поділитися" соціальних мереж можуть відстежувати активність користувачів навіть на сторонніх сайтах.
- **Розширення браузера:** Деякі розширення браузера можуть збирати дані користувачів без їхньої явної згоди [11].

1.3. Сучасні методи захисту приватності

Існує ряд методів, які користувачі можуть застосувати для захисту своєї приватності в інтернеті:

- **Використання приватних пошукових систем:** Такі як DuckDuckGo, Startpage, які не зберігають історію пошуку користувачів і не передають їхні дані третім особам.
- **Встановлення блокувальників реклами:** Ці розширення для браузерів блокують відображення реклами, трекерів і шкідливого програмного забезпечення, які можуть відстежувати активність користувачів.
- **Використання VPN-сервісів:** Віртуальна приватна мережа шифрує трафік користувача та приховує його реальну IP-адресу, що ускладнює його геолокацію та відстеження [12].
- **Режим інкогніто/приватного перегляду:** Доступний у більшості веб-браузерів, цей режим обмежує збереження історії пошуку, файлів cookie та інших даних про веб-перегляди. Однак, слід пам'ятати, що режим інкогніто не приховує вашу активність від веб-сайтів, які ви відвідуєте, або вашого інтернет-провайдера.
- **Двофакторна автентифікація:** Додатковий рівень безпеки для захисту

облікових записів від несанкціонованого доступу.

- **Регулярне очищення історії браузера, кешу та файлів cookie:** Видалення цих даних може допомогти обмежити обсяг інформації, доступної для відстеження [10].
- **Вибір веб-сайтів та онлайн-сервісів із чіткою політикою конфіденційності:** Обирайте сайти, які обмежують збір та використання особистих даних і надають зрозумілу інформацію про те, як вони використовують ваші дані.
- **Використання менеджерів паролів:** Зберігають паролі в зашифрованому вигляді, що підвищує безпеку та зменшує ризик їх викрадення.
- **Ознайомлення з налаштуваннями конфіденційності:** У веб-браузерах та соціальних мережах існують налаштування конфіденційності, які дозволяють користувачам обмежити збір та використання їхніх даних. Ознайомтеся з цими налаштуваннями та налаштуйте їх відповідно до ваших потреб.
- **Підтримка законодавчих ініціатив щодо захисту приватності в інтернеті:** Лобіювання та підтримка законів, які захищають право користувачів на конфіденційність [15].

Ефективність методів захисту залежить від:

- **Комплексного підходу:** Застосування кількох методів одночасно забезпечує більш надійний захист.
- **Оновлення програмного забезпечення та налаштувань:** Оновлення браузерів, розширень, операційних систем та антивірусного програмного забезпечення закриває відомі вразливості безпеки.
- **Обізнаність користувачів:** Розуміння ризиків для приватності та методів захисту дозволяє користувачам приймати усвідомлені рішення щодо їхньої онлайн-активності [8].

1.4. Файли Cookie: та їхня роль у збереженні та передачі даних

Файли cookie - це невід'ємна частина сучасного Інтернету, що забезпечує зручність та персоналізацію користувацького досвіду. Ці текстові файли, які веб-сайти розміщують на пристроях користувачів, відіграють важливу роль у збереженні та передачі різноманітної інформації. У цьому розділі ми глибоко зануримося в аналіз файлів cookie, розглядаючи їх функції, типи, вплив на приватність, методи керування, етичні аспекти та правові норми [2].

1.4.1. Функції та типи файлів cookie

Існує кілька типів cookie, які виконують різні функції:

- **Сеансові cookie:** тимчасові файли, що видаляються після закриття браузера. Їх використовується для зберігання тимчасових даних, таких як ідентифікатор сеансу користувача або дані, введені в форми.
- **Постійні cookie:** зберігаються на пристрої користувача протягом певного часу, який визначається веб-сайтом (зазвичай від кількох днів до кількох років). Ці файли використовуються для зберігання даних, які не втрачають актуальність після закриття браузера, наприклад, налаштування користувача або мова інтерфейсу.
- **Сторонні cookie:** розміщуються на веб-сайті не самим веб-сайтом, а третьою стороною, наприклад, рекламною мережею. Їх використовується для відстеження користувачів на різних сайтах з метою персоналізації реклами [3].

1.4.2. Збереження даних користувача

Файли cookie можуть використовуватися для зберігання різноманітних даних про користувача:

- **Дані про аутентифікацію:** Файли cookie можуть зберігати дані про

аутентифікацію, такі як ім'я користувача та пароль, щоб користувачам не потрібно було вводити їх повторно при кожному відвідуванні веб-сайту.

- **Налаштування користувача:** Файли cookie можуть зберігати налаштування користувача, такі як мова інтерфейсу, розмір шрифту та колірну тему, щоб зробити веб-сайт більш зручним для користувача.
- **Історія переглядів:** Файли cookie можуть зберігати історію переглядів користувача, щоб рекомендувати йому подібний контент або продукти.
- **Дані про місцезнаходження:** Файли cookie можуть використовуватися для визначення місцезнаходження користувача, щоб показувати йому релевантну рекламу або інформацію.
- **Дані про аналітику:** Файли cookie можуть використовуватися для збору даних про те, як користувачі взаємодіють з веб-сайтом, щоб покращити його дизайн та функціональність [16].

1.4.3. Механізми роботи та вплив на приватність

Веб-сайти можуть використовувати дані з cookie для:

- **Відстеження користувачів на різних сайтах:** Веб-сайти можуть використовувати сторонні cookie для відстеження користувачів на різних сайтах з метою персоналізації реклами.
- **Створення профілів користувачів:** Веб-сайти можуть використовувати дані з cookie для створення профілів користувачів, які містять інформацію про їхні інтереси, звички та демографічні дані. Ця інформація може використовуватися для таргетування реклами та контенту.
- **Персоналізація реклами та вмісту веб-сайтів:** Файли cookie можуть використовуватися для персоналізації реклами та контенту веб-сайтів, щоб користувачі бачили те, що їм, ймовірно, буде цікаво [17].

1.4.4. Методи блокування та видалення файлів cookie

Користувачі можуть керувати використанням файлів cookie, налаштовуючи браузери для:

- **Блокування сторонніх cookie:** Більшість браузерів дозволяють користувачам блокувати сторонні cookie, що може допомогти захистити їхню приватність.
- **Блокування всіх cookie:** Користувачі також можуть заблокувати всі cookie, але це може призвести до проблем з роботою деяких веб-сайтів.
- **Регулярне видалення cookie:** Користувачі можуть регулярно видаляти cookie, щоб захистити свою приватність [18].

1.4.5. Переваги та недоліки використання файлів cookie

Переваги:

- **Покращення користувацького досвіду:** Файли cookie можуть зробити веб-сайти більш зручними для користувачів, наприклад, запам'ятовуючи їхні налаштування або дані про аутентифікацію.
- **Зручність користування веб-сайтами:** Файли cookie можуть зробити веб-сайти більш зручними для користувачів, наприклад, автоматично заповнюючи форми.
- **Функціональність веб-сайтів:** Файли cookie можуть бути необхідними для роботи деяких веб-сайтів, наприклад, для додавання товарів у кошик у магазині.

Недоліки:

- **Ризики для приватності:** Файли cookie можуть використовуватися для відстеження користувачів та збору їхніх особистих даних.
- **Відстеження активності користувачів:** Файли cookie можуть використовуватися для відстеження активності користувачів на різних веб-сайтах.

- **Вплив на швидкість завантаження веб-сторінок:** Файли cookie можуть уповільнювати завантаження веб-сторінок [19].

Висновоки до розділу 1

У першому розділі бакалаврської роботи було глибоко проаналізовано теоретичні аспекти захисту приватності в Інтернеті. Звернуто увагу на контроль за особистою інформацією, безпеку, анонімність та конфіденційність, враховано виклики, такі як збільшення обсягу персональних даних та відсутність єдиного регулювання, і загрози від веб-браузерів, включаючи кіберзлочинність та урядове стеження.

Описано сучасні методи захисту, такі як приватні пошукові системи, VPN-сервіси та блокувальники реклами, з підкресленням важливості комплексного підходу, оновлень програмного забезпечення та освіченості користувачів.

Також досліджено роль файлів cookie в онлайн середовищі, включаючи їхню сутність, функції та вплив на приватність користувачів. Особлива увага приділена різноманітним функціям, від збереження даних до відстеження історії переглядів та аналітики.

Розглянуто різні типи файлів cookie та їхній вплив на приватність користувачів, у тому числі їхню здатність відстежувати користувачів та персоналізувати рекламу.

Також розглянуто методи керування файлами cookie, включаючи блокування та видалення, і їхні впливи на користувацький досвід та приватність.

РОЗДІЛ 2. ПРОЕКТУВАННЯ ПЛАГІНУ ДЛЯ ЗАХИСТУ ПРИВАТНОСТІ

2.1. Вимоги до плагіну

Плагін "CookieShield" створений для надання користувачам контролю над файлами cookie їхніх веб-браузерів. Нижче наведено ключові вимоги до функціональності та характеристик плагіну:

Функціональність:

- Додавання, видалення та редагування cookie: Користувачі повинні мати можливість додавати, видаляти та редагувати файли cookie з їхніх веб-браузерів.
- Пошук cookie: Плагін повинен надавати можливість шукати конкретні файли cookie за різними параметрами, такими як домен, шлях або назва.
- Захист та блокування cookie: Користувачі повинні мати можливість захищати свою приватність, блокуючи небажані файли cookie або деякі типи cookie з певних доменів.
- **Детальна інформація про cookie:** Плагін повинен надавати детальну інформацію про кожен файл cookie, включаючи його назву, значення, домен, шлях, час життя та інші атрибути.
- **Імпорт та експорт cookie:** Користувачі повинні мати можливість імпортувати та експортувати файли cookie, щоб ділитися ними з іншими або створювати резервні копії.
- **Підтримка декількох браузерів:** Плагін повинен бути сумісний з популярними веб-браузерами, такими як Chrome, Firefox, Edge, Safari та Opera [20].

Характеристики:

- **Простий та інтуїтивно зрозумілий інтерфейс:** Графічний інтерфейс плагіну має бути простим та зрозумілим, щоб користувачі могли легко керувати своїми файлами cookie.
- **Швидка та ефективна робота:** Плагін не повинен значно впливати на

продуктивність веб-браузера.

- **Безпечне зберігання cookie:** Всі файли cookie повинні зберігатися в зашифрованому форматі для забезпечення їх безпеки та конфіденційності.
- **Відкритий код:** Код плагіну має бути відкритим, щоб будь-хто міг його перевірити та внести свій вклад [21].

2.2. Архітектура та основні компоненти

Плагін "CookieShield" має наступну архітектуру та ключові компоненти:

- **Інтерфейс користувача:** Графічний інтерфейс, що дозволяє користувачам легко керувати файлами cookie та виконувати різноманітні операції з ними.
- **Менеджер cookie:** Основний модуль, що відповідає за взаємодію з веб-браузером та операції з файлами cookie, включаючи додавання, видалення, редагування, пошук, захист та блокування.
- **Сховище cookie:** Механізм для зберігання cookie в зашифрованому форматі, щоб забезпечити їх безпеку та конфіденційність [18].

2.3. Методи захисту приватності, що використовуються

"CookieShield" використовує різні методи для захисту приватності користувачів:

- **Шифрування cookie:** Всі файли cookie зберігаються у зашифрованому форматі для запобігання несанкціонованому доступу.
- **Блокування небажаних cookie:** Користувачі можуть вказувати конкретні домени або типи cookie, які слід блокувати для забезпечення більшої приватності.
- **Індивідуальне керування cookie:** "CookieShield" надає користувачам повний контроль над їхніми файлами cookie, дозволяючи їм вибирати,

які дані зберігати та які видаляти.

- **Видалення cookie при закритті браузера:** Користувачі можуть налаштувати плагін на автоматичне видалення всіх cookie при закритті веб-браузера [16].

Висновки до розділу 2

У другому розділі розглянуто процес проектування плагіну "CookieShield" для захисту приватності користувачів в онлайн середовищі. Визначено ключові вимоги до плагіну, включаючи функціональні можливості, такі як додавання, видалення, редагування, пошук, захист та блокування файлів cookie.

Описано архітектуру та основні компоненти плагіну, такі як інтерфейс користувача, менеджер файлів cookie та їхнє сховище.

Описано методи захисту приватності, включаючи шифрування файлів cookie, блокування небажаних файлів та автоматичне видалення їх при закритті браузера.

РОЗДІЛ 3. МЕТОДИКА ТА ЗАСОБИ РОЗРОБКИ ПЛАГІНУ

3.1. Вибір технологічного стеку

Плагіни (англ. Plug-in) – це інструменти, які можуть модифікувати функціонал вже існуючих програм або додавати нові можливості до них. У веб-браузері Firefox особливо поширені плагіни, оскільки команда розробників Mozilla працювала над створенням браузера, що мав мінімалістичний дизайн, щоб уникнути зростання кількості помилок та складності коду, але при цьому зберігаючи можливість розширення. Це дозволяє користувачам додавати в браузер ті функції, які вони вважають важливими [1].

Плагін зазвичай використовується для розширення базового функціоналу програми, додаючи нові можливості. Наприклад, за допомогою плагінів можна додати такі функції, як інструменти для веб-розробки, менеджери закладок, спеціалізовані клієнтські програми для роботи з конкретними веб-ресурсами, FTP-клієнти, електронна пошта, налаштування рухів миші, зміна проксі-серверів і багато іншого [4].

Зазвичай розширення використовують наступні технології: CSS, DOM, XPCOM, XP, XUL та Connect XPI. Багато з них можуть змінювати вміст веб-сторінки під час її відображення на екрані. Наприклад, існують програми, які блокують рекламний контент на веб-сторінках, або навпаки, програми, які додають додатковий вміст до основного змісту сторінки. Загалом, це досягається шляхом зміни або доповнення таблиць каскадних стилів [5].

Веб-переглядач, або браузер, є програмою, призначеною для взаємодії з мережею Інтернет, пошуку, обробки та відображення веб-сайтів, а також для переміщення користувача між веб-сторінками у пошуку інформації. Це інструмент, який дозволяє користувачеві переглядати вміст веб-сайтів та взаємодіяти з ним [23].

Сучасні веб-переглядачі, крім підтримки HTTP-протоколу для перегляду гіпертекстового контенту, також зазвичай надають можливість роботи з

електронною поштою, участь у відеоконференціях, доступ до файлових архівів FTP, а також підтримують голосовий та відеозв'язок. Вони зберігають історію відвідувань користувачем інформаційних ресурсів для подальшого пошуку, а також надають можливість зберігання веб-ресурсів у вигляді закладок для швидкого доступу [25].

3.1.1. Мова гіпертекстової розмітки HTML:

HTML (Hypertext Markup Language) є основною мовою розмітки для створення веб-сторінок та веб-додатків. Вона дозволяє створювати структуру сторінки та визначати взаємозв'язки між різними елементами на ній [6].

HTML-документи, як правило, містять текст, зображення, гіперпосилання та інші компоненти. Розмітка визначає структуру документа, включаючи параграфи, заголовки, списки, таблиці та інші елементи. Окрім того, за допомогою HTML можна вбудовувати медіа-файли, такі як зображення, аудіо та відео.

HTML дозволяє створювати гіперпосилання, які дозволяють користувачам переходити з однієї веб-сторінки на іншу. Це робить його незамінним інструментом для розробки веб-сайтів та веб-додатків.

При розробці інтерфейсу користувача за допомогою HTML створюються різноманітні елементи, такі як кнопки, поля вводу, тексти, вікна тощо (рис. 3.1).

```
<html>
    <button id="my-button">Натисніть мене</button>
</html>
```

Рисунок 3.1 – HTML-код

Такий підхід дозволяє забезпечити зручну взаємодію користувача з веб-сторінкою або веб-додатком, а також створити чітку та зрозумілу структуру для їх відображення у веб-браузері [6].

3.1.1 Каскадні таблиці стилів (CSS):

CSS (Cascading Style Sheets) - це мова опису стилів, яка використовується для форматування веб-сторінок. Її основна функція полягає у візуальному оформленні елементів HTML, задаючи їм кольори, шрифти, розміри, відступи та інші візуальні властивості.

Переваги використання CSS:

- **Економія часу:** CSS дозволяє стилізувати елементи HTML один раз, а потім повторно використовувати цей код на багатьох сторінках.
- **Швидше завантаження:** CSS зменшує обсяг HTML-коду, адже атрибути стилів винесені в окремі файли.
- **Просте обслуговування:** Зміни в стилі можна внести централізовано, що оновить всі сторінки, де використовується цей стиль.
- **Кращі стилі:** CSS пропонує значно ширший набір атрибутів, ніж HTML, що дає більше можливостей для дизайну.
- **Мультиплатформність:** CSS дозволяє адаптувати веб-сторінки для різних пристроїв, таких як мобільні телефони чи планшети.
- **Сучасний веб-стандарт:** CSS є загальноприйнятим стандартом для стилізації веб-сторінок, роблячи їх сумісними з сучасними та майбутніми браузерами [6].

Як працює CSS:

- **Селектори:** CSS використовує селектори для визначення, до яких елементів HTML буде застосовано стиль. Селектори можуть бути засновані на типі елемента, його атрибутах, розташуванні в DOM-дереві або інших факторах.
- **Правила:** Правила CSS складаються з селектора та блоку оголошення, де описуються візуальні властивості елемента.
- **Каскадність:** CSS правила можуть каскадуватися, тобто на один елемент може впливати кілька правил. Браузер визначає, яке правило має більший пріоритет, і використовує його.

- **Спадкування:** Деякі властивості CSS можна успадковувати від одного елемента до його нащадків [6].

CSS в плагінах:

CSS використовується для стилізації інтерфейсу користувача плагінів. Він визначає зовнішній вигляд елементів HTML, роблячи його візуально привабливим і зручним для користувачів (рис. 3.2).

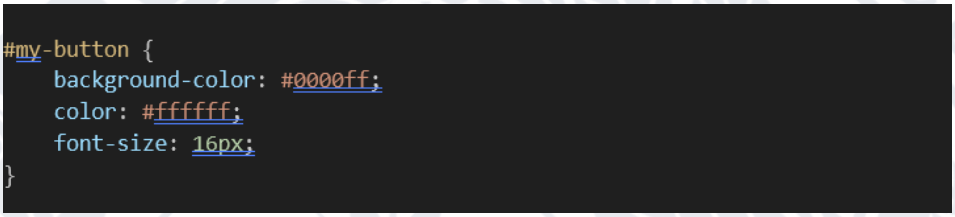


Рисунок 3.2 – CSS-код

3.1.2 JavaScript:

JavaScript - це потужна мова програмування, що використовується для створення динамічних веб-сторінок. Вона дає можливість реалізувати інтерактивність, обробляти дії користувачів, маніпулювати DOM (Document Object Model) та взаємодіяти з сервером [16].

Переваги JavaScript**:**

- **Мультипарадигменна:** JavaScript підтримує об'єктно-орієнтований, імперативний та функціональний стилі програмування.
- **Універсальна:** JavaScript використовується не лише у веб-розробці, але й у мобільних додатках, серверних програмах, іграх та інших сферах.
- **Простота вивчення:** JavaScript має лаконічний синтаксис, схожий на інші популярні мови.

JavaScript в плагінах:

JavaScript використовується для реалізації функціональності плагінів (рис. 3.3). Це може включати:

- **Обробка подій:** JavaScript може реагувати на дії користувача, такі як

клікання по кнопках, введення тексту або прокручування сторінки.

- **Зміна DOM:** JavaScript може змінювати вміст веб-сторінки, додавати або видаляти елементи, змінювати стилі та анімацію.
- **Взаємодія з сервером:** JavaScript може надсилати запити на сервер, отримувати та обробляти дані [16].

```
document.getElementById("my-button").onclick = function() {
    // Функціональність кнопки
};
```

Рисунок 3.3 – JavaScript-код

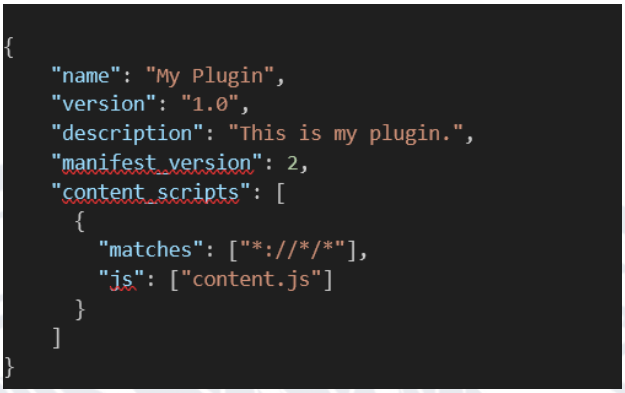
3.1.3 JSON:

JSON (JavaScript Object Notation) - це формат обміну даними, який використовується для зберігання та передачі інформації. Він легкий, читабельний людьми та машинно-зрозумілий, що робить його популярним вибором для веб-розробки [15].

JSON в плагінах:

JSON використовується для опису метаданих плагіну в файлі manifest.json (рис.3.4). Цей файл містить інформацію про:

- **Назву плагіну:** Користувацьке ім'я плагіну.
- **Версію плагіну:** Ідентифікатор версії плагіну.
- **Опис плагіну:** Короткий опис функціональності плагіну.
- **Автора плагіну:** Ім'я або організацію, яка розробила плагін.
- **Механізми підключення:** Визначає, як плагін підключається до веб-сторінок.
- **Інші параметри:** Додаткові налаштування та конфігурації плагіну.



```
{
    "name": "My Plugin",
    "version": "1.0",
    "description": "This is my plugin.",
    "manifest_version": 2,
    "content_scripts": [
      {
        "matches": ["*://*/*"],
        "js": ["content.js"]
      }
    ]
}
```

Рисунок 3.4 – JSON-код

3.1.4 Середовище розробки Visual Studio Code (VS Code):

Visual Studio Code (VS Code) - це безкоштовне, багатофункціональне середовище розробки (IDE), яке використовується для написання, налагодження та тестування коду. VS Code підтримує широкий спектр мов програмування та пропонує безліч розширень, що робить його популярним вибором як для новачків, так і для досвідчених розробників [22].

Переваги VS Code:

- **Безкоштовне та відкрите джерело:** VS Code доступний безкоштовно для всіх платформ (Windows, macOS, Linux) та має відкритий код, що дає можливість спільноті розробників вносити свій вклад у його розвиток.
- **Легкий та швидкий:** VS Code має невеликий розмір та швидко запускається, що робить його зручним для роботи з проектами будь-якого розміру.
- **Підтримка багатьох мов:** VS Code підтримує понад 30 мов програмування з коробки, а також пропонує безліч розширень для підтримки інших мов.
- **Розширений набір інструментів:** VS Code пропонує широкий спектр інструментів для розробки, таких як автодоповнення коду, підсвічування синтаксису, форматування коду, інтеграція з системами керування версіями (Git, SVN), відладчик та багато

іншого.

- **Розширення та налаштування:** VS Code має велику екосистему розширень, що дозволяє персоналізувати ваше середовище розробки [22].

VS Code використовується як основне середовище для написання коду плагінів. VS Code надає такі можливості:

- **Автодоповнення коду:** VS Code пропонує автодоповнення коду для JavaScript, JSON та інших мов програмування, що використовується в розробці плагінів.
- **Підсвічування синтаксису:** VS Code підсвічує синтаксис коду, що робить його більш читабельним і зручним для редагування.
- **Відладчик:** VS Code має вбудований відладчик, який дозволяє покроково виконувати код плагіну, знаходити та виправляти помилки.
- **Інтеграція з Git:** VS Code може інтегруватися з Git, що дозволяє легко відстежувати зміни в коді плагіну та співпрацювати з іншими розробниками [24].
- **Розширення:** VS Code пропонує безліч розширень для покращення робочого процесу розробки плагінів, таких як linting, formatting, and testing tools [6].

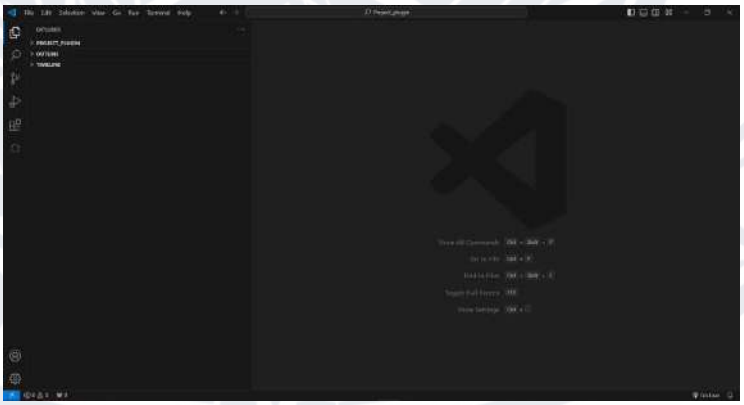


Рисунок 3.5 – Середовище розробки Visual Studio Code

Висновки до розділу 3

У третьому розділі розглянуто технології та інструменти, використані для розробки плагіну "CookieShield". Вибір технологічного стеку був обґрунтований простотою, широкою підтримкою та гнучкістю.

Архітектура плагіну включала різні компоненти, такі як manifest.json, background.js, та файли contentScript.js, HTML та CSS для ін'єкції на веб-сторінки та керування файлами cookie.

РОЗДІЛ 4. АРХІТЕКТУРНІ ТА СТРУКТУРНІ ОСОБЛИВОСТІ ПЛАГІНУ

4.1 Архітектура плагіну

Зазвичай, плагіни складаються з компактних наборів HTML, CSS, JavaScript коду, зображень та інших файлів, які використовуються на веб-платформі та налаштовуються для взаємодії з браузером. Розширення розробляються з використанням веб-технологій і можуть користуватися тими ж API-інтерфейсами, які надає браузер для взаємодії з мережею [22].

Плагіни мають різноманітний функціонал, який може включати зміну веб-контенту, який бачать користувачі, взаємодіювати з ним або розширювати його, а також змінювати поведінку самого браузера.

Розробка плагіну дозволяє розширити функціональність веб-браузера без необхідності докладного вивчення його власного коду. Для створення плагіну використовуються такі технології, як HTML, CSS, JavaScript, Ajax, а також бібліотека JQuery. За допомогою цих технологій розробляються структурні файли, що містять код для реалізації плагіну [16].

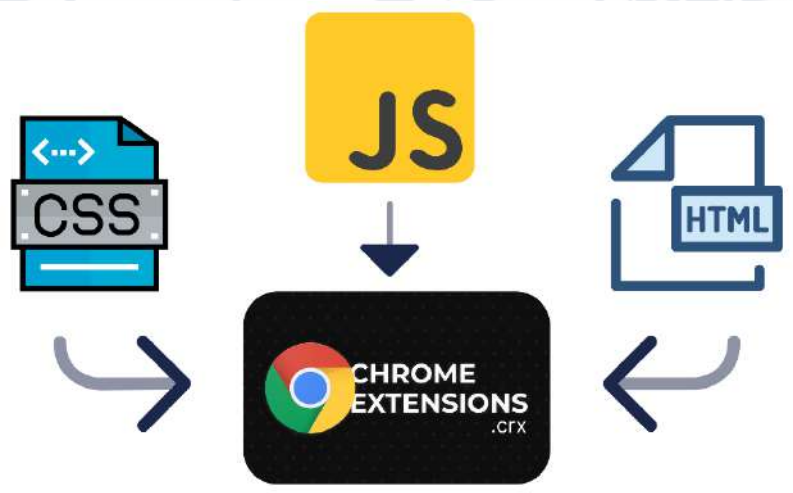


Рисунок 4.1 – Архітектура розширення

Відповідно до конкретних потреб і особливостей втілення, структура плагіну може залишатися незмінною. Хоча архітектура розширення буде

визначатися його функціональністю, практично всі розширення мають загальні складові:

- Опис (Manifest) - Опис структури та параметрів розширення.
- Сценарій фону (Background Script) - Сценарій, що виконується в фоновому режимі та керує функціональністю розширення.
- Елементи користувацького інтерфейсу (User Interface Elements) - Елементи користувацького інтерфейсу, такі як кнопки, панелі або спливаючі вікна.
- Сценарій вмісту (Content Script) - Сценарій, який взаємодіє з вмістом веб-сторінок, зазвичай використовується для модифікації або збільшення функціоналу сторінок.
- Сторінка параметрів (Options Page) - Сторінка налаштувань, де користувач може змінювати параметри розширення.

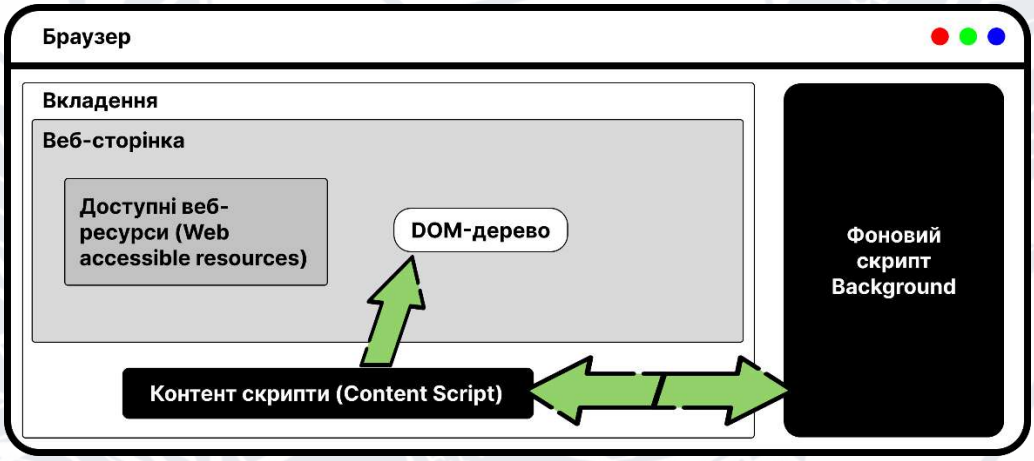


Рисунок 4.2 – Архітектура стандартного розширення

4.2. Аналіз файлової структури розширення

4.2.1. Manifest.json

Manifest.json - це основний файл, який містить інформацію про необхідні дозволи для правильної роботи плагіну, а також дані про підключені файли,

метадані, такі як версія, автор, опис та налаштування безпеки.

Основні скрипти, які включає файл, включають такі елементи:

- Manifest_version - версія маніфесту файлу, що визначається.
- Description – опис плагіну.
- Permission - масив з переліком необхідних дозволів для коректної роботи плагіну.
- Content_scripts - масив файлів, які підключаються як скрипти контенту.
- Background - опис файлів, які запускаються у фоновому режимі.
- Browser_action - оперативний документ, який автоматично активується при взаємодії з іконкою у панелі інструментів.
- Icons - зображення іконки зі стандартними розмірами, такими як 28 пікселів, 48 пікселів або 128 пікселів.

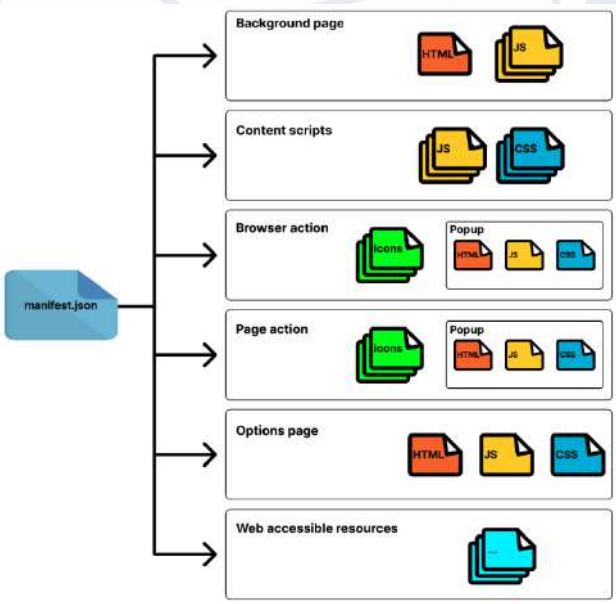


Рисунок 4.3 – Взаємозв'язок між файлом manifest.json та іншими файлами.

У маніфесті можуть бути вказані лише певні посилання, але окрім них розширення може містити і додаткові сторінки та допоміжні файли [7].

Маніфест у форматі JSON містить інформацію про програму, таку як назву, автора, іконку та опис. Його головна мета - забезпечити встановлення веб-

додатку на екрані пристрою, надаючи користувачеві швидкий доступ та розширені можливості.

4.2.2. Фоновий скрипт (Background)

Background.js - це файл, що містить основну логіку плагіну. Його особливість полягає в тому, що він автоматично активується при запуску браузера і залишається активним в його оперативній пам'яті як фоновий процес протягом усієї сесії.

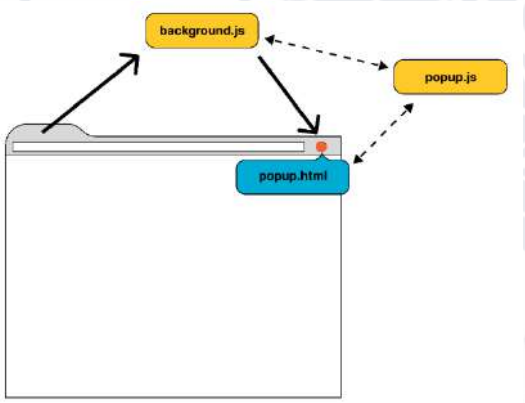


Рисунок 4.3 – Обов'язки Background.js.

Background.js можна підключити у файлі manifest.json, додавши наступний фрагмент коду:

```
"background": {
    "scripts": ["background.js"]
}
```

У 2012 році, з метою оптимізації ресурсів, які використовуються, було створено Event Pages.

Event Pages автоматизує роботу браузера за рахунок заміни постійної роботи фонового скрипту, івент-сторінка запускається лише у випадках, коли це необхідно, наприклад, для обробки конкретної події. Після цього очищується пам'ять, до моменту, коли виникає необхідність обробки події повторно [9].

З погляду програмування, обидва ці підходи є еквівалентними. Необхідно

лише визначити властивість persistent, яка знаходиться в маніфесті. Якщо ви використовуєте івент-сторінку, потрібно вказати "false", так як стандартне значення "true" [20].

4.2.3. Скрипти контенту(Content scripts)

Content scripts - це код, написаний мовою JavaScript, який працює в контексті веб-сторінки, відповідно цей скрипт не працює на рівні фонового скрипту.

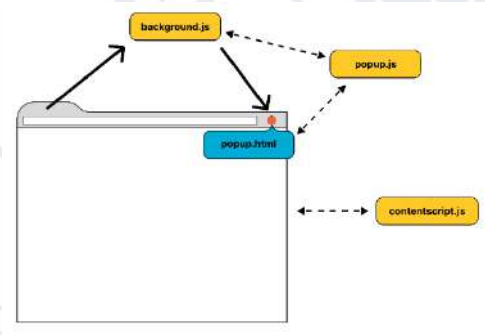


Рисунок 4.4 – Робота скрипта контенту

Контент-скрипти використовують обмежені API, і вони ізольовані від функцій та змінних, оголошених на фоновій сторінці або в інших скриптах на веб-сторінки (рис. 4.4). Вони можуть зчитувати та змінювати DOM, який переглядає користувач у веб-переглядачі [20].

Контент-скрипти можуть взаємодіяти зі своїм основним розширенням, обмінюючись повідомленнями та зберігаючи дані за допомогою API сховища (див. мал. 4.5).

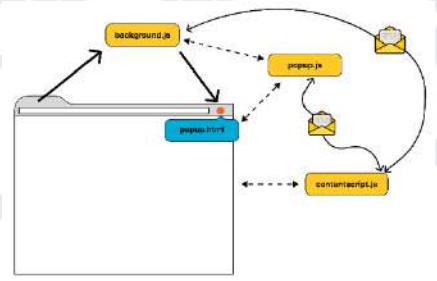


Рисунок 4.5 – Робота Content Script з обміном API

Таким чином, повноцінний доступ до скриптів можливий лише через DOM-дерево сторінки [24]. Контент-скрипти можуть ініціювати події та змінювати DOM. Крім того, є можливість додавати теги script на сторінку і підключати потрібні файли (рис. 4.6).

Таким чином, для отримання повного доступу до скрипту, необхідно мати доступ до DOM-дерева веб-сторінки[24]. Дані скрипти можуть ініціювати події та модифікувати DOM-дерево (додавання тегів, підключення файлів).

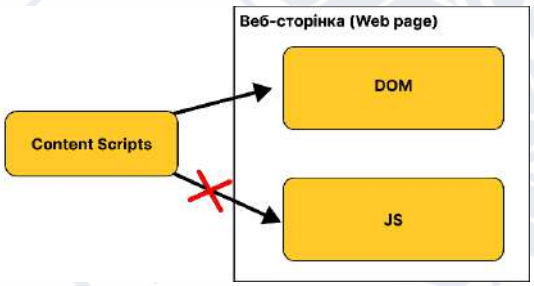


Рисунок 4.6 – Можливості скриптів контенту

4.2.4. Елементи користувацького інтерфейсу

Index.html - це файл, який описує ключову структуру плагіну. Він містить декларацію типу документу, шапку та тіло документа. Активація файлу відбувається при кліку на клавішу плагіну, яка є іконкою, та відображається в поточному вкладені.

Main.css – файл, який вміщає в себе каскадні таблиці стилів (CSS), для коректного відображення. Підключення файлу відбувається шляхом використання тегу <link>. Мета даного делегування файлів полягає в автоматизації та розділені файлів, які вміщують основний код логіки та стилі оформлення [20].

Також, файл reset.css дозволяє "скинути" стандартні значення стилів з метою забезпечення єдиності і консистентності вигляду елементів на різних

браузерах. Файл змінює (або "скидає") стандартні стилі, що застосовуються до різних HTML-елементів, і встановлює власні значення, які дозволяють здійснити більш точний контроль над оформленням. Це допомагає уникнути проблем зі сумісністю стилів між різними браузерами і забезпечити більш передбачуваний результат.

Головні компоненти, які використовуються під час створення плагіна, включають:

- Manifest.json
- Background.js
- Popup.html
- Popup.js

Додаткові компоненти:

- Jquery.js
- object-watch.js
- i18n_translator.js

4.2.5. Політика безпеки контенту

Розширення браузерів використовують так звану політику безпеки контенту (Content Security Policy), яка вміщує в собі правила та вимого. Ці правила необхідні для забезпечення безпеки розширення та контролю контенту, який завантажується [10].

Якщо маніфест використовує версію 2, плагін має обмеження:

- Використання функцій eval та подібних заборонено.
- Inline JavaScript не буде виконуватися.
- Можливість завантажувати лише локальні скрипти та ресурси.

Якщо веб-браузер підтримує Content Security Policy (CSP), то він зможе працювати з сервером, в якого є підтримка CSP. Проте, якщо веб-браузер не підтримує CSP, то він ігнорує дану політику, та продовжує свою діяльність використовуючи стандартні правила завантаження контенту [20].

Конфігурація CSP включає додавання HTTP-заголовка Content-Security-Policy на сторінку і налаштування його. При цьому використовується список джерел, з яких користувачеві дозволено отримувати контент [26].

4.2.6. JSON (JavaScript Object Notation)

JavaScript Object Notation (JSON) – це формат представлення даних, побудова якого полягає з JavaScript об’єктів. Головна мета даного формату, це передача даних між сервером та клієнтом. [26].

JSON має розширення .json. Дані, які мають інший формат необхідно помістити в лапки, щоб була можливість розмістити їх в JSON файлі [11].

Переваги використання формату JSON для передачі даних включають наступне:

- Зниження загального трафіку на 28% завдяки швидкості передачі даних.
- Підтримка всіх основних типів даних, що можуть легко серіалізуватися (перетворюватися) з будь-якої мови програмування.
- Мінімальне семантичне уявлення, що дозволяє значно зменшити розмір відповіді сервера та навантаження на нього [21].

Висновки до розділу 4

У четвертому розділі детально вивчено архітектуру та файли, які використовує плагін "CookieShield". Розглянуті архітектурні складові, такі як файл Manifest.json, та фоновий скрипт, що відповідає за фонову роботу плагіну.

Також детально розглянуто Content Script, що взаємодіє з DOM веб-сторінок, а також роль елементів інтерфейсу користувача.

РОЗДІЛ 5. СТВОРЕННЯ ПЛАГІНУ: РОЗРОБКА

5.1. Огляд файлової структури плагіну

Плагін структурується за допомогою HTML розмітки, CSS стилізації та JavaScript логіки, що визначає його поведінку. Додаткові можливості розробника надаються підключеними бібліотеками.

Для структуризації проекту, усі файли, які використовує плагін, розміщено в відокремлених папках (рис.5.1).

5.1.1. Manifest.json

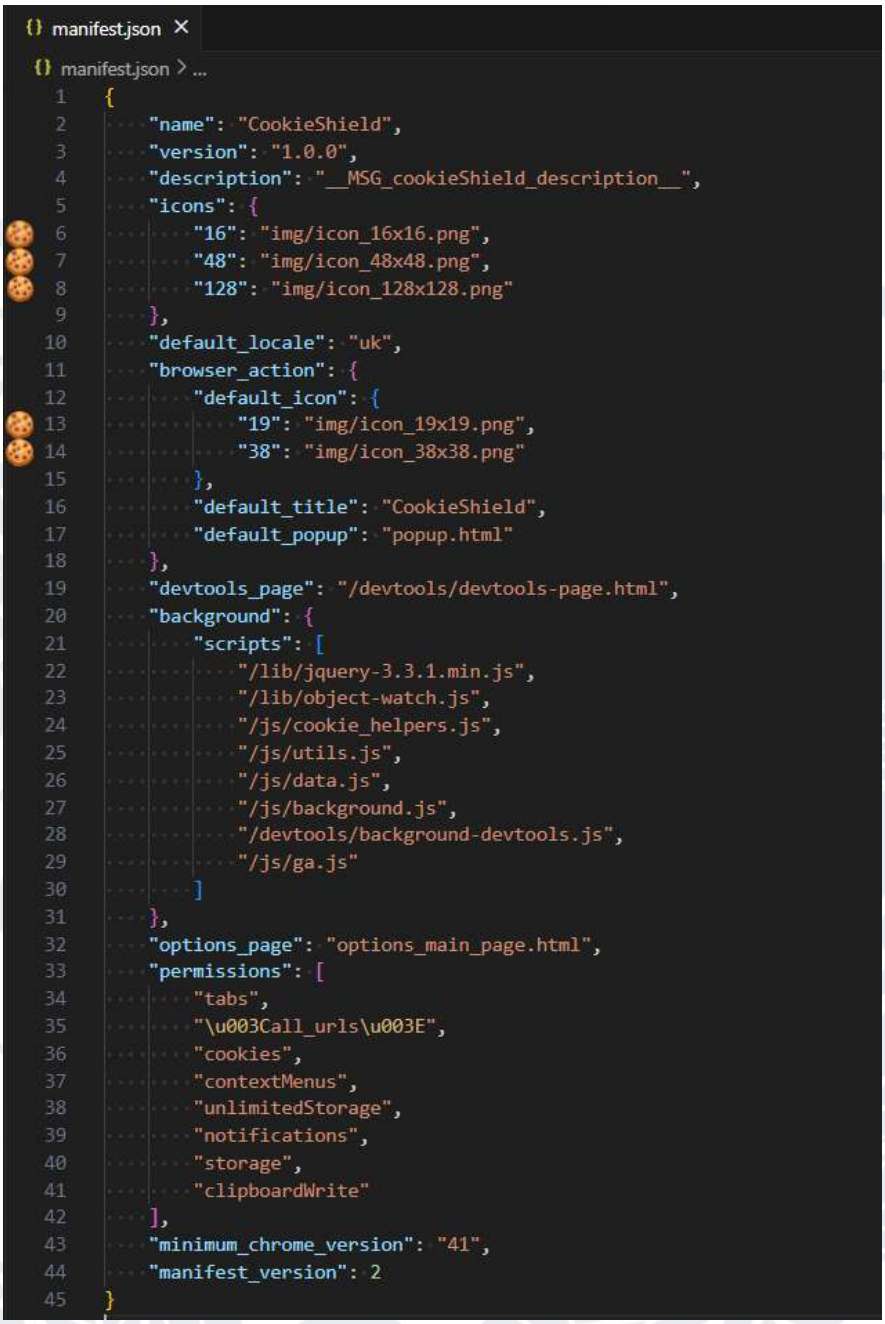
Файл Manifest.json є ключовим елементом, який вміщає конфігурацію про необхідні доступи для правильної роботи плагіну (рис.5.2).

Включені в файл скрипти включають:

- Manifest_version: Версія маніфесту.
- Name: Ім'я плагіну.
- Author: Інформація про розробника.
- Version: Поточна версія.
- Web accessible resources: Підключені файли.
- Description: Опис плагіну.
- Permissions: Потрібні доступи для роботи плагіну.
- Content_scripts: Скрипти, що включаються як контент.
- Background: Скрипти для фонового режиму.
- Browser_action: Скрипти, які виконуються при натисканні на іконку в панелі інструментів.
- Icons: Іконка зі стандартним розміром.

Рисунок 5.1 – Структура проекту

Manifest.json, вміщає ключову інформацію про проект (авторство, іконка,опис і тд.).



manifest.json

```
{
    "name": "CookieShield",
    "version": "1.0.0",
    "description": "__MSG_cookieShield_description__",
    "icons": {
        "16": "img/icon_16x16.png",
        "48": "img/icon_48x48.png",
        "128": "img/icon_128x128.png"
    },
    "default_locale": "uk",
    "browser_action": {
        "default_icon": {
            "19": "img/icon_19x19.png",
            "38": "img/icon_38x38.png"
        },
        "default_title": "CookieShield",
        "default_popup": "popup.html"
    },
    "devtools_page": "/devtools/devtools-page.html",
    "background": {
        "scripts": [
            "/lib/jquery-3.3.1.min.js",
            "/lib/object-watch.js",
            "/js/cookie_helpers.js",
            "/js/utils.js",
            "/js/data.js",
            "/js/background.js",
            "/devtools/background-devtools.js",
            "/js/ga.js"
        ]
    },
    "options_page": "options_main_page.html",
    "permissions": [
        "tabs",
        "\u003Call_urls\u003E",
        "cookies",
        "contextMenus",
        "unlimitedStorage",
        "notifications",
        "storage",
        "clipboardWrite"
    ],
    "minimum_chrome_version": "41",
    "manifest_version": 2
}
```

Рисунок 5.2 – Файл manifest.json

5.1.2. Фоновий скрипт (Background)

Background.js - це файл, який містить основну логіку роботи плагіну. Одна з ключових особливостей цього скрипту полягає в тому, що активація відбувається під час запуску браузера і залишається в його оперативній пам'яті як фоновий процес протягом усієї сесії (рис.5.3).

Рисунок 5.3 – Обов’язки фонового скрипту

Комбінація клавіш "Shift+Esc" надає можливість перегляду списку подій, які виконуються в поточний момент браузера.

В даному списку відображається background-сторінка плагіну (рис.5.4).

```
background.js
js > background.js > setContextMenu
1   var showContextMenu = undefined;
2
3   updateCallback = function () {
4       if (showContextMenu !== preferences.showContextMenu) {
5           showContextMenu = preferences.showContextMenu;
6           setContextMenu(showContextMenu);
7       }
8   };
9
10
11  localStorage.setItem("option_panel", "null");
12
13  var currentVersion = chrome.runtime.getManifest().version;
14  var oldVersion = data.lastVersionRun;
15
16  data.lastVersionRun = currentVersion;
17
18  if (oldVersion !== currentVersion) {
19      var opt = {
20          type: "basic",
21          title: "CookieShield",
22          message: _getMessage("updated"),
23          iconUrl: "/img/icon_128x128.png",
24          isClickable: true
25      };
26      chrome.notifications.create("", opt, function (notificationId) {
27      });
28  }
29
30  setContextMenu(preferences.showContextMenu);
31
32  chrome.cookies.onChanged.addListener(function (changeInfo) {
```

Рисунок 6.4 – Файл Background.js

5.1.3. Popup.html

Popup.html - це файл, що містить загальну структуру розмітки плагіну. У цьому файлі описується тип документу, заголовок та тіло документа. Він включає елементи, які відповідають за зовнішній вигляд плагіну.

Popup.html відображається у відкритому вікні при натисканні на зображення плагіну.

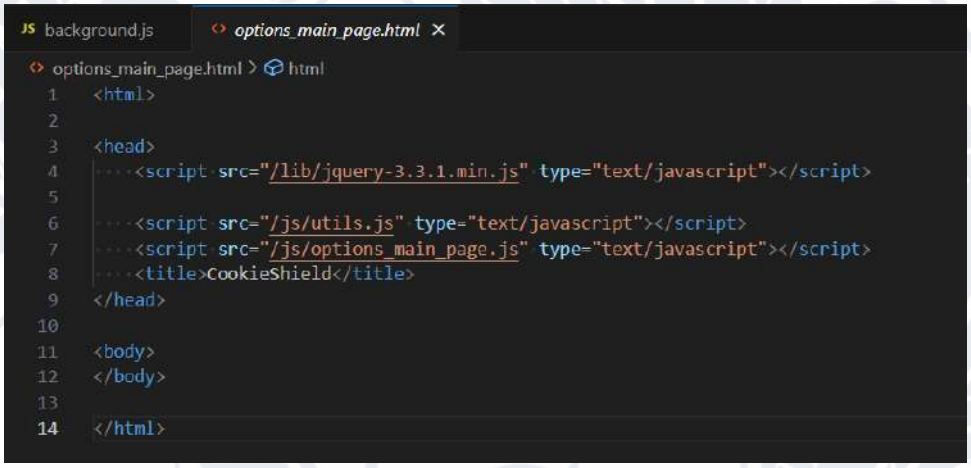


Рисунок 5.5 – Файл Popup.html

5.1.4. Файли каскадних таблиць стилів

Файли вміщають стилі для коректного відображення елементів, які відносяться до структури плагіну.

Всі файли даного розширення об’єднанні в папку "css" (рис. 5.6).

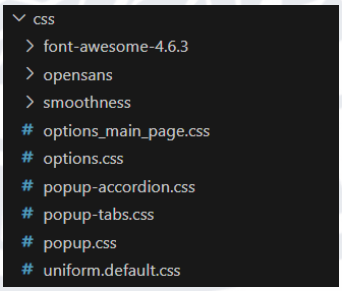


Рисунок 5.6 – CSS-файли

5.2. Вікно "Popup"

5.2.1. Отримання даних Cookies

Для отримання даних про куки необхідно скористатися вбудованими функціями браузера. У функції **createList** (рис.5.7) використовуються методи розширення браузера Chrome, щоб отримати доступ до списку кукісів. Після виклику функції **getAllCookieStores**, отримані дані про зберігання кукісів використовуються для фільтрації кукісів на поточній вкладці. Потім застосовуються фільтри до кукісів за допомогою функції **getAll**.

Отримані кукіси додатково фільтруються залежно від переданих параметрів, таких як ім'я, домен, безпека та сесійність. Крім того, для деяких кукісів може встановлюватися властивість **isProtected**, яка вказує на те, що куки є захищеними та недоступними для зміни.

Після фільтрації кукісів вони відображаються у відповідному розділі веб-сторінки. Якщо кукісів немає або жоден з них не відповідає заданим критеріям, виконується відповідний обробник. В іншому випадку, кукіси сортуються згідно з вказаними у налаштуваннях користувача параметрами та відображаються у відповідному порядку.

```
function createList(filters, isSeparateWindow) {
    var filteredCookies = [];

    if (filters === null)
        filters = {};

    var filterURL = {};
    if (filters.url !== undefined)
        filterURL.url = filters.url;
    if (filters.domain !== undefined)
        filterURL.domain = filters.domain;

    if (!isSeparateWindow) {
        $('#submitDiv').css({
            'bottom': 0
        });
    } else {
        $('#submitDiv').addClass("submitDivSepWindow");
    }

    chrome.cookies.getAllCookieStores(function (cookieStores) {
        for (let x = 0; x < cookieStores.length; x++) {
            if (cookieStores[x].tabIds.indexOf(currentTabID) != -1) {
                filterURL.storeId = cookieStores[x].id;
                break;
            }
        }

        chrome.cookies.getAll(filterURL, function (cks) {
            var currentC;
            for (var i = 0; i < cks.length; i++) {
                currentC = cks[i];

                if (filters.name != undefined && currentC.name.toLowerCase().indexOf(filters.name.toLowerCase()) === -1)
```

Рисунок 5.7 – Функція createList

5.2.2. Генерація таблиці з даними про вхідні кукі

Отримавши список куків функцією **createAccordionList** (рис.5.8), результат зберігається у відповідній змінній. Далі, для виведення кожного елементу списку у вигляді аккордеону, використовується метод accordion() бібліотеки jQuery UI.

Під час створення кожного елементу аккордеону, відбувається формування відповідних полів для відображення даних про куки. Наприклад, для кожного елементу куки формуються поля для ідентифікації, імені, значення, домена, шляху, ідентифікатора зберігання, а також інших характеристик.

Крім того, відповідно до налаштувань користувача, виводяться додаткові відомості, такі як домен, безпека, та інші.

Кожен елемент аккордеону містить відповідні дані куки та можливість редагування їх параметрів.

Після створення аккордеону (рис.5.9), викликається зазначена функція з колбеком для оновлення інтерфейсу або виконання інших дій залежно від контексту.

```
function createAccordionList(cks, callback, callbackArguments) {
    let createAccordionCallback = callback;
    let createAccordionCallbackArguments = callbackArguments;

    try {
        $("#cookiesList").accordion("destroy");
    } catch (e) {
        console.warn(e.message)
    }

    if (cks === null)
        cks = cookieList;
    for (var i = 0; i < cks.length; i++) {
        currentC = cks[i];

        var domainText = "";
        if (preferences.showDomain) {
            domainText = currentC.domain;
            if (preferences.showDomainBeforeName) {
                domainText = domainText + " | ";
            } else {
                domainText = " | " + domainText;
            }
        }

        var titleText;
        if (preferences.showDomainBeforeName) {
            titleText = $("<p/>").text(domainText).append($("<b/>").text(currentC.name));
            if (currentC.isProtected)
                $(":first-child", titleText).css("color", "green");
        } else {
            titleText = $("<p/>").append($("<b/>").text(currentC.name)).append($("<span/>").text(domainText));
        }

        var titleElement = $("<h3/>").append($("<a/>").html(titleText.html()).attr("href", "#"));
```

Рисунок 5.8 – Функція createAccordionList

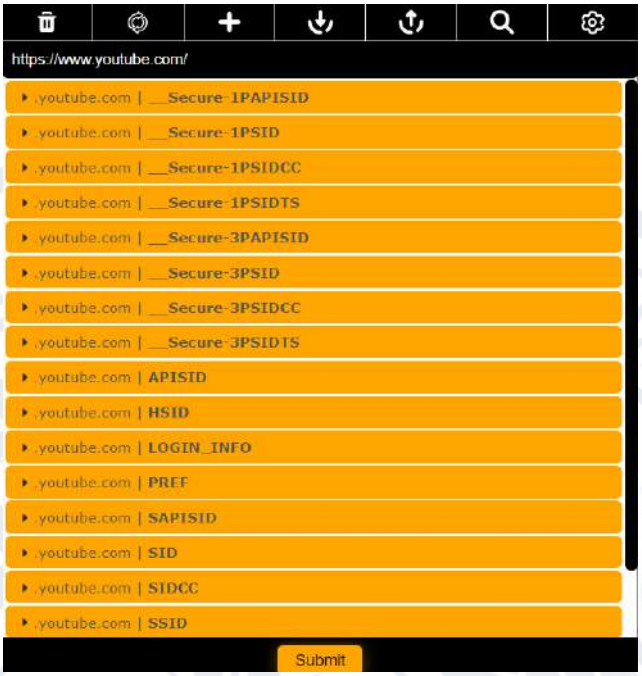


Рисунок 5.9 – Таблиця з даними про кукі

5.2.3. Пошук по таблиці

У вікні "Popup" вкладено функціонал пошуку за іменем або доменом кука у таблиці даних. Користувач може вводити текст у текстове поле, щоб здійснити пошук за іменем або доменом кука.

Механізм пошуку базується на порівнянні введеного тексту з іменем або доменом кука в таблиці. Для забезпечення коректної роботи пошуку весь текст, який користувач вносить в поле для пошуку буде зведено до нижнього регістру.

Пошук виконується тільки під час введення більше ніж двох символів у поле пошуку, що сприяє ефективній фільтрації результатів. Для порівняння тексту та пошуку використовується метод indexOf(), який повертає індекс елемента, що відповідає умові. В результаті знайдений елемент з таблиці відображається на екрані (рис.5.10).

Таким чином, користувач може швидко знаходити необхідні куки за їх іменем або доменом, забезпечуючи зручну навігацію та управління даними.

Рисунок 5.10 – Реалізація функції пошуку з таблиці

5.2.4. Оновлення інформації про кукі файли

У вікні "Popup" ми можемо активувати функцію оновлення даних натисканням клавіші з відповідною іконкою на панелі інструментів плагіну. При кліку на іконку користувач запускає обробник подій, який виконує очищення поля пошуку та виклик функції clearNewCookieData ().

Функція clearNewCookieData () (рис.5.11) відповідає за оновлення списку кукісів. clearNewCookieData () надсилає запит для отримання актуальних даних про куки, який описаний відповідним пунктом в документації.

```
function clearNewCookieData() {
    var cookieForm = $("#newCookie");
    $(".index", cookieForm).val("");
    $(".name", cookieForm).val("");
    $(".value", cookieForm).val("");
    $(".domain", cookieForm).val(getHost(getUrlOfCookies()));
    $(".hostOnly", cookieForm).prop("checked", false);
    $(".path", cookieForm).val("/");
    $(".secure", cookieForm).prop("checked", false);
    $(".httpOnly", cookieForm).prop("checked", false);
    $(".session", cookieForm).prop("checked", false);

    var expDate = new Date();
    expDate.setFullYear(expDate.getFullYear() + 1);
    $(".expiration", cookieForm).val(expDate);

    $.uniform.update();
}
```

Рисунок 5.11 – Функція clearNewCookieData

5.2.5. Видалення поточних куків

При кліку на клавішу з відповідною іконкою на панелі інструментів плагіну, яка має ідентифікатор deleteAllButton (рис.5.12), ми запускаємо обробник подій, який виконує функцію для видалення всіх поточних куків.

Спочатку перевіряється, чи список куків не є порожнім. Після цього викликається функція deleteAll(), яка видаляє всі куки. Після видалення куків оновлюється кількість видалених куків у змінній data.nCookiesDeleted, і викликається функція doSearch() для оновлення відображення списку куків.

```
$("#deleteAllButton").unbind().click(function () {
    if (cookieList.length === 0)
        return false;

    var okFunction = function () {
        nCookiesDeletedThisTime = cookieList.length;
        deleteAll(cookieList, getUrlOfCookies());
        data.nCookiesDeleted += nCookiesDeletedThisTime;
        doSearch();
    }
    startAlertDialog(_getMessage("Alert_deleteAll"), okFunction);
});
```

Рисунок 5.12 – Обробний події deleteAllButton

5.2.6. Імпорт/Експорт куків

Для імпорту куків просто клацніть на кнопку "Імпортувати куки" (рис.5.13), вставте куки у форматі JSON і клацніть кнопку "Підтвердити зміни куки". Для отримання прикладу прийнятого формату JSON просто експортуйте кілька куків і перегляньте скопійований текст. Імпорт або експорт куків корисний для обміну куків між різними ПК, тестування веб-сайтів та резервного копіювання.

Рисунок 5.13 – Застосування імпорту

Для експорту куків клацніть на кнопку "Експортувати куки" (рис.5.14). Куки поточної сторінки тепер скопійовано до буфера обміну. Ви можете вставити їх у файл, електронний лист тощо. В налаштуваннях розширення ви можете змінити формат, у якому експортуються куки. Доступні формати:

- JSON
- Файл куків HTTP Netscape
- Розділені крапкою з комою пари ім'я=значення
- Perl::LWP

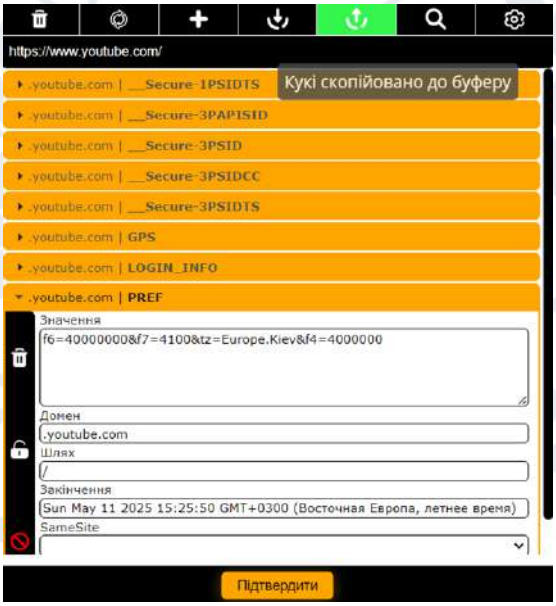


Рисунок 5.14 – Застосування експорту

5.2.7. Редагування куків

Після клацання на значок куки ви побачите куки (якщо вони є), збережені браузером для поточної вкладки. Виберіть куки, які ви хочете відредагувати, клацнувши на їх назву у списку. Змініть значення, які вам потрібні, і клацніть кнопку "Підтвердити" (рис.5.15).

Також доступні такі опції для редагування куків:

- **Блокування:** Блокування файлів cookie може бути корисним з багатьох причин, наприклад, для захисту вашої конфіденційності, зміни поведінки браузера під час відвідування певних веб-сайтів, тестування веб-розробки тощо. Щоб заблокувати файли cookie, ви можете натиснути кнопку "Заблокувати файли cookie". Після цього ви можете вибрати, за якими параметрами слід блокувати файли cookie.
- **Захист:** Щоб захистити файл cookie, ви можете натиснути кнопку "Установити лише для читання". Захист файлу cookie означає, що веб-сайт, який створив файл cookie, більше не зможе змінити його значення.
- **Зміна файлів:** Натиснувши піктограму cookie, ви побачите файли cookie (якщо такі є), збережені браузером для поточної вкладки. Виберіть файл cookie, який потрібно редагувати, клацнувши на його ім'я у списку. Змініть потрібні значення та натисніть кнопку "Підтвердити зміни файлів cookie".
- **Властивості:** Нижче наведені властивості файлів cookie разом з їх поясненнями (рис. 5.15).

Властивість	Тип	Пояснення
name	string	Назва куки.
value	string	Значення куки.
domain	string	Домен куки (наприклад, “www.google.com”, “example.com”).
hostOnly	boolean	Логічне значення, що вказує, чи є кука кукою тільки для хоста (тобто, хост запиту повинен точно відповідати домену куки).
path	string	Шлях куки.
secure	boolean	Логічне значення, яке вказує, чи позначена кука як безпечна (тобто, її область дії обмежена безпечними каналами, зазвичай HTTPS).
httpOnly	boolean	Логічне значення, яке вказує, чи позначена кука як HttpOnly (тобто, кука недоступна для сценаріїв на клієнтському боці).
session	boolean	Логічне значення, яке вказує, чи є кука сеансовою кукою, на відміну від постійної куки з датою закінчення.
expirationDate	double	Дата закінчення куки у форматі кількості секунд з початку епохи UNIX. Не надається для сеансових куки.
storeId	string	Ідентифікатор сховища куки, що містить цю куку, як надано у функції getAllCookieStores ().

Рисунок 5.15 – Властивості кукі

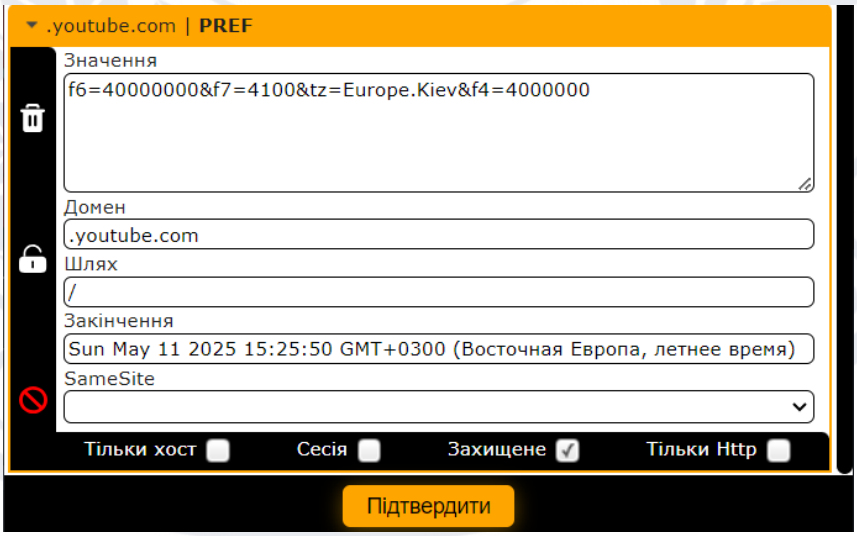


Рисунок 5.16 – Застосування редагування

5.3. Вікно "Options"

5.3.1. Відкриття Options

Після кліку на кнопку "Options", ви будете перенаправлені на сторінку налаштувань, де зможете налаштувати плагін (рис.5.17).

Перелік налаштувань включає в себе:

- Завжди запитувати підтвердження: Показувати сповіщення для підтвердження дій.
- Показувати надписи біля іконок: Відображати підписи біля іконок.
- Оновлювати сторінку після підтвердження змін кукі: Автоматично оновлювати сторінку після внесення змін в куки.
- Не оновлювати кеш під час оновлення сторінки: Заборонити оновлення кешу при перезавантаженні сторінки.
- Показувати "Редагувати кукі" в контекстному меню браузера: Додати опцію "Редагувати кукі" до контекстного меню браузера.
- Показувати у панелі розробника: Додати плагін у панель розробника.
- Показувати кнопку "Блокувати і видалити всі": Відображати кнопку "Блокувати і видалити всі".
- Показувати ім'я домену біля кожного імені кукі: Відображати ім'я домену біля кожного імені кукі.
- Показувати ім'я домену перед іменем кукі: Відображати ім'я домену перед іменем кукі.
- Вибрати іншу мову: Змінити мову інтерфейсу плагіна.
- Сортувати кукі: Обрати тип сортування кукі.
- Вибери необхідний формат для експорту кукі: Обрати формат експорту кукі.
- Перевизначити максимальний вік для будь-якого кукі: Встановити максимальний вік для кукі.

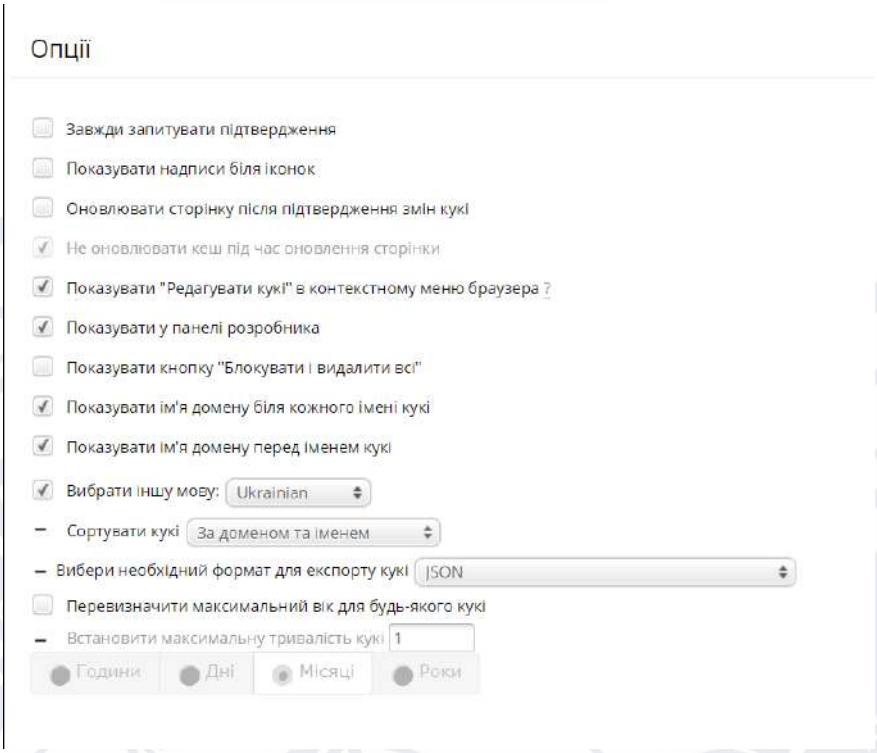


Рисунок 5.17 – Налаштування плагіну

5.3.2 Вікно "Блоковані кукі"

Після кліку на кнопку "Блоковані кукі", ви потрапите до списку заблокованих кукі (рис.5.18).

Рисунок 5.18 – Блоковані кукі

Вікно блокованих куків включає в себе:

- Заблокувати кукі: Кнопка для додавання нового правила блокування.
- Таблиця правил блокування:
  - Домен: Домен, для якого застосовується правило блокування.
  - Ім'я: Назва куки, яка буде заблокована.
  - Значення: Значення куки, яке буде заблоковане.

- Немає жодних правил блокування: Повідомлення, яке відображається, якщо немає жодних правил блокування.
- Форма для додавання нового правила блокування:
  - Домен: Поле для введення домену.
  - Ім'я: Поле для введення назви куки.
  - Значення: Поле для введення значення куки.

5.3.3. Вікно "Захищені кукі"

Після кліку на кнопку "Захищені кукі", ви потрапите до списку захищених кукі (рис.5.19).

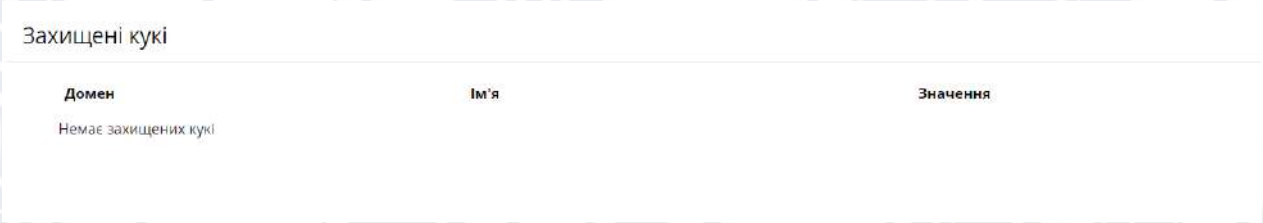


Рисунок 5.19 – Захищені кукі

Вікно захищених куків включає в себе:

- Таблиця захищених кукі:
  - Домен: Домен, для якого застосовується правило захисту.
  - Ім'я: Назва куки, яка є захищеною.
  - Значення: Значення куки, яке є захищеним.
- Немає захищених кукі: Повідомлення, яке відображається, якщо немає жодних захищених кукі.

Висновки до розділу 5

У шостому розділі детально описано процес розробки та функціональні можливості плагіну "CookieShield". Розглянуто архітектурний аспект, звернувши

увагу на структуру плагіну, включаючи файли Manifest.json, Background.js, Content Script та елементи інтерфейсу користувача.

Проаналізовано різні аспекти безпеки, такі як Content Security Policy (CSP), та JSON, який використовується для передачі даних. Функціональні можливості "CookieShield" ретельно проаналізовано, охоплюючи широкий спектр дій відображення, пошуку, фільтрації, оновлення, видалення, блокування та захисту куків.

Надано можливість імпорту та експорту куків у різних форматах. Налаштування плагіну дозволяють користувачам налаштовувати його поведінку відповідно до їхніх потреб.

ВИСНОВКИ

У ході виконання бакалаврської роботи було глибоко досліджено теоретичні аспекти захисту приватності в Інтернеті, а також детально проаналізовано роль файлів cookie в онлайн середовищі. Особливу увагу було приділено процесу проектування і розробки плагіну "CookieShield", який призначений для забезпечення захисту приватності користувачів. Під час роботи було розглянуто кілька ключових методів захисту, що дозволило сформувати комплексний підхід до захисту особистих даних в онлайн середовищі.

Проведений аналіз ролі файлів cookie виявив їхні функції та вплив на приватність користувачів. Це дало можливість визначити ефективні методи керування цими файлами, що є критично важливим для забезпечення конфіденційності даних. Під час проектування та розробки плагіну "CookieShield" були визначені ключові вимоги та функціональні можливості, які забезпечують високий рівень захисту приватності користувачів.

В процесі аналізу технологій та інструментів, використаних для розробки плагіну, було встановлено, що вибраний технологічний стек є простим, гнучким та ефективним для досягнення поставлених цілей. Архітектура плагіну "CookieShield", описана у роботі, включає різноманітні компоненти, що дозволяють ефективно керувати файлами cookie та забезпечувати їхню безпеку.

Усі ці аспекти, детально розглянуті у роботі, сприяли успішній розробці та функціонуванню плагіну "CookieShield". Плагін надає користувачам зручний і ефективний інструмент для захисту їхньої приватності в онлайн середовищі. Ретельно проаналізовані теоретичні та практичні аспекти захисту приватності стали основою для створення надійного та функціонального продукту, який відповідає сучасним вимогам щодо безпеки особистих даних.

СПИСОК ВИКОРИСТАНИХ ПОСИЛАНЬ

1. Блог проекту Mozilla. URL: http://blog.mozilla.org/blog/2008/07/02/were-official (дата звернення 19.03.2024).
2. Кукі (інтернет). URL: https://uk.wikipedia.org/wiki/Кукі_(інтернет) (дата звернення 11.02.2024).
3. Що таке файли cookies та навіщо вони потрібні. URL: https://ssl.com.ua/blog/ukr/what-are-cookies/ (дата звернення 09.12.2023).
4. Статистичний портал StatCounter. URL: http://gs.statcounter.com/#browser-ww-monthly-201002-201305 (дата звернення 03.04.2024).
5. Звіт "Порівняльний аналіз безпеки браузерів 2019" компанії NSS Labs. URL: https://www.nsslabs.com/reports/categories/test-reports/browser-security (дата звернення 13.05.2024).
6. Стівен Шафер. HTML, XHTML та CSS. Біблія користувача, 5-е видання = Біблія HTML, XHTML та CSS, Видання. — М.: "Діалектика", 2010. — 656 с.
7. Bootstrap. URL: https://getbootstrap.com/docs/5.0/customize/overview/ (дата звернення 21.11.2023).
8. Кібербезпека: Все Що Необхідно Знати Кожному Користувачу Мережі Інтернет. URL: https://www.ukraine-lifehacker.com/kiberbezpeka-vse-shcho-neobkhidno-znaty (дата звернення 07.01.2024).
9. Вплив використання файлів cookie на конфіденційність користувачів в Інтернеті. URL: https://beitrans.com/polityka-konfidentsiynosti-ta-failiv-cookie/ (дата звернення 26.02.2024).
10. Захист конфіденційності в сучасних веб-браузерах. URL: https://journals.nupp.edu.ua/sunz/article/view/3058 (дата звернення 27.05.2024).
11. Аналіз методів захисту приватності в веб-браузерах.URL: https://irek.ase.md/xmlui/bitstream/handle/123456789/2039/STAVER%20Mihaela_Conf_September%2024th-25th_%202021_Vol_2.pdf?sequence=1&isAllowed=y (дата звернення 27.10.2023).

12. Захист приватності в Інтернеті: Посібник для користувачів. URL: http://cyberpeace.org.ua/files/zahist_prava_na_privatnist__koristuvaciv_internet.pdf (дата звернення 03.03.2024).
13. "Приватність в Інтернеті". URL: https://en.wikipedia.org/wiki/Privacy (дата звернення 03.04.2024).
14. "Як захистити свою приватність в Інтернеті". URL: https://www.nytimes.com/privacy (дата звернення 28.03.2024).
15. "Privacy perceptions and norms in youth and adults". URL: https://www.researchgate.net/publication/331662461_Privacy_perceptions_and_norms_in_youth_and_adults (дата звернення 17.05.2024).
16. "Three Browser Storage Mechanism". URL: https://www.telerik.com/blogs/three-browser-storage-mechanism (дата звернення 13.12.2023).
17. "Cookie Consent: Ensuring User Privacy in the Digital Age". URL: https://medium.com/@samsmithuk972/cookie-consent-ensuring-user-privacy-in-the-digital-age-2ab8d9558d8a (дата звернення 01.12.2023).
18. "Cookie Management". URL: https://www.hydrao.com/blog/cookie-management/?lang=en (дата звернення 16.11.2024).
19. "Exploring the Pros and Cons of Using Cookies in Web Development". URL: https://medium.com/@thecodingblog/exploring-the-pros-and-cons-of-using-cookies-in-web-development-3b89893c2f0 (дата звернення 19.10.2023).
20. "How Do You Secure Your Web Projects: Plugins & Extensions". URL: https://www.linkedin.com/advice/0/how-do-you-secure-your-web-projects-plugins-extensions (дата звернення 01.04.2024).
21. "The architecture of the dynamic browser extension privacy detection framework". URL: https://www.researchgate.net/figure/The-architecture-of-the-dynamic-browser-extension-privacy-detection-framework_fig3_349826311 (дата звернення 14.10.2023).
22. Jack M. KHTML vs. Gecko vs. Trident vs.Presto: Behind the Browser / M. Jack//ECT NewsNetwork : news sites TechNewsWorld.com. URL: https://www.linuxinsider.com/story/khtml-vs-gecko-vs-trident-vs-presto-behind-

the-browser-59309.html (дата звернення 03.03.2024).

23. "Інформаційна та Кібербезпека: соціотехнічний аспект" / В. Л. Бурячок, В. Б. Толубко, В. О. Хорошко, С. В. Толюпа, 2015. 288 с.
24. "Кібербезпека мереж наступного покоління" / О.О. Вараксін, Є.В. Василіу, С.М. Горохов, В.Й. Кільдишев, В. Г. Кононович, 2013. 240 с.
25. "Кібербезпека: ризики та заходи" / Ю.П. Лісовська, 2019. 272 с.
26. "Кібербезпека: освіта, наука, техніка" / І.С. Пазиніна, Р.О. Корчомний, 2022. 166 с.